Complexity, Algorithms, and Heuristics of Influence Maximization

by

Biaoshuai Tao

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2020

Doctoral Committee:

Associate Professor Chris Peikert, Co-Chair
Assistant Professor Grant Schoenebeck, Co-Chair
Professor Wei Chen
Professor Seth Pettie
Assistant Professor Daniel Romero

Biaoshuai Tao

bstao@umich.edu

ORCID iD: 0000-0003-4098-844X

Acknowledgments

First, I would like to thank my advisor Grant Schoenebeck. He has provided me invaluable advice in my research and trained me in various useful research skills. He has also provided me all the opportunities of learning, visiting, travelling and conferences that I have ever requested.

In addition to my advisor, I would like to thank Wei Chen. As an expert in influence maximization, he has kindly given me many invaluable suggestions and shown me new research sub-areas related to this thesis.

I would also like to thank my other coauthors and collaborators on this thesis: Fang-Yi Yu and Binghui Peng. It has been a very pleasant experience to work with them.

I thank the other members of my thesis committee, Chris Peikert, Seth Pettie and Daniel Romero, for their helpful feedback on this dissertation and throughout graduate school.

I would also like to thank all of my coauthors and collaborators on my other research areas unrelated to this thesis. Especially, I would like to thank Xiaohui Bei for his helpful suggestions and ideas for my work in fair division. I would also like to thank Yuqing Kong, Hongjun Wu, Ning Chen, Fedor Duzhin, Guangda Huzhang, Jiajun Wu, Endong Yang, Xia Hua, Andrew Gitlin, Laura Balzano and John Lipor who have worked with me on various other research areas.

I thank all of my friends and family. Thanks to my parents, my wife, and my parents-in-law, for their support during these five years.

Finally, I would like to give my best wishes to my new-born child Jingce Tao (nickname: little bamboo).

Table of Contents

Chapter

Part II Nonsubmodular Influence Maximization 126

List of Figures

List of Tables

List of Appendices

ABSTRACT

People often adopt improved behaviors, products, or ideas through the influence of friends. This is modeled by *cascades*. One way to spread such positive elements through society is to identify those most influential agents—those that cause the maximum spread, and initiate the spread by seeding them. However, this strategy has a key difficulty: finding these influential seed nodes. This is difficult even if both the network structure and the way the cascade spreads are known. In the *influence maximization problem*, a central planner is given a graph and a limited budget k, and he needs to pick k seeds such that the expected total number of infected vertices in the graph at the end of the cascade is maximized. This problem plays a central role in viral marketing, outbreak detection, rumor controls, etc.

This thesis focuses on computational complexity, approximability and algorithm/heuristic design aspects of the influence maximization problem, with both *submodular* and *nonsubmodular* diffusion models. The first part of the thesis studies submodular influence maximization mainly in the computational complexity and algorithm analysis aspects, which includes some breakthroughs in understanding the approximability of submodular influence maximization and the theoretical performance of the well-studied greedy algorithm. The second part of the thesis focuses on nonsubmodular influence maximization. New sociologically founded nonsubmodular diffusion models are proposed, and we show how the seeding strategy for nonsubmodular diffusion models is fundamentally different compared to submodular diffusion models.

CHAPTER 1

Introduction

In social networks, *cascades* model the phenomena that agents receive certain information, adopt certain products, or take up certain political opinions from their neighbors due to their influence. We can model a cascade on a graph as a stochastic mapping from a subset of vertices, called the *seeds*, to another set of vertices that always contain the seed vertices, called the *infected vertices.* A graph models the social network, the seeds represent the initial set of agents that have a certain property (e.g., adopted a certain product, endorsed a political bill, etc.), and the infected vertices represent those agents that eventually obtain the said property, due to the direct or indirect influence of the initial agents. This process is normally broken down into iterations. In the initial iteration, only the seeds are considered infected. In each future iteration, a not-yet-infected vertex will be infected with certain probability, and this probability is decided by the set of its infected neighbors and the diffusion model considered. The cascade terminates when there is no new infection in an iteration.

In the *influence maximization problem* (InfMax), initially posed by Domingos and Richardson [26, 62], a central planner is given a graph and a limited budget k, and he needs to pick k seeds such that the expected total number of infected vertices in the graph at the end of the cascade is maximized. This problem plays a central role in viral marketing—a marketing strategy of advertising products by giving the products to a certain number of users for free in the hope that they will recommend these products to their friends. It also plays an important role in outbreak detection, rumor controls, etc. InfMax has been studied extensively in the past literature both theoretically and practically (cf. Chen et al. [19], Li et al. [54]).

1.1 Diffusion Models: Submodular versus Nonsubmodular

Perhaps the two best known diffusion models are the *independent cascade model* and the *linear threshold model*, both introduced in the seminal paper Kempe et al. [44]. These two models are studied almost exclusively in the past literature of this field. In the independent cascade model, a newly-infected (or seed) vertex u infects each of its not-yet-infected neighbors v with a certain fixed probability p_{uv} independently. See Sect. 2.1.3 for details about the independent cascade model. In the linear threshold model, in the case the graph is unweighted, each non-seed vertex has a threshold independently sampled uniformly from the interval $[0, 1]$, and becomes infected when the fraction of its infected neighbors exceeds this threshold; in the case the graph is edge-weighted, it is assumed that the sum of the weights of all incoming edges for each vertex is at most 1, and a vertex is infected if the sum of the weights of all the incoming edges connecting from its infected neighbors exceeds its threshold, where, again, the threshold is sampled uniformly from the interval $[0, 1]$. See Sect. 2.1.4 for details about the linear threshold model.

Both models were shown to be *submodular* [44], meaning that the marginal influence from a newly infected vertex u to its not-yet-infected neighbor v decreases as v have an increasing number of existing infected neighbors. More formally, letting T be the set of v's infected neighbors and letting $P(T)$ be the probability that v will be infected given the set of infected neighbors T, a diffusion model is submodular if $P(T_1 \cup \{u\}) - P(T_1) \geq P(T_2 \cup \{u\}) - P(T_2)$ whenever $T_1 \subseteq T_2$. It is widely known that, for the InfMax problem with a submodular diffusion model, the greedy algorithm which iteratively picks the seed that has the most marginal influence achieves a $(1-1/e)$-approximation. This includes the independent cascade model and the linear threshold model. On the hardness side, it is shown by Kempe et al. [44] that InfMax is NP-hard to approximate to within a factor of $(1 - 1/e + \varepsilon)$ for any ε for the independent cascade model, and finding the exact optimal solution for InfMax for the linear threshold model is NP-hard, even for undirected graphs. Refer to Sect. 1.2.1 for more details about these results.

Diffusion models that violate the submodularity property are called *nonsubmodular* cascades (or sometimes complex cascades). A well-known nonsubmodular diffusion model, which is also the most extreme one, is the *r-complex contagion* [10, 11, 28, 39], where a node is infected if and only if at least r of its neighbors are infected, also known as the *bootstrap percolation*, or the *fixed threshold model* (we will define these terms slightly differently in this thesis, and see Sect. 2.1.5 for details). In nonsubmodular contagion models, like the r-complex contagion model, the marginal probability of being infected may increase as more neighbors are infected. For example, if $r = 2$, then the first infected neighbor has zero marginal impact, but the second infected neighbor causes this vertex to become infected with probability 1. Unlike submodular contagions, nonsubmodular contagions can require well-connected regions to spread [9].

InfMax becomes qualitatively different in nonsubmodular diffusion models. The greedy approaches, in particular, can perform poorly in the nonsubmodular setting [2]. Moreover, in contrast to the submodular case which has efficient constant approximation algorithms, for general nonsubmodular cascades, it is NP-hard even to approximation influence maximization to within an $n^{1-\varepsilon}$ factor of optimal for any constant $\varepsilon > 0$ [45] (where n is the number of vertices), and the inapproximability results have been extended to several more restrictive nonsubmodular models [20, 53]. The intrinsic reason why nonsubmodular influence maximization is hard is that one needs to take into account the potential synergy of multiple seeds. This is in sharp contrast to submodular influence maximization, where the submodularity enables a seed-picker to consider placing seeds one at a time in a myopic way, as is done in the greedy algorithm.

Motivations for different diffusion models Submodular diffusion models have been studied mostly in the past literature for many reasons. They are conceptually simple, natural and mathematically clean. As remarked by Kempe et al. [44], the independent cascade model is conceptually the simplest model based on the work in interacting particle systems from probability theory [27, 55], and has also been studied in the context of marketing [35, 34]. The linear threshold model can be motivated in two different natural ways. Firstly, as remarked by [44], it can be viewed as a variant to those threshold models (such as r-complex contagion) with uncertainty to the thresholds of the vertices, which naturally models that the advertiser does not know how likely it is for each individual agent to adopt a new product. Secondly, if we want that a vertex's infection probability is linear in terms of the infected neighbors, the linear threshold model becomes the simplest model to capture this linearity.

Another important reason that submodular diffusion models are popular is its tractability. As we have mentioned, the greedy algorithm always achieves a $(1-1/e)$-approximation for submodular InfMax, while nonsubmodular InfMax is substantially harder to approximate even under very restrictive simple models. Our result in Chapter 6 shows that InfMax is inapproximable for almost all nonsubmodular diffusion models, including those that are "very close to" submodular ones. This provides strong evidences that restrictive assumptions on diffusion model along cannot make InfMax tractable, and we need to additionally make some assumptions on the network topology (which motivates our work in Chapter 7 and 8).

Despite the intractability of nonsubmodular InfMax, many sociological studies reveal that the social influence in our daily life can be nonsubmodular [4, 50, 63, 75, 32]. Specifically, the submodularity is violated in a particular way: the second infected neighbor usually has more marginal influence than the first (see Sect. 6.1 for more details). This also suggests that the cascade in *social* networks is more complex than it is suggested by the independent cascade model: the probability that u infects v *depends* on the set of v's existing infected neighbors, although the independent cascade model and other submodular diffusion models may more accurately describe non-social cascades, like virus spreading.

1.2 Related Work

1.2.1 On Approximability of Influence Maximization

Kempe et al. [44] showed that the simple greedy algorithm that iteratively chooses the seed with the maximum marginal influence obtains a $(1-1/e)$ factor approximation to InfMax with both the independent cascade model and the linear threshold model. The same set of authors later extended this result to a family of submodular cascades, called the *decreasing cascade model*, which captures the independent cascade model and the linear threshold model as special cases [45]. Mossel and Roch [58] showed that local submodularity implies global submodularity: if the diffusion model is submodular, then the global influence function that maps a set of seeds to the expected total number of infections is a submodular function (see Theorem 2.4 for details). This implies that the greedy algorithm achieve a $(1-1/e)$ factor approximation whenever the diffusion model is submodular, as the greedy algorithm achieves $(1-1/e)$ approximation whenever the objective function is monotone and submodular, which is known by Nemhauser et al. [59] and also remarked by Kempe et al. [44].

On the hardness or inapproximability side, Kempe et al. [44] showed that InfMax on both the independent cascade model and the linear threshold model is NP-hard.

For the independent cascade model, Kempe et al. [44] considered a reduction from the maximum coverage problem preserving the approximation factor. Since Feige [29] showed that the maximum coverage problem is NP-hard to approximate within factor $(1 - 1/e + \varepsilon)$ for any constant $\varepsilon > 0$, the same inapproximability factor holds for the independent cascade InfMax. Therefore, the gap between upper bound and lower bound for the independent cascade InfMax is closed (up to lower order terms). Note that, however, the reduction depends heavily on the directed nature of the graph, and there is no known hardness result for the independent cascade InfMax with undirected graphs. For the linear threshold model, Kempe et al. [44] reduced the problem from the vertex cover problem such that the graph, being undirected, is the same in the two problems. It is easy to see that we can infect all the vertices in the graph with probability 1 by k seeds if and only if the graph admits a vertex cover of size k (the k seeds are exactly the vertex cover). This reduction shows InfMax with the linear threshold model is NP-hard even for undirected graphs, but does not imply any hardness of approximation. In summary, we do not know any hardness of approximation result for InfMax with the independent cascade model for undirected graphs, nor do we know any hardness of approximation result for InfMax with the linear threshold model even for directed graphs. In Chapter 3, we show that InfMax is APX-hard for both diffusion models with undirected graphs, which provides the thus far missing APX-hardness result.

If the diffusion model can be nonsubmodular, Kempe et al. [44] showed that InfMax is NP-hard to approximate to within a factor of $n^{1-\varepsilon}$ for any $\varepsilon > 0$, where n is the number of vertices. Many work after this [12, 53, 73] are dedicated to studying specific nonsubmodular models (or study optimization on nonsubmodular objective functions), and unfortunately most of those models, even for those very restrictive models, admit polynomial inapproximability factors (n^τ for some constant $\tau > 0$, or even worse, $n^{1-\varepsilon}$ for any $\varepsilon > 0$). Our work in Chapters 6, 7 and 8 studies the approximability of nonsubmodular InfMax on various diffusion models and network structures, and provides a general intuition on how the seeding strategy is different for the nonsubmodular InfMax (compared with the one for submodular InfMax).

1.2.2 Greedy Algorithm And Its Approximation Guarantee

The *greedy algorithm* iteratively picks a seed that has the maximum marginal gain to the expected total number of infected vertices. We will describe it formally in Sect. 2.2. For InfMax, nearly all the known algorithms are based on the greedy algorithm. Some of them improve the running time of the original greedy algorithm by skipping vertices that are known to be suboptimal [51, 37], while the others improve the scalability of the greedy algorithm by using more scalable algorithms to approximate the expected total influence [8, 71, 72, 22, 60, 40] or computing a score of the seeds that is closely related to the expected total influence [15, 18, 17, 38, 43, 31]. Due to its prevalence, the greedy algorithm will be one of the main focuses in this thesis. We remark that there do exist InfMax algorithms that are not based on greedy [7, 33, 2], but they are typically for nonsubmodular diffusion models.

On the approximation guarantee, we know that the greedy algorithm always achieves a $(1-1/e)$-approximation for InfMax with the independent cascade model, the linear threshold model, or even the general submodular diffusion model, as mentioned earlier. It is easy to see that this is tight if the network is directed, and this is true for both the independent cascade model and the linear threshold model. If the graph is undirected, Khanna and Lucier [47] showed that the greedy algorithm achieves a $(1 - 1/e + c)$-approximation for some constant $c > 0$. No previous result is known for the exact performance of the greedy algorithm for the linear threshold model, and our results in Chapter 4 fill in this missing piece.

1.2.3 Adaptive Influence Maximization

The InfMax problem has recently been studied in the *adaptive* setting, where the seed-picker can observe the spread whenever a seed is chosen and chooses the seeds iteratively and adaptively. In the *full-adoption feedback model*, after selecting each seed, the seed-picker observes all the infected vertices until the cascade dies out. In the *myopic feedback model*, the seed-picker only observes whether each neighbor of the chosen seed is infected.

The greedy algorithm can be naturally extended to this adaptive setting, and the adaptive greedy algorithm has been studied before. Golovin and Krause [36] showed that InfMax with the independent cascade model and full-adoption feedback is *adaptive submodular*, which implies that the adaptive greedy algorithm obtains a $(1 - 1/e)$-approximation to the adaptive optimal solution. On the other hand, InfMax for the independent cascade model with myopic feedback, as well as InfMax for the linear threshold model with both feedback models, are not adaptive submodular. In particular, the adaptive greedy algorithm fails to obtain a $(1-1/e)$-approximation for the independent cascade model with myopic feedback [61]. Previous work has also been focused on improving the running time of the adaptive greedy algorithm [40].

To measure the power of adaptivity, the *adaptivity gap* was proposed [36], which is defined by the ratio between the performance of the *optimal* adaptive algorithm and the performance of the *optimal* non-adaptive algorithm. Peng and Chen [61] showed that the adaptivity gap for the independent cascade model with myopic feedback is at most 4 and at least $e/(e-1)$, and they also showed that both the adaptive and non-adaptive greedy algorithms perform a $0.25(1-1/e)$-approximation to the adaptive optimal solution. The adaptivity gap for the independent cascade model with full-adoption feedback, as well as the adaptivity gap for the linear threshold model with both feedback models, are still open problems, although there is some partial progress [14]. There are some partial progress on this: Chen and Peng [14] showed that the adaptive gap for the independent cascade model with the full-adoption feedback is upper-bounded by a constant if the graph is an in-arborescence (a directed tree with the root being the sink), an out-arborescence (a directed tree with the root being the source), or a directed bipartite graph (where all edges are from one side of the vertices to another).

1.2.4 Empirical Work

Following the work of Kempe, Kleinberg, and Tardos [44, 45], there was extensive work to solve InfMax based on the alternative implementations of the greedy algorithm designed to be efficient and scalable. Leskovec et al. [51] and Goyal et al. [37] proposed algorithms CELF and CELF++ respectively which improve the greedy algorithm by skipping some of the vertices in each iteration that are known to be suboptimal, but still relying on Monte-Carlo method to evaluate the influence of a given seed set. Although the $(1-1/e)$ approximation guarantee is secured, algorithms based on Monte-Carlo method are limited in their scalability [3].

To avoid using Monte-Carlo method to evaluate the influence of a given seed set, another category of algorithms are based on constructing reverse reachable sets (more details have been discussed in Sect. 2.2.2). Theoretically, these algorithms achieve $(1-1/e-\varepsilon)$ approximation, and arbitrarily small $\varepsilon > 0$ can be obtained by sampling more reverse reachable sets.[1]

Another method similar to reverse reachable sets sampling is *snapshots* sampling, based on which the algorithms StaticGreedy [22] and PMC [60] were invented. Instead of considering the set of vertices that can infect a particular vertex, these algorithms sample a sufficient number of "snapshots" of the graph, each of which is obtained by

[1] There was a mistake in the algorithm IMM which makes IMM possibly fail to achieve the said approximation guarantee. Chen [13] spotted and corrected this mistake.

including each edge with probability proportional to its edge-weight.

Other than the above-mentioned algorithms that preserve the theoretical approximation guarantee, heuristics have also been proposed, and most of them are based on computing a “score” for a seed set that are closely correlated to the expected number of infected vertices. These algorithms include the Degree Discount Heuristics [15], PMIA [16], LDAG [18], SIMPATH [38], IRIE [43] and EaSyIM [31]. Our heuristic in Sect. 3.6 is perhaps most related to the Degree Discount Heuristics, as both heuristics only examine the neighbors or a restrictive local area around the seeds. As we will see in Sect. 3.6, by exploiting our theoretically provable upper bounds in Sect. 3.5, our heuristics produce seeds with better qualities.

A benchmarking study of most of these above-mentioned algorithms has been done by Arora et al. [3], and it was observed that TIM$^+$, IMM and PMC provide the seeds with the best qualities while also maintain moderate scalability. As a result, we will mainly compare the performance of our heuristics to those reverse-reachable-set-based algorithms.

Rather than working on the independent cascade model and the linear threshold model as it is in all the work above, empirical study for nonsubmodular InfMax has also been studied. For example, Angell and Schoenebeck [2] provided a dynamic programming based InfMax algorithm which is fundamentally different than most algorithms that are more or less based on greedy.

1.3 Overview of this Thesis

In Chapter 2, we introduce basic diffusion models and notations related to InfMax that are used throughout the entire thesis. The remaining part of the thesis is then split into two parts. The first part focuses on InfMax with submodular diffusion model. It includes Chapter 3, Chapter 4 and Chapter 5. The second part of this thesis focuses on InfMax with nonsubmodular diffusion models. It includes Chapter 6, Chapter 7 and Chapter 8.

In general, my work on submodular InfMax in the first part mainly focuses on those fundamental theoretical problems about complexity and approximability. My work on nonsubmodular InfMax in the second part, on the other hand, mainly focuses on new diffusion models and network models that are well motivated by empirical/sociological studies, and studies how the seeding strategy is fundamentally different for nonsubmodular InfMax.

1.3.1 Part I: Submodular Influence Maximization

In Chapter 3, we show that InfMax with both the independent cascade model and the linear threshold model are APX-hard (i.e., there exists a constant $\tau > 0$ such that InfMax is NP-hard to approximate within a factor of $(1 - \tau)$), even for undirected graphs. This is one of the major breakthroughs in understanding the approximability of submodular InfMax. As mentioned in Sect. 1.2.1, it provides the thus far missing hardness-of-approximation results, and it rules out the possibility of Polynomial Time Approximation Scheme (PTAS).

In both Chapters 4 and 5, we study the standard greedy algorithm that has been used almost exclusively in the past literature. In Chapter 4, we discuss the approximation guarantee for the greedy algorithm. In particular, we show that, for the linear threshold model with undirected graphs, the greedy algorithm achieves at least a $(1-(1-1/k)^k+\Omega(1/k^3))$-approximation and at most a $(1-(1-1/k)^k+O(1/k^{0.2}))$-approximation (recall that k is the number of the seeds). This indicates that the approximation guarantee for the greedy algorithm in general submodular InfMax, $(1 - 1/e)$, is still asymptotically correct under this special case. This is in sharp contract to the result from Khanna and Lucier [47] mentioned in Sect. 1.2.2 for the independent cascade model.

In Chapter 5, we study InfMax and the greedy algorithm in the adaptive setting. We define the *greedy adaptivity gap* which is given by the ratio of the performance of the adaptive greedy algorithm versus the performance of the conventional nonadaptive greedy algorithm. The greedy adaptivity gap provides a better measurement for the power of adaptivity in the practical sense than the adaptivity gap (mentioned in Sect. 1.2.3, which compares the adaptive *optimal* algorithm to the nonadaptive *optimal* algorithm): firstly, our APX-hardness results in Chapter 3 indicate that it is hard to achieve, or even approach to, the optimal solution in practice; secondly, as mentioned in Sect. 1.2.2, the greedy algorithm is the one that is used exclusively in practice. In Chapter 5, we give some characterizations on the greedy adaptivity gap. In particular, we show that the infimum of the greedy adaptivity gap is $(1 - 1/e)$ for both the independent cascade model and the linear threshold model with both the full-adoption feedback model and the myopic feedback model (see Sect. 1.2.3 for informal definitions of the two feedback model, and see Sect. 5.2.1 for formal definitions), meaning that the adaptive greedy algorithm can only achieve a $(1-1/e)$-fraction of the performance of the nonadaptive greedy algorithm. This shows that, surprisingly, the adaptive greedy algorithm can perform worse than the nonadaptive greedy algorithm. Nevertheless, it never perform too bad. On the other hand, for general submodular diffusion model with the full-adoption feedback, the supremum of the greedy adaptivity gap is infinity, which indicates that the adaptive greedy algorithm can perform significantly better than its nonadaptive counterpart under this setting.

Chapter 3 is based on the paper [67], Chapter 4 is based on the paper [70], and Chapter 5 is based on the paper [21].

1.3.2 Part II: Nonsubmodular Influence Maximization

In Chapter 6, we point out that a few sociological studies show that many cascade processes in our real life are nonsubmodular. Based on the observations of these studies, we propose a new nonsubmodular diffusion model called the *2-quasi-submodular diffusion model*. This diffusion model can be arbitrarily close to a submodular model by adjusting the parameters. We show that, even when the parameters of the 2-quasi-submodular diffusion model are fixed in advance, InfMax is still NP-hard to approximate to within factor n^τ, where $\tau > 0$ is a constant depending on the parameters of the model. This result can be viewed as a threshold result: if the model is submodular, InfMax admits a $(1 - 1/e)$-approximation algorithm; if the model is only slightly nonsubmodular, no constant-factor or even subpolynomial-factor approximation algorithm is possible (unless $\mathrm{P} = \mathrm{NP}$).

Since even strong assumptions on the diffusion models fail to make InfMax tractable, another natural approach is to see if assumptions on network structures can. In Chapter 7, we propose the (stochastic) hierarchical blockmodel, which is a special case of the well-known (stochastic) blockmodel where the communities in the network form a hierarchical structure. We study the algorithmic complexity of InfMax under the (stochastic) hierarchical blockmodel with the standard r-complex contagion model. We show that InfMax is NP-hard to approximate to within factor $n^{1-\varepsilon}$ for any constant $\varepsilon > 0$ under this setting, if the thresholds r of different vertices need not be the same. On the other hand, in Chapter 8, we show that, under some further mild assumptions, InfMax is tractable if all vertices have a same threshold r. In particular, we show that the optimal seeding strategy is to put all the seeds in a single community. This justifies the intuition that putting seeds together creates synergy for nonsubmodular InfMax, which is in sharp contrast to the seeding strategy for submodular InfMax where seeds are put far apart to avoid waste of influence.

Both Chapter 6 and 7 are based on the papers [66] and [68]. Chapter 8 is based on the paper [69].

CHAPTER 2

Preliminaries

Unless otherwise specified, all graphs in this thesis are simple, unweighted and directed by default. Given a graph $G = (V, E)$ and a vertex $v \in V$, let $\Gamma(v)$ and $\deg(v)$ be the set of in-neighbors and the in-degree of v respectively. Let $\Gamma^o(v)$ and $\deg^o(v)$ be the set of out-neighbors and the out-degree of v respectively. In this thesis, we will consistently use $n = |V|$ to denote the number of vertices in the graph. If a graph $G = (V, E)$ is undirected, it is treated as a directed graph with anti-parallel edges such that $(u, v) \in E$ if and only if $(v, u) \in E$.

2.1 Cascade and Diffusion Models

In general a *cascade* on a graph is a stochastic mapping from a subset of vertices—the *seeds*, to another set of vertices that always contain the seed vertices—the *infected vertices*. In the next five subsections, we define the InfMax problem with various different diffusion models. The notations defined there are summarized in Table 2.1, which will be used throughout the thesis.

2.1.1 General Threshold Model and Influence Maximization

The cascades we study in this thesis all belong to the general threshold model [44], which captures the local decision-making of vertices.

Definition 2.1 (Kempe et al. [44]). The *general threshold model*, $I_{G,F}$, is defined by a graph $G = (V, E)$ and for each vertex v a monotone *local influence function* $f_v : \{0, 1\}^{|\Gamma(v)|} \to [0, 1]$ with $f_v(\emptyset) = 0$. Let $F = \{f_v \mid v \in V\}$.

On an input seed set $S \subseteq V$, $I_{G,F}(S)$ outputs a set of infected vertices as follows:

notation	meaning
$G = (V, E)$	a directed simple graph with vertex set V and edge set E
$\Gamma(v)$	set of v's in-neighbors
$\deg(v)$	in-degree of v
$\Gamma^o(v)$	set of v's out-neighbors
$\deg^o(v)$	out-degree of v
n	total number of vertices, $\lvert V\rvert$
S	set of seeds
k	number of seeds
f_v	local influence function (Definition 2.1)
$F = \{f_v \mid v \in V\}$	collection of local influence functions for all vertices
$I_{G,F}$	general threshold model with G and F (Definition 2.1)
$\sigma_{G,F}$	global influence function: $\sigma_{G,F}(S) = \mathbb{E}\left[\lvert I_{G,F}(S)\rvert\right]$
Trig_v	triggering set of vertex v (Definition 2.5)
$\mathcal{D}_v$	distribution of v's triggering set (Definition 2.5)
$D = \{\mathcal{D}_v \mid v \in V\}$	collection of distribution of triggering sets for all vertices
$I_{G,D}$	triggering model with G and D (Definition 2.5)
ϕ	realization (see the second paragraph after Definition 2.5)
`ICM`	independent cascade model (Definition 2.7)
`UICM`	uniform independent cascade model (Definition 2.9)
`WICM`	weighted independent cascade model (Definition 2.10)
`LTM`	linear threshold model (Definition 2.11)
`ULTM`	uniform linear threshold model (Definition 2.13)
`BP`	bootstrap percolation (Definition 2.15)
$R = \{r_v \mid v \in V\}$	collection of thresholds for all vertices for `BP`

Table 2.1: Notations used in this thesis

1. Initially, only vertices in S are infected, and for each vertex v the threshold θ_v is sampled uniformly at random from the interval $(0, 1]$ independently.[1]
2. In each subsequent round, a vertex v becomes infected if the influence of its infected in-neighbors, $IN_v \subseteq \Gamma(v)$, exceeds its threshold: $f_v(IN_v) \geq \theta_v$.
3. After a round where no additional vertices are infected, the set of infected vertices is the output.

$I_{G,F}$ in Definition 2.1 can be viewed as a random function $I_{G,F} : \{0,1\}^n \to \{0,1\}^n$. In addition, if the threshold θ_v is fixed for each vertex v, then $I_{G,F}$ is deterministic. Let $\sigma_{G,F}(S) = \mathbb{E}[|I_{G,F}(S)|]$ be the *expected* total number of infected vertices due to the influence of S, where the expectation is taken over the samplings of the thresholds of all vertices. We refer to $\sigma_{G,F}(\cdot)$ as the *global influence function.* Sometimes we write $\sigma(\cdot)$ with the parameters G, F omitted, when there is no confusion. Because each f_v is monotone, it is straightforward to see that σ is monotone.

Definition 2.2. The InfMax problem is an optimization problem which takes as inputs $G = (V, E)$, F, and an integer k, and the goal is to output a seed set $S \subseteq V$ subject to the budget constraint $|S| \leq k$ that maximizes $\sigma_{G,F}(S)$.

In this thesis, we will consistently use $k = |S|$ to denote the number of seeds, or the budget, which is a part of the input to the InfMax instance.

Submodularity A function Φ mapping from a set of elements to a non-negative value is *submodular* if $\Phi(A \cup \{v\}) - \Phi(A) \geq \Phi(B \cup \{v\}) - \Phi(B)$ for any two sets A, B with $A \subseteq B$ and any element $v \notin B$. We formally define the submodularity of a diffusion model, to which we have referred multiple times in the introduction, as follows.

Definition 2.3. A diffusion model characterized by the general threshold model $I_{G,F}$ is *submodular* if the local influence function f_v is a submodular function for each $v \in V$. A diffusion model characterized by the general threshold model is *nonsubmodular* if this is not satisfied.

The following result says that local submodularity implies global submodularity, which is a crucial characterization that eventually establishes the tractability of submodular InfMax: as long as the diffusion model is submodular, the greedy algorithm always achieves a $(1 - 1/e)$-approximation.

[1] The rationale of sampling thresholds *after* the seeds' selection is to capture the scenario that the seed-picker does not have the full information on the agents in a social network, and this setting is standard in the InfMax literature.

Theorem 2.4 (Mossel and Roch [58])**.** *If the general threshold model $I_{G,F}$ is submodular, then the function $\sigma_{G,F}(\cdot)$ is submodular.*

2.1.2 Triggering Model

A common diffusion model that is strictly less general than the general threshold model is the *triggering model.* The triggering model is still general enough to include the well-studied independent cascade model and linear threshold model as special cases. As we will see later, the triggering model is always submodular, and it enables an useful interpretation of the cascade process called the *live edges realization.* This enables a class of powerful and highly-scalable InfMax algorithms discussed in Sect. 2.2.2.

Definition 2.5 (Kempe et al. [44])**.** The *triggering model*, $I_{G,D}$, is defined by a graph $G = (V, E)$ and for each vertex v a distribution $\mathcal{D}_v$ over the subsets of its in-neighbors $\{0, 1\}^{|\Gamma(v)|}$. Let $D = \{\mathcal{D}_v \mid v \in V\}$.

On an input seed set $S \subseteq V$, $I_{G,D}(S)$ outputs a set of infected vertices as follows:

1. Initially, only vertices in S are infected. Each vertex v samples a subset of its in-neighbors $\text{Trig}_v \subseteq \Gamma(v)$ from $\mathcal{D}_v$ independently. We call Trig_v the *triggering set* of v.
2. In each subsequent round, a vertex v becomes infected if a vertex in Trig_v is infected in the previous round.
3. After a round where no additional vertices are infected, the set of infected vertices is the output.

Given v, its triggering set Trig_v, and an in-neighbor $u \in \Gamma(v)$, we say that the edge (u, v) is *live* if $u \in \text{Trig}_v$, and we say that (u, v) is *blocked* if $u \notin \text{Trig}_v$. It is easy to see that, when the triggering sets for all vertices are sampled, $I_{G,D}(S)$ is the set of all vertices that are reachable from S when removing all blocked edges from the graph.

We define a *realization* of a graph $G = (V, E)$ as a function $\phi : E \to \{\mathtt{L}, \mathtt{B}\}$ such that $\phi(e) = \mathtt{L}$ if $e \in E$ is live and $\phi(e) = \mathtt{B}$ if $e \in E$ is blocked. Let $I^{\phi}_{G,D} : \{0, 1\}^n \to \{0, 1\}^n$ be the deterministic function corresponding to the triggering model $I_{G,D}$ with vertices' triggering sets following realization ϕ. We write $\phi \sim D$ to indicate that a realization ϕ is sampled according to $D = \{\mathcal{D}_v\}$.

The theorem below says that the triggering model is a special case of the general threshold model, and it is always submodular.

Theorem 2.6 (Kempe et al. [44])**.** *The triggering model is a special case of the general threshold model $I_{G,F}$ where each $f_v \in F$ is a submodular function.*

The theorem above says that the submodularity of the general threshold model $I_{G,F}$ is a necessary condition for it to be a triggering model. However, this is *not* a sufficient condition. A diffusion model that is captured by a submodular general threshold model but not the triggering model, named *decreasing cascade model*, was discovered in the full version of [44]; this indicates that even the general threshold model with submodular local influence functions is strictly more general than the triggering model. Salek et al. [65] completely characterized the necessary and sufficient condition under which a general threshold model can be captured by a triggering model.

2.1.3 Independent Cascade Model

The diffusion model that has been studied *the most* often is the *independent cascade model*. From now on, we will use the acronym `ICM` to denote the independent cascade model. In `ICM`, each vertex u attempts only once to infect each of its not-yet-infected out-neighbor v with probability $w(u, v)$, where $w(u, v)$ is a parameter for the edge (u, v) that is given as an input to the instance.

Definition 2.7. The *independent cascade model* $\texttt{ICM}_G$ is defined by a directed edge-weighted graph $G = (V, E, w)$ such that $0 \leq w(u, v) \leq 1$ for each $(u, v) \in E$. On input seed set $S \subseteq V$, $\texttt{ICM}_G(S)$ outputs a set of infected vertices as follows:

1. Initially, only vertices in S are infected.
2. In each subsequent round, each vertex u infected in the previous round infects each (not yet infected) out-neighbor v with probability $w(u, v)$ independently.
3. After a round where there is no additional infected vertices, $\texttt{ICM}_G(S)$ outputs the set of infected vertices.

It is easy to see that `ICM` is a special case of both the triggering model and the general threshold model.

Theorem 2.8 (Kempe et al. [44])**.** *$\texttt{ICM}_G$ with $G = (V, E, w)$ is equivalent to the general threshold model $I_{G,F}$ with $f_v(IN_v) = 1 - \prod_{u \in IN(v)}(1 - w(u, v))$. It is also equivalent to the triggering model $I_{G,D}$ where $\mathcal{D}_v$ is defined such that* Trig_v *includes each $u \in \Gamma(v)$ with probability $w(u, v)$ independently.*

Since ICM is a special case of the triggering model, Theorem 2.6 implies that ICM is a submodular diffusion model.

ICM on unweighted graphs ICM in Definition 2.7 is defined upon edge-weighted graphs. However, in many practical scenarios, only unweighted graphs are available as inputs to InfMax instances. We discuss the following two common ways to assign weights to an unweighted graph. In the *uniform independent cascade model*, all the edges have the same weight; in the *weighted independent cascade model*, each edge (u, v) has weight $w(u, v) = 1/\deg(v)$.

Definition 2.9. The *uniform independent cascade model* $\texttt{UICM}_{G,p}$ for $G = (V, E)$ is a special case of the independent cascade model $\texttt{ICM}_G$ with $G = (V, E, w)$, where $w(u, v) = p$ for each $(u, v) \in E$ and $p \in [0, 1]$ is an input parameter.

Definition 2.10. The *weighted independent cascade model* $\texttt{WICM}_G$ for $G = (V, E)$ is a special case of the independent cascade model IC_G with $G = (V, E, w)$, where $w(u, v) = 1/\deg(v)$ for each $(u, v) \in E$.

As mentioned at the beginning of this chapter, when we restrict our attention to undirected graphs, the undirected graph is “bidirected”, i.e., it is viewed as a special directed graph with each undirected edge of the graph being viewed as two anti-parallel edges.

It should then be noticed that modeling cascade process on undirected graphs does not necessarily imply that the influence from u to v is the same as the influence from v to u on an edge (u, v). Although they are the same in the uniform independent cascade model, they are not necessarily the same in the weighted independent cascade model. In the weighted independent cascade model, if u has a larger degree than v, it is less likely that v will infect u than u will infect v, as $w(v, u) = \frac{1}{\deg(u)} < \frac{1}{\deg(v)} = w(u, v)$.

2.1.4 Linear Threshold Model

The second most well-studied diffusion model is the *linear threshold model*, which we will denote by the acronym LTM from now on. The basic idea behind LTM is that the influence from the in-neighbors of a vertex is additive.

Definition 2.11. The *linear threshold model* $\texttt{LTM}_G$ is defined by a directed edge-weighted graph $G = (V, E, w)$ such that $\sum_{u:u\in\Gamma(v)} w(u, v) \leq 1$ for each $v \in V$. On input seed set $S \subseteq V$, $\texttt{LTM}_G(S)$ outputs a set of infected vertices as follows:

1. Initially, only vertices in S are infected, and for each vertex v a *threshold* θ_v is sampled uniformly at random from $(0,1]$ independently.

2. In each subsequent round, a vertex v becomes infected if $\sum_{u:u\in IN_v} w(u,v) \geq \theta_v$, where again IN_v is the set of v's infected neighbors.

3. After a round where there is no additional infected vertices, $\texttt{LTM}_G(S)$ outputs the set of infected vertices.

The constraint $\sum_{u\in\Gamma(v)} w(u,v) \leq 1$ in the definition above is crucial for the linear threshold model, as otherwise the probability a vertex v is infected is no longer "linear" in terms of the influence from its infected in-neighbors, and the resultant model becomes fundamentally different.

It is straightforward to see why LTM is a special case of the general threshold model. It may be less obvious that LTM is a special case of the triggering model and what does the distribution of the triggering set for each vertex look like. Kempe et al. [44] gave the following characterization.

Theorem 2.12 (Kempe et al. [44]). *$\texttt{LTM}_G$ with $G = (V,E,w)$ is equivalent to the general threshold model $I_{G,F}$ with $f_v(IN_v) = \sum_{u\in IN(v)} w(u,v)$. It is also equivalent to the triggering model $I_{G,D}$ where $\mathcal{D}_v$ is defined as follows: order v's in-neighbors $u_1,\ldots,u_T$ arbitrarily, sample a real number γ in $[0,1]$ uniformly, and*

$$\mathrm{Trig}_v = \begin{cases} \{u_t\} & \text{if } \gamma \in \left[\sum_{i=1}^{t-1} w(u_i,v), \sum_{i=1}^{t} w(u_i,v)\right) \\ \emptyset & \text{if } \gamma \geq \sum_{i=1}^{T} w(u_i,v) \end{cases}.$$

Intuitively, Trig_v *includes at most one of v's in-neighbors such that each u_t is included with probability $w(u_t,v)$.*

For an intuition of the theorem above, consider a not-yet-infected vertex v and a set of its infected neighbors $IN_v \subseteq \Gamma(v)$. v will be infected by vertices in IN_v with probability $\sum_{u:u\in IN_v} w(u,v)$, as $\Pr\left(\theta_v \leq \sum_{u:u\in IN_v} w(u,v)\right) = \sum_{u:u\in IN_v} w(u,v)$. In the case where v becomes infected, we can attribute its infection to exactly one of its infected neighbors. The infection will be attributed to neighboring infected vertex u with probability equal to $w(u,v)$ (in which case $\mathrm{Trig}_v = \{u\}$). Overall, the probability that v includes an incoming edge from $\{(u,v) : u \in IN_v\}$ is exactly $\sum_{u:u\in IN_v} w(u,v)$.

Since LTM is a special case of the triggering model, Theorem 2.6 implies that LTM is a submodular diffusion model.

`LTM` on unweighted graphs Again, `LTM` in Definition 2.11 is defined upon edge-weighted graphs. Similar to the scenario in `ICM`, we can also define `LTM` for unweighted graphs.

Definition 2.13. The *uniform linear threshold model* $\texttt{ULTM}_G$ for $G = (V, E)$ is a special case of the linear threshold model $\texttt{LTM}_G$ with $G = (V, E, w)$, where $w(u, v) = 1/\deg(v)$ for each $(u, v) \in E$.

The weight assignments in the definition of the uniform linear threshold model is similar to the one of the *weighted* independent cascade model. The readers may wondering why we do not define a linear threshold model with all the weights of the edges being the same, as it is in the uniform independent cascade model. In fact, this definition for the linear threshold model is unnatural. More generally, any model assigning the weights of the edges satisfying $\forall u, v : w(u, v) = w(v, u)$ is unnatural for `LTM`, because it disallows the case that a popular vertex exercises significant influence over many somewhat lonely vertices. Consider an extreme example where the graph is a star, with a center u and T leaves $v_1, \ldots, v_T$. The constraint $\sum_{i=1}^{T} w(v_i, u) \leq 1$ (see Definition 2.11 and the remark immediately following the definition) implies that there exists at least one v_i such that $w(v_i, u) \leq \frac{1}{n}$, and furthermore, $w(u, v_i) = w(v_i, u) \leq \frac{1}{n}$. In this case, even if u is the only neighbor of v_i, u still has very limited influence to v_i just because u has a lot of other neighbors. In reality, it is unnatural to assume that a node's being popular reduces its influence to its neighbors.

We remark that it is the constraint $\sum_{u \in \Gamma(v)} w(u, v) \leq 1$ that is unique for `LTM` that makes the above model with symmetrically weighted graphs unnatural. For `ICM` which does not have this constraint, it is much more natural to consider graphs with symmetric edge weights $\forall u, v : w(u, v) = w(v, u)$.

Again, when we restrict our attention for undirected graphs, the undirected graph is "bidirected". For the reason discussed above, it is unnatural to consider a weighted undirected graph with $w(u, v) = w(v, u)$ for all pairs $\{u, v\}$ (of course, we can assign weights to an undirected graph such that the two anti-parallel edges have different weights, but this model is then no more specific to `LTM` with the general directed graphs, as we can assign $w(u, v) > 0$ and $w(v, u) = 0$ to simulate directed edge (u, v)). Therefore, we assume the following.

Assumption 2.14. When considering $\texttt{LTM}_G$ with G being an undirected graph, the uniform linear threshold model $\texttt{ULTM}_G$ is automatically assumed: an undirected edge (u, v) is viewed as two directed edges (u, v) and (v, u), with $w(u, v) = 1/\deg(v)$ and $w(v, u) = 1/\deg(u)$.

The same as it is in the weighted independent cascade model, if u has a larger degree than v, it is less likely that v will infect u than u will infect v.

As a final remark, the triggering model corresponding to ULTM has a very neat description: each vertex v chooses exactly one incoming edge being live uniformly at random. This characterization will be used multiple times in this thesis.

2.1.5 Bootstrap Percolation

The triggering model and two of its special cases, ICM and LTM, are all submodular diffusion models. In this subsection, we define a typical nonsubmodular diffusion model called the *bootstrap percolation*, which we have mentioned briefly in the introduction chapter. Some existing literature uses the phrases *bootstrap percolation* and *r-complex contagion* interchangeably. However, in this thesis, we will define them to be slightly different where "bootstrap percolation" refers to a more general model where vertices can have different thresholds and "r-complex contagion" refer to the special case where all the vertices have the same threshold r.

Definition 2.15. The *bootstrap percolation* $\mathtt{BP}_{G,R}$ is defined by a directed unweighted graph $G = (V, E)$ with $R = \{r_v \in \mathbb{Z}^+ \mid v \in V\}$. On input seed set $S \subseteq V$, $\mathtt{BP}_{G,R}(S)$ outputs a set of infected vertices as follows:

1. Initially, only vertices in S are infected.
2. In each subsequent round, a vertex v becomes infected if the number of v's infected neighbors is at least r_v.
3. After a round where there is no additional infected vertices, $\mathtt{BP}_{G,R}(S)$ outputs the set of infected vertices.

Definition 2.16. The *r-complex contagion* is a special case of the bootstrap percolation $\mathtt{BP}_{G,R}$ with $r_v = r$ for each $v \in V$.

Traditionally, the bootstrap percolation is defined on unweighted graphs. However, it is also natural to define it for edge-weighted graphs $G = (V, E, w)$. In the edge-weighted graph setting, r_v can be a real number that is not an integer, and a vertex v is infected if the sum of the weights of the edges connecting from v's infected neighbors exceeds the threshold r_v: $\sum_{u \in IN_v} w(u, v) \geq r_v$.

The bootstrap percolation, especially under the edge-weighted graph setting, may appear to be very similar to LTM. However, there is a fundamental difference. The

```
  **Input:** $G = (V, E)$; $F = \{f_v \mid v \in V\}$; $k \in \mathbb{Z}^+$
  **Output:** a seed set $S \subseteq V$ satisfying $|S| \leq k$
1 initialize $S = \emptyset$
2 **for** *each of k iterations* **do**
3     find $s \in \mathrm{argmax}_{s \in V}(\sigma(S \cup \{s\}) - \sigma(S))$ with tie broken arbitrarily
4     $S \leftarrow S \cup \{s\}$
5 **end**
6 **return** $S$
```

Algorithm 2.1: The greedy algorithm

threshold θ_v in `LTM` is sampled *after* the seeds are chosen, which is not a part of the input parameters. The threshold r_v in `BP`, on the other hand, is fixed as an input parameter. In particular, given a seed set $S \subseteq V$, $\mathtt{LTM}_G(S)$ is a random set where the randomness comes from the sampling of the thresholds, while $\mathtt{BP}_{G,R}(S)$ is deterministic. This key difference makes `BP` nonsubmodular.

It is easy to see that `BP` is still a special case of the general threshold model $I_{G,F}$. In particular, we have

$$f_v(IN_v) = \begin{cases} 0 & \text{if } |IN_v| < r_v \\ 1 & \text{if } |IN_v| \geq r_v \end{cases},$$

and

$$f_v(IN_v) = \begin{cases} 0 & \text{if } \sum_{u \in IN_v} w(u, v) < r_v \\ 1 & \text{if } \sum_{u \in IN_v} w(u, v) \geq r_v \end{cases}$$

if graphs are edge-weighted.

Since `BP` is nonsubmodular, it cannot be a special case of the triggering model due to Theorem 2.6.

2.2 Greedy Algorithm

The *greedy algorithm* iteratively picks a seed that has the maximum marginal gain in the objective function $\sigma(\cdot)$. The formal description is given in Algorithm 2.1.

However, it has been shown that, given $S \subseteq V$, it is #P-hard to compute $\sigma(S)$ even for `LTM` [18] and `ICM` [16]. This obstacle prevents us from executing Line 3 in Algorithm 2.1. However, it is easy to see that a simple Monte-Carlo method can approximate $\sigma(S)$ arbitrarily close with arbitrarily high probability. In other words, there is a Fully Polynomial-Time Randomized Approximation Scheme (FPRAS) to compute $\sigma(S)$. If the diffusion model is a triggering model, a powerful and highly scalable method using the technique of *reverse-reachable sets* is available. We discuss these two methods in the next two subsections.

As an important remark, the greedy algorithm based on either of the two methods can achieve a $(1-1/e-\varepsilon)$-approximation with probability at least $1-\delta$ for any $\varepsilon > 0$ and $\delta > 0$, and the algorithm runs in a time that is polynomial in terms of the input size, $1/\varepsilon$ and $\log(1/\delta)$.

2.2.1 Monte-Carlo Method

The Monte-Carlo method works as follows. Given general threshold model $I_{G,F}$ and a seed set S, we sample the threshold θ_v for each vertex v uniformly at random and independently from $[0, 1]$ (as described in Definition 2.1), and compute the number of infected vertices $|I_{G,F}(S)|$ with given $\{\theta_v \mid v \in V\}$. This experiment is repeated for a sufficient number of times independently, and $\sigma(S)$ is estimated by taking an average over the values of $|I_{G,F}(S)|$ from those experiments. Kempe et al. [44] showed that, in order to approximate $\sigma(S)$ with multiplicative error at most ε with probability at least $1-\delta$, $O(kn\varepsilon^{-2}\ln(n/\delta))$ experiments is sufficient. This number is recently known to be improvable [64]. When implemented in practice, Kempe et al. [44] recommended $10,000$ experiments.

2.2.2 Reverse-Reachable-Set-Based Algorithms

The Monte-Carlo in the previous subsection is widely-known to be unscalable to large graphs. If the diffusion model is the triggering model, a highly scalable method using the technique of *reverse-reachable sets* is available. This technique was first invented by Borgs et al. [8], and the algorithm RIS is invented based on this. This technique has later been improved, and more algorithms based on this were invented later on, including TIM$^+$ [71], IMM [72], EPIC [40], and so on.

In all those reverse-reachable-set-based algorithms, a sufficient number of reverse reachable sets are sampled. Each reverse reachable set is sampled as follows: first, a vertex v is sampled uniformly at random; second, sample the live edges in the graph where each vertex chooses a triggering set according to the triggering model (undirected graphs are treated as directed graphs with anti-parallel edges); lastly, the reverse reachable set consists of exactly those vertices from which v is reachable.

After collecting sufficiently many reverse reachable sets, the algorithms choose k seeds that attempt to cover as many reverse reachable sets as possible (we say a reverse reachable set is covered if it contains at least 1 seed), and this is done by a greedy maximum coverage way: iteratively select the seed that maximizes the extra number of reverse reachable sets covered by this seed.

The meat of those reverse-reachable-set-based algorithms is that, given a seed set S, the probability that a randomly sampled reverse reachable set is covered by S is exactly the probability that a vertex selected uniformly at random from the graph is infected by S. Therefore, when sufficiently many reverse reachable sets are sampled, the fraction of the reverse reachable sets covered by S is a good approximation to $\sigma(S)/|V|$.

Part I

Submodular Influence Maximization

CHAPTER 3

On Approximability of Submodular Influence Maximization

In this chapter, we study InfMax in undirected networks, specifically focusing on the three special cases of `ICM` and `LTM`: `UICM`, `WICM` and `ULTM`. We prove APX-hardness (NP-hardness of approximation within factor $(1 - \tau)$ for some constant $\tau > 0$) for each of the three models (which implies APX-hardness for the more general `ICM` and `LTM`), which improves the previous NP-hardness lower bound for `LTM`. No previous NP-hardness or hardness-of-approximation result was known for `ICM` (for undirected graphs).

As part of the hardness proof, we show some natural properties of these cascades on undirected graphs. We show that $\sigma(S)$ is upper-bounded by the size of the edge cut of S for `WICM` and `ULTM`.

Motivated by our upper bounds, we present a suite of highly scalable local greedy heuristics for InfMax on both `WICM` and `ULTM` on undirected graphs that, in practice, find seed sets which on average obtain 97.52% of the performance of the much slower greedy algorithm for `ULTM`, and 97.39% of the performance of the greedy algorithm for `WICM`. Our heuristics also outperform other popular local heuristics, such as the degree discount heuristic by Chen et al. [15].

3.1 Introduction

Among these social networks on which InfMax algorithms are tested (e.g, those mentioned in Sect. 1.2.4), most of them are undirected. For example, Arora et al. [3] performs a thoughtful benchmarking study of those InfMax algorithms, and the four main graphs considered, Nethept, HepPh, DBLP, YouTube, are all undirected. Undirected datasets are used in most, if not all, of the empirical work in Sect. 1.2.4, and are used exclusively in [15, 18, 31, 37, 38][1] in particular. Peer relationships are, in some sense, undirected by definition. Moreover, these models typically try to capture word-of-mouth influence among peers as opposed to a more directional advertisements.

Unfortunately, theoretical results for InfMax on undirected graphs are lacking. The original greedy algorithm in [44], for example, does not take advantage of the undirected nature of the graph in any meaningful way. On the positive side, as mentioned in Sect. 1.2.2, for `ICM` with undirected graphs, Khanna and Lucier [47] improve the analysis of the greedy algorithm to show it obtains an approximation ratio slightly better than $(1-1/e)$. This is one of the very few existing results that take advantage of the undirected nature of the graph. On the hardness side, the inapproximability results for InfMax on undirected graphs are almost completely lacking in the literature. For `LTM`, the previously mentioned NP-hardness result in [44], which is almost 16 years old, is the only known hardness result to the best of our knowledge. There is still a gap between $(1-1/e)$ and 1 for both directed and undirected graphs. For `ICM`, although the inapproximability factor $(1-1/e+\varepsilon)$ in [44] matches the approximation guarantee $(1-1/e)$ of the greedy algorithm, the hardness/inapproximability results for undirected graphs are completely missing.[2] In particular, before this work, it was not even known whether computing the exact optimal seeds is NP-hard.

It is worth noticing that, as we mentioned before, given a seed set S, computing the exact value of $\sigma(S)$ is #P-hard for both `LTM` [18] and `ICM` [16]. This can be easily extended to the #P-hardness of InfMax. However, if we assume oracle access to $\sigma(\cdot)$, a setting commonly adopted in most work (including [44]), the complexity of computing the exact optimal seed set is still unknown. Moreover, we have seen in Sect. 2.2.1 that a simple Monte-Carlo method provides a FPRAS for computing $\sigma(S)$. Therefore, the #P-hardness result cannot even rule out a polynomial time approximation scheme (PTAS) for InfMax. In this chapter, we will assume that $\sigma(\cdot)$ can be accessed by an oracle.

Our results As our main results, we show that InfMax on both `ICM` and `LTM` with undirected graphs is APX-hard: there exists a constant $\tau > 0$ such that approximating InfMax to within factor $(1-\tau)$ is NP-hard. The APX-hardness holds even for the special cases `UICM`, `WICM` and `ULTM`.

[1] Directed graphs are treated as undirected graphs in [18].

[2] An APX-hardness result for the independent cascade InfMax with undirected graphs was presented by Khanna and Lucier [47]. However, their APX-hardness result is applied to a more general non-standard setting. We discussed this in details in Sect. 3.2.

We prove this result by using the PCP theorems and a novel coupling approach. Our coupling approach reveals an upper bound on $\sigma(S)$ for each of the three cascade models: UICM, WICM and ULTM. An especially interesting upper bound for both ULTM and WICM on undirected graphs is that, *the expected number of vertices infected by S is no more than the number of edges between vertices in S and vertices outside S.* This is an indication that the cascade on these two models on undirected graphs is diminishing and somehow characterized by its local behaviors on S and neighbors of vertices in S: no matter how large and dense the graph G is outside S, the cascade is limited as long as S has limited connections to the remaining vertices. This indicates that, although WICM is similar to UICM in definition, it is more similar to ULTM in behaviors.[3] This also theoretically justifies the phenomenon that those reverse-reachable-set-based algorithms (Sect. 2.2.2) are much more scalable on ULTM and WICM than UICM, as observed by Arora et al. [3] (See Sect. 3.5.5 for details.).

Motivated by this observation, we present a family of InfMax heuristics we called the *local greedy heuristics*, which iteratively select the seeds only based on the local features of S: the number of edges goes out from S, the number of vertices that are connected to S, and a combination of both. Our heuristics are highly scalable, since we have only checked the neighbors of the seeds, instead of analyzing the influence of the seeds in the graph globally (as it is in the Monte-Carol method). Moreover, our heuristics, although being extremely simple, produce seeds with almost the same quality as the seeds output by those greedy-based algorithms in the state-of-art (being 97.52% of the greedy-based algorithms on average for ULTM, and 97.39% for WICM), outperform the *degree discount heuristic* proposed by [15] which iteratively selects a vertex with highest degree and removes it from the graph, and significantly outperform the naïve algorithm that picks the seeds with the highest degrees.

3.2 Additional Related Work

The work by Khanna and Lucier [47] mentioned in Sect. 1.2.1 is most relevant to our work. Khanna and Lucier [47] studied the independent cascade InfMax on undirected graphs. They worked on a more general model where the seed-picker is only allowed to pick seeds from a prescribed subset of vertices and the vertices are weighted so that the objective is to maximize the expected total weight of all the infected vertices. Under this generalized setting, Khanna and Lucier [47] showed that the greedy algorithm achieves a $(1-1/e+c)$-approximation for some constant

[3] Similar observations have been made before by Kempe et al. [44], Chen et al. [15].

$c > 0$, and InfMax on this model is APX-hard. In their APX-hardness reduction, a bipartite graph is constructed such that the vertices on the left-hand side have weight 0 and the vertices on the right-hand side have weight 1, and the seed-picker is restricted to pick seeds among the left-hand side vertices. These features are not allowed in the standard InfMax setting (where the seed-picker can seed on any vertices, and the expected *number* of infected vertices is the objective), and extending the same APX-hardness result to the less general, standard setting requires significantly more insight.

A weaker version of our upper bound for `ULTM` on undirected graphs was presented by Lim et al. [56], where it was shown that the expected number of infected vertices by a single seed is upper-bounded by the degree of this seed plus one. In addition, Lim et al. [56] observed that this upper bound is tight when the graph is a tree. We have the same observation in Lemma 3.13 and Corollary 3.15. Our upper bound generalizes Lim et al.'s upper bound to the setting with multiple seeds, which is crucial for showing the APX-hardness of the InfMax problem.

3.3 Dinur's PCP Theorem

We will use the following version of the PCP theorem which is due to Dinur [25].

Theorem 3.1. *There exist a universal constant integer d and a universal constant $\gamma \in (0, 1)$ such that, given a* 3SAT *instance ϕ where each variable appears in at most d clauses, it is NP-hard to distinguish between the following two cases:*

- YES*: ϕ has a satisfiable truth-assignment;*
- NO*: for any truth-assignment, at most $(1-\gamma)$ fraction of clauses can be satisfied.*

Dinur's PCP theorem straightforwardly implies Theorem 3.2.

Theorem 3.2 (Inapproximability of IndSet)**.** *There exist a universal constant integer d and a universal constant $\gamma \in (0, 1)$ such that, given an undirected graph $G = (V, E)$ where the degree of each vertex is at most d and the number of vertices $|V|$ is a multiple of* 3*, it is NP-hard to distinguish between the following two cases:*

- YES*: G has an independent set of size n;*
- NO*: any subgraph of G induced by n vertices contains at least γn edges,*

where $n = \frac{|V|}{3}$.

Proof. Given a 3SAT instance ϕ with n clauses such that each variable appears in at most d' clauses, we construct the IndSet instance $G_\phi = (V_\phi, E_\phi)$ as follows: G_ϕ contains $3n$ vertices $\{v_{ij} \mid i = 1, \ldots, n; j = 1, 2, 3\}$ where v_{ij} corresponds to the j-th literal in the i-th clause of ϕ; $(v_{ij}, v_{i'j'}) \in E_\phi$ if $i = i'$ or $v_{ij}, v_{i'j'}$ correspond to two literals such that one is the negation of the other. It is easy to see that if each variable in ϕ appears in at most d' clauses, then each vertex in G_ϕ has degree at most $d = d' + 1$. If the 3SAT instance is a YES instance, then ϕ is satisfiable and has an assignment which satisfies all the n clauses. This assignment, by including one true literal in each clause, yields an independent set of size n in G. If the 3SAT instance is a NO instance, Dinur's PCP theorem implies that there exists a constant $\gamma \in (0, 1)$ such that no assignment satisfies even $(1 - \gamma)n$ clauses. This means that no independent set of size more than $(1 - \gamma)n$ exists, because by including one true literal in each clause for a given assignment yields an independent set with the same number of vertices as satisfying clauses. As a result, any subgraph of size n contains at least γn edges: if the number of edges in certain subgraph with size n is less than γn, removing one endpoint for each edge gives an independent set of size more than $n - \gamma n$, which is a contradiction. □

Dinur's PCP theorem also implies the APX-hardness of the vertex cover problem (VertexCover) on graphs with constant degree-bound. We will need it for the proof of Theorem 3.10.

Theorem 3.3 (Inapproximability of VertexCover). *There exist a universal constant integer d and a universal constant $\gamma \in (0, 1)$ such that, given an integer k and an undirected graph $G = (V, E)$ where the degree of each vertex is bounded by d, it is NP-hard to distinguish between the following two cases:*

- YES*: G has a vertex cover of size k;*
- NO*: all vertex covers of G have sizes at least $(1 + \gamma)k$.*

Proof. Fix $\gamma > 0$ from Theorem 3.1. We will show that Theorem 3.3 is true for substituting γ with $\gamma/2$. Given a 3SAT instance ϕ with n clauses, we construct the VertexCover instance (G_ϕ, k) with $k = 2n$ and G_ϕ being the same graph as it is in the proof of Theorem 3.2. By Theorem 3.2 (or following the same arguments in the proof of Theorem 3.2, if ϕ is a YES instance, G_ϕ has an independent set of size n, picking the remaining $2n$ vertices that are not in this independent set gives a vertex cover of size $k = 2n$. If ϕ is a NO instance, Dinur's PCP theorem implies that no assignment satisfies even $(1 - \gamma)n$ clauses. This means that no independent set of size $(1-\gamma)n$ exists (because by including one true literal in each clause for a given assignment yields an independent set with the same number of vertices as satisfying clauses). Thus, all vertex covers of G have sizes at least $3n-(1-\gamma)n=(2+\gamma)n$. This implies the YES and NO instances differ by a factor of $1+\gamma/2$. □

3.4 APX-hardness of Influence Maximization

All our APX-hardness results are built upon the upper bounds on $\sigma(S)$ in Theorem 3.4, 3.5 and 3.6. Our upper bound for `ULTM` and `WICM` in Theorem 3.5 and 3.6 are particularly interesting even on their own, and they will further be considered in Sect. 3.6. Both of them show that $\sigma(S)$ is upper-bounded by the number of edges between S and $V\setminus S$, which reveals that $\sigma(S)$ is somehow characterized by its local behaviors on S and neighbors of vertices in S. In particular, the expected number of new infections caused by a single seed is upper-bounded by the degree of this seed. As we will see later in Sect. 3.5.2 and 3.5.3, the expected number of infections caused by a single seed is exactly the degree of this seed if the graph is a tree (Lemma 3.13 and 3.16). This implies that trees provide the most number of infections, and adding more edges to a tree may only reduce the total number of infections.

Notice that, although Theorem 3.4 can be adapted for directed graphs with d in the theorem being the maximum *out-degree*, Theorem 3.5 and 3.6 only holds for undirected graphs. `ULTM` and `WICM` on directed graphs has a fundamentally different nature. The local characterization no longer applies and the cascade process is no longer diminishing. As a simple example, if a seed s is put at one end of a directed line, $G=(V,E)$ where $E=\{(s,v_1),(v_1,v_2),\ldots,(v_{N-1},v_N)\}$, then, for both the `ULTM` and `WICM`, $\sigma(\{s\})=N+1$ as all the vertices in G will be infected with probability 1, even if the degree of s is only 1.

Theorem 3.4. *Given a uniform independent cascade model* $\texttt{UICM}_{G,p}$ *with a seed set* $S\subseteq V$*, and assuming* $p<\frac{1}{d}$ *where* d *is the maximum degree over all vertices, we have* $\sigma_{G,p}(S)\le|S|+\frac{|E(S,V\setminus S)|p}{1-pd}$.

Theorem 3.5. *Given a uniform linear threshold model* $\texttt{ULTM}_G$ *with an undirected graph* $G=(V,E)$ *and a seed set* $S\subseteq V$*, we have* $\sigma_G(S)\le|E(S,V\setminus S)|+|S|$*, where* $E(S,V\setminus S)$ *is the set of edges between* S *and* $V\setminus S$*.*

Theorem 3.6. *Given a weighted independent cascade model* $\texttt{WICM}_G$ *with an undirected graph* $G=(V,E)$ *and a seed set* $S\subseteq V$*, we have* $\sigma_G(S)\le|E(S,V\setminus S)|+|S|$*, where* $E(S,V\setminus S)$ *is the set of edges between* S *and* $V\setminus S$*.*

We defer the proofs of these three theorems to Sect. 3.5, and in this section we use them to prove our main APX-hardness results.

The theorem below shows that InfMax on undirected graphs with ICM is APX-hard, even if all edges have the same weight (i.e., the model UICM).

Theorem 3.7. *There exist universal constants $\tau, p \in (0,1)$ and a function $T : \mathbb{Z}^+ \to \mathbb{R}^+$ such that, considering the* InfMax *problem $(G = (V, E), k)$ with a uniform independent cascade model* UICM$_{G,p}$ *on an undirected graph G, it is NP-hard to distinguish between the following two cases:*

- YES*: there exists a seed set S with $|S| = k$ such that $\sigma_{G,p}(S) \geq T(|V|)$;*
- NO*: for any seed set S with $|S| = k$, we have $\sigma_{G,p}(S) \leq (1-\tau)T(|V|)$.*

The intuition behinds the following proof of Theorem 3.7 is as follows: given an IndSet instance G_ϕ, we first make vertices in G_ϕ have the same degree d by adding dummy vertices; by making the value of p small, we can approximate $\sigma_{G,p}(S)$ by only counting the number of edges going out from S; if G_ϕ has an independent set of size n, then the number of out-going edges is nd; otherwise, Theorem 3.2 indicates that there are at most $nd - \gamma n$ out-going edges for any seed set of size n; this yields the APX-hardness result since γ is a constant.

Proof. The reduction is as follows. Given an IndSet instance $G_\phi = (V_\phi, E_\phi)$ with $|V_\phi| = 3n$ and degree bound d (and a potential independent set of size n is regarded), the undirected graph $G = (V, E)$ is obtained by modifying $G_\phi = (V_\phi, E_\phi)$ as follows: let d be the maximum degree over vertices in G_ϕ, for each $v \in V_\phi$ with $\deg(v) < d$, add $d - \deg(v)$ dummy vertices and make them connect to v only. This makes all the original $3n$ vertices in V_ϕ have the same degree d. The number of seeds k is set to n. We will decide the values for p and τ, and the function $T(\cdot)$ later. In the remaining part of this proof, we will view G_ϕ as a subgraph of G so that $V \setminus V_\phi$ are exactly those dummy vertices.

When the IndSet instance G_ϕ is a YES instance, there is an independent set of size n in G_ϕ. If we pick the corresponding n vertices in G as seeds, denoted by S, the expected number of infected vertices after the first round of the cascade is at least $ndp(1-p)^{d-1}$. This is because the total number of edges between S and $V \setminus S$ is exactly nd (since S is an independent set and each $s \in S$ has degree d), and the expected number of infected vertices for the first round is

$$\begin{aligned}\sum_{v\in\partial S}\Pr(v\text{ is infected}) &= \sum_{v\in\partial S}\left(1-(1-p)^{\delta_v}\right)\\ &\qquad(\text{where } \partial S=\{v\in V\setminus S\mid \exists s\in S:(s,v)\in E\}\text{ and }\delta_v=\Gamma(v)\cap S)\\ &= \sum_{v\in\partial S}\sum_{i=0}^{\delta_v-1}p(1-p)^i\\ &\geq \sum_{v\in\partial S}\sum_{i=0}^{\delta_v-1}p(1-p)^{d-1} \qquad\qquad (\text{since } i\leq\delta_v-1\leq d-1)\\ &= ndp(1-p)^{d-1},\end{aligned}$$

where the last equality follows since $\sum_{v\in\partial S}\sum_{i=0}^{\delta_v-1}1$ is exactly the number of edges between S and $V\setminus S$, which is nd. Therefore, by only consider the first round of the cascade, we have

$$\sigma_{G,p}(S)\geq n+ndp(1-p)^{d-1}, \tag{3.1}$$

for certain $S\subseteq V$ if G_ϕ is a YES instance.

When the IndSet instance G_ϕ is a NO instance, there are at least γn edges between any set of n vertices in G_ϕ. We first note that we can assume without loss of generality that seeds are chosen in $V_\phi\subseteq V$, as seeding any $w\in V\setminus V_\phi$ is always less beneficial.[4] Next, consider any seed set $S\subseteq V_\phi\subseteq V$ with $|S|=n$. We have $|E(S,V\setminus S)|\leq nd-\gamma n=n(d-\gamma)$. By Theorem 3.4, if we set p such that $p<\frac{1}{d}$, this implies

$$\sigma_{G,p}(S)\leq n+np\frac{d-\gamma}{1-dp}, \tag{3.2}$$

for any S with $|S|=n$.

Next, the inequality

$$d(1-p)^{d-1}>\frac{d-\gamma}{1-dp}$$

holds for sufficiently small p, as the left-hand side has limit d and the right-hand side has limit $d-\gamma<d$ when $p\to 0$. We choose the value of p with $p>0$ such that this inequality still holds. Notice that p is a universal constant, since d and γ in the

[4] To see this, for (u,w) such that $u\in V_\phi$ and $w\in V\setminus V_\phi$, if u is not seeded, seeding u is strictly more beneficial than seeding w; if u is seeded, seeding w adds only $1-p$ to $\sigma_G(S)$, while seeding any unseeded vertices in V_ϕ is more beneficial.

equality are universal constants. Let

$$\tau = 1 - \frac{1 + p\frac{d-\gamma}{1-dp}}{1 + dp(1-p)^{d-1}}.$$

Our choice of p makes sure $\tau > 0$, and τ is a constant since p, d and γ are all constants. Finally, choosing $T(n) = n + ndp(1-p)^{d-1}$, it is straightforward to check that Eqn. (3.1) and Eqn. (3.2) imply the theorem. □

The theorem below shows that the linear threshold InfMax is also APX-hard, even with undirected graphs. The ideas for the proof of Theorem 3.8 is similar to the ones for proving Theorem 3.7. The difficulty here is, we cannot adjust the value of p, as the weight of each edge is decided by the degree of one of its incident vertices in ULTM. However, we can link to each vertex a large number D (still being a constant) of dummy vertices to artificially reduce the weights of edges.

Theorem 3.8. *There exists a universal constant $\tau \in (0,1)$ and a function $T : \mathbb{Z}^+ \to \mathbb{R}^+$ such that, considering the* InfMax *problem $(G = (V,E), k)$ with the linear threshold model $\texttt{ULTM}_G$ on an undirected graph G, it is NP-hard to distinguish between the following two cases:*

- YES*: there exists a seed set S with $|S| = k$ such that $\sigma_G(S) \geq T(n)$;*
- NO*: for any seed set S with $|S| = k$, we have $\sigma_G(S) \leq (1-\tau)T(n)$.*

Proof. Let $G_\phi = (V_\phi, E_\phi)$ be the IndSet instance, with d being the upper bound of the degrees and $|V_\phi| = 3n$. We construct the InfMax instance $G = (V, E)$ as follows. Let D be a sufficiently large integer to be decided later. For each vertex $v \in V_\phi$, create $D + d - \deg(v)$ dummy vertices and connect them to v. Again, we will view G_ϕ as a subgraph of G. By our construction, all vertices in V_ϕ have degree $D + d$ in G. Set $k = n$.

When the IndSet instance G_ϕ is a YES instance, there is an independent set of size n in G_ϕ. If we pick the corresponding n vertices in G as seeds, denoted by S, the expected number of infected vertices after two rounds of the cascade is at least $n + nD + \frac{ndD}{d+D}$. To see this, first notice that, if a vertex $v \in V_\phi$ is infected, then all the dummy vertices connected to it will be infected in the next round, as these vertices have degree 1. Let $\partial S = \{v \in V_\phi \setminus S \mid \exists s \in S : (s,v) \in E_\phi\}$ be the set of all non-dummy vertices that are connected to a seed, and $\partial^d S = \{v \in V \setminus V_\phi \mid \exists s \in S : (s,v) \in E\}$ be the set of all dummy vertices that are connected to a seed. By our construction, we have $|E(S, V_\phi \setminus S)| + |\partial^d S| = n(d + D)$ (since the left hand side is exactly the number of edges between S and $V \setminus S$ and each vertex in the independent set S has degree exactly $d+D$) and $|\partial^d S| \geq nD$ (since each vertex in V_ϕ has at least D dummy neighbors). For each $v \in \partial S$, let δ_v be the number of v's neighbors in S. In the first round, all vertices in $\partial^d S$ will be infected with probability 1, and each vertex $v \in \partial S$ will be infected with probability $\frac{\delta_v}{d+D}$. If a vertex in ∂S is infected, it will infect all the dummy vertices connected to it in the second round, and there are at least D such dummy vertices. Thus, the total number of vertices that are infected in the first two rounds is at least

$$\begin{aligned}\sum_{v\in\partial^d S} 1 + \sum_{v\in\partial S} \frac{\delta_v}{d+D}\cdot D &= \left|\partial^d S\right| + |E(S, V_\phi \setminus S)| \frac{D}{d+D} \\ &= nD + \left(\left|\partial^d S\right| - nD\right) + |E(S, V_\phi \setminus S)| \frac{D}{d+D} \\ &\geq nD + \left(\left|\partial^d S\right| - nD\right) \frac{D}{d+D} + |E(S, V_\phi \setminus S)| \frac{D}{d+D} \\ &\qquad\qquad \text{(since } |\partial^d S| \geq nD) \\ &= nD + nd\frac{D}{d+D}. \\ &\qquad\qquad \text{(since } |E(S, V_\phi \setminus S)| + |\partial^d S| = n(d+D))\end{aligned}$$

Therefore, we have

$$\sigma_G(S) \geq n + nD + \frac{ndD}{d+D}. \tag{3.3}$$

When the IndSet instance G_ϕ is a NO instance, there are at least γn edges between any set of n vertices in G_ϕ. We first note that we can assume without loss of generality that seeds are chosen in $V_\phi \subseteq V$.[5] Next, consider any seed set $S \subseteq V_\phi \subseteq V$ with $|S| = n$. We have $|E(S, V \setminus S)| \leq n(d+D) - \gamma n$. By Theorem 3.5, this implies

$$\sigma_G(S) \leq n + nD + nd - n\gamma, \tag{3.4}$$

for any S with $|S| = n$.

Lastly, it is possible to choose D such that

$$1 + D + \frac{dD}{d+D} > 1 + D + d - \gamma,$$

as $\frac{dD}{d+D}$, having limit d when $D \to \infty$, can be larger than $d - \gamma$ for sufficiently large

[5] To see this, for (u, v) such that $u \in V_\phi$ and $v \in V \setminus V_\phi$, if u is not seeded, seeding u is strictly more beneficial than seeding v; if u is seeded, v will be infected with probability 1, while seeding any unseeded vertices in V_ϕ is more beneficial.

D. We choose D satisfying this, and notice that D is a universal constant since d and γ are universal constants. Finally, letting

$$\tau = 1 - \frac{1 + D + d - \gamma}{1 + D + \frac{dD}{d+D}} \qquad \text{and} \qquad T(n) = n + nD + \frac{ndD}{d+D},$$

Eqn. (3.3) and Eqn. (3.4) imply the theorem. □

The APX-hardness for `WICM` can be proved in a way similar to proving Theorem 3.8.

Theorem 3.9. *There exists a universal constant $\tau \in (0,1)$ and a function $T : \mathbb{Z}^+ \to \mathbb{R}^+$ such that, considering the* InfMax *problem $(G = (V, E), k)$ with the weighted independent cascade model $\texttt{WICM}_G$ on an undirected graph G, it is NP-hard to distinguish between the following two cases:*

- YES*: there exists a seed set S with $|S| = k$ such that $\sigma_G(S) \geq T(n)$;*
- NO*: for any seed set S with $|S| = k$, we have $\sigma_G(S) \leq (1 - \tau)T(n)$.*

Proof. Let $G_\phi = (V_\phi, E_\phi)$ be the IndSet instance, with d being the upper bound of the degrees and $|V_\phi| = 3n$. We construct the InfMax instance $G = (V, E)$ in exactly the same way as it is in the proof of Theorem 3.8, and the corresponding constant D is decided later.

When the IndSet instance G_ϕ is a YES instance, there is an independent set of size n in G_ϕ. Again, we consider picking the corresponding n vertices in G as seeds, denoted by S, and find a lower bound for $\sigma_G(S)$. Let ∂S and $\partial^d S$ be the same as they are in the proof of Theorem 3.8. For each $v \in \partial S$, let δ_v be the number of v's neighbors in S again. In the first round, all vertices in $\partial^d S$ will be infected with probability 1, and each vertex $v \in \partial S$ will be infected with probability $1 - \left(1 - \frac{1}{d+D}\right)^{\delta_v}$. If a vertex in ∂S is infected, it will infect all the dummy vertices connected to it in the second round, and there are at least D such dummy vertices. Putting together, we have

$$\begin{aligned}
\sigma_G(S) &\geq n + \sum_{v \in \partial^d S} 1 + \sum_{v \in \partial S} \left(1 - \left(1 - \frac{1}{d+D}\right)^{\delta_v}\right) D \\
&= n + \left|\partial^d S\right| + \sum_{v \in \partial S} \sum_{i=0}^{\delta_v - 1} \frac{1}{d+D} \left(1 - \frac{1}{d+D}\right)^i D \\
&\geq n + \left|\partial^d S\right| + \sum_{v \in \partial S} \sum_{i=0}^{\delta_v - 1} \frac{D}{d+D} \left(1 - \frac{1}{d+D}\right)^d && (\text{since } i < \delta_v \leq d) \\
&= n + \left|\partial^d S\right| + |E(S, V_\phi \setminus S)| \frac{D}{d+D} \left(1 - \frac{1}{d+D}\right)^d \\
& && \left(\text{since } \sum_{v \in \partial S} \sum_{i=0}^{\delta_v - 1} 1 = |E(S, V_\phi \setminus S)|\right) \\
&\geq n + nD + \left(\left|\partial^d S\right| - nD + |E(S, V_\phi \setminus S)|\right) \frac{D}{d+D} \left(1 - \frac{1}{d+D}\right)^d \\
& && (\text{since } |\partial^d S| \geq nD) \\
&= n + nD + \frac{ndD}{d+D} \left(1 - \frac{1}{d+D}\right)^d . \\
& && (\text{since } |E(S, V_\phi \setminus S)| + |\partial^d S| = n(d+D))
\end{aligned}$$

When the IndSet instance G_ϕ is a NO instance, the same analysis in the proof of Theorem 3.8, coupled with Theorem 3.6, implies that

$$\sigma_G(S) \leq n + nD + nd - n\gamma,$$

for any S with $|S| = n$.

Lastly, it is possible to choose D such that

$$1 + D + \frac{dD}{d+D} \left(1 - \frac{1}{d+D}\right)^d > 1 + D + d - \gamma,$$

as $\frac{dD}{d+D}\left(1 - \frac{1}{d+D}\right)^d$, having limit d when $D \to \infty$, can be larger than $d - \gamma$ for sufficiently large D. We choose D satisfying this, and notice that D is a universal constant since d and γ are universal constants. Finally, letting

$$\tau = 1 - \frac{1 + D + d - \gamma}{1 + D + \frac{dD}{d+D}\left(1 - \frac{1}{d+D}\right)^d} \qquad \text{and} \qquad T(n) = n + nD + \frac{ndD}{d+D}\left(1 - \frac{1}{d+D}\right)^d,$$

the theorem is implied. □

We conclude this section by presenting the following APX-hardness result that has *almost perfect completeness*: in the case of a YES instance, the number of infected vertices is $N - o(N)$ deterministically. However, it only holds for general directed graphs. The proof of the theorem makes use of Dinur's PCP theorem as well, but does not depend on the upper bound in Theorem 3.5 as well as any results in Sect. 3.5.

Theorem 3.10. *Consider the* InfMax *problem* $(G = (V, E), k)$ *with* ULTM*. There exists a constant* $\tau \in (0, 1)$ *such that it is NP-hard to distinguish between the following two cases:*

- YES*: there exists* $S \subseteq V$ *with* $|S| = k$ *such that* $\Pr(\texttt{ULTM}_G(S) \geq N - o(N)) = 1$*;*
- NO*: for any seed set* S *with* $|S| = k$*, we have* $\sigma_G(S) \leq (1 - \tau)N$*.*

Proof. We consider a reduction from VertexCover. Given a VertexCover instance $(\overline{G} = (\overline{V}, \overline{E}), \overline{k})$, we will construct an InfMax instance $(G = (V, E, w), k)$. Let $\bar{n} = |\overline{V}|$, $\bar{m} = |\overline{E}|$, and $W = 2\bar{m}^5\bar{n}^5$ (W can be any sufficiently large even number in this reduction, as long as it is bounded by a polynomial of $\bar{m}$ and $\bar{n}$). We consider exclusively those $\overline{G}$ such that each $v \in \overline{V}$ has degree at most d. Let γ and the notion of YES/NO instance be defined as they are in Theorem 3.3. We further assume that $\overline{k} \geq \bar{m}/d$. This can be assume without loss of generality: the total number of edges that can possibly be covered by $\overline{k}$ vertices is at most $\overline{k}d$, so a set of less than $\bar{m}/d$ vertices cannot be a vertex cover and the VertexCover instance is a trivial NO instance if $\overline{k} < \bar{m}/d$.

The number of seeds k is set to $k = \bar{m} + \overline{k}$. The graph G contains $N = \bar{n} + \bar{m}(2 + W)$ vertices and is defined as follows.

- For each $\bar{v} \in \overline{V}$, construct a vertex $v \in V$; for each $(\bar{u}, \bar{v}) \in \overline{E}$, construct $2 + W$ vertices: $e^u_{u,v}, e^v_{u,v}, w^1_{u,v}, w^2_{u,v}, \ldots, w^W_{u,v}$.
- For each $(\bar{u}, \bar{v}) \in \overline{E}$, construct $4 + W$ edges:
  - $(u, e^v_{u,v}), (v, e^u_{u,v}), (e^u_{u,v}, e^v_{u,v}), (e^v_{u,v}, e^u_{u,v})$, each of which has weight 0.5, and
  - $(e^u_{u,v}, w^1_{u,v}), \ldots, (e^u_{u,v}, w^{\frac{W}{2}}_{u,v}), (e^v_{u,v}, w^{\frac{W}{2}+1}_{u,v}), \ldots, (e^u_{u,v}, w^W_{u,v})$, each of which has weight 1.

For each $(\bar{u}, \bar{v}) \in \overline{E}$, we have constructed a *gadget* shown in Fig. 3.1, for which we will denote by $G_{u,v}$, that contains $4 + W$ vertices $u, v, e^u_{u,v}, e^v_{u,v}, w^1_{u,v}, w^2_{u,v}, \ldots, w^W_{u,v}$ and

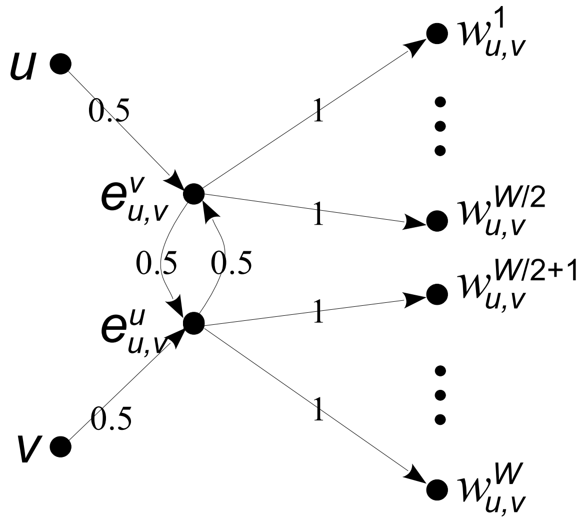


Figure 3.1: The edge gadget

$4+W$ edges defined above. Notice that any two gadgets can only share at most a single vertex v which corresponds to a certain $\bar{v} \in \overline{V}$. More importantly, the infected vertices in $G_{u,v} \setminus \{v\}$ will never infect v as v is connected to the rest of $G_{u,v}$ by a directed edge. This ensures that the infection of any one or more of $e^u_{u,v}, e^v_{u,v}, w^1_{u,v}, w^2_{u,v}, \ldots, w^W_{u,v}$ in a gadget $G_{u,v}$ has no impact on the other gadgets. Thus, after a seed set is fixed, we can analyze the cascade in each gadget independently. Finally, it is easy to see that the construction has a polynomial size, so this is a polynomial time reduction.

If the VertexCover instance is a YES instance, we assume without loss of generality that $\{\bar{v}_1, \ldots, \bar{v}_{\overline{k}}\}$ is a vertex cover of $\overline{G}$. Consider the seed set S of size $k = \bar{m} + \bar{k}$ defined as follows:

- S includes $v_1, \ldots, v_{\overline{k}}$;
- for each gadget $G_{u,v}$, S includes exactly one of $e^u_{u,v}, e^v_{u,v}$: S includes $e^u_{u,v}$ if $u \in \{v_1, \ldots, v_{\overline{k}}\}$, S includes $e^v_{u,v}$ if $v \in \{v_1, \ldots, v_{\overline{k}}\}$, and pick an arbitrary one of $e^u_{u,v}, e^v_{u,v}$ to be included in S if $u, v \in \{v_1, \ldots, v_{\overline{k}}\}$. Notice that we have either $u \in \{v_1, \ldots, v_{\overline{k}}\}$ or $v \in \{v_1, \ldots, v_{\overline{k}}\}$ since $\{\bar{v}_1, \ldots, \bar{v}_{\overline{k}}\}$ is a vertex cover of $\overline{G}$.

It is easy to see that each gadget $G_{u,v}$ contains at least $3 + W \geq W$ infected vertices after the cascade: assume without loss of generality that $u, e^u_{u,v} \in S$; then $e^v_{u,v}$, having all the two in-neighbors $u, e^u_{u,v}$ infected, will be infected with probability 1; finally, the infection of both $e^u_{u,v}$ and $e^v_{u,v}$ will infect all of $w^1_{u,v}, \ldots, w^W_{u,v}$ with probability 1. Therefore, with probability 1, the total number of infected vertices is at least $\bar{m}W = N - o(N)$.

If the VertexCover instance is a NO instance, a vertex cover of $\overline{G}$ has size at least $(1+\gamma)\overline{k}$. Consider an arbitrary seed set S' with $|S'| = k = \bar{m}+\bar{k}$. We can assume that S' does not contain any of $w_{u,v}^1, \ldots, w_{u,v}^W$ in each gadget $G_{u,v}$, as seeding $e_{u,v}^u$ is strictly better than seeding any of $w_{u,v}^1, \ldots, w_{u,v}^{\frac{W}{2}}$ and seeding $e_{u,v}^v$ is strictly better than seeding any of $w_{u,v}^{\frac{W}{2}+1}, w_{u,v}^W$ (if $e_{u,v}^u$ or $e_{u,v}^v$ is also seeded, we can remove the seeds from $w_{u,v}^1, \ldots, w_{u,v}^W$ without changing the number of infected vertices). Moreover, we can assume that S' does not contain *both* $e_{u,v}^u$ and $e_{u,v}^v$ in each gadget $G_{u,v}$, as in this case either switching $e_{u,v}^u$ to u or switching $e_{u,v}^v$ to v will cause one more infection and thus strictly better: for example, $e_{u,v}^u$ will still be infected by u and $e_{u,v}^v$ if we seed u instead of $e_{u,v}^u$ (if either u or v is already seeded, we can just remove $e_{u,v}^u$ or $e_{u,v}^v$ from S' without changing the number of infected vertices).

We say that $G_{u,v}$ is *active* if we have either $u, e_{u,v}^u \in S'$ or $v, e_{u,v}^v \in S'$. From previous analyses, an active gadget contains at least $3 + W$ infected vertices. A gadget can fail to be active in the following two different ways:

Failure I none of $e_{u,v}^u, e_{u,v}^v$ is chosen in the seed set, and

Failure II none of u, v is chosen in the seed set.

Keeping our justified assumptions $\{e_{u,v}^u, e_{u,v}^v\} \not\subseteq S'$ and $w_{u,v}^1, \ldots, w_{u,v}^W \notin S'$, by analyzing the cascade, it is straightforward to see that the expected number of infected vertices in $G_{u,v}$ is at most $\frac{3}{4}W + O(1)$ if one of the above two failures occur, and the number of infected vertices is 0 if both failures occur. To calculate an upper-bound of $\sigma_G(S')$, we only need to count the number of failures in all the gadgets, where the occurrences of both failures in a single gadget are counted as two failures. Since the occurrence of a single failure causes a lost of $\frac{1}{4}W + O(1)$ infected vertices and the occurrences of both failures in a gadget imply a lost of $W + O(1)$ infected vertices which is even more than $2 \times (\frac{1}{4}W + O(1))$, this way of counting only underestimates the number of uninfected vertices

Next, we aim to show that the total number of failures is at least $\gamma\bar{k}$. Let n_1, n_2 be the number of Failure I and II respectively. The number of seeds in the "inner parts" of all the $\bar{m}$ gadgets, $\bigcup_{u,v:(\bar{u},\bar{v})\in\overline{E}}\{e_{u,v}^u, e_{u,v}^v\}$, is $\bar{m} - n_1$, since exactly a seed is needed for one of $e_{u,v}^u, e_{u,v}^v$ in each $G_{u,v}$ to avoid a Failure I. As a results, the number of seeds allocated for those vertices representing vertices of $\overline{G}$, $\{v : \bar{v} \in \overline{V}\}$, is $k - (\bar{m} - n_1) = \overline{k} + n_1$. This symbolizes picking a set of $\overline{k} + n_1$ vertices in the VertexCover instance. By Theorem 3.3, we need $\max\{\gamma\bar{k} - n_1, 0\}$ additional vertices to form a vertex cover. This implies that there are at least $\max\{\gamma\bar{k} - n_1, 0\}$

uncovered edges for the picked set of $\bar{k} + n_1$ vertices. Thus, $n_2 \geq \max\{\gamma\bar{k} - n_1, 0\}$. Therefore, the total number of failures is lower-bounded by

$$n_1 + n_2 \geq n_1 + \max\{\gamma\bar{k} - n_1, 0\} \geq \gamma\bar{k},$$

implying

$$\sigma_G(S') \leq N - (n_1 + n_2)\left(\frac{1}{4}W + O(1)\right) \leq \left(m - \frac{\gamma\bar{k}}{4}\right)W + o(W)$$

$$\leq \left(1 - \frac{\gamma}{4d}\right)mW + o(W),$$

where the last inequality is due to our assumption $\bar{k} \geq \bar{m}/d$. We conclude the theorem by noticing $N = mW + o(W)$ and letting τ be a constant strictly less than $\frac{\gamma}{4d}$. □

3.5 Lift, Coupling and Upper Bounds

In this section, we define the *lift* of an undirected graph G with respect to a vertex set $A \subseteq V$, which is a new undirected graph $\widehat{G}_A$ that has the same vertex set A with G plus a lot of new vertices. We will then define a coupling between sampling live-edges in G and sampling live-edges in $\widehat{G}_A$ (refer to Sect. 2.1.2 for details about live-edges). Given the seed set S, this coupling reveals an upper bound of $\sigma_G(S)$ for each of UICM, WICM and ULTM on undirected graphs. In particular, we will show that, for $A = S$ being the seed set, $\sigma_G(S) \leq \sigma_{\widehat{G}_A}(S)$ for all these models. These bounds will imply Theorem 3.4, 3.5 and 3.6 in Sect. 3.4, and the upper bounds in Theorem 3.5 and 3.6 are also the key in the InfMax heuristic presented in Sect. 3.6.

Let

$$\mathcal{P}_A = \{P = ((v_1, v_2), (v_2, v_3), \ldots, (v_{t-1}, v_t)) : v_1 \in A; v_2, \ldots, v_t \notin A; \forall i \neq j : v_i \neq v_j\}$$

be the set of all simple paths P that start from vertices in A but never come back to A.

Definition 3.11. Given an undirected graph $G = (V, E)$ and $A \subseteq V$, the *lift* of G with respect to A, denoted by $\widehat{G}_A = (\widehat{V}, \widehat{E})$, is an undirected graph defined as follows.

- The vertex set is $\widehat{V} = A \cup V_P$, where $V_P = \{v_P : P \in \mathcal{P}_A\}$ is the set of vertices corresponding to the simple paths in $\mathcal{P}_A$.

- For each $u \in A$ and $v_P \in V_P$, include $(u, v_P) \in \widehat{E}$ if P is a path of length 1 that starts from u; for each $v_{P_1}, v_{P_2} \in V_P$, include (v_{P_1}, v_{P_2}) if $|P_2| = |P_1| + 1$ and P_2, P_1 share the first $|P_1|$ common edges (or $|P_1| = |P_2| + 1$ and P_1, P_2 share the first $|P_2|$ common edges, since $\widehat{G}_A$ is undirected); do not include any edge between any two vertices in A.

It is easy to see that the same vertex set A is in both $\widehat{G}_A$ and G, and $\widehat{G}_A$ is a *forest* with roots being exactly those vertices in A. The vertices in the tree rooted at $u \in A$ in $\widehat{G}_A$ correspond to all the paths in $\mathcal{P}_A$ starting at u. For any path $P \in \mathcal{P}_A$ with v being its ending vertex, $\deg(v_P)$ in $\widehat{G}_A$ is less than or equal to $\deg(v)$ in G. Specifically, $\deg(v) - \deg(v_P)$ equals to exactly the number of v's neighbors that are in A. Figure 3.2 shows an example of G and $\widehat{G}_A$ (please ignore the third graph in Fig. 3.2 at this moment).

Next, we present a coupling argument to show that $\sigma_G(S) \leq \sigma_{\widehat{G}_S}(S)$ for all the diffusion models considered. Intuitively, we somehow have separated different "trends" of the cascade on G by considering the corresponding cascade process on G's tree-like counterpart $\widehat{G}_S$. If a seed s infects vertices $v_1, v_2, \ldots, v_t$ one by one along the path

$$P_t := ((s, v_1), (v_1, v_2), \ldots, (v_{t-1}, v_t)),$$

it corresponds to the case that the same seed s in the lift $\widehat{G}_S$ infects $v_{P_1}, v_{P_2}, \ldots, v_{P_t}$ one by one, where

$$\begin{aligned}
P_1 &:= ((s, v_1)),\\
P_2 &:= ((s, v_1), (v_1, v_2)),\\
&\vdots\\
P_t &:= ((s, v_1), (v_1, v_2), \ldots, (v_{t-1}, v_t)).
\end{aligned}$$

We will define the coupling describing the above correspondence. Moreover, as we will see later, $\widehat{G}_S$ contains many more vertices than G, which potentially produces more infected vertices.

Let $\Psi : E \to 2^{\widehat{E}}$ be the function mapping an undirected edge in G to its counterparts in $\widehat{G}_A$:

$$\Psi(e) = \begin{cases} \emptyset & \text{if } e = (u, v) \text{ for } u, v \in A \\ \{(u, v_P) \mid P = ((u, v))\} & \text{if } e = (u, v) \text{ for } u \in A, v \notin A \\ \{(v_{P_1}, v_{P_2}) \mid P_2 = (P_1, e)\} & \text{Otherwise.} \end{cases}$$

Notice that in the above definition, $\Psi(e)$ contains only a single edge (u, v_P) with $P = ((u, v))$ being the length-one path connecting u, v if $u \in A$ and $v \notin A$, while $\Psi(e)$ contains the set of all (v_{P_1}, v_{P_2}) such that P_2 is obtained by appending e to P_1. Let $\Phi : V \to 2^{\widehat{V}}$ represent the vertex correspondence:

$$\Phi(v) = \begin{cases} \{v\} & \text{if } v \in A \\ \{v_P \mid P \text{ ends at } v\} & \text{Otherwise.} \end{cases}$$

From our definition, it is easy to see that $\Psi(e_1) \cap \Psi(e_2) = \emptyset$ if $e_1 \neq e_2$, and $\Phi(u) \cap \Phi(v) = \emptyset$ if $u \neq v$. Moreover, since $\mathcal{P}_A$ contains only paths, for any vertex v and edge e in G, each path in $\widehat{G}_A$ connecting a root in A to a leaf (recall that $\widehat{G}_A$ is a forest with the set of roots exactly A) can intersect each of $\Psi(e)$ and $\Phi(v)$ at most once.[6] We will use this fact multiple times later.

3.5.1 Upper Bound for Uniform Independent Cascade Model

We consider `UICM` with p being fixed for the weight of all edges in this subsection.

Lemma 3.12. *Given a uniform independent cascade model* $\mathtt{UICM}_{G,p}$ *with an undirected graph* $G = (V, E)$ *and a seed set* $S \subseteq V$*, we have*

$$\sigma_{G,p}(S) \leq \sigma_{\widehat{G}_S,p}(S).$$

Proof. We will define a coupling between the process of revealing live-edges in G and the process of revealing live-edges in $\widehat{G}_S$. We shall ignore those edges that are internal to S in G, as they do not affect the number of infected vertices. Let χ_G be the edge-revelation process in G, and $\chi_{\widehat{G}_S}$ be the edge-revelation process in $\widehat{G}_S$, where each edge is revealed with probability p independently in both processes. We will couple χ_G with another edge-revelation process $\chi'_{\widehat{G}_S}$ of $\widehat{G}_S$.

We consider the following coupling. For each undirected edge e in G, we reveal both of its corresponding two anti-parallel directed edges, and meanwhile reveal the two anti-parallel edges for each undirected edge in $\Psi(e)$ in the following way:

[6] To see this for each $\Psi(e)$, suppose for the sake of contradiction that the path from v_P to the root contains two edges $(v_{P_1}, v_{P_2}), (v_{P_3}, v_{P_4})$ such that $(v_{P_1}, v_{P_2}), (v_{P_3}, v_{P_4}) \in \Psi(e)$ for some edge e. Assume without loss of generality that the order of the four vertices on the path according to the distances to the root is $(v_{P_1}, v_{P_2}, v_{P_3}, v_{P_4})$. It is easy to see from our construction that $P_1 \subsetneq P_2 \subsetneq P_3 \subsetneq P_4$. As a result, $(v_{P_1}, v_{P_2}), (v_{P_3}, v_{P_4}) \in \Psi(e)$ implies that P_2 is the path obtained by appending e to P_1, and P_4, containing P_2, P_3, is obtained by appending e to P_3, which further implies that P_4 is a path that uses the edge e twice, contradicting to our definition that $\mathcal{P}_S$ contains only simple paths.

The corresponding claim for each $\Phi(v)$ can be shown similarly.

- if $e = (u, v)$ for $u \in S$ and $v \notin S$, then for each undirected edge $(u, v_P) \in \Psi(e)$ where $P = ((u, v))$, make (u, v_P) live if and only if (u, v) is live in χ_G, and make (v_P, u) live if and only if (v, u) is live in χ_G;
- if $e = (u, v)$ for $u, v \notin S$, then
  - for each undirected edge $(v_{P_1}, v_{P_2}) \in \Psi(e)$ where P_1 ends at u and $P_2 = (P_1, e)$, make (v_{P_1}, v_{P_2}) live if and only if (u, v) is live in χ_G, and make (v_{P_2}, v_{P_1}) live if and only if (v, u) is live in χ_G;
  - for each undirected edge $(v_{P_1}, v_{P_2}) \in \Psi(e)$ where P_1 ends at v and $P_2 = (P_1, e)$, make (v_{P_1}, v_{P_2}) live if and only if (v, u) is live in χ_G, and make (v_{P_2}, v_{P_1}) live if and only if (u, v) is live in χ_G.

This defines a coupling between χ_G and $\chi'_{\widehat{G}_S}$. Notice that the two processes $\chi'_{\widehat{G}_S}$ and $\chi_{\widehat{G}_S}$ are not the same, because the edges in $\chi'_{\widehat{G}_S}$ are not revealed independently. However, we will show that the expected number of vertices that are reachable from S by live edges is the same in both $\chi'_{\widehat{G}_S}$ and $\chi_{\widehat{G}_S}$, which implies that the expected number of vertices that are reachable from S by live edges in $\chi'_{\widehat{G}_S}$ is still exactly $\sigma_{\widehat{G}_S}(S)$.

To see this, it suffices to show that, for each $v_P \in \widehat{V}$, all the directed edges in the path connecting from the root of the tree v_P is in to the vertex v_P are sampled independently, since this would imply that the probability v_P is connected to a seed is the same in both $\chi'_{\widehat{G}_S}$ and $\chi_{\widehat{G}_S}$, and the total number of vertices reachable from S by live edges is the same by the linearity of expectation. We only need to show that there do not exists two edges on this path that are in the same set $\Psi(e)$ for some $e \in E$, since edges in $\Psi(e_1)$ are revealed independently to the revelations of edges in $\Psi(e_2)$ whenever $e_1 \neq e_2$ (this is because e_1, e_2 are revealed independently in χ_G). This is true since paths in $\mathcal{P}_S$ are all simple paths, as remarked earlier.

To conclude the lemma, we will show that the number of the vertices reachable from S in χ_G is always at most the number of vertices reachable from S in $\chi'_{\widehat{G}_S}$. It is easy to see that, if $v \in V$ is connected to S by a path P consisting of live edges, the vertex $v_P \in \widehat{V}$ is also connected to S in $\widehat{G}_S$ by live edges. Thus, there exists at least one vertex in $\Phi(u)$ connected to S by live edges in $\widehat{G}_S$ for each vertex $u \in V$ connected to S by live edges in G. The lemma follows from that $\Phi(u)$'s are non-overlapping. □

Now we are ready to prove Theorem 3.4.

Proof of Theorem 3.4. Fix an arbitrary $p \in [0, 1/d)$ (as assumed in the theorem statement). For each seed $s \in S \subseteq V$, let δ_s be the number of s's neighbors in $V \setminus S$.

Let T be a forest such that each vertex $s \in S$ is the root with δ_s children, and each child is a root of an infinite full d-ary subtree. Clearly, $\widehat{G}_S$ is a subgraph of T, so $\sigma_{T,p}(S) \geq \sigma_{\widehat{G}_S,p}(S) \geq \sigma_{G,p}(S)$ by Lemma 3.12. For each tree in T rooted at $s \in S$, the probability that a vertex on level i is infected is p^i, and there are $\delta_s d^{i-1}$ vertices on level i. Therefore, we have

$$\sigma_{T,p}(S) = |S| + \sum_{s \in S} \sum_{i=1}^{\infty} \delta_s d^{i-1} p^i = |S| + \sum_{i=1}^{\infty} |E(S, V \setminus S)| d^{i-1} p^i = |S| + \frac{|E(S, V \setminus S)| p}{1 - pd},$$

which implies the theorem. □

Lemma 3.12 and Theorem 3.4 can be generalized to directed graphs as well. The lift $\widehat{G}_A$ should then be defined as the directed forest such that the root for each tree is the source and the leaves are the sinks. With the same proofs, Lemma 3.12 and Theorem 3.4 follow for directed graphs, where d in Theorem 3.4 becomes the maximum *out-degree*. Since we only need Theorem 3.4 for undirected graphs and the generalization to directed graphs is straightforward, we omit the details for directed graphs. Notice that, however, the results in the next section for LTM do not generalize to directed graphs.

3.5.2 Upper Bound for Uniform Linear Threshold Model

In the edge-revelations for ULTM, the incoming edges of a single vertex are revealed dependently, with one of them being live. This suggests that we should couple the two edge-revelation processes by vertices instead of edges.

The following lemma has been shown by Lim et al. [56]. We include a simple proof here for the completeness.

Lemma 3.13. *Consider a uniform linear threshold model $\texttt{ULTM}_T$ with the graph T being an undirected tree. Let $S = \{s\}$ be a seed set containing only a single seed s. We have $\sigma_T(S) = \deg(s) + 1$.*

Proof. We assume without loss of generality that T is rooted at s. Consider an arbitrary vertex $v \neq s$ at the second last level with children $v_1, \ldots, v_t$ being leaves of T. We have $\deg(v) = t + 1$. Suppose v's parent u is infected by s with probability x ($x = 1$ if $u = s$). Then v will be infected with probability $x/(t+1)$, and all of $v_1, \ldots, v_t$, having degree 1, will be infected with probability 1 if v is infected. Therefore, the expected number of infected vertices in the subtree rooted at v is $\frac{x}{t+1}(1+t) + (1 - \frac{x}{t+1}) \cdot 0 = x$. This suggests that, if we contract the subtree rooted at

v to a single vertex v, the expected total number of infected vertices stays the same for this change of the graph T, since the degree of v becomes 1 after this contraction, making the infection probability of v be the same as that of u, which is x. We can keep doing this contraction until T becomes a star with center s, and the expected number of infected vertices remains the same. The lemma follows. □

Lemma 3.14 is proved similarly as Lemma 3.12, except that we will couple the two processes by vertices instead of by edges: for each v in the original graph, we reveal all its incoming edges simultaneously, and this is coupled with the revelations of all incoming edges for each vertex in $\Phi(v)$. Although the corresponding vertex in $\widehat{G}_S$ may have a less degree: $\deg(v) - \deg(v_P) > 0$ for certain $v_P \in \Phi(v)$ (this makes the weights of the incoming edges inconsistent), we can add dummy vertices to make the degrees consistent, and Lemma 3.13 ensures this modification does not change the expected number of infected vertices.

Lemma 3.14. *Given a uniform linear threshold model $\texttt{ULTM}_G$ with an undirected graph $G = (V, E)$ and a seed set $S \subseteq V$, we have*

$$\sigma_G(S) \leq \sigma_{\widehat{G}_S}(S).$$

Proof. First of all, we modify the graph $\widehat{G}_S$ such that $\deg(v_P) = \deg(v)$ for each $v_P \in \Phi(v)$. We have seen that $\deg(v) - \deg(v_P) \geq 0$, and so we can add $(\deg(v) - \deg(v_P))$ dummy vertices that connect to v_P only. Notice that Lemma 3.13 ensures this modification does not change the expected number of infected vertices: for each $s \in S$, the degree of s in $\widehat{G}_S$ will be the same as the degree of s in G after removing all internal edges of S (the vertices that are adjacent to s in $\widehat{G}_S$ are exactly those paths of length 1 in G, which corresponds to exactly the neighbors of s in G); thus, the tree rooted at s will have exactly $\deg(s) + 1$ infected vertices in expectation, with or without modification (the resultant graph is still a forest after connecting those dummy vertices). We will let $\widehat{G}_S$ be this modified graph from now on.

We will define a coupling between the process of revealing live-edges in G and the process of revealing live-edges in $\widehat{G}_S$. We shall ignore those edges that are internal to S in G, as they do not affect the number of infected vertices. Let χ_G be the edge-revelation process in G, and $\chi_{\widehat{G}_S}$ be the edge-revelation process in $\chi_{\widehat{G}_S}$, where in both processes, each edge is viewed as two anti-parallel directed edges, and we always reveal all the incoming edges for a vertex simultaneously by choosing exactly one incoming edge uniformly at random. Again, we will couple χ_G with another edge-revelation process $\chi'_{\widehat{G}_S}$ of $\widehat{G}_S$.

We consider the following coupling. In each iteration where all the incoming edges of v, denoted by $(u_1, v), (u_2, v), \ldots, (u_{\deg(v)}, v)$, are revealed such that exactly one of them is live, we reveal all the incoming edges for each $v_P \in \Phi(v)$ as follows.

- For each P' such that $v_{P'}$ is a neighbor of v_P, there exists $u_i \in \{u_1, \ldots, u_{\deg(v)}\}$ such that either that P' is obtained by appending (v, u_i) to P or that P is obtained by appending (u_i, v) to P'. Reveal the directed edge $(v_{P'}, v_P)$ such that it is live if and only if (u_i, v) is live in G.

- If there is a live edge $(v_{P'}, v_P)$ revealed in the above step, make all the remaining directed edges connecting the dummy vertices to v_P not be live. If no live edge is revealed in the above step, choose exactly one of the directed edges connecting the dummy vertices to v_P as the live edge uniformly at random.

This defines a coupling between χ_G and $\chi'_{\widehat{G}_S}$. It is easy to check that each $v_P \in \widehat{V}$ chooses exactly one of its incoming edges uniformly at random in this coupling, which is the same as it is in the process $\chi_{\widehat{G}_S}$. The difference is that, there are dependencies between the revelations of incoming edges for different vertices in $\widehat{G}_S$: if both $v_P, v_{P'} \in \widehat{V}$ belongs to the same $\Phi(v)$ for some $v \in V$, the incoming edges for v_P and $v_{P'}$ are revealed in the same iteration and in the same way.

Same as it is in the proof of Lemma 3.12, although the two processes $\chi'_{\widehat{G}_S}$ and $\chi_{\widehat{G}_S}$ are not the same, we will show that the expected number of vertices that are reachable from S by live edges is the same in both $\chi'_{\widehat{G}_S}$ and $\chi_{\widehat{G}_S}$. It suffices to show that, for each $v_P \in \widehat{V}$, all the *vertices* in the path connecting v_P to the root of the tree v_P is in are considered independently (meaning that the incoming edges for v_{P_1} on the path are revealed independently to the revelations of the incoming edges of v_{P_2}), since this would imply that the probability v_P is connected to a seed is the same in both $\chi'_{\widehat{G}_S}$ and $\chi_{\widehat{G}_S}$, and the total number of vertices reachable from S by live edges is the same by the linearity of expectation. We only need to show that there do not exist two vertices on this path that are in the same set $\Phi(v)$ for some $v \in V$, since the incoming edges of each $v_{P_1} \in \Phi(v_1)$ are revealed independently to the revelations of the incoming edges of each $v_{P_2} \in \Phi(v_2)$ whenever $v_1 \neq v_2$. This is true due to that all the paths in $\mathcal{P}_S$ are simple paths, as remarked in the paragraph below where we define function $\Phi(\cdot)$.

Following the same analysis before, we can show that the number of the vertices reachable from S in χ_G is always at most the number of vertices reachable from S in $\chi'_{\widehat{G}_S}$. The lemma concludes here. □

Now we are ready to prove Theorem 3.5.

Proof of Theorem 3.5. For each $s \in S$, let n_s be the number of s's incident edges with the other ends in $V \setminus S$. It is easy to see that $\widehat{G}_S$ is a forest such that each $s \in S$ has exactly degree n_s. By Lemma 3.14 and Lemma 3.13,

$$\sigma_G(S) \leq \sigma_{\widehat{G}_S}(S) = \sum_{s \in S}(n_s + 1) = |E(S, V \setminus S)| + |S|,$$

which implies the theorem. □

In particular, if we take $S = \{s\}$ in the theorem above, $\deg(s) + 1$ in Lemma 3.13 becomes an upper bound.

Corollary 3.15. *Consider a uniform linear threshold model* $\mathtt{ULTM}_G$ *with an undirected graph* G*. Let* $S = \{s\}$ *be a seed set containing only a single seed* s*. We have* $\sigma_G(S) \leq \deg(s) + 1$.

This suggests that, in `ULTM` on undirected graphs, trees provide the most number of infections, and adding more edges to a tree may only reduce the total number of infections.

3.5.3 Upper Bound for Weighted Independent Cascade Model

Firstly, Lemma 3.13 holds for `WICM` as well, with exactly the same proof.

Lemma 3.16. *Consider a weighted independent cascade model* $\mathtt{WICM}_T$ *with the graph* T *being an undirected tree. Let* $S = \{s\}$ *be a seed set containing only a single seed* s*. We have* $\sigma_T(S) = \deg(s) + 1$.

Next, we can show the following lemma by combining ideas in the proofs of both Lemma 3.12 and 3.14.

Lemma 3.17. *Given a weighted independent cascade model* $\mathtt{WICM}_G$ *with an undirected graph* $G = (V, E)$ *and a seed set* $S \subseteq V$*, we have*

$$\sigma_G(S) \leq \sigma_{\widehat{G}_S}(S).$$

Proof. We first modify the graph $\widehat{G}_S$ in the same way as described in the first paragraph of the proof of Lemma 3.14. Then, the remaining part of the coupling follows the one in the proof of Lemma 3.12. Notice that the modification of $\widehat{G}_S$ ensures that

$v \in V$ has the same degree as the degree of each vertex in $\Phi(v)$. This validates the coupling between χ_G and $\chi'_{\widehat{G}_S}$. The remaining part of the proof is the same as the proof of Lemma 3.12. □

Applying the same arguments in the proof of Theorem 3.5, Lemma 3.17 implies Theorem 3.6. The following corollary of Theorem 3.6, as a counterpart to Corollary 3.15, shows that trees also provide the most number of infections for `WICM`.

Corollary 3.18. *Consider a weighted independent cascade model* $\texttt{WICM}_G$ *with an undirected graph* G. *Let* $S = \{s\}$ *be a seed set containing only a single seed* s. *We have* $\sigma_G(S) \leq \deg(s) + 1$.

3.5.4 Refined Upper Bounds for ULTM and WICM

The upper bounds in Theorem 3.5 and 3.6 can be further refined, and the refined upper bounds will yield a local greedy heuristic with better performance than the heuristic based on upper bound in Theorem 3.5 and 3.6.

Theorem 3.19. *Given a uniform linear threshold model* $\texttt{ULTM}_G$ *with an undirected graph* $G = (V, E)$ *and a seed set* $S \subseteq V$, *we have*

$$\sigma_G(S) \leq |S| + \sum_{v \in V \setminus S} \frac{\delta_v}{\deg(v)} \left(1 + \deg(v) - \delta_v\right),$$

where δ_v *is the number of* v*'s neighbors in* S.

Theorem 3.20. *Given a weighted independent cascade model* $\texttt{WICM}_G$ *with an undirected graph* $G = (V, E)$ *and a seed set* $S \subseteq V$, *we have*

$$\sigma_G(S) \leq |S| + \sum_{v \in V \setminus S} \frac{\delta_v}{\deg(v)} \left(1 + \deg(v) - \delta_v\right),$$

where δ_v *is the number of* v*'s neighbors in* S.

Notice that both theorem above indeed provide a tighter upper bound compared to Theorem 3.5, as

$$|E(S, V \setminus S)| = \sum_{v \in \partial S} \delta_v \geq \sum_{v \in V \setminus S} \frac{\delta_v}{\deg(v)} \left(1 + \deg(v) - \delta_v\right).$$

Towards proving Theorem 3.5 and 3.6, we have considered the lift of G with respect to S such that $\widehat{G}_S$ is a forest rooted at vertices in S, and we have shown that

$\sigma_G(S) \leq \sigma_{\widehat{G}_S}(S)$ by coupling. We need a slightly more complicated way to "lift" G in order to show Theorem 3.19, in which we make those vertices in ∂S to be the roots of the trees, where $\partial S = \{v \in V \setminus S \mid \exists s \in S : (s, v) \in E\}$ is the set of all non-seed vertices that are adjacent to a seed.

Similar to the definition of $\mathcal{P}_A$, let

$$\mathcal{P}_{\partial A} = \{P = ((v_1, v_2), (v_2, v_3), \ldots, (v_{t-1}, v_t)) : v_1 \in \partial A; v_2, \ldots, v_t \notin A; \forall i, j : v_i \neq v_j\}$$

be the set of all simple paths P that start from vertices in ∂A but never reach a vertex in A (we allow the paths passing a vertex in ∂A in the middle).

Definition 3.21. Given an undirected graph $G = (V, E)$ and $A \subseteq V$, the *boundary lift* of G with respect to A, denoted by $\widehat{G}^b_A = (\widehat{V}, \widehat{E})$, is an undirected graph defined as follows.

- The vertex set is $\widehat{V} = A \cup \partial A \cup V_P$, where $V_P = \{v_P : P \in \mathcal{P}_{\partial A}\}$ is the set of vertices corresponding to the simple paths in $\mathcal{P}_{\partial A}$.
- For each $u \in A$ and $v \in \partial A$, include $(u, v) \in \widehat{E}$ if $(u, v) \in E$; for each $u \in \partial A$ and $v_P \in V_P$, include $(u, v_P) \in \widehat{E}$ if P is a path of length 1 that starts from u; for each $v_{P_1}, v_{P_2} \in V_P$, include (v_{P_1}, v_{P_2}) if $|P_2| = |P_1| + 1$ and P_2, P_1 share the first $|P_1|$ common edges (or $|P_1| = |P_2| + 1$ and P_1, P_2 share the first $|P_2|$ common edges, since $\widehat{G}_A$ is undirected); do not include any edge between any two vertices in A; do not include any edge between any two vertices in ∂A.

It is easy to see that the same vertex set $A \cup \partial A$ is in both $\widehat{G}^b_A$ and G. Moreover, for each $v \in \partial A$, if removing those edges from v to vertices in A, the connected component that contains v is a tree rooted at v. Figure 3.2 shows an example of G, $\widehat{G}_A$ and $\widehat{G}^b_A$.

Corresponding, the definitions for $\Psi(\cdot)$ and $\Phi(\cdot)$ are changed accordingly as follows.

$$\Psi(e) = \begin{cases} \emptyset & \text{if } e = (u, v) \text{ for } u, v \in A \\ \{(u, v)\} & \text{if } e = (u, v) \text{ for } u \in A, v \in \partial A \\ \{(v_{P_1}, v_{P_2}) \mid P_2 = (P_1, e)\} & \text{Otherwise.} \end{cases}$$

$$\Phi(v) = \begin{cases} \{v\} & \text{if } v \in A \cup \partial A \\ \{v_P \mid P \text{ ends at } v\} & \text{Otherwise.} \end{cases}$$

It is straightforward to check that all the properties of the two functions in Sect. 3.5 continue to hold here.

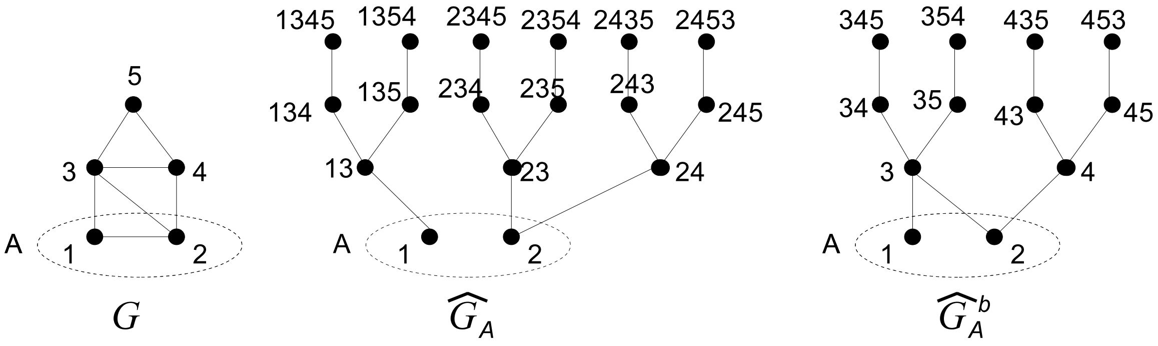


Figure 3.2: The lift and the boundary lift of G

Proof of Theorem 3.19. By the same coupling in the proof of Lemma 3.14, we can show that

$$\sigma_G(S) \le \sigma_{\widehat{G}_S^b}(S).$$

For each $v \in \partial S$, we denote its neighbors as $s_1, \dots, s_{\delta_v}, u_1, \dots, u_{\deg(v)-\delta_v}$, where $s_1, \dots, s_{\delta_v}$ are v's neighbors in S and $u_1, \dots, u_{\deg(v)-\delta_v}$ are v's remaining neighbors. As we have mentioned, if we remove the δ_v edges $(s_1, v), \dots, (s_{\delta_v}, v)$, v is the root of a tree, with $u_1, \dots, u_{\deg(v)-\delta_v}$ being all the children of v. Since this tree contains no seed, we can apply the contraction argument in the proof of Lemma 3.13 to iteratively contract the leaves of this tree, until at the stage where the tree only contains v and $u_1, \dots, u_{\deg(v)-\delta_v}$. We do this contraction for each $v \in \partial S$. It is then simple to compute $\sigma_{\widehat{G}_S^b}(S)$: each $v \in \partial S$ is infected with probability $\frac{\delta_v}{\deg(v)}$, and it will further infect all its $\deg(v) - \delta_v$ children if infected. Therefore,

$$\sigma_{\widehat{G}_S^b}(S) = |S| + \sum_{v \in \partial S} \frac{\delta_v}{\deg(v)} \left(1 + \deg(v) - \delta_v\right).$$

The theorem follows from $\sigma_G(S) \le \sigma_{\widehat{G}_S^b}(S)$. □

Proof of Theorem 3.20. By the same coupling in the proof of Lemma 3.17, we can show that

$$\sigma_G(S) \le \sigma_{\widehat{G}_S^b}(S).$$

For each $v \in \partial S$, we denote its neighbors as $s_1, \dots, s_{\delta_v}, u_1, \dots, u_{\deg(v)-\delta_v}$, where $s_1, \dots, s_{\delta_v}$ are v's neighbors in S and $u_1, \dots, u_{\deg(v)-\delta_v}$ are v's remaining neighbors. By the same arguments in the previous proof, we can iteratively contract the leaves of this tree, until at the stage where the tree only contains v and $u_1, \dots, u_{\deg(v)-\delta_v}$. We do this contraction for each $v \in \partial S$. It is then simple to compute $\sigma_{\widehat{G}_S^b}(S)$: each

$v \in \partial S$ is infected with probability $1 - \left(1 - \frac{1}{\deg(v)}\right)^{\delta_v} \leq \frac{\delta_v}{\deg(v)}$, and it will further infect all its $\deg(v) - \delta_v$ children if infected. Therefore,

$$\sigma_{\widehat{G}_S^b}(S) = |S| + \sum_{v \in \partial S} \frac{\delta_v}{\deg(v)} \left(1 + \deg(v) - \delta_v\right).$$

The theorem follows from $\sigma_G(S) \leq \sigma_{\widehat{G}_S^b}(S)$. □

3.5.5 Discussions about Scalability of Reverse-Reachable-Set-Based Algorithms

As mentioned in Sect. 1.2.4 and Sect. 2.2, reverse-reachable-set-based algorithms give $(1 - 1/e - \varepsilon)$ approximation for InfMax, and are considered as the state-of-the-art in that they produce high quality seeds (almost as good as the greedy algorithm) and are moderately scalable.

Arora et al. [3] observed that these algorithms are much more scalable on `ULTM` and `WICM` than on `UICM`. In particular, Arora et al. [3] pointed out that IMM requires significantly more memory for `UICM` compared to `WICM` (see Fig. 1 in their paper). This indicates that the average size of a random reverse reachable set for `UICM` is significantly larger.

Notice that most of the networks used by Arora et al. [3] are undirected. Our upper bounds in Corollary 3.15 and 3.18 provide a sound theoretical justification of this phenomenon. For `UICM`, p is usually set to 0.1 in the simulations. If a graph contains a lot of well-connected vertices with degrees significantly higher than 10, as it is the case in large social networks, then the size of a random reverse reachable set is likely to be large. On the other hand, Corollary 3.15 and 3.18 suggest that, for `ULTM` and `WICM`, the expected size of a reverse reachable set is upper-bounded by the degree of the randomly chosen vertex plus 1, which is small.

3.6 Highly Scalable Heuristics with Empirical Good Performance

For both `ULTM` and `WICM` with undirected graphs, Theorem 3.5 (Theorem 3.6) and Theorem 3.19 (Theorem 3.20) provide upper bounds based only on the local graph structure around the seed set S and the vertices that are adjacent to the seeds. Towards proving both theorems, the (boundary) lift of the graph G with respect to

```
input : An undirected graph $G = (V, E)$ and an integer $k$
initialize $S = \emptyset$
for $i = 1, \ldots, k$ do
    find $v \in V \setminus S$ maximizing $f(S \cup \{v\}) - f(S)$ ;        // follow Table 3.1
    update $S \leftarrow S \cup \{v\}$
end
return $S$
```

Algorithm 3.1: Local greedy heuristic with $f \in \{f_v, f_e, f_b, f_{ve}^{\lambda}, f_{vb}^{\lambda}\}$

the seed set S has been considered, where the network G has been modified to a forest. We have also seen that tree structures provide the maximum numbers of infections, and adding more edges can only decrease the expected number of infections. Since most of the networks in our real life are sparse, it is very possible that tree structures also provide good *approximations* to the number of infected vertices, and the seeds S maximizing $E(S, V \setminus S)$ or $\sum_{v \in \partial S} \frac{\delta_v}{\deg(v)} (1 + \deg(v) - \delta_v)$ as appeared in the two upper bounds are likely to provide good expected number of infections. This motivates our *local greedy heuristics* in this section.

3.6.1 Local Greedy Heuristics

We consider three local features of a seed set S:

Vertex Cut: Number of vertices adjacent to S, denoted by $f_v(S) = |\partial S|$.

Edge Cut: Number of edges between S and $V \setminus S$, denoted by $f_e(S) = |E(S, V \setminus S)|$.

Boundary Expansion: Second term in the upper bound in Theorem 3.19, denoted by $f_b(S) = \sum_{v \in \partial S} \frac{\delta_v}{\deg(v)} (1 + \deg(v) - \delta_v)$.

We also consider mixtures of them:

Mixture of Vertex Cut and Edge Cut: $f_{ve}^{\lambda}(S) = \lambda f_v(S) + (1 - \lambda) f_e(S)$, and

Mixture of Vertex Cut and Boundary Expansion:
$f_{vb}^{\lambda}(S) = \lambda f_v(S) + (1 - \lambda) f_b(S)$,

where λ is a parameter set in the interval $(0, 1)$.

For a chosen objection function $f \in \{f_v, f_e, f_b, f_{ve}^{\lambda}, f_{vb}^{\lambda}\}$, we iteratively choose the seed that maximizes the marginal gain of f, the algorithm is shown in Algorithm 3.1. Notice that the marginal gain $f(S \cup \{v\}) - f(S)$ can be easily computed for each of the five objective functions as shown in Table 3.1 (where the marginal gain for

Objective Function	How to Compute Marginal Gain $f(S \cup \{v\}) - f(S)$
f_v	$+1$ for each v's neighbor in $V \setminus S$; -1 if $v \in \partial S$
f_e	$+1$ for each v's neighbor in $V \setminus S$; -1 for each v's neighbor in S
f_b	equals to $1 - \frac{\delta_v}{\deg(v)}(1 + \deg(v) - \delta_v) + \sum_{u \in \gamma(v) \setminus S} \left(1 - 2\frac{\delta_u}{\deg(u)}\right)$
f_{ve}^{λ}	equals to $\lambda\left(f_v(S \cup \{v\}) - f_v(S)\right) + (1-\lambda)\left(f_e(S \cup \{v\}) - f_e(S)\right)$
f_{vb}^{λ}	equals to $\lambda\left(f_v(S \cup \{v\}) - f_v(S)\right) + (1-\lambda)\left(f_b(S \cup \{v\}) - f_b(S)\right)$

Table 3.1: Computing marginal gain for each objective function

f_b in the table follows from straightforward calculations, which are omitted here). In particular, the marginal gains for all these objectives can be computed by only looking at v's neighbors, and we only need to check each neighbor's degree and the number of each neighbor's neighbors that are already chosen as seeds, which are all local properties that can be stored and updated in a look-up table.

3.6.2 The Heuristic DegreeDiscountIC

We will compare our heuristics to the degree discount heuristics in [15] which iteratively finds the vertex with highest degree and removes it from the graph[7], as they are similar to our heuristics in that only local features of the seeds are ever looked at. This algorithm is called "the single degree discount heuristic" in [15], for which we will name it `DegreeDiscount` in this chapter.

Another variant of this heuristic designed specifically for `UICM`, named "degree discount IC", was also proposed by Chen et al. [15], for which we will name it `DegreeDiscountIC`. The ideas and motivations of `DegreeDiscountIC` are as follows. In `DegreeDiscount`, the score of a candidate seed is computed by its degree minus the number of its neighbors that are already selected as seeds in the previous iterations. In other words, each edge connecting the candidate seed to the existing seed is "discounted" by 1. Intuitively, these edges, connecting to the existing seeds, play no role in further infections, so they should be discounted from the degree of the candidate seed. However, when considering `UICM` especially with small parameter p, discounting by 1 is too much and thus inaccurate. The heuristic `DegreeDiscountIC` shown in Algorithm 3.2 fixes this inaccuracy by using a more delicate estimation of this discount. Readers interested in more details of this heuristic can refer to [15].

[7] Notice the difference between our edge cut heuristic and the degree discount heuristic: if we start from the sum of the degrees of all the seeds, each internal edge is punished by -1 in the degree discount heuristic, and is punished by -2 in our edge cut heuristic (when adding all the degrees, each edge has been over-counted twice when computing the edge cut).

```
input : An undirected graph $G = (V, E)$, parameter $p \in [0, 1]$, and an integer $k$
initialize $S = \emptyset$
for each vertex $v$ do
    compute its degree $d_v$
    $dd_v \leftarrow d_v$
    initialize $t_v \leftarrow 0$
end
for $i = 1$ to $k$ do
    select $u = \arg\max_v \{dd_v \mid v \in V \setminus S\}$
    $S \leftarrow S \cup \{u\}$
    for each neighbor $v$ of $u$ and $v \in V \setminus S$ do
        $t_v \leftarrow t_v + 1$
        $dd_v \leftarrow d_v - 2t_v - (d_v - t_v)t_v p$
    end
end
return $S$
```

Algorithm 3.2: DegreeDiscountIC

Dataset	Number of Vertices	Number of Edges	Average Degree
CA-GrQc	5,242	14,490	5.53
ego-facebook	1,034	26,749	51.74
Nethept	15,233	31,387	4.12
CA-HepPh	12,008	118,505	19.73
DBLP	317,080	1,049,866	6.62
com-YouTube	1,134,890	2,987,624	5.26
LiveJournal	3,997,962	34,681,189	17.35

Table 3.2: Datasets for experiments

Chen et al. [15] observed that, surprisingly, the seeds output by this variant, although aiming for UICM, works well for LTM as well, with the parameter p set to 0.01.

3.6.3 Experimental Setup

We implement the experiments on seven undirected graphs, shown in Table 3.2. All of our datasets come from [49], and these networks are also popular choices in other empirical work.

All the experiments are implemented on a laptop with an Intel i7 processor and a 16GB memory. We compare the best one of our five local greedy heuristics (with our five different objective functions) to the standard greedy algorithm implemented

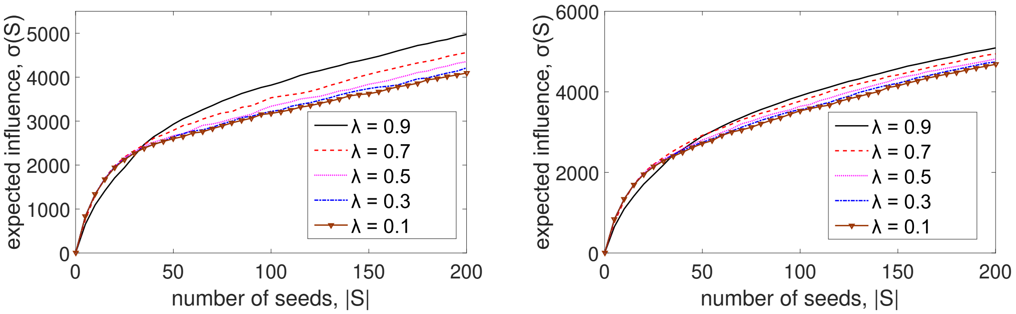


Figure 3.3: Comparing the performances of the local greedy heuristics with objective functions f_{ve}^{λ} (left) and f_{vb}^{λ} (right) with $\lambda = 0.9, 0.7, 0.5, 0.3, 0.1$ on dataset CA-HepPh under `ULTM`.

using 1,000,000 reverse reachable sets. Note that this is more than the number of the reverse reachable sets that are generated when standard heuristics (e.g., RIS, TIM$^+$ or IMM) are implemented in practice.[8] We also compare our heuristics to the simple degree heuristic that chooses the k vertices with highest degrees, and the two versions of the degree discount heuristic mentioned in the last section, `DegreeDiscount` and `DegreeDiscountIC`, where p is set to 0.01 for `DegreeDiscountIC`.

On those small networks (the first six networks in Table 3.2), we set the number of seeds $k = 200$ with the graph plotted for every seed (except for Fig. 3.3 where we are comparing different values of λ for f_{ve}^{λ} and f_{vb}^{λ}). For some large networks (the last three networks in Table 3.2), we set the number of seeds $k = 2000$.

3.6.4 Results

Figure 3.3 shows the performance of Algorithm 3.1 with the objective functions f_{ve}^{λ} and f_{vb}^{λ} for $\lambda = 0.9, 0.7, 0.5, 0.3, 0.1$ respectively, on the dataset CA-HepPh with `ULTM`. We notice that the seeds quality decreases as the value of λ decreases, for both f_{ve}^{λ} and f_{vb}^{λ}. This indicates that the local greedy heuristics with more emphasis on vertex cut are better. We observe the same phenomenon for the remaining six datasets. We will only consider $\lambda = 0.9$ for both f_{ve}^{λ} and f_{vb}^{λ} from now on.

Although it seems that our observation in Fig. 3.3 indicates that vertex cut is a better predictor for the seeds quality, setting $\lambda = 1$, i.e., implementing Algorithm 3.1 with the vertex cut objective f_v, is outperformed by both $f_{ve}^{0.9}$ and $f_{vb}^{0.9}$. Figure 3.4 compares the performances of $f_v, f_e, f_b, f_{ve}^{0.9}, f_{vb}^{0.9}$ on the dataset CA-HepPh, and the

[8] Our implementation with 1,000,000 reverse reachable sets produce slightly better quality of seed sets than those of TIM$^+$ and IMM as reported in [3], where TIM$^+$ and IMM are tested on the four networks, Nethept, HepPh, DBLP, Youtube, which we also used in our experiment.

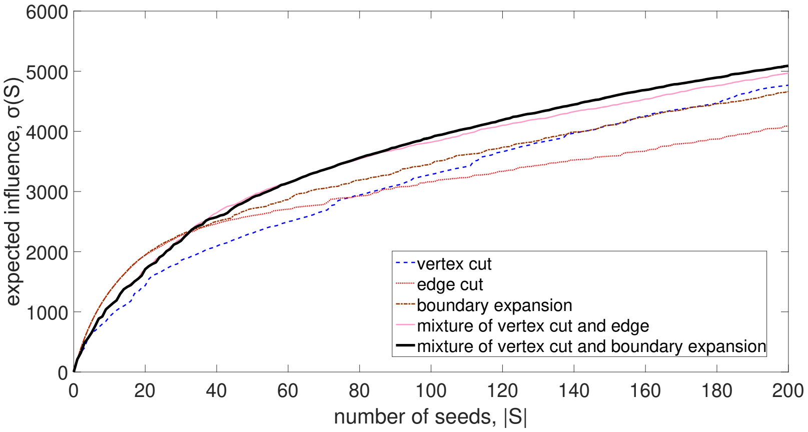


Figure 3.4: Comparing the performances of the local greedy heuristics with the five objective functions $f_e, f_v, f_{ve}^{0.9}, f_b, f_{vb}^{0.9}$ on CA-HepPh under `ULTM`.

mixture of boundary expansion and vertex cut $f_{vb}^{0.9}$ performs the best and the edge cut f_e performs the worst. These have been observed in the remaining six datasets as well, except for ego-facebook where the vertex cut is worse than the edge cut, while $f_{vb}^{0.9}$ is still the best. Notice that it is not surprising that f_b performs better than f_e nor that $f_{vb}^{0.9}$ performs better than $f_{ve}^{0.9}$, as we have remarked after Theorem 3.20 that the boundary expansion provides a better upper bound than the edge cut. As a result, we recommend using $f_{vb}^{0.9}$ in practice.

Finally, we compare, on all the datasets, the performance of $f_{vb}^{0.9}$ to the greedy maximum coverage algorithm on 1,000,000 reverse reachable sets, the two degree discount heuristics `DegreeDiscount` and `DegreeDiscountIC`, and the algorithm that simply picks the k vertices with highest degrees. We perform this comparison for `ULTM`, `WICM`, and `UICM`. Since the local characterization Theorem 3.5, Theorem 3.6, Theorem 3.19, and Theorem 3.20 do not hold for `UICM`, the performance of our local greedy heuristics is better on `ULTM` and `WICM` than on `UICM`, as we will see next.

Linear Threshold Model Figure 3.5 shows the results on the seven datasets. In all the experiments, our heuristic has slightly worse but almost the same performance with the greedy maximum coverage algorithm on reverse reachable sets, and our heuristic outperforms the two degree discount heuristics and the heuristic picking k vertices with highest degrees. In particular, our algorithm outperforms the two degree discount heuristics and the max-degree heuristic significantly in some of the datasets. On average, for `ULTM`, the performance of our heuristic is 97.52% of the performance of the much slower greedy algorithm (on 1,000,000 reverse reachable sets).[9] However, our heuristic does not require any MCMC simulations or reverse reachable sets samplings, and thus is much more scalable.

Weighted independent cascade model The results for the weighted independent cascade model are shown in Fig. 3.6. On average, the performance of the local greedy heuristic with $f_{vb}^{0.9}$ is 97.39% of the performance of the greedy maximum coverage algorithm on 1,000,000 reverse reachable sets.

Comparing Fig. 3.5 and Fig. 3.6, we can see that the shapes of the curves are very similar in each of the nine figures, although the numbers of infected vertices in both models are different (look at the values on the y-axis). This again suggests that the dynamics in both `ULTM` and `WICM` are similar.

Uniform independent cascade model The results for the uniform independent cascade model are shown in Fig. 3.7. The parameter p is set to 0.01. Since `DegreeDiscountIC` is specially designed for `UICM`, we will only compare our heuristic to `DegreeDiscountIC`, not `DegreeDiscount`. Our heuristic has similar performance with `DegreeDiscountIC` in all the six datasets. It is slightly worse than `DegreeDiscountIC` on CA-GrQc, Nethept, DBLP, it matches the performance of `DegreeDiscountIC` on ego-facebook and CA-HepPh, and it performs slightly better on com-YouTube. On average, the performance of our heuristic is 86.26% of the greedy algorithm (with 1,000,000 reverse reachable sets). This indicates that the local greedy heuristic is less promising for `UICM`. As mentioned earlier, this is not surprising, as our local characterization does not applied to this model.

3.7 Conclusion

We have seen that InfMax is APX-hard for both `LTM` and `ICM` on undirected graphs. For `LTM`, there is still a gap between the upper bound $(1-1/e)$ and the lower bound $(1-\tau)$, while the gap is slightly smaller for `ICM` with upper bound $(1-1/e+c)$ and lower bound $(1-\tau)$. A natural open problem is to close these gap by either designing an approximation algorithm, taking advantage of the undirected nature of the graph, that achieves approximation guarantee better than $(1-1/e)$ (or $(1-1/e+c)$ for `ICM`), or to show that the problem is hard to approximate with a larger gap. Another

[9] This percentage is averaging over the experiments with exactly 200 seeds. If including those with 2000 seeds, the percentage is 97.20%.

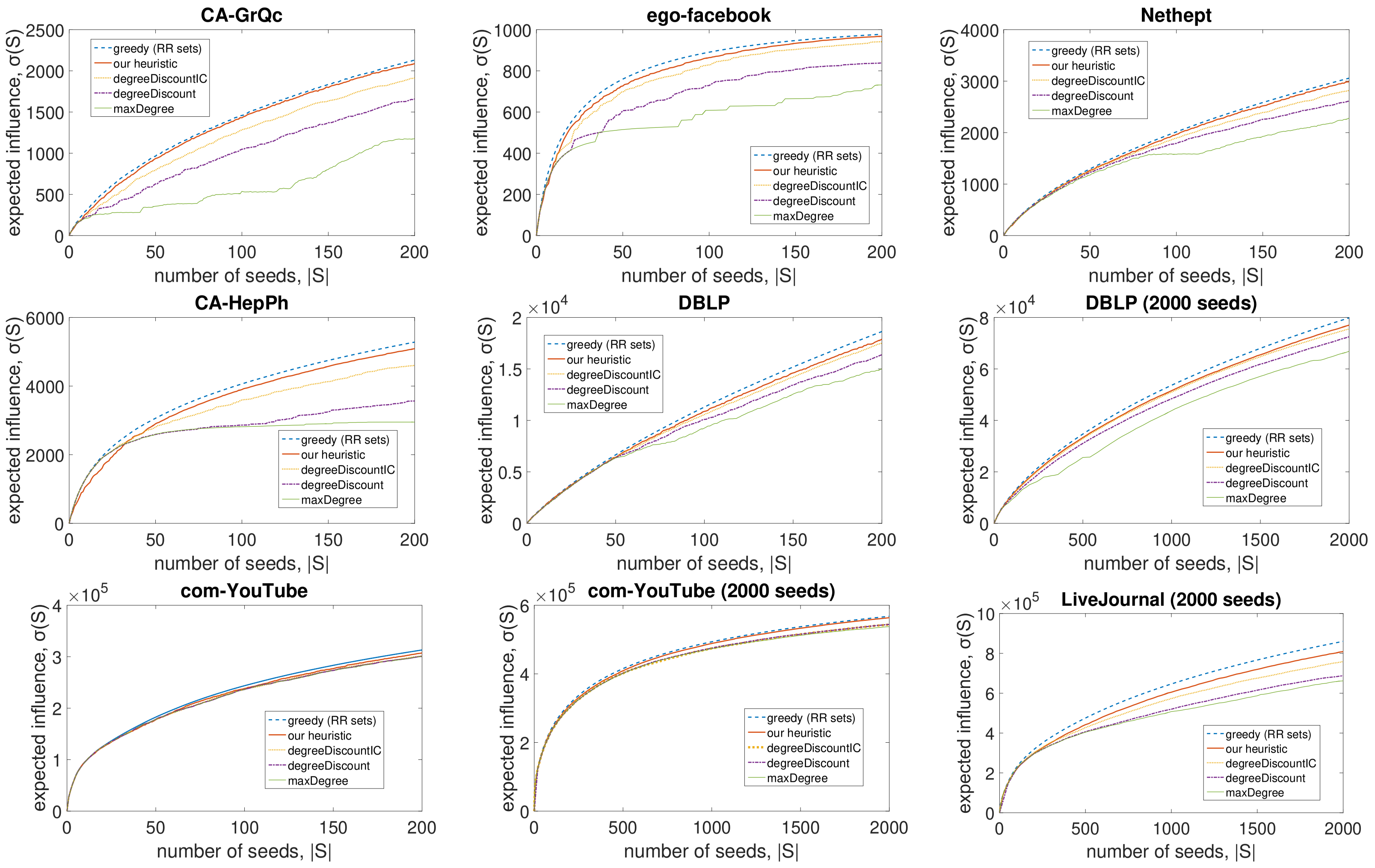


Figure 3.5: Comparing the local greedy heuristic with $f_{vb}^{0.9}$ to the greedy maximum coverage algorithm on 1,000,000 reverse reachable sets (labelled as “greedy (RR sets)”), `DegreeDiscount`, `DegreeDiscountIC`, and the algorithm that simply pick k vertices with highest degrees (labelled as “maxDegree”). The diffusion model is `ULTM`.

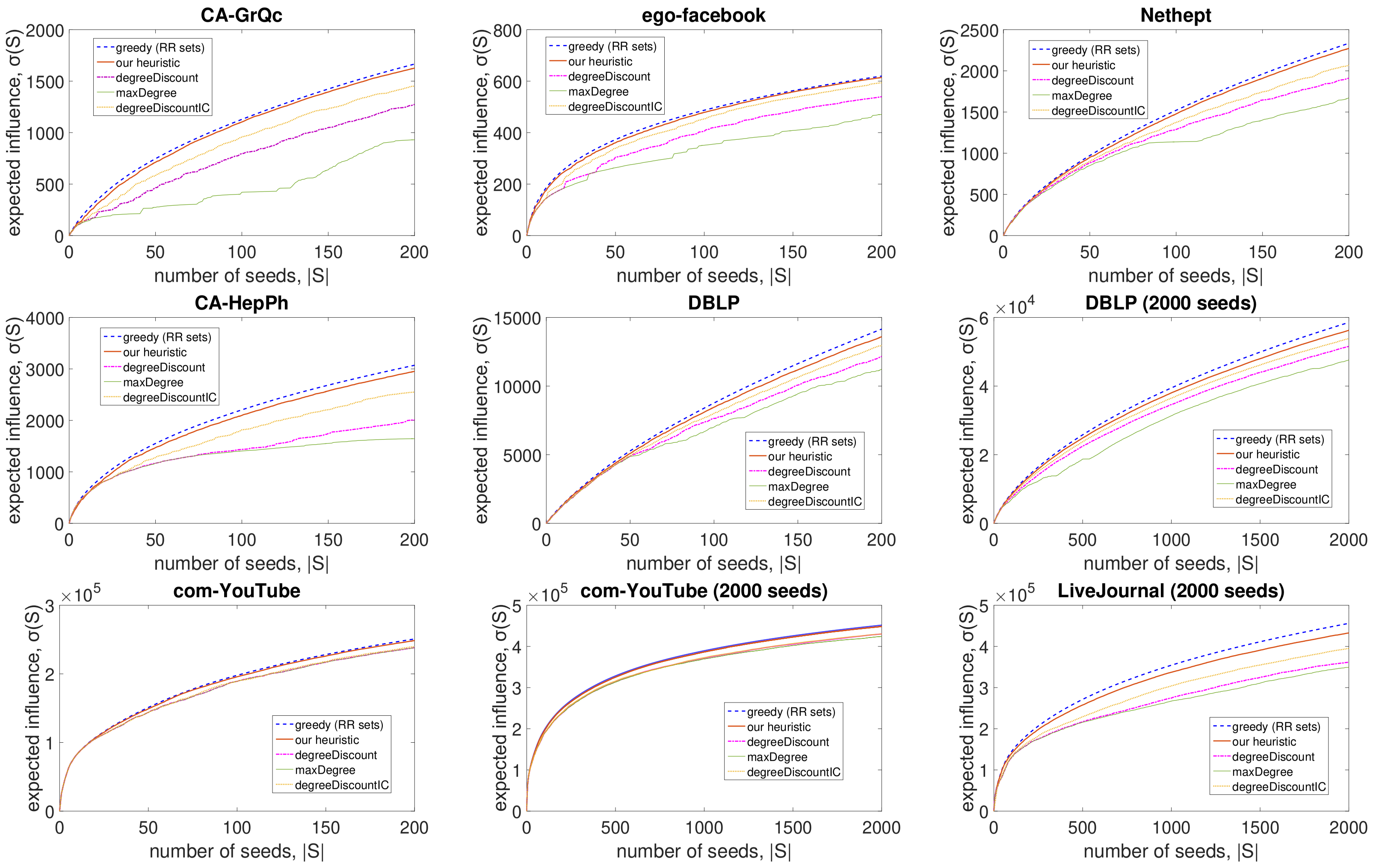


Figure 3.6: Comparing the local greedy heuristic with $f_{vb}^{0.9}$ to the greedy maximum coverage algorithm on 1,000,000 reverse reachable sets (labelled as "greedy (RR sets)"), `DegreeDiscount`, `DegreeDiscountIC`, and the algorithm that simply pick k vertices with highest degrees (labelled as "maxDegree"). The diffusion model is `WICM`.

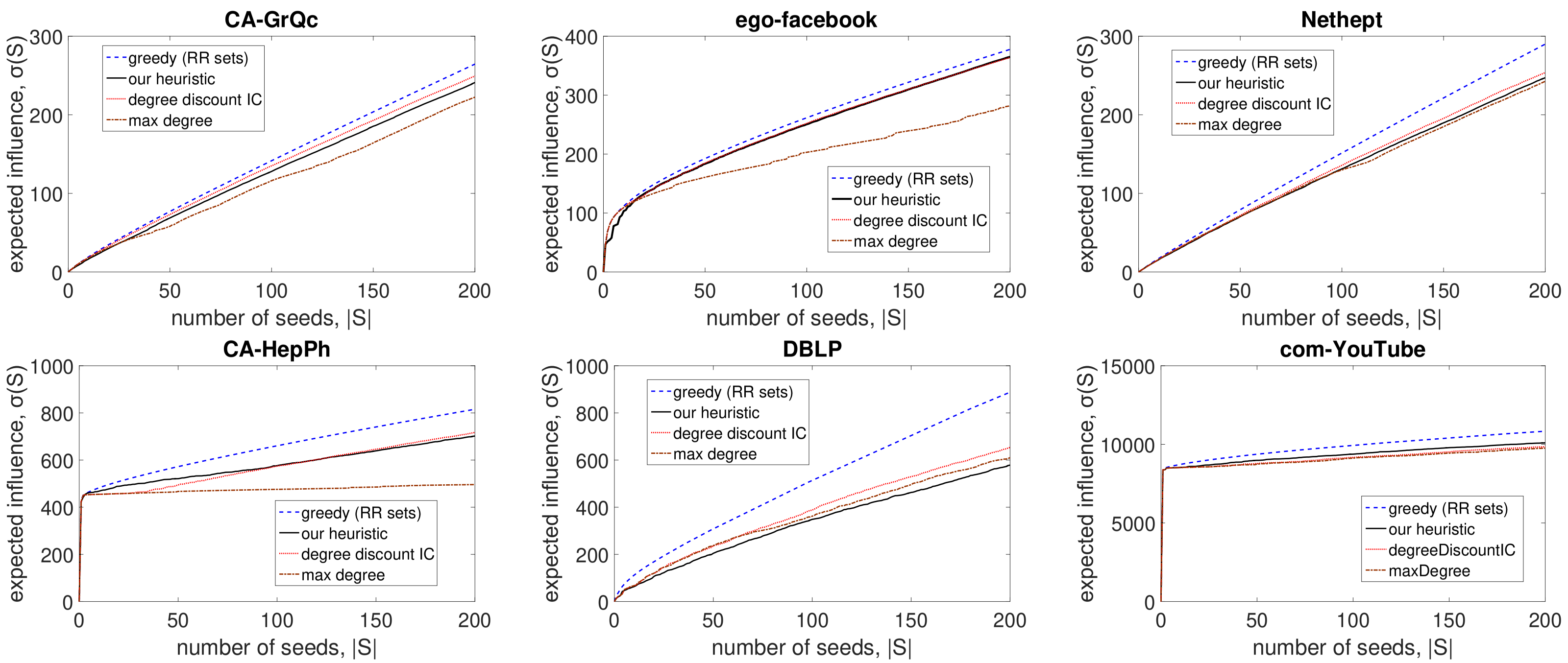


Figure 3.7: Comparing the local greedy heuristic with $f_{vb}^{0.9}$ to the greedy maximum coverage algorithm on 1,000,000 reverse reachable sets (labelled as "greedy (RR sets)"), `DegreeDiscountIC`, and the algorithm that simply pick k vertices with highest degrees (labelled as "maxDegree"). The diffusion model is `UICM` with $p = 0.01$.

interesting future direction is to study the same problem for `LTM` with directed graphs. Notice that the same gap still exists even for this setting.

CHAPTER 4

On Approximation Ratio of Greedy Algorithm

In this chapter, we consider InfMax on undirected graphs under LTM (equivalently, undirected graphs under ULTM by Assumption 2.14). On the one hand, we prove that the greedy algorithm always achieves a $(1-(1-1/k)^k+\Omega(1/k^3))$-approximation, showing that the greedy algorithm does slightly better on undirected graphs than the generic $(1-(1-1/k)^k)$ bound which also applies to directed graphs. On the other hand, we show that substantial improvement on this bound is impossible by presenting an example where the greedy algorithm can obtain at most a $(1-(1-1/k)^k+O(1/k^{0.2}))$ approximation.

This result stands in contrast to the previous work on ICM. Like LTM, the greedy algorithm obtains a $(1-(1-1/k)^k)$-approximation on directed graphs in ICM. However, Khanna and Lucier [47] showed that, in undirected graphs, the greedy algorithm performs substantially better: a $(1-(1-1/k)^k+c)$ approximation for constant $c>0$ (as mentioned multiple times in many chapters of this thesis). Our results show that, surprisingly, no such improvement occurs in LTM.

Finally, we show that, under LTM, the approximation ratio $(1-(1-1/k)^k)$ is tight if 1) the graph is directed or 2) the vertices are weighted. In other words, under either of these two settings, the greedy algorithm cannot achieve $(1-(1-1/k)^k+f(k))$-approximation for any positive function $f(k)$. The result in setting 2) is again in a sharp contrast to Khanna and Lucier's $(1-(1-1/k)^k+c)$-approximation result for ICM, where the $(1-(1-1/k)^k+c)$ approximation guarantee can be extended to the setting where vertices are weighted.

We also discuss some possible generalizations of the model ULTM to the edge-weighted settings (that violates Assumption 2.14, but still reasonable), and whether our results extend to those more generalized settings.

4.1 Introduction

We have remarked in Sect. 2.2, for InfMax, nearly all the known algorithms are based on the greedy algorithm. Therefore, improving the approximation guarantee of the standard greedy algorithm improves the approximation guarantees of most InfMax algorithms in the literature in one shot!

We have seen that both ICM and LTM are submodular, and the greedy algorithm achieves a $(1-(1-1/k)^k)$-approximation, or, a $(1-1/e)$-approximation for any k. A natural and important question is, can we show that the greedy algorithm can perform better than a $(1-(1-1/k)^k)$-approximation through a more careful analysis?

To answer this question, it is helpful to notice that InfMax is a special case of the Max-k-Coverage problem: given a collection of subsets of a set of elements and a positive integer k, find k subsets that cover maximum number of elements (see details in Sect. 4.2.1). For Max-k-Coverage, it is well known that the greedy algorithm cannot overcome the $(1-(1-1/k)^k)$ barrier: for any positive function $f(k)$ which may be infinitesimal, there exists a Max-k-Coverage instance where the greedy algorithm cannot achieve $(1-(1-1/k)^k+f(k))$-approximation. Thus, to hope that the greedy algorithm can overcome this barrier for InfMax, we need to find out what makes InfMax more special and exploit those InfMax features that are not in Max-k-Coverage.

Unfortunately, InfMax with ICM for general directed graphs is nothing more special than Max-k-Coverage, as it can simulate any Max-k-Coverage instance: set the probability that u infects v to be 1 for all edges (u,v) (i.e., a vertex will be infected if it contains an infected in-neighbor); use a vertex to represent a subset in the Max-k-Coverage instance, and use a clique of size m to represent an element; create a directed edge from the vertex representing the subset to an arbitrary vertex in the clique representing the element if this subset contains this element. It is easy to see that this simulates a Max-k-Coverage instance if m is sufficiently large. Therefore, the greedy algorithm cannot achieve a $(1-(1-1/k)^k+f(k))$-approximation for any positive function $f(k)$. This implies we must use properties beyond mere submodularity (a property shared by Max-k-Coverage) to improve the algorithmic analysis.

As mentioned in Sect. 3.2, Khanna and Lucier [47] showed that the $(1-(1-1/k)^k)$ barrier can be overcome if we restrict the graphs to be undirected in ICM. They proved that the greedy algorithm for InfMax with ICM for undirected graphs achieves a $(1-(1-1/k)^k+c)$-approximation for some constant $c>0$ that does not even depend on k.[1] This means greedy produces a $(1-1/e+c)$ algorithm for any k. Moreover, this result holds for the more general setting where 1) there is a prescribed set of vertices $V' \subseteq V$ as a part of input to the InfMax instance such that the seeds can only be chosen among vertices in V' and 2) a positive weight is assigned to each vertex such that the objective is to maximize the total weight of infected vertices (instead of the total number of infected vertices). This result is remarkable, as many of the social networks in our daily life are undirected by their nature (for example, friendship, co-authorship, etc.). Knowing that the $(1-(1-1/k)^k)$ barrier can be overcome for `ICM`, a natural question is, what is the story for `LTM`?

Our results We show that Khanna and Lucier's result on `ICM` can only be partially extended to `LTM`. Our first result is an example showing that the greedy algorithm can obtain at most a $(1-(1-1/k)^k+O(1/k^{0.2}))$-approximation for InfMax on undirected graphs under `LTM`. This shows that, up to lower order terms, the approximation guarantee $1-(1-1/k)^k$ is tight. In particular, no analogue of Khanna and Lucier's $(1-1/e+c)$ result is possible if $c > 0$ is a constant. For the greedy algorithm, we define the *approximation surplus at k* be the additive term after $1-(1-1/k)^k$ in the approximation ratio. Our result can then be equivalently stated as the approximation surplus at k for the linear threshold model is $O(1/k^{0.2})$.

For our second result, we prove that the greedy algorithm does achieve a $(1-(1-1/k)^k+\Omega(1/k^3))$-approximation under the same setting (`LTM` with undirected graphs). This indicates that the greedy algorithm can overcome the $(1-(1-1/k)^k)$ barrier by a lower order term. In particular, the barrier is overcome for constant k. We remark that the approximation surplus $\Omega(1/k^3)$ does not depend on the number of vertices/edges in the graph, so this improvement is not diminishing as the size of the graph grows.

Finally, we extend our results to other InfMax settings. Firstly, we show that the approximation ratio $(1-(1-1/k)^k)$ is tight if we consider general directed graphs. That is, the greedy algorithm cannot achieve a $(1-(1-1/k)^k+f(k))$-approximation for any positive function $f(k)$. Secondly, while still considering undirected graphs, we consider the two generalizations considered by Khanna and Lucier [47]. We show that our result that the greedy algorithm achieves a $(1-(1-1/k)^k+\Omega(1/k^3))$-approximation can be extended to the setting where the seeds can only be picked

[1] Khanna and Lucier [47] only claimed that the greedy algorithm achieves a $(1-1/e+c)$-approximation. However, c being a constant implies that there exists k_0 such that $1-(1-1/k)^k < 1-1/e+c/2$ for all $k \geq k_0$ (notice that $(1-(1-1/k)^k)$ is decreasing and approaches to $1-1/e$); the greedy algorithm will then achieve a $(1-(1-1/k)^k+c/2)$-approximation for $k \geq k_0$.

	Linear Threshold		Independent Cascade	
Approximation Surplus	at least $\Omega(1/k^3))$ at most $O(1/k^{0.2}))$	less than $f(k)$ for any $f(k) > 0$	at least some constant $c > 0$	less than $f(k)$ for any $f(k) > 0$
Directed Graph		✓		✓
Undirected Graph	✓		✓	
Undirected Graph with Weighted Vertices		✓	✓	
Undirected Graph with Prescribed Seed Set	✓		✓	

Table 4.1: Approximation surplus of the greedy algorithm under different settings.

from a prescribed vertex set. However, it cannot be extended to the setting where the vertices are weighted, in which case the approximation ratio of $(1 - (1 - 1/k)^k)$ is tight, as it is in directed graphs. These results, as well as the corresponding result for the independent cascade model by Khanna and Lucier [47], are summarized in Table 4.1.

We have defined LTM for *unweighted*, undirected graphs based on Assumption 2.14. We discuss alternative versions and extensions of LTM to edge-weighted graphs (that violates Assumption 2.14), and discuss how our results extend to these settings.

4.2 Preliminaries

In this chapter, we will use the following equivalent definition for ULTM. Recall that, for undirected graphs with LTM (which are the subjects mainly studied in this chapter), ULTM is automatically assumed (Assumption 2.14).

Definition 4.1. The *(uniform) linear threshold model LT_G* is defined by a directed graph $G = (V, E)$. On input seed set $S \subseteq V$, $LT_G(S)$ outputs a set of infected vertices as follows:

1. Initially, only vertices in S are infected, and for each vertex v a *threshold* $\theta_v \in \mathbb{Z}^+$ is sampled uniformly at random from $\{1, 2, \ldots, \deg(v)\}$ independently. If $\deg(v) = 0$, set $\theta_v = \infty$.

2. In each subsequent iteration, a vertex v becomes infected if v has at least θ_v infected in-neighbors.

3. After an iteration where there are no additional infected vertices, $LT_G(S)$ outputs the set of infected vertices.

As we also remarked, under `ULTM`, each vertex chooses one of its incoming edges being live uniformly at random. We summarize this result again in the theorem below, as this is a crucial observation used in the proofs in this chapter.

Theorem 4.2 (Claim 2.6 in [44])**.** *Let $\widehat{LT}_G(S) \subseteq V$ be the set of vertices that are reachable from S when each vertex v picks exactly one of its incoming edges uniformly at random to be included in the graph and vertices pick their incoming edges independently. Then $\widehat{LT}_G(S)$ and $LT_G(S)$ have the same distribution. Those picked edges are called "live edges".*

Once again, when considering undirected graphs, those live edges in Theorem 4.2 are still directed. Whenever we mention a live edge in the remaining part of this chapter, it should always be clear that this edge is directed.

Remark 4.3. Since each vertex can choose only one incoming edge as being live, *if a vertex v is reachable from a vertex u after sampling all the live edges, then there exists a unique simple path consisting of live edges connecting u to v.*

Remark 4.4. When considering the probability that a given vertex v will be infected by a given seed set S, we can consider a "reverse random walk without repetition" process. The random walk starts at v, and it chooses one of its neighbors (in-neighbors for directed graphs) uniformly at random and moves to it. The random walk terminates when it reaches a vertex that has already been visited or when it reaches a seed. By analogizing each move in the reverse random walk to selecting one incoming live edge, Theorem 4.2 implies that the probability that this random walk reaches a seed is exactly the probability that v will be infected by seeds in S.

Given a set of vertices A and a vertex v, let $A \to v$ be the event that v is reachable from A after sampling live edges. Alternatively, this means that the reverse random walk from v described in Remark 4.4 reaches a vertex in A. If A is the set of seeds, then $\Pr(A \to v)$ is exactly the probability that v will be infected. Intuitively, $A \to v$ can be seen as the event that "A infects v". We set $\Pr(A \to v) = 1$ if $v \in A$. In this chapter, we mean $A \to v$ when we say v *reversely walks to* A or v *is reachable from* A. In particular, the reachability is in terms of the live edges, not the original edges.

Given a set of vertices A, a vertex v, and a set of vertices B such that $B \cap (A \cup \{v\}) = \emptyset$, let $A \xrightarrow{\cancel{B}} v$ be the event that the reverse random walk from v reaches a vertex in A before reaching any vertex in B. By definition, $A \xrightarrow{\cancel{B}} v$ is the same as $A \to v$ if $B = \emptyset$.

Again, let $\sigma(S)$ be the *expected* total number of infected vertices due to the influence of S, $\sigma(S) = \mathbb{E}[|LT_G(S)|]$, where the expectation is taken over the samplings of thresholds of all vertices, or equivalently, over the choices of incoming live edges of all vertices. By the linearity of expectation, we have $\sigma(S) = \sum_{v \in V} \Pr(S \to v)$. In this chapter, we adopt the standard assumption $\sigma(\cdot)$ can be accessed by an oracle.

Remark 4.4 straightforwardly implies the following lemma, which describes a negative correlation between the event that $\{u\}$ infects v and the event that u is infected by another seed set. Some other properties for the linear threshold are presented in Sect. 4.4.2. We decide to introduce Lemma 4.5 in the preliminary section because this negative correlation property is a signature property that makes `LTM` quite different from `ICM`. In `ICM`, knowing the existence of certain connections between vertices only makes it more likely that another pair of vertices are connected. Intuitively, this is because, in `ICM`, each vertex does not "choose" one of its incoming edges, but rather, each incoming edge is included with certain probability independently. In addition, Lemma 4.5 holds for directed graphs, while all the lemmas in Sect. 4.4.2 hold only for undirected graphs.

Lemma 4.5. *For any three sets of vertices A, B_1, B_2 with $A \cap B_1 = A \cap B_2 = \emptyset$ and any two vertices $u, v \notin A \cup B_1 \cup B_2$, we have* $\Pr(A \xrightarrow{\cancel{B_1}} u) \geq \Pr(A \xrightarrow{\cancel{B_1}} u \mid \{u\} \xrightarrow{\cancel{A \cup B_2}} v)$.

Proof. Consider any simple path p from u to v. If $u \xrightarrow{\cancel{A \cup B_2}} v$ happens with all edges in p being live, then $\Pr(A \xrightarrow{\cancel{B_1}} u) \geq \Pr(A \xrightarrow{\cancel{B_1}} u \mid p \text{ is live})$. This is apparent by noticing Remark 4.4: if p is already live, then the reverse random walk starting from u should reach A without touching any vertices on p (if the random walk touches a vertex in p, it will follow the reverse direction of p and eventually go back to u), which obviously happens with less probability compared to the case without restricting that the random walk cannot touch vertices on p.

Noticing this, the remaining part of the proof is trivial:

$$\Pr\left(A \xrightarrow{\cancel{B_1}} u \mid u \xrightarrow{\cancel{A \cup B_2}} v\right) = \sum_p \frac{\Pr(A \xrightarrow{\cancel{B_1}} u \mid p \text{ is live}) \Pr(p \text{ is live})}{\Pr(u \xrightarrow{\cancel{A \cup B_2}} v)}$$

$$\leq \Pr(A \xrightarrow{\cancel{B_1}} u) \sum_p \frac{\Pr(p \text{ is live})}{\Pr(\{u\} \xrightarrow{\cancel{A \cup B_2}} v)} = \Pr\left(A \xrightarrow{\cancel{B_1}} u\right),$$

where the summation is over all simple paths p connecting u to v without touching any vertices in $A \cup B_2$, and Remark 4.3 ensures that the events "p is live" over all possible such p's form a partition of the event $u \xrightarrow{\cancel{A \cup B_2}} v$. □

4.2.1 Influence Maximization Is A Special Case of Max-k-Coverage

In this section, we establish that InfMax is a special case of the well-studied Max-k-Coverage problem, a folklore that is widely known in the InfMax literature. This section also introduces some key intuitions that will be used throughout the chapter. We will only discuss `ULTM` for the purpose of this chapter, although InfMax in general can also be viewed as a special case of Max-k-Coverage.

Definition 4.6. The Max-k-Coverage problem is an optimization problem which takes as input a universe of elements $U = \{e_1, \ldots, e_N\}$, a collection of subsets $\mathcal{M} = \{S_1, \ldots, S_M : S_i \subseteq U\}$ and an positive integer k, and outputs a collection of k subsets that maximizes the total number of covered elements: $\mathcal{S} \in \underset{\mathcal{S}\subseteq\mathcal{M}, |\mathcal{S}|=k}{\operatorname{argmax}} \left| \bigcup_{S \in \mathcal{S}} S \right|$. Given $\mathcal{S} \subseteq \mathcal{M}$, we denote $\operatorname{val}(\mathcal{S}) = \left| \bigcup_{S \in \mathcal{S}} S \right|$.

It is well-known that the greedy algorithm (that iteratively selects a subset that maximizes the marginal increment of $\operatorname{val}(\cdot)$) achieves a $(1-(1-1/k)^k)$-approximation for Max-k-Coverage. On the other hand, this approximation guarantee is tight: for any positive function $f(k) > 0$ which may be infinitesimal, there exists a Max-k-Coverage instance such that the greedy algorithm cannot achieve $(1 - (1 - 1/k)^k + f(k))$-approximation.[2] We will review some properties of Max-k-Coverage in Sect. 4.4.1 that will be used in our analysis for InfMax.

InfMax with `ULTM` can be viewed as a special case of Max-k-Coverage in that an instance of InfMax can be transformed into an instance of Max-k-Coverage. Given an instance of InfMax $(G = (V, E), k)$, let H be the set of all possible live-edge samplings. That is, H is the set of directed graphs on V that are subgraphs of G where each vertex has in-degree equal to 1. In particular, $|H| = \prod_{v \in V} \deg(v)$.[3] We create an instance of Max-k-Coverage by letting the universe of elements be

[2] Our result in Sect. 4.5 says that the greedy algorithm cannot achieve a $(1 - (1 - 1/k)^k + f(k))$-approximation for the linear threshold InfMax with *directed* graphs, which provides a proof of this, since, as we will see soon, InfMax is a special case of Max-k-Coverage.

[3] Of course, vertices with in-degree 0 should be excluded from this product. Whenever we write this product next time, we always refer to the one excluding vertices with in-degree 0.

$V \times H$, i.e., pairs of vertices and live-edge samplings, (v, g), where $v \in V$ and $g \in H$. We then create a subset for each vertex $v \in V$. The subset corresponding to $v \in V$ contains (u, g) if u is reachable from v in g. Since $\sigma(S) = \sum_{v \in V} \Pr(S \to v) = \sum_{v \in V} \frac{|\{g\colon v \text{ is reachable from } S \text{ under } g\}|}{\prod_{w \in V} \deg(w)} = \frac{|\{(v,g)\colon v \text{ is reachable from } S \text{ under } g\}|}{\prod_{w \in V} \deg(w)}$, $\sigma(S)$ equals to the total number of elements covered by "subsets" in S, divided by $\prod_{v \in V} \deg(v)$. As a result, $\sigma(S)$ is proportional to the total number of covered elements if viewing S as a collection of subsets. This establishes that InfMax is a special case of Max-k-Coverage. We denote by $\Sigma(S) = \{(u, g) : u \text{ is reachable from } S \text{ under } g\}$ the set of "elements" that the "subsets" in S cover, and we have $\sigma(S) = |\Sigma(S)| / \prod_{v \in V} \deg(v)$ as discussed above.

Having established the connection between InfMax and Max-k-Coverage, we take a closer look at the intersection, union and difference of two subsets. Let S_1, S_2 be two seed sets. $\Sigma(S_1) \cup \Sigma(S_2)$ contains all those (u, g) such that u is reachable from either S_1 or S_2 under g. Clearly, $\sigma(S_1 \cup S_2) = |\Sigma(S_1 \cup S_2)| / \prod_{v \in V} \deg(v) = |\Sigma(S_1) \cup \Sigma(S_2)| / \prod_{v \in V} \deg(v)$. The first equality holds by definition which holds for set intersection and set difference as well. The last equality, however, does not hold for set intersection and set difference.

$\Sigma(S_1) \cap \Sigma(S_2)$ contains all those (u, g) such that u is reachable from both S_1 and S_2 under g. We have $|\Sigma(S_1) \cap \Sigma(S_2)| / \prod_{v \in V} \deg(v) = \sum_{v \in V} \Pr((S_1 \to v) \wedge (S_2 \to v))$. For the special case where $S_1 = \{u_1\}$ and $S_2 = \{u_2\}$, by Remark 4.3, the event $(S_1 \to v) \wedge (S_2 \to v)$ can be partitioned into two disjoint events: 1) v reaches u_2 before u_1 in the reverse random walk, $(\{u_1\} \xrightarrow{\cancel{\{v\}}} u_2) \wedge (\{u_2\} \xrightarrow{\cancel{\{u_1\}}} v)$, and 2) v reaches u_1 before u_2 in the reverse random walk, $(\{u_2\} \xrightarrow{\cancel{\{v\}}} u_1) \wedge (\{u_1\} \xrightarrow{\cancel{\{u_2\}}} v)$. For general S_1, S_2 with $S_1 \cap S_2 = \emptyset$, the event $(S_1 \to v) \wedge (S_2 \to v)$ can be partition into two disjoint events with respect to which of S_1, S_2 that v reversely reaches first.

Similarly, $\Sigma(S_1) \setminus \Sigma(S_2)$ contains all those (u, g) such that u is reachable from S_1 but not from S_2 under g, we have $|\Sigma(S_1) \setminus \Sigma(S_2)| / \prod_{v \in V} \deg(v) = \sum_{v \in V} \Pr((S_1 \to v) \wedge \neg(S_2 \to v))$.

4.3 Upper Bound on Approximation Guarantee

In this section, we show that the approximation guarantee for the greedy algorithm on InfMax is at most $(1 - (1 - 1/k)^k + O(1/k^{0.2}))$ with the linear threshold model on undirected graphs. In other words, the approximation surplus is $O(1/k^{0.2})$. This shows that the approximation guarantee $(1 - 1/e)$ cannot be asymptotically improved, even if undirected graphs are considered.

Before we prove our main theorem in this section, we need the following lemma characterizing the cascade of a single seed on a complete graph which is interesting on its own.

Lemma 4.7. *Let G be a complete graph with n vertices and S be a set containing a single vertex. We have $\sigma(S) < 3\sqrt{n}$ for `ULTM`.*

The proof of Lemma 4.7 is in Appendix A.1. The intuition behind this lemma is simply the birthday paradox. Consider the reverse random walk starting from any particular vertex v. At each step, instead of choosing one of the remaining $n-1$ vertices uniformly at random, we add a self-loop and assume that the random walk chooses one of the n vertices in the graph uniformly at random. The effect of this change can be ignored if n is large. Then, by the birthday paradox, with a high probability, it takes approximately $\sqrt{n}$ steps for a reverse random walk to visit a vertex that has been visited before. The probability that a particular vertex v is infected is then the probability that the random walk reaches the seed before reaching a visited vertex, which is approximately $1-(1-1/n)^{\sqrt{n}} \approx 1/\sqrt{n}$. Finally, by the linearity of expectation, the total number of infected vertices is about $\sqrt{n}$.

The remaining part of this section proves the following theorem.

Theorem 4.8. *Consider* InfMax *on `ULTM` with undirected graphs. There exists an instance where the greedy algorithm only achieves a $(1-(1-1/k)^k+O(1/k^{0.2}))$-approximation.*

The InfMax instance mentioned in Theorem 4.8 is shown below.

Example 4.9. The example is illustrated in Fig. 4.1. Given the number of seeds k, we construct the undirected graph $G=(V,E)$ with $k\lceil k^{1.2}\rceil + \lfloor (1-\frac{100}{k^{0.2}})k^{1.8}\rfloor$ vertices as follows. Firstly, construct k cliques $C_1,\ldots,C_k$ of size $\lceil k^{1.2}\rceil$, and in each clique C_i label an arbitrary vertex u_i . Secondly, construct k vertices $v_1,\ldots,v_k$. For each $i=1,\ldots,k$, create $\lceil k^{0.8}(1-1/k)^{i-1}\rceil - 1$ vertices and connect them to v_i. For each i, those $\lceil k^{0.8}(1-1/k)^{i-1}\rceil - 1$ vertices combined with v_i form a star of size $\lceil k^{0.8}(1-1/k)^{i-1}\rceil$, and we will use D_i to denote the i-th star. Thirdly, we continue creating ℓ of these kinds of stars $D_{k+1},\ldots,D_{k+\ell}$ centered at $v_{k+1},\ldots,v_{k+\ell}$ such that $|D_{k+1}|=\cdots=|D_{k+\ell-1}|=\lceil k^{0.8}(1-1/k)^k\rceil, |D_{k+\ell}|\leq\lceil k^{0.8}(1-1/k)^k\rceil$, and $\sum_{i=1}^{k+\ell}|D_i| = \lfloor(1-\frac{100}{k^{0.2}})k^{1.8}\rfloor$. In other words, we keep creating stars of the same size $\lceil k^{0.8}(1-1/k)^k\rceil$ until we reach the point where the total number of vertices in all those stars is $\lfloor(1-\frac{100}{k^{0.2}})k^{1.8}\rfloor$, where the last star created may be "partial" and have a size smaller than $\lceil k^{0.8}(1-1/k)^k\rceil$. Notice that $|D_1|\geq|D_2|\geq\cdots\geq|D_k|\geq$

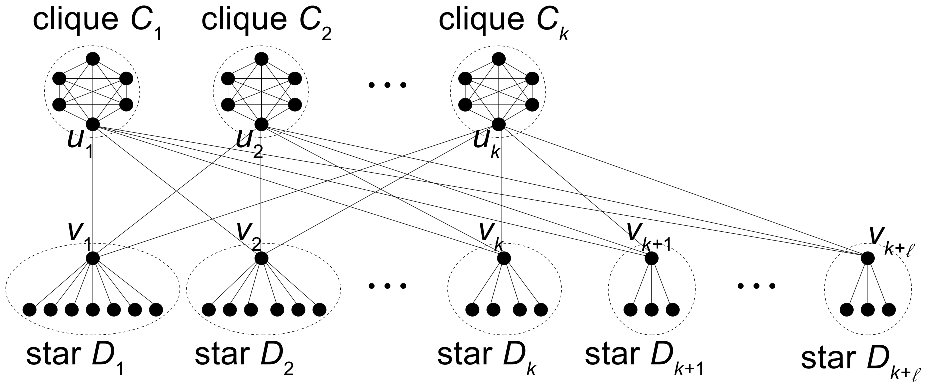


Figure 4.1: The tight example.

$|D_{k+1}| = \cdots = |D_{k+\ell-1}| \geq |D_{k+\ell}| = \Theta(k^{0.8})$.[4] Finally, create $k \times (k+\ell)$ edges $\{(u_i, v_j) : i = 1, \ldots, k; j = 1, \ldots, k+\ell\}$.

Proof Sketch of Theorem 4.8 We want that the greedy algorithm picks the seeds $v_1, \ldots, v_k$, while the optimal seeds are $u_1, \ldots, u_k$. The purpose of constructing a clique C_i for each u_i is to simulate directed edges (u_i, v_j) (such that, as mentioned earlier, each u_i will be infected with $o(1)$ probability even if all of $v_1, \ldots, v_{k+\ell}$ are infected, and the total number of infections among the cliques is negligible so that the "gadget" itself is not "heavy"). In the optimal seeding strategy, each v_i will be infected with probability $1 - o(1)$, as the number of edges connecting to the seeds $u_1, \ldots, u_k$ is k, which is significantly more than the number of edges inside D_i (which is at most $\lceil k^{0.8} \rceil$). Therefore, $\sigma(\{u_1, \ldots, u_k\}) \approx \sum_{i=1}^{k+\ell} |D_i| = \lfloor (1 - \frac{100}{k^{0.2}}) k^{1.8} \rfloor$, which is slightly less than $k^{1.8}$. Moreover, each $\sigma(\{u_i\})$ is approximately $\frac{1}{k}$ of $\sigma(\{u_1, \ldots, u_k\})$, which is slightly less than $k^{0.8}$

The greedy algorithm would pick v_1 as the first seed, as $\sigma(v_1)$ is at least $\lceil k^{0.8} \rceil$ (by only accounting for the infected vertices in D_1) which is slightly larger than each $\sigma(\{u_i\})$. After picking v_1 as the first seed, the marginal increment of $\sigma(\cdot)$ by choosing each of $u_1, \ldots, u_k$ becomes approximately $\frac{1}{k} \sum_{i=2}^{k+\ell} |D_i| = \frac{1}{k}(-|D_1| + \sum_{i=1}^{k+\ell} |D_i|)$, which is slightly less than $\frac{1}{k}(-\lceil k^{0.8} \rceil + k^{1.8}) \approx |D_2|$. On the other hand, noticing that v_1 infects each of $u_1, \ldots, u_k$ as well as v_2 with probability $o(1)$, the marginal increment of $\sigma(\cdot)$ by choosing v_2 is approximately $|D_2|$, which is slightly larger than the marginal increment by choosing any u_i based on our calculation above. Thus, the greedy algorithm will continue to pick v_2. In general, we have designed the sizes of

[4] These inequalities may not be strict. In fact, $|D_1|$ may be equal to $|D_2|$ as $k^{0.8} - k^{0.8}(1 - 1/k) = 1/k^{0.2} < 1$.

$D_1, D_2, \ldots, D_k$ such that they are just large enough to make sure the greedy algorithm will pick $v_1, v_2, \ldots, v_k$ one by one.

Our construction of cliques $C_1, \ldots, C_k$ makes sure that each of $u_1, \ldots, u_k$ will be infected with $o(1)$ probability even if all of $v_1, \ldots, v_k$ are seeded. Therefore, $\sigma(\{v_1, \ldots, v_k\}) \approx \sum_{i=1}^{k} |D_i| = \sum_{i=1}^{k} \lceil k^{0.8}(1-1/k)^{i-1} \rceil \leq k + \sum_{i=1}^{k} k^{0.8}(1-1/k)^{i-1} = k + k^{1.8}(1-(1-1/k)^k)$. On the other hand, we have seen that $\sigma(\{u_1, \ldots, u_k\})$ is just slightly less than $k^{1.8}$. To be more accurate, $\sigma(\{u_1, \ldots, u_k\}) \approx (1 - \frac{100}{k^{0.2}})k^{1.8}$. Dividing $\sigma(\{v_1, \ldots, v_k\})$ by $\sigma(\{u_1, \ldots, u_k\})$ gives us the desired upper bound on the approximation ratio in Theorem 4.8. The numbers $0.2, 0.8, 1.2$ on the exponent of k are optimized for getting the tightest bound while ensuring that the greedy algorithm still picks $v_1, \ldots, v_k$.

A full proof of Theorem 4.8 is available in Appendix A.2.

4.4 Lower Bound on Approximation Guarantee

In this section, we prove that the greedy algorithm can obtain at least a $(1-(1-1/k)^k + \Omega(1/k^3))$-approximation to $\max_{S \subseteq V:|S|=k} \sigma(S)$, stated in Theorem 4.10. This indicates that the barrier $1-(1-1/k)^k$ can be overcome if k is a constant. We have seen that InfMax is a special case of Max-k-Coverage in Sect. 4.2.1, and it is known that the greedy algorithm cannot overcome the barrier $1-(1-1/k)^k$ in Max-k-Coverage. Theorem 4.10 shows that InfMax with `ULTM` on undirected graphs has additional structure. To prove Theorem 4.10, we first review in Sect. 4.4.1 some properties of Max-k-Coverage that are useful to our analysis, and then we prove Theorem 4.10 in Sect. 4.4.2 by exploiting some special properties of InfMax that are not satisfied in Max-k-Coverage.

Theorem 4.10. *Consider* InfMax *on undirected graphs with* `ULTM`*. The greedy algorithm achieves a* $(1-(1-1/k)^k + \Omega(1/k^3))$*-approximation.*

4.4.1 Some Properties of Max-k-Coverage

In this section, we list some of the properties of Max-k-Coverage which will be used in proving Theorem 4.10. The proofs of the lemmas in this section are all standard, and are deferred to the appendix. For all the lemmas in this section, we are considering a Max-k-Coverage instance $(U, \mathcal{M}, k)$, where $\mathcal{S} = \{S_1, \ldots, S_k\}$ denotes the k subsets output by the greedy algorithm and $\mathcal{S}^* = \{S_1^*, \ldots, S_k^*\}$ denotes the optimal solution.

Lemma 4.11. *If $S_1 \in \mathcal{S}^*$, then* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - \frac{1}{k})^k + \frac{1}{4k^2}) \operatorname{val}(\mathcal{S}^*)$.

Lemma 4.12. *If* $\frac{|S_1 \cap (\bigcup_{i=1}^k S_i^*)|}{\operatorname{val}(\mathcal{S}^*)} \notin [\frac{1}{k} - \varepsilon, \frac{1}{k} + \varepsilon]$ *for some $\varepsilon > 0$ which may depend on k, then* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - 1/k)^k + \varepsilon/4) \operatorname{val}(\mathcal{S}^*)$.

Lemma 4.13. *If* $\sum_{i=1}^k |S_i^*| > (1 + \varepsilon) \operatorname{val}(\mathcal{S}^*)$ *for some $\varepsilon > 0$ which may depend on k, then* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - \frac{1}{k})^k + \frac{\varepsilon}{8k}) \operatorname{val}(\mathcal{S}^*)$.

Lemma 4.14. *If* $|S_1 \setminus (\bigcup_{i=1}^k S_i^*)| > \varepsilon \operatorname{val}(\mathcal{S}^*)$ *for some $\varepsilon > 0$ which may depend on k, then* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - 1/k)^k + \varepsilon/16) \operatorname{val}(\mathcal{S}^*)$.

Lemma 4.15. *If there exists $S_i^* \in \mathcal{S}^*$ such that* $|S_i^*| < (\frac{1}{k} - \varepsilon) \operatorname{val}(\mathcal{S}^*)$ *for some $\varepsilon > 0$ which may depend on k, then* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - \frac{1}{k})^k + \frac{\varepsilon}{8k}) \operatorname{val}(\mathcal{S}^*)$.

4.4.2 Proof of Theorem 4.10

We begin by proving some properties that are exclusively for InfMax.

Lemma 4.16. *Given a subset of vertices $A \subseteq V$, a vertex $v \notin A$ and a neighbor $u \in \Gamma(v)$ of v, with probability at most $\frac{|A|}{|A|+1}$, there is a simple live path from a vertex in A to vertex v such that the last vertex in the path before reaching v is not u.*

Proof. We consider all possible reverse random walks starting from v, and define a mapping from those walks that eventually reach A to those that do not. For each reverse random walk that reaches a vertex $a \in A$, $v \leftarrow w_1 \leftarrow \cdots \leftarrow w_{\ell-1} \leftarrow w_\ell \leftarrow a$ (with $w_1, \ldots, w_\ell \notin A$), we map it to the random walk $v \leftarrow w_1 \leftarrow \cdots \leftarrow w_{\ell-1} \leftarrow w_\ell \leftarrow w_{\ell-1}$, i.e., the one with the last step moving back. Notice that the latter reverse random walk visits $w_{\ell-1}$ more than once, and thus will not reach A. Specifically, for those reverse random walks that reach A in one single step $v \leftarrow a$ (in the case v is adjacent to $a \in A$), we map it to the reverse random walk $v \leftarrow u$, which are excluded from the event that "there is a simple live path from a vertex in A to vertex v such that the last vertex in the path before reaching v is not u" (if $v \leftarrow u$, then every path that reaches v should then reach u in the penultimate step).

It is easy to see that at most $|A|$ different reverse random walks that reach A can be mapped to a same random walk that does not reach A. In order to make different reverse random walks have the same image in the mapping, they must share the same path $v \leftarrow w_1 \leftarrow \cdots \leftarrow w_\ell$ except for the last step. The last step, which moves to a vertex in A, can only have $|A|$ different choices. For the special reverse random walks that move to A in one step, there are at most $|A|$ of them, which are mapped to the random walk $v \leftarrow u$.

It is also easy to see that each random walk happens with the same probability as its image does. This is because w_ℓ chooses its incoming edges uniformly, so choosing a happens with the same chance as choosing w_ℓ. Specifically, v chooses its incoming edge (a, v) with the same probability as (u, v).

Since we have defined a mapping that maps at most $|A|$ disjoint sub-events in the positive case to a sub-event in the negative case with the same probability, the lemma follows. □

Lemma 4.17. *Given a subset of vertices $A \subseteq V$ and two different vertices $u, v \notin A$, we have* $\Pr(A \to u \mid \{u\} \overset{\cancel{A}}{\to} v) \leq \frac{|A|}{|A|+1}$.

Proof. Let $w_1, \ldots, w_t$ enumerate all the neighbors of u that are not in A. For each $i = 1, \ldots, t$, let E_i be the event that the reverse random walk starting from v reaches u without touching A and its last step before reaching u is at w_i. Clearly, $\{E_1, \ldots, E_t\}$ is a partition of $\{u\} \overset{\cancel{A}}{\to} v$. Conditioning on the event E_i, if $A \to u$ happens, the reverse random walk from u to A cannot touch w_i, since w_i has already chosen its incoming edge (u, w_i) in the case E_i happens. Therefore, by Lemma 4.5 and Lemma 4.16, $\Pr(A \to u \mid E_i) = \Pr(A \xrightarrow{\cancel{\{w_i\}}} u \mid E_i) \leq \Pr(A \xrightarrow{\cancel{\{w_i\}}} u) \leq \frac{|A|}{|A|+1}$.[5] We have

$$\Pr(A \to u \mid \{u\} \overset{\cancel{A}}{\to} v) = \frac{\sum_{i=1}^{t} \Pr(A \to u \mid E_i) \Pr(E_i)}{\Pr(\{u\} \overset{\cancel{A}}{\to} v)}$$

$$\leq \frac{|A|}{|A|+1} \frac{\sum_{i=1}^{t} \Pr(E_i)}{\Pr(\{u\} \overset{\cancel{A}}{\to} v)} = \frac{|A|}{|A|+1},$$

which concludes this lemma. □

Finally, we need the following lemma, which is a special case of Corollary 3.15.

Lemma 4.18. *For any $v \in V$, we have* $\sigma(\{v\}) \leq \deg(v) + 1$.

[5] Rigorously speaking, the statement of Lemma 4.5 does not directly imply $\Pr(A \xrightarrow{\cancel{\{w_i\}}} u \mid E_i) \leq \Pr(A \xrightarrow{\cancel{\{w_i\}}} u)$. However, the proof of Lemma 4.5 can be adapted to show this. Instead of summing over all simple paths p from u to v in the summation of the last inequality in the proof, we sum over all simple paths from u to v *such that u first moves to w_i*. The remaining part of the proof is the same. The idea here is that, the event v reversely walks to u is negatively correlated to the event that u reversely walks to A, as the latter walk cannot hit the vertices on the path $u \to v$ if there is already a path from u to v.

A proof of a more generalized version of the lemma above, which extends this lemma to the linear threshold model *with slackness* (see Append. A.4 for definition of this model), is included in Appendix A.5 for completeness. The proof is mostly identical to the arguments in Sect. 3.5.2.

Now we are ready to show Theorem 4.10. In the remaining part of this section, we use $S = \{v_1, \ldots, v_k\}$ and $S^* = \{u_1, \ldots, u_k\}$ to denote the seed sets output by the greedy algorithm and the optimal seed set respectively. Recall that we have established that InfMax is a special case of Max-k-Coverage in Sect. 4.2.1, and $v_1, \ldots, v_k, u_1, \ldots, u_k$ can be viewed as subsets in Max-k-Coverage. Thus, the lemmas in Sect. 4.4.1 can be applied here.

First of all, if $v_1 \in S^*$, Lemma 4.11 implies Theorem 4.10 already. In particular, Lemma 4.11 implies that $|\Sigma(S)| \geq (1-(1-1/k)^k+1/4k^2)|\Sigma(S^*)|$ (refer to Sect. 4.2.1 for the definition of $\Sigma(\cdot)$), which implies $\sigma(S) \geq (1-(1-1/k)^k + 1/4k^2)\sigma(S^*)$ by dividing $\prod_{w \in V} \deg(w)$ on both side of the inequality. Therefore, we assume $v_1 \notin S^*$ from now on.

Next, we analyze the intersection between $\Sigma(\{v_1\})$ and $\Sigma(S^*)$. As an overview of the remaining part of our proof, suppose the barrier $1-(1-1/k)^k$ cannot be overcome, Lemma 4.13 and Lemma 4.15 imply that $\Sigma(\{u_1\}), \ldots, \Sigma(\{u_k\})$ must be almost disjoint and almost balanced, Lemma 4.12 implies that $\Sigma(\{v_1\})$ must intersect approximately $1/k$ fraction of $\Sigma(S^*)$, and Lemma 4.14 implies that $\Sigma(\{v_1\}) \setminus \Sigma(S^*)$ should not be large. We will prove that these conditions cannot be satisfied at the same time.

The intersection $\Sigma(\{v_1\}) \cap \Sigma(S^*)$ consists of all the tuples (w, g) such that w is reachable from both v_1 and S^* under the live-edge realization g. Consider the reverse random walk starting from w. There are three different disjoint cases: 1) w reaches v_1 first, and then reaches a vertex in S^*; 2) w reaches a vertex in S^*, and then reaches v_1; 3) w visits more than one vertex in S^*, and then reaches v_1. The three terms in the following equation, which are named C_1, C_2, C_3, correspond to these three cases respectively.

$$\begin{aligned}\frac{|\Sigma(\{v_1\}) \cap \Sigma(S^*)|}{\prod_{w \in V} \deg(w)} &= \sum_{w \in V} \Pr\left((S^* \to v_1) \wedge \left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right)\right) && (C_1)\\ &+ \sum_{w \in V} \sum_{i=1}^{k} \Pr\left(\left(\{v_1\} \xrightarrow{\cancel{S^*}} u_i\right) \wedge \left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right)\right) && (C_2)\\ &+ \sum_{w \in V} \sum_{i \neq j} \Pr\left((\{v_1\} \to u_j) \wedge \left(\{u_j\} \xrightarrow{\cancel{S^*}} u_i\right) \wedge \left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right)\right)\\ & && (C_3)\end{aligned}$$

Notice that this decomposition assumes $v_1 \notin S^*$.

Firstly, we show that C_1 cannot be too large if the barrier $1 - (1 - 1/k)^k$ is not overcome. Intuitively, C_1 describes those w that first reversely reaches v_1 and then reversely reaches a vertex in S^*. Lemma 4.17 tells us that v_1 will reversely reach S^* with at most probability $k/(k+1)$ conditioning on w reversely reaching v_1. This implies that, if w reversely reaches v_1, v_1 will not reversely reach S^* with probability at least $1/(k+1)$, which is at least $1/k$ of the probability that v_1 reversely reaches S^*. Therefore, whenever we have a certain number of elements in $\Sigma(\{v_1\}) \cap \Sigma(S^*)$ that corresponds to C_1, we have at least $1/k$ fraction of this number in $\Sigma(\{v_1\}) \setminus \Sigma(S^*)$. Lemma 4.14 implies that the $1-(1-1/k)^k$ barrier can be overcome if $|\Sigma(\{v_1\}) \setminus \Sigma(S^*)|$ is large.

Proposition 4.19. *If $C_1 > \frac{9}{10k} \cdot \sigma(S^*)$, then $\sigma(S) \geq (1 - (1 - \frac{1}{k})^k + \frac{1}{640k^2}) \cdot \sigma(S^*)$.*

Proof. If $w = v_1$, $\{v_1\} \xrightarrow{\cancel{S^*}} w$ happens automatically, and $\Pr((\{v_1\} \xrightarrow{\cancel{S^*}} w) \wedge (S^* \to v_1)) = \Pr(S^* \to v_1)$. Substituting this into C_1, we have

$$\begin{aligned} C_1 &= \Pr(S^* \to v_1) + \sum_{w \in V \setminus \{v_1\}} \Pr\left((S^* \to v_1) \wedge \left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right)\right)\\ &\leq 1 + \sum_{w \in V \setminus \{v_1\}} \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right) \cdot \Pr\left(S^* \to v_1 \mid \{v_1\} \xrightarrow{\cancel{S^*}} w\right)\\ &\leq 1 + \sum_{w \in V \setminus \{v_1\}} \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right) \cdot k \Pr\left(\neg(S^* \to v_1) \mid \{v_1\} \xrightarrow{\cancel{S^*}} w\right)\\ & && \text{(Lemma 4.17)}\\ &= 1 + k \sum_{w \in V \setminus \{v_1\}} \Pr\left(\left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right) \wedge \neg\left(S^* \to v_1\right)\right), \end{aligned}$$

where the penultimate step is due to Lemma 4.17 from which we have $\Pr(S^* \to v_1 \mid$

$\{v_1\} \xrightarrow{\cancel{S^*}} w) \leq \frac{k}{k+1}$, which implies $\Pr(\neg(S^* \to v_1) \mid \{v_1\} \xrightarrow{\cancel{S^*}} w) \geq \frac{1}{k+1}$, which further implies $\Pr(S^* \to v_1 \mid \{v_1\} \xrightarrow{\cancel{S^*}} w) \leq k \cdot \Pr(\neg(S^* \to v_1) \mid \{v_1\} \xrightarrow{\cancel{S^*}} w)$.

Notice that the summation $\sum_{w \in V \setminus \{v_1\}} \Pr((\{v_1\} \xrightarrow{\cancel{S^*}} w) \wedge \neg(S^* \to v_1))$ describes those (w, g) such that w is reachable from v_1 but not S^* under realization g, which corresponds to elements in $\Sigma(\{v_1\}) \setminus \Sigma(S^*)$. Therefore, we have

$$\frac{|\Sigma(\{v_1\}) \setminus \Sigma(S^*)|}{\prod_{w \in V} \deg(w)} \geq \sum_{w \in V \setminus \{v_1\}} \Pr\left(\left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right) \wedge \neg\left(S^* \to v_1\right)\right) \geq \frac{C_1 - 1}{k}.$$

If $\sigma(S^*) \leq \frac{8}{7}k$, we can see that $\sigma(S) \geq k \geq \frac{7}{8}\sigma(S^*) > (1 - (1 - \frac{1}{k})^k + \frac{1}{640k^2})\sigma(S^*)$ and the proposition is already implied. Thus, we assume $\sigma(S^*) > \frac{8}{7}k$ from now on.

If we have $C_1 > \frac{9}{10k}\sigma(S^*)$ as given in the proposition statement, we have $C_1 - 1 > \frac{9}{10k}\sigma(S^*) - \frac{7}{8k}\sigma(S^*) = \frac{1}{40k}\sigma(S^*) = \frac{1}{40k}\frac{|\Sigma(S^*)|}{\prod_{w \in V} \deg(w)}$. Putting together,

$$\frac{|\Sigma(\{v_1\}) \setminus \Sigma(S^*)|}{\prod_{w \in V} \deg(w)} \geq \frac{C_1 - 1}{k} > \frac{1}{40k^2}\frac{|\Sigma(S^*)|}{\prod_{w \in V} \deg(w)},$$

which yields $|\Sigma(\{v_1\}) \setminus \Sigma(S^*)| > \frac{1}{40k^2}|\Sigma(S^*)|$. Lemma 4.14 implies $|\Sigma(S)| \geq (1 - (1 - \frac{1}{k})^k + \frac{1}{640k^2})|\Sigma(S^*)|$, which further implies this proposition. □

Secondly, we show that C_2 cannot be too large if the barrier $1 - (1 - 1/k)^k$ is not overcome. To show this, we first show that there exists $u_i \in S^*$ such that $\Pr(\{v_1\} \to u_i) \geq \frac{C_2}{\sigma(S^*)}$, and then show that this implies that $|\Sigma(\{v_1\}) \setminus \Sigma(S^*)|$ is large by accounting for v_1's influence to u_i's neighbors.

Proposition 4.20. *If* $C_2 > \frac{1}{100k} \cdot \sigma(S^*)$, *then* $\sigma(S) \geq (1 - (1 - \frac{1}{k})^k + \frac{1}{64000k^3})\sigma(S^*)$.

Proof. We give an outline of the proof first. Assume $u_1 \in \operatorname*{argmax}_{u_i \in S^*} \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} u_i\right)$ without loss of generality. The proof is split into two steps.

- Step 1: We will show that $\sum_{w \in \Gamma(u_1) \setminus S^*} \Pr(\{v_1\} \xrightarrow{\cancel{S^*}} w) = \Omega\left(\frac{1}{k^2}\right)\sigma(S^*)$ if we have $C_2 > \frac{1}{100k} \cdot \sigma(S^*)$ in the proposition statement. Notice that the summation consists of the neighbors of u_1 (that are not in S^*) that reversely reaches v_1, which is a lower bound to $\sigma(v_1)$ (v_1 may infect much more vertices than only the neighbors of u_1). To show this, we first find an upper bound of C_2 in terms of this summation: $\frac{C_2}{\sigma(S^*)} \leq \frac{1}{\deg(u_1)} \sum_{w \in \Gamma(u_1) \setminus S^*} \Pr(\{v_1\} \xrightarrow{\cancel{S^*}} w)$. This will imply that $\sum_{w \in \Gamma(u_1) \setminus S^*} \Pr(\{v_1\} \xrightarrow{\cancel{S^*}} w) = \Omega\left(\frac{1}{k^2}\right)\sigma(S^*)$ if assuming $C_2 > \frac{1}{100k} \cdot \sigma(S^*)$, because $\deg(u_1)$ is (approximately) an upper bound to $\sigma(\{u_1\})$ by Lemma 4.18, and $\sigma(\{u_1\})$ is approximately $\frac{1}{k}\sigma(S^*)$ (otherwise, the proposition holds directed by Lemma 4.15).

- Step 2: We will show that $\Pr(\neg(S^* \to v_1) \mid \{v_1\} \xrightarrow{\cancel{S^*}} w) \geq \frac{1}{2(k+1)}$ for each $w \in \Gamma(u_1)\backslash S^*$. This says that, for each of u_1's neighbor w, if it reversely reaches v_1, it will not reach S^* with a reasonably high probability. Correspondingly, a reasonably large fraction of $\Sigma(\{v_1\})$ will not be in $\Sigma(S^*)$. By Lemma 4.14, this proposition is concluded.

Step 1 Firstly, based on the first vertex in S^* that w reversely reaches, we can decompose $\sigma(S^*)$ as follows:

$$\sigma(S^*) = \sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right).$$

Next, we have

$$\begin{aligned}
\frac{C_2}{\sigma(S^*)} &= \frac{\sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right) \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} u_i \mid \{u_i\} \xrightarrow{\cancel{S^*}} w\right)}{\sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right)} \\
&\leq \frac{\sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right) \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} u_i\right)}{\sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right)} \qquad \text{(Lemma 4.5)} \\
&\leq \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} u_1\right) \cdot \frac{\sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right)}{\sum_{w\in V}\sum_{i=1}^{k} \Pr\left(\{u_i\} \xrightarrow{\cancel{S^*}} w\right)} \\
&= \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} u_1\right) \\
&= \frac{1}{\deg(u_1)} \sum_{w\in\Gamma(u_1)\backslash S^*} \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} w\right).
\end{aligned}$$

For the last step, v_1 needs to first connect to one of u_1's neighbors before connecting to u_1. Notice that these neighbors may include v_1 itself. In this special case $w = v_1 \in \Gamma(u_1) \setminus S^*$, we have $\Pr(\{v_1\} \xrightarrow{\cancel{S^*}} w) = 1$ and u_1 chooses its incoming live edge to be (v_1, u_1) with probability $\frac{1}{\deg(u_1)}$, which is also a valid term in the summation above.

If $C_2 > \frac{1}{100k} \cdot \sigma(S^*)$ as suggested by the proposition statement, we have

$$\sum_{w \in \Gamma(u_1) \setminus S^*} \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} w \right) \geq \frac{\deg(u_1) C_2}{\sigma(S^*)} > \frac{\deg(u_1)}{100k}$$

$$\geq \frac{\deg(u_1) + 1}{200k} \geq \frac{\sigma(\{u_1\})}{200k} \geq \frac{9\sigma(S^*)}{2000k^2},$$

where the penultimate step is due to Lemma 4.18 and the last step is based on the assumption $\sigma(\{u_1\}) \geq \frac{9}{10k}\sigma(S^*)$. Notice that we can assume this without loss of generality, as otherwise Lemma 4.15 implies that $|\Sigma(S)| \geq (1-(1-\frac{1}{k})^k + \frac{1}{80k^2})|\Sigma(S^*)|$, which directly implies this proposition.

Step 2 If $w \neq v_1$, Lemma 4.17 implies that $\Pr(\neg(S^* \to v_1) \mid \{v_1\} \xrightarrow{\cancel{S^*}} w) \geq \frac{1}{k+1} > \frac{1}{2(k+1)}$. If $w = v_1$, then u_1 and v_1 are adjacent. Notice that $\deg(v_1) \geq 2$, for otherwise $\sigma(\{u_1\}) > \sigma(\{v_1\})$ so v_1 cannot be the first seed picked by the greedy algorithm. Therefore, v_1 reversely reaches u_1 in one step with probability at most $\frac{1}{2}$. If v_1 reversely reaches a vertex in S^* such that the first step of the reverse random walk is not towards u_1, Lemma 4.16 implies that the probability this happens is at most $\frac{k}{k+1}$. Putting together, for $w = v_1$, $\Pr(S^* \to v_1 \mid \{v_1\} \xrightarrow{\cancel{S^*}} w) \leq \frac{1}{2} + \frac{1}{2} \cdot \frac{k}{k+1}$. Therefore, it is always true that $\Pr(\neg(S^* \to v_1) \mid \{v_1\} \xrightarrow{\cancel{S^*}} w) \geq \frac{1}{2(k+1)}$.

Finally, we consider $\Sigma(\{v_1\}) \setminus \Sigma(S^*)$ by only accounting for those vertices in $\Gamma(u_1) \setminus S^*$.

$$\begin{aligned}
\frac{|\Sigma(\{v_1\}) \setminus \Sigma(S^*)|}{\prod_{w \in V} \deg(w)} &\geq \sum_{w \in \Gamma(u_1) \setminus S^*} \Pr\left(\left(\{v_1\} \xrightarrow{\cancel{S^*}} w \right) \wedge \neg\left(S^* \to v_1\right) \right) \\
&\geq \sum_{w \in \Gamma(u_1) \setminus S^*} \frac{1}{2(k+1)} \Pr\left(\{v_1\} \xrightarrow{\cancel{S^*}} w \right) \\
&> \frac{1}{2(k+1)} \cdot \frac{9\sigma(S^*)}{2000k^2} && \text{(result from Step 1)} \\
&> \frac{1}{4000k^3} \frac{|\Sigma(S^*)|}{\prod_{w \in V} \deg(w)}.
\end{aligned}$$

By Lemma 4.14, this implies $|\Sigma(S)| \geq (1 - (1 - \frac{1}{k})^k + \frac{1}{64000k^3})|\Sigma(S^*)|$, which further implies this proposition. □

Finally, we prove that C_3 cannot be too large if the greedy algorithm does not overcome the $1 - (1 - 1/k)^k$ barrier. Informally, this is because C_3 corresponds to a subset of the intersection among $\Sigma(\{u_1\}), \ldots, \Sigma(\{u_k\})$, and Lemma 4.13 implies that it cannot be too large.

Proposition 4.21. *If $C_3 > \frac{1}{k^2} \cdot \sigma(S^*)$, then $\sigma(S) \geq (1-(1-\frac{1}{k})^k + \frac{1}{8k^3})\sigma(S^*)$.*

Proof. Notice that $C_3 \prod_{w\in V} \deg(w)$ is at most the number of tuples (w,g) such that w is reachable from more than one vertex in S^* under g. It is easy to see that

$$C_3 \prod_{w\in V} \deg(w) \leq \left(\sum_{i=1}^{k} |\Sigma(\{u_i\})|\right) - |\Sigma(S^*)|$$

because: 1) each (w,g) such that w is reachable by more than one vertex in S^* under g is counted at most once by $C_3 \prod_{w\in V} \deg(w)$, exactly once by $\Sigma(S^*)$, and at least twice by $\sum_{i=1}^{k} \Sigma(\{u_i\})$, so the contribution of each such (w,g) to the right-hand side of the inequality is at least the contribution of it to the left-hand side; 2) each (w,g) such that w is reachable by exactly one vertex in S^* under g is not counted by $C_3 \prod_{w\in V} \deg(w)$ and is counted exactly once by both $\sum_{i=1}^{k} \Sigma(\{u_i\})$ and $\Sigma(S^*)$, so the contribution of such (w,g) is the same on both sides of the inequality; 3) each (w,g) such that g is not reachable from S^* contributes 0 to both sides of the inequality. Observing this inequality, if $C_3 > \frac{1}{k^2} \cdot \sigma(S^*)$, we have

$$\left(\sum_{i=1}^{k} |\Sigma(\{u_i\})|\right) - |\Sigma(S^*)| > \frac{1}{k^2}\sigma(S^*) \prod_{w\in V} \deg(w) = \frac{1}{k^2}|\Sigma(S^*)|.$$

Lemma 4.13 implies $|\Sigma(S)| \geq (1-(1-\frac{1}{k})^k+\frac{1}{8k^3})|\Sigma(S^*)|$, which implies this proposition.

□

With Proposition 4.19, 4.20 and 4.21, if $\sigma(S) = (1-(1-1/k)^k + o(1/k^3))\sigma(S^*)$, it must be that

$$\frac{|\Sigma(\{v_1\}) \cap \Sigma(S^*)|}{\prod_{w\in V} \deg(w)} = C_1 + C_2 + C_3 \leq \left(\frac{1}{k^2} + \frac{9}{10k} + \frac{1}{100k}\right)\sigma(S^*) < \frac{\frac{92}{100k}|\Sigma(S^*)|}{\prod_{w\in V} \deg(w)}.$$

However, Lemma 4.12 would have implied $\sigma(S) \geq (1-(1-\frac{1}{k})^k + \frac{8}{400k})\sigma(S^*)$, which is a contradiction. This finishes proving Theorem 4.10.

4.5 On Other Alternative Models

In this section, we first consider the linear threshold InfMax on more general models. Naturally, Theorem 4.8 holds if the model is more general. We study if Theorem 4.10

still holds. We consider whether the barrier $1-(1-1/k)^k$ can still be overcome. Subsequently, we consider alternative models of LTM on undirected graphs that are not ULTM, those that violate Assumption 2.14.

Directed graphs If we consider InfMax with LTM on general directed graphs, Theorem 4.10 no longer holds, even for ULTM. Moreover, for any positive function $f(k)$ which may be infinitesimal, there is always an example where the greedy algorithm achieves less than a $(1-(1-1/k)^k+f(k))$-approximation. Example 4.9 can be easily adapted to show this. Firstly, all the $k(k+\ell)$ edges (u_i,v_j) become directed, so the cliques associated with those u_i's are not even needed. We replace each C_i by a single vertex u_i. Secondly, $D_1,\ldots,D_{k+\ell}$ become directed stars such that the directed edges in each star D_i are from v_i to the remaining vertices in the star. Lastly, we change the size of the star so that $|D_i|=\lceil m(1-\frac{1}{k})^{i-1}\rceil$ for $i=1,\ldots,k$ and $|D_{k+1}|=\cdots=|D_{k+\ell-1}|=\lceil m(1-\frac{1}{k})^k\rceil$, where ℓ and $|D_{k+\ell}|$ are set such that $\sum_{i=1}^{k+\ell}|D_i|=mk-2k$ and m is a large integer which can be set significantly larger than $1/f(k)$.

Now each u_i has in-degree 0, so will never be infected unless seeded. Each v_j has in-degree exactly k, and each u_i will contribute $1/k$ to v_j's infection probability. Straightforward calculations reveal that the greedy algorithm will pick $S=\{v_1,\ldots,v_k\}$ so that $\sigma(S)=\sum_{i=1}^{k}\lceil m(1-\frac{1}{k})^{i-1}\rceil\leq mk(1-(1-k)^k)+k$. On the other hand, the optimal solution is $S^*=\{u_1,\ldots,u_k\}$, and $\sigma(S^*)=k+(mk-2k)=mk-k$. We have $\frac{\sigma(S)}{\sigma(S^*)}=\frac{mk(1-(1-k)^k)+k}{mk-k}$, which can be less than $(1-(1-1/k)^k+f(k))$ when m is sufficiently large.

Prescribed seed set Khanna and Lucier [47] considered the more generalized setting where the seed set S can only be a subset of a prescribed vertex set $V'\subseteq V$, where V' is a part of the input of the instance, and showed that their result for the independent cascade model can be extended to this setting. It is straightforward to check that our proof for Theorem 4.10 can also be extended to this setting. In particular, all the lemmas in Sect. 4.4.1 hold for the generalized Max-k-Coverage setting where $\mathcal{S}$ must be a subset of a prescribed candidate set $\mathcal{M}'\subseteq\mathcal{M}$, with the proofs being exactly the same. Basically, the proofs in Sect. 4.4 do not rely on that each vertex in V is a valid seed choice, so restricting that the seeds can only be chosen from V' does not invalidate any propositions or lemmas.

Weighted vertices Another generalization Khanna and Lucier [47] considered is to allow that each vertex v has a positive weight $\omega(v)$, and the objective of InfMax is to find the seed set that maximizes the expected total *weight* of infected vertices. Khanna and Lucier [47] showed that the greedy algorithm can still achieve a $(1-(1-1/k)^k+c)$ approximation (for some constant $c>0$) for this generalized model. We show that, for `LTM`, the story is completely different. If vertices are weighted, for any positive function $f(k)$ which may be infinitesimal, there is always an example where the greedy algorithm achieves less than a $(1-(1-1/k)^k+f(k))$-approximation (for the linear threshold InfMax with undirected graphs). Thus, Theorem 4.10 fails to extend to this setting. While the settings with and without weighted vertices are not very different in `ICM`, they are quite different for `LTM`.

Again, Example 4.9 can be easily adapted to show our claim. Let $m \gg k$ be a very large number. Firstly, change the size of each clique C_i to $m^{0.1}$. Secondly, instead of connecting each v_i to a lot of vertices to form a star, we let v_i have a very high weight (so each star D_i is replaced by a single vertex v_i). Specifically, let $\omega(v_i)=m(1-1/k)^{i-1}$ for each $i=1,\ldots,k$, let $\omega(v_{k+1})=\cdots=\omega(v_{k+\ell-1})=m(1-1/k)^k$, and let ℓ and $\omega(v_{k+\ell})$ be such that $\sum_{i=1}^{k+\ell}\omega(v_i)=mk-m^{0.1}k$. Let the weight of all the remaining vertices be 1. The greedy algorithm will pick $\{v_1,\ldots,v_k\}$, and the expected total weight of infected vertices is $o(m^{0.1})+\sum_{i=1}^{k}\omega(v_i)=m(1-(1-1/k)^k)+o(m^{0.1})$. The optimal seeds are $u_1,\ldots,u_k$, with expected total weight of infected vertices being at least $mk-m^{0.1}k$. We have $\frac{\sigma(S)}{\sigma(S^*)}\leq\frac{mk(1-(1-1/k)^k)+o(m^{0.1})}{mk-m^{0.1}k}$, which is less than $(1-(1-1/k)^k+f(k))$ when m is sufficiently large.

Alternative models for linear threshold model So far, we have been following Assumption 2.14 such that the graph becomes automatically unweighted if we are dealing with `LTM` for undirected graphs. Can we define `LTM` for undirected graphs in a more general way that allows edge-weighted graphs?

For undirected graphs, `ULTM` is a special case of `LTM` by assigning weights to the edges in the graph (that is originally unweighted) as follows: $w(u,v)=\frac{1}{\deg(v)}$. If the undirected graph $G=(V,E,w')$ is originally weighted, then a natural extension is to define $w(u,v)=\frac{w'(u,v)}{\sum_{u\in\Gamma(v)}w'(u,v))}$.

In Appendix A.4, we discuss alternative or more general ways to define a linear threshold model on undirected graphs. In particular, we show that in the above undirected weighted version of `LTM`, the greedy algorithm cannot achieve a $(1-(1-1/k)^k+f(k))$-approximation for any positive function $f(k)$. (See the subsection "weighted undirected graphs with normalization" in Appendix A.4.) We also consider a version where we require all incoming edges of a vertex v to have the same weight but the total weight is allowed to be strictly less than 1. In this case, all our results (Theorem 4.8 and Theorem 4.10) still hold. (See the subsection "Unweighted undirected graphs with slackness" in Appendix A.4.)

4.6 Conclusion and Open Problems

We have seen that the greedy algorithm for InfMax with `ULTM` on undirected graphs can overcome the $1-(1-1/k)^k$ barrier by an additive term $\Omega(1/k^3)$ as shown in Theorem 4.10. However, Theorem 4.8 suggests that, unlike the case for `ICM`, the greedy algorithm cannot overcome the $(1-1/e)$ barrier for $k \to \infty$ for `ULTM`. Moreover, we have seen in Sect. 4.5 that the approximation guarantee $1-(1-1/k)^k$ is tight if the vertices are weighted, which is different from Khanna and Lucier's result for `ICM`. This again suggests that there are fundamental differences between these two models.

The tight example in Example 4.9 has a significant limitation: it cannot scale to large $\sigma(S^*)$. Notice that, to make the example work, we have to make the size of each D_i be $o(k)$ and $\sigma(S^*) = o(k^2)$. Otherwise, $\{u_1, \ldots, u_k\}$ will not be able to infect each v_i with probability $1 - o(1)$. If this happens, each seed in the seed set $\{v_1, \ldots, v_k\}$ output by the greedy algorithm will not be connected from $\{u_1, \ldots, u_k\}$ with a constant probability. In the Max-k-Coverage view, this will imply that $\Sigma(S) \setminus \Sigma(S^*)$ contains a significant number of elements, which will make the greedy algorithm overcome the $1-1/e$ barrier. Therefore, a natural question is, if $\sigma(S^*)$ is large enough, say, $\sigma(S^*) = \omega(k^2)$, can the $1-1/e$ barrier be overcome? We believe it can be overcome, and we make the following conjecture.

Conjecture 4.22. *Consider* InfMax *problem* $(G = (V, E), k)$ *with* `ULTM` *on undirected graphs. If* $\max_{S: S \subseteq V, |S| \leq k} \sigma(S) = \omega(k^2)$*, there exists a constant* $c > 0$ *such that the greedy algorithm achieves a* $(1 - 1/e + c)$*-approximation.*

Other than to prove (or disprove) the conjecture above, another open problem is to further close the gap for InfMax with undirected graphs. Right now, the gap between $(1-1/e)$ [44] and $(1-\tau)$ (Chapter 3) is still large. Designing an approximation algorithm that achieves significantly better than a $(1-1/e)$-approximation and proving a stronger APX-hardness result are two interesting and important directions for future work.

CHAPTER 5

Adaptive Influence Maximization and Greedy Adaptivity Gap

In this chapter, we consider the *adaptive influence maximization problem*: InfMax where seeds are chosen *iteratively* and *adaptivity*. In the *full-adoption feedback model*, after selecting each seed, the seed-picker observes all the resulting adoptions. In the *myopic feedback model*, the seed-picker only observes whether each neighbor of the chosen seed adopts. Motivated by the extreme success of greedy-based algorithms/heuristics for InfMax, we propose the concept of *greedy adaptivity gap*, which compares the performance of the adaptive greedy algorithm to its non-adaptive counterpart. Our first result shows that, for submodular InfMax, the adaptive greedy algorithm can perform up to a $(1-1/e)$-fraction worse than the non-adaptive greedy algorithm, and that this ratio is tight. More specifically, on one side we provide examples where the performance of the adaptive greedy algorithm is only a $(1-1/e)$ fraction of the performance of the non-adaptive greedy algorithm in four settings: for both feedback models and both `ICM` and `LTM`. On the other side, we prove that in any submodular cascade, the adaptive greedy algorithm always outputs a $(1-1/e)$-approximation to the expected number of adoptions in the optimal non-adaptive seed choice. Our second result shows that, for the general submodular diffusion model with full-adoption feedback, the adaptive greedy algorithm can outperform the non-adaptive greedy algorithm by an unbounded factor. Finally, we propose a risk-free variant of the adaptive greedy algorithm that always performs no worse than the non-adaptive greedy algorithm.

5.1 Introduction

In this chapter, we study the *adaptive influence maximization problem*, where seeds are selected iteratively and feedback is given to the seed-picker after selecting each seed. Two different feedback models have been studied in the past: the *full-adoption feedback model* and the *myopic feedback model* [36]. In the full-adoption feedback model, the seed-picker sees the entire diffusion process of each selected seed, and in the myopic feedback model the seed-picker only sees whether each neighbor of the chosen seed is infected.

Past literature focused on the *adaptivity gap*—the ratio between the performance of the *optimal* adaptive algorithm and the performance of the *optimal* non-adaptive algorithm [36, 61, 14]. However, even in the non-adaptive setting, InfMax is known to be APX-hard (see Chapter 3). As a result, in practice, it is not clear whether the adaptivity gap can measure how much better an adaptive algorithm can do.

In this chapter, we define and consider the *greedy adaptivity gap*, which is the ratio between the performance of the adaptive greedy algorithm and the non-adaptive greedy algorithm. We focus on the gap between the greedy algorithms for three reasons. First, as we mentioned, the APX-hardness of InfMax renders the practical implications of the adaptivity gap unclear. Second, as we remarked multiple times, the greedy algorithm is used almost exclusively in the context of influence maximization. Third, the iterative nature of the original greedy algorithm naturally extends to the adaptive setting.

Our results We show that, for the general submodular diffusion models, with both the full-adoption feedback model and the myopic feedback model, the infimum of the greedy adaptivity gap is exactly $(1 - 1/e)$ (Sect. 5.3). In addition, this result can be extended to both `ICM` and `LTM`. This is proved in two steps.

As the first step, in Sect. 5.3.1, we show that there are InfMax instances where the adaptive greedy algorithm can only produce $(1 - 1/e)$ fraction of the influence of the solution output by the non-adaptive greedy algorithm. This result is surprising: one would expect that the adaptivity is always helpful, as the feedback provides more information to the seed-picker, which makes the seed-picker refine the seed choices in future iterations. Our result shows that this is not the case, and the feedback, if overly used, can make the seed-picker act in a more myopic way, which is potentially harmful.

As the second step, in Sect. 5.3.2, we show that the adaptive greedy algorithm always achieves a $(1 - 1/e)$-approximation of the non-adaptive optimal solution, so its performance is always at least a $(1 - 1/e)$ fraction of the performance of the non-adaptive greedy algorithm. In particular, combining the two steps, we see that when the adaptive greedy algorithm output only obtains a (nearly) $(1-1/e)$-fraction

model	AG	GAG inf	GAG sup
ICM, full-adoption	at least $e/(e-1)$ [14]	$1-1/e$ (Thm 5.6)	unknown
ICM, myopic	at least $e/(e-1)$, at most 4 [61]	$1-1/e$ (Thm 5.6)	at most $4e/(e-1)$ [61]
LTM, full-adoption	unknown	$1-1/e$ (Thm 5.6)	unknown
LTM, myopic	unknown	$1-1/e$ (Thm 5.6)	unknown
GSDM, full-adoption	∞ (Thm 5.15)	$1-1/e$ (Thm 5.6)	∞ (Thm 5.14)
GSDM, myopic	at least $e/(e-1)$ (implied by [61])	$1-1/e$ (Thm 5.6)	unknown

Table 5.1: Results for the adaptivity gap (AG), the infimum of the greedy adaptivity gap (GAG inf) and the supremum of the greedy adaptivity gap (GAG sup), where GSDM stands for general submodular diffusion model.

of the performance of the non-adaptive greedy algorithm, the non-adaptive greedy algorithm is (almost) optimal. This worst-case guarantee indicates that the adaptive greedy algorithm will never be too bad.

As the second result, in Sect. 5.4, we show that the supremum of the greedy adaptivity gap is infinity, for the general submodular diffusion model with full-adoption feedback. This indicates that the adaptive greedy algorithm can perform significantly better than its non-adaptive counterpart. We also show, with almost the same proof, that the adaptivity gap in this setting (general submodular model with full-adoption feedback) is also unbounded.

All the results above hold for the "exact" deterministic greedy algorithm where a vertex with the exact maximum marginal influence is chosen as a seed in each iteration. However, most variants of the greedy algorithm used in practice are randomized algorithms that find a seed with a marginal influence *close to* the maximum *with high probability* in each iteration. In Sect. 5.5, we discuss how our results for the exact greedy algorithm can be adapted to those greedy algorithms used in practice.

Finally, in Sect. 5.6, we propose a risk-free but more conservative variant of the adaptive greedy algorithm, which always performs at least as well as the non-adaptive greedy algorithm. In Sect. 5.7, we compare this variant of the adaptive greedy algorithm with the adaptive greedy algorithm and the non-adaptive greedy algorithm by implementing experiments on social networks in our real life.

We summarize the existing results about the adaptivity gap (see Sect. 1.2.3 for details) and our new results about the greedy adaptivity gap in Table 5.1.

5.2 Preliminary

We consider the triggering model (Sect. 2.1.2) for this chapter. A more general way to capture submodular diffusion models is the general threshold model (Sect. 2.1.1) with *submodular local influence functions.* All our results hold under this setting as well. We will discuss this in Appendix B.

5.2.1 Adaptive Influence Maximization

In the remaining part of this subsection, we define the adaptive version of the influence maximization problem. We will define two different models: the *full-adoption feedback model* and the *myopic feedback model.* Suppose a seed set $S \subseteq V$ is chosen by the seed-picker, and an underlying realization ϕ is given but not known by the seed-picker. Informally, in the full-adoption feedback model, the seed-picker sees all the vertices that are infected by S in all future iterations, i.e., the seed-picker sees $I_{G,D}^{\phi}(S)$. In the myopic feedback model, the seed-picker only sees the states of S's neighbors, i.e., whether each vertex in $\{v \mid \exists s \in S : s \in \Gamma(v)\}$ is infected.

Define a *partial realization* as a function $\varphi : E \to \{\mathtt{L}, \mathtt{B}, \mathtt{U}\}$ such that $\varphi(e) = \mathtt{L}$ if e is known to be live, $\varphi(e) = \mathtt{B}$ if e is known to be blocked, and $\varphi(e) = \mathtt{U}$ if the status of e is not yet known. We say that a partial realization φ *is consistent with* the full realization ϕ, denoted by $\phi \simeq \varphi$, if $\phi(v) = \varphi(v)$ whenever $\varphi(v) \neq \mathtt{U}$. For the ease of notation, for an edge $(u, v) \in E$, we will write $\phi(u, v), \varphi(u, v)$ instead of $\phi((u, v)), \varphi((u, v))$.

Definition 5.1. Given a triggering model $I_{G=(V,E),D}$ with a realization ϕ, the *full-adoption feedback* is a function $\Phi_{G,D,\phi}^{\mathtt{f}}$ mapping a seed set $S \subseteq V$ to a partial realization φ such that

- $\varphi(u, v) = \phi(u, v)$ for each $u \in I_{G,D}^{\phi}(S)$, and
- $\varphi(u, v) = \mathtt{U}$ for each $u \notin I_{G,D}^{\phi}(S)$.

Definition 5.2. Given a triggering model $I_{G=(V,E),D}$ with a realization ϕ, the *myopic feedback* is a function $\Phi_{G,D,\phi}^{\mathtt{m}}$ mapping a seed set $S \subseteq V$ to a partial realization φ such that

- $\varphi(u, v) = \phi(u, v)$ for each $u \in S$, and
- $\varphi(u, v) = \mathtt{U}$ for each $u \notin S$.

An *adaptive policy* π is a function that maps a seed set S and a partial realization φ to a vertex $v=\pi(S,\varphi)$, which corresponds to the next seed the policy π would choose given φ and S being the set of seeds that has already been chosen. Naturally, we only care about $\pi(S,\varphi)$ when $\varphi=\Phi^{\mathfrak{f}}_{G,D,\phi}(S)$ or $\varphi=\Phi^{\mathfrak{m}}_{G,D,\phi}(S)$, although we define π that specifies an output for any possible inputs S and φ. Notice that we have defined π as a deterministic policy for simplicity, and our results hold for randomized policies. Let Π be the set of all possible adaptive policies.

Notice that an adaptive policy π completely specifies a seeding strategy in an iterative way. Given an adaptive policy π and a realization ϕ, let $\mathcal{S}^{\mathfrak{f}}(\pi,\phi,k)$ be the first k seeds selected according to π with the underlying realization ϕ under the full-adoption feedback model. By on our definition, $\mathcal{S}^{\mathfrak{f}}(\pi,\phi,k)$ can be computed as follows:

1. initialize $S=\emptyset$;
2. update $S=S\cup\{\pi(S,\Phi^{\mathfrak{f}}_{G,D,\phi}(S))\}$ for k iterations;
3. output $\mathcal{S}^{\mathfrak{f}}(\pi,\phi,k)=S$.

Define $\mathcal{S}^{\mathfrak{m}}(\pi,\phi,k)$ similarly for the myopic feedback model, where $\Phi^{\mathfrak{m}}_{G,D,\phi}(S)$ instead of $\Phi^{\mathfrak{f}}_{G,D,\phi}(S)$ is used in Step 2 above.

Let $\sigma^{\mathfrak{f}}(\pi,k)$ be the expected number of infected vertices given that k seeds are chosen according to π, i.e., $\sigma^{\mathfrak{f}}(\pi,k)=\mathbb{E}_{\phi\sim F}[|I^{\phi}_{G,D}(\mathcal{S}^{\mathfrak{f}}(\pi,\phi,k))|]$. Define $\sigma^{\mathfrak{m}}(\pi,k)$ similarly for the myopic feedback model.

Definition 5.3. The *adaptive influence maximization problem* (adaptive InfMax) is an optimization problem which takes as inputs $G=(V,E)$, D, and $k\in\mathbb{Z}^+$, and outputs an adaptive policy π that maximizes the expected total number of infections: $\pi\in\operatorname{argmax}_{\pi\in\Pi}\sigma^{\mathfrak{f}}(\pi,k)$ or $\pi\in\operatorname{argmax}_{\pi\in\Pi}\sigma^{\mathfrak{m}}(\pi,k)$ (depending on the feedback model used).

5.2.2 Adaptivity Gap and Greedy Adaptivity Gap

The adaptivity gap is defined as the ratio between the performance of the optimal adaptive policy and the performance of the optimal non-adaptive seeding strategy. In this chapter, we only consider the adaptivity gap for triggering models.

Definition 5.4. The *adaptivity gap with full-adoption feedback* is

$$\sup_{G,D,k}\frac{\max_{\pi\in\Pi}\sigma^{\mathfrak{f}}(\pi,k)}{\max_{S\subseteq V,|S|\leq k}\sigma(S)}.$$

The *adaptivity gap with myopic feedback* is defined similarly.

The (non-adaptive) *greedy algorithm* iteratively picks a seed that has the maximum marginal gain to the objective function $\sigma(\cdot)$, which has been defined in Sect. 2.2. Let $S^g(k)$ be the set of k seeds output by the (non-adaptive) greedy algorithm.

The *greedy adaptive policy* π^g is defined as $\pi^g(S,\varphi)=s$ such that

$$s \in \operatorname*{argmax}_{s\in V} \mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}\left(S\cup\{s\}\right)\right| - \left|I^{\phi}_{G,D}\left(S\right)\right|\right],$$

with tie broken in an arbitrary consistent order.

Definition 5.5. Given a triggering model $I_{G,D}$ and $k\in\mathbb{Z}^+$, the *greedy adaptivity gap with full-adoption feedback* is $\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}$. The *greedy adaptivity gap with myopic feedback* is defined similarly.

Notice that, unlike the adaptivity gap in Definition 5.4, we leave G,D,k unspecified (instead of taking a supremum over them) when defining the greedy adaptivity gap. This is because we are interested in both supremum and infimum of the ratio $\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}$. Notice that the infimum of the ratio $\frac{\max_{\pi\in\Pi}\sigma^{\mathfrak{f}}(\pi,k)}{\max_{S\subseteq V,|S|\leq k}\sigma(S)}$ in Definition 5.4 is 1: the optimal adaptive policy is at least as good as the optimal non-adaptive policy, as the non-adaptive policy can be viewed as a special adaptive policy; on the other hand, it is easy to see that there are InfMax instances such that the optimal adaptive policy is no better than non-adaptive one (for example, a graph containing k vertices but no edges). For this reason, we only care about the supremum of this ratio.

5.3 Infimum of Greedy Adaptivity Gap

In this section, we show that the infimum of the greedy adaptivity gap for the triggering model is exactly $(1-1/e)$, for both the full-adoption feedback model and the myopic feedback model. This implies that the greedy adaptive policy can perform even worse than the conventional non-adaptive greedy algorithm, but it will never be significantly worse. Moreover, we show that this result also holds for both `ICM` and `LTM`.

Theorem 5.6. *For the full-adoption feedback model,*

$$\inf_{G,D,k:\ I_{G,D}\text{ is }\texttt{ICM}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}=\inf_{G,D,k:\ I_{G,D}\text{ is }\texttt{LTM}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}=\inf_{G,D,k}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}=1-\frac{1}{e}.$$

The same result holds for the myopic feedback model.

In Sect. 5.3.1, we show by providing examples that the greedy adaptive policy in the worst case will only achieves $(1-1/e+\varepsilon)$-approximation of the expected number of infected vertices given by the non-adaptive greedy algorithm, for both `ICM` and `LTM`.

In Sect. 5.3.2, we shows that the greedy adaptive policy has performance at least $(1-1/e)$ of the performance of the non-adaptive optimal seeds (Theorem 5.10). Theorem 5.10 provides a lower bound on the greedy adaptivity gap for the triggering model and is also interesting on its own. At the end of Sect. 5.3.2, we prove Theorem 5.6 by putting the results from Sect. 5.3.1 and Sect. 5.3.2 together.

5.3.1 Tight Examples

In this subsection, we show that the adaptive greedy algorithm can perform worse than the non-adaptive greedy algorithm by a factor of $(1-1/e+\varepsilon)$, for both `ICM` and `LTM` and any $\varepsilon > 0$. This may be surprising, as one would expect that the feedback provided to the seed-picker will refine the seed choices in the future iterations. Here, we provide some intuitions why adaptivity can sometimes hurt. Suppose there are two promising sequences of seed selections, $\{s, u_1, \ldots, u_k\}$ and $\{s, v_1, \ldots, v_k\}$, such that

- s is the best seed which will be chosen first;
- $\{s, u_1, \ldots, u_k\}$ has a better performance;
- the influence of $u_1, \ldots, u_k$ are non-overlapping, the influence of $v_1, \ldots, v_k$ are non-overlapping, but the influence of u_i, v_j overlaps for each i, j; moreover, if u_1 is picked as the second seed, the greedy algorithm, adaptive or not, will continue to pick $u_2, \ldots, u_k$, and if v_1 is picked as the second seed, $v_2, \ldots, v_k$ will be picked next;

Now, suppose there is a vertex t elsewhere which can be infected by both s and v_1, such that

- if t is infected by s, which slightly reduces the marginal influence of v_1, v_1 will be less promising than u_1;
- if t is not infected by s, v_1 is more promising than u_1;
- in average, when there is no feedback, v_1 is still less promising than u_1, even after adding the increment in t's infection probability to v_1's expected marginal influence.

In this case, the non-adaptive greedy algorithm will "go to the right trend" by selecting u_1 as the second seed; the adaptive greedy algorithm, if receiving feedback that t is not infected by s, will "go to the wrong trend" by selecting v_1 next.

As a high-level description of the lesson we learned, both versions of the greedy algorithms are intrinsically myopic, and the feedback received by the adaptive policy may make the seed-picker act in a more myopic way, which could be more hurtful to the final performance.

We will assume in the rest of this section that vertices can have positive integer weights, as justified in the following remark.

Remark 5.7. For both `ICM` and `LTM`, we can assume without loss of generality that each vertex has a positive integer weight, so that, in InfMax, we are maximizing the expected total weight of the infected vertices instead of maximizing the expected number of infected vertices as before. Suppose we want to make a vertex v have weight $W \in \mathbb{Z}^+$. We can construct $W-1$ vertices $w_1, \ldots, w_{W-1}$, and create $W-1$ directed edges $(v, w_1), \ldots, (v, w_{W-1})$ with weight 1. (Recall from Definition 2.7 and Definition 2.11 that the graphs in both `ICM` and `LTM` are edge-weighted, and the weights of edges completely characterize the collection of triggering set distributions F.) It is straightforward that, for both `ICM` and `LTM`, each of $w_1, \ldots, w_{W-1}$ will be infected with probability 1 if v is infected. In addition, both the greedy algorithm and the greedy adaptive policy will never pick any of $w_1, \ldots, w_{W-1}$ as seeds, as seeding v is strictly better. Therefore, we can consider the subgraph consisting of $v, w_1, \ldots, w_{W-1}$ as a gadget that representing a vertex v having weight W.

Lemma 5.8. *For any $\varepsilon > 0$, there exists G, D, k such that $I_{G,D}$ is an `ICM` and*

$$\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S^g(k))} \leq 1 - \frac{1}{e} + \varepsilon, \qquad \frac{\sigma^{\mathfrak{m}}(\pi^g, k)}{\sigma(S^g(k))} \leq 1 - \frac{1}{e} + \varepsilon.$$

Proof. We will construct an InfMax instance $(G = (V, E, w), k+1)$ with $k+1$ seeds allowed. Let $W \in \mathbb{Z}^+$ be a sufficiently large integer divisible by $k^{2k}(k-1)$ and whose value are to be decided later. Let $\Upsilon = W/k^2$. The vertex set V contains the following weighted vertices:

- a vertex s that has weight $2W$;
- a vertex t that has weight $4k\Upsilon$;
- $2k$ vertices $u_1, \ldots, u_k, v_1, \ldots, v_k$ that have weight 1;

- $2k^2$ vertices $\{w_{ij} \mid i = 1, \ldots, 2k; j = 1, \ldots, k\}$
  - $w_{11}, \ldots, w_{1k}$ have weight $\frac{W}{k}$;
  - for each $i \in \{2, \ldots, k\}$, $w_{i1}, \ldots, w_{ik}$ have weight $\frac{1}{k}(1-\frac{1}{k})^{i-1}W + \frac{4k-2}{k-1}\Upsilon$;
  - for each $i \in \{k+1, \ldots, 2k\}$, $w_{i1}, \ldots, w_{ik}$ have weight $\frac{1}{k}(1-\frac{1}{k})^{k}W$.

The edge set E is specified as follow:

- create two edges (v_1, t) and (s, t);
- for each $i = 1, \ldots, k$, create $2k$ edges $(u_i, w_{1i}), (u_i, w_{2i}), \ldots, (u_i, w_{(2k)i})$, and create k edges $(v_i, w_{i1}), (v_i, w_{i2}), \ldots, (v_i, w_{ik})$.

For the weights of edges, all the edges have weight 1 except for the edge (s, t) which has weight $1/k$.

It is straightforward to check that

$$\begin{aligned}
\sigma(\{s\}) &= \overline{w}(s) + \frac{1}{k}\overline{w}(t) = 2W + 4\Upsilon, && (5.1)\\
\forall i \in \{1, \ldots, k\} : \sigma(\{u_i\}) &= \overline{w}(u_i) + \sum_{j=1}^{2k}\overline{w}(w_{ji}) = 1 + W + (4k-2)\Upsilon, && (5.2)\\
\sigma(\{v_1\}) &= \overline{w}(v_1) + \overline{w}(t) + \sum_{j=1}^{k}\overline{w}(w_{1j}) = 1 + 4k\Upsilon + W, && (5.3)\\
\forall i \in \{2, \ldots, k\} : \sigma(\{v_i\}) &= \overline{w}(v_i) + \sum_{j=1}^{k}\overline{w}(w_{ij})\\
&= 1 + \left(1 - \frac{1}{k}\right)^{i-1} W + \frac{k(4k-2)}{k-1}\Upsilon, && (5.4)
\end{aligned}$$

and the influence of the remaining vertices are significantly less than these.

Since s has the highest influence, both the greedy algorithm and the greedy adaptive policy will choose s as the first seed.

The non-adaptive greedy algorithm will choose $u_1, \ldots, u_k$ iteratively for the next k seeds, and the expected number of infected vertices by the seeds chosen by non-adaptive greedy algorithm is

$$\sigma(\{s, u_1, \ldots, u_k\}) = \overline{w}(s) + \frac{1}{k}\overline{w}(t) + \sum_{i=1}^{k}\overline{w}(u_i) + \sum_{i=1}^{2k}\sum_{j=1}^{k}\overline{w}(w_{ij}) = (k+2)W + o(W). \tag{5.5}$$

To show the former claim, letting $U_i = \{s, u_1, \ldots, u_i\}$ and $U_0 = \{s\}$, and supposing without loss of generality that the non-adaptive greedy algorithm has chosen U_i for the first $(i+1)$ seeds (notice the symmetry of $u_1, \ldots, u_k$), it suffices to show that, for any vertex x, we have

$$\sigma(U_i \cup \{u_{i+1}\}) - \sigma(U_i) \geq \sigma(U_i \cup \{x\}) - \sigma(U_i). \tag{5.6}$$

To consider an x that makes the right-hand side large, it is easy to see that we only need to consider one of $u_{i+1}, \ldots, u_k, v_1, v_2$, as the remaining vertices clearly have less marginal influence. By symmetry, $\sigma(U_i \cup \{u_{i+1}\}) = \sigma(U_i \cup \{u_{i+2}\}) = \cdots = \sigma(U_i \cup \{u_k\})$. Therefore, we only need to consider x being v_1 or v_2. It is straightforward to check that

$$\sigma(U_i \cup \{u_{i+1}\}) - \sigma(U_i) = 1 + W + (4k-2)\Upsilon, \tag{5.7}$$

$$\sigma(U_i \cup \{v_1\}) - \sigma(U_i) = 1 + (4k-4)\Upsilon + \frac{k-i}{k}W \leq 1 + W + (4k-4)\Upsilon, \tag{5.8}$$

$$\begin{aligned}\sigma(U_i \cup \{v_2\}) - \sigma(U_i) &= 1 + \frac{k-i}{k}\left(1 - \frac{1}{k}\right)W + \frac{(k-i)(4k-2)}{k-1}\Upsilon \\ &\leq 1 + W - \frac{W}{k} + (4k+5)\Upsilon.\end{aligned} \tag{5.9}$$

Recall that $\Upsilon = W/k^2$, straightforward calculations show that $\sigma(U_i \cup \{u_{i+1}\}) - \sigma(U_i)$ is maximum.

For the greedy adaptive policy, we have seen that s will be the first seed chosen. The second seed picked by the greedy adaptive policy will depend on whether t is infected by s. Notice that the status of t is available to the policy in both the full-adoption feedback model and the myopic feedback model, so the arguments here, as well as the remaining part of this proof, apply to both feedback models. By straightforward calculations, the greedy adaptive policy will pick v_1 as the next seed if t is not infected by s, and the policy will pick a seed from $u_1, \ldots, u_k$ otherwise.

In the latter case, the policy will eventually pick the seed set $\{s, u_1, \ldots, u_k\}$, which will infect vertices with a total weight of

$$\overline{w}(s) + \overline{w}(t) + \sum_{i=1}^{k} \overline{w}(u_i) + \sum_{i=1}^{2k}\sum_{j=1}^{k} \overline{w}(w_{ij}) = (k+2)W + o\left(W\right)$$

with probability 1 (notice that we are in the scenario that t has been infected by s).

In the former case, we can see that the third seed picked by the policy will be v_2

instead of any of $u_1, \dots, u_k$. In particular, v_2 contributes $1 + (1 - \frac{1}{k})W + \frac{k(4k-2)}{k-1}\Upsilon$ infected vertices. On the other hand, since $w_{11}, \dots, w_{ik}$ have already been infected by v_1, the marginal contribution for each u_i is $\sigma(\{u_i\}) - \overline{w}(w_{1i}) = 1 + W + (k-1) \cdot \frac{4k-2}{k-1}\Upsilon - \frac{1}{k}W$, which is less than the contribution of v_2. By similar analysis, we can see that the greedy adaptive policy in this case will pick the seed set $\{s, v_1, \dots, v_k\}$, which will infect vertices with a total weight of

$$\overline{w}(s) + \overline{w}(t) + \sum_{i=1}^{k} \overline{w}(v_i) + \sum_{i=1}^{k} \sum_{j=1}^{k} \overline{w}(w_{ij}) = \left(2 + \sum_{i=1}^{k} \left(1 - \frac{1}{k}\right)^{i-1}\right) W + o\left(W\right)$$

$$= \left(2 + k\left(1 - \left(1 - \frac{1}{k}\right)^{k}\right)\right) W + o\left(W\right)$$

in expectation.

Since t will be infected with probability $\frac{1}{k}$, the expected weight of infected vertices for the greedy adaptive policy is

$$\frac{1}{k}\left((k+2)W + o\left(W\right)\right) + \left(1 - \frac{1}{k}\right) \cdot \left(\left(2 + k\left(1 - \left(1 - \frac{1}{k}\right)^{k}\right)\right) W + o\left(W\right)\right)$$

$$\leq \left(3 + k\left(1 - \left(1 - \frac{1}{k}\right)^{k}\right)\right) W + o\left(W\right).$$

Putting this together with Eqn. (5.5), both $\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}$ and $\frac{\sigma^{\mathfrak{m}}(\pi^g,k)}{\sigma(S^g(k))}$ in this case are at most

$$\frac{\left(3 + k\left(1 - \left(1 - \frac{1}{k}\right)^{k}\right)\right) W + o\left(W\right)}{(k+2)W + o\left(W\right)},$$

which has limit $1 - 1/e$ when both W and k tend to infinity. □

Lemma 5.9. *For any $\varepsilon > 0$, there exists G, D, k such that $I_{G,D}$ is an* `LTM` *and*

$$\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))} \leq 1 - \frac{1}{e} + \varepsilon, \qquad \frac{\sigma^{\mathfrak{m}}(\pi^g,k)}{\sigma(S^g(k))} \leq 1 - \frac{1}{e} + \varepsilon.$$

Proof. We will construct an InfMax instance $(G = (V, E, w), k+1)$ with $k+1$ seeds allowed. Let $W \in \mathbb{Z}^+$ be a sufficiently large integer divisible by $k^{2k}(k-1)$ and whose value are to be decided later. Let $\Upsilon = W/k^2$. The vertex set V contains the following weighted vertices:

- a vertex s that has weight $2W$;

- a vertex t that has weight $4k\Upsilon$;
- k vertices $u_1, \dots, u_k$ that have weight 1;
- k vertices $v_1, \dots, v_k$ such that $\overline{w}(v_1) = W$ and $\overline{w}(v_i) = W(1-\frac{1}{k})^{i-1} + \frac{4k^2-2k}{k-1}\Upsilon$ for each $i = 2, \dots, k$;
- k vertices $v_{k+1}, \dots, v_{2k}$ such that $\overline{w}(v_{k+1}) = \cdots = \overline{w}(v_{2k}) = W(1-\frac{1}{k})^k$.

The edge set E and the weights of edges are specified as follow:

- create two edges (v_1, t) and (s, t) with weights $1 - \frac{1}{k}$ and $\frac{1}{k}$ respectively;
- create $2k^2$ edges $\{(u_i, v_j) \mid i = 1, \dots, k; j = 1, \dots, 2k\}$, each of which has weight $\frac{1}{k}$.

It is easy to check that the weights of the incoming edges for each vertex v satisfy $\sum_{u\in\Gamma(v)} w(u, v) \leq 1$, as required in Definition 2.11.

The remaining part of the analysis is similar to the proof of Lemma 5.8. The first seed chosen by both algorithms is s. After this, each u_i has marginal influence $1 + \frac{1}{k}\sum_{i=1}^{2k} \overline{w}(v_i) = 1 + W + (4k-2)\Upsilon$. Since t is infected by s with probability $\frac{1}{k}$, the marginal influence of v_1 without any feedback is $(1-\frac{1}{k})\overline{w}(t) + \overline{w}(v_1) = W + (4k-4)\Upsilon$. If t is known to be infected, the marginal influence of v_1 is W; if t is known to be uninfected, then seeding v_1 will infect t with probability 1, and the marginal influence of v_1 is $W + 4k\Upsilon$. By comparing these values, the non-adaptive greedy algorithm will pick one of $u_1, \dots, u_k$ as the second seed, and the greedy adaptive policy will pick v_1 as the second seed if t is not infected and one of $u_1, \dots, u_k$ as the second seed if t is infected. (Notice that $\overline{w}(v_1) > \overline{w}(v_2) > \cdots > \overline{w}(v_k) > \overline{w}(v_{k+1}) = \cdots = \overline{w}(v_{2k})$.)

Simple analyses show that the non-adaptive greedy algorithm will choose seeds $\{s, u_1, \dots, u_k\}$, which will infect all of $v_1, \dots, v_{2k}$ with probability 1, and the adaptive greedy policy will choose $\{s, v_1, \dots, v_k\}$ with a very high probability $1 - \frac{1}{k}$, which will leave $v_{k+1}, \dots, v_{2k}$ uninfected. Since $s, v_1, \dots, v_{2k}$ are the only vertices with weight $\Theta(W)$ and we have both $\sum_{i=1}^{k} \overline{w}(v_i) = (1-(1-\frac{1}{k})^k)W + o(W)$ and $\sum_{i=1}^{2k} \overline{w}(v_i) = W + o(W)$, the lemma follows by taking the limit $W \to \infty$ and $k \to \infty$. □

5.3.2 Lower Bound

Theorem 5.10. *For a triggering model $I_{G,D}$, we have both*

$$\sigma^{\mathsf{f}}(\pi^g, k) \geq \left(1 - \frac{1}{e}\right) \max_{S\subseteq V, |S|\leq k} \sigma(S) \qquad \text{and} \qquad \sigma^{\mathsf{m}}(\pi^g, k) \geq \left(1 - \frac{1}{e}\right) \max_{S\subseteq V, |S|\leq k} \sigma(S).$$

For a high-level idea of the proof, let S with $|S|=i$ be the seeds picked by π^g for the first i iterations and S^* be the optimal non-adaptive seed set: $S^* \in \operatorname{argmax}_{|S'|\leq k}\sigma(S')$. Given S as the existing seeds and any feedback (myopic or full-adoption) corresponding to S, we can show that the marginal increment to the expected influence caused by the $(i+1)$-th seed picked by π^g is at least $1/k$ of the marginal increment to the expected influence caused by S^*. Then, a standard argument showing that the greedy algorithm can achieve a $(1-1/e)$-approximation for any submodular monotone optimization problem can be used to prove this theorem.

Theorem 5.10 is implied by the following three propositions. In the remaining part of this section, we let S^* be an optimal seed set for the non-adaptive InfMax: $S^* \in \max_{S\subseteq V,|S|\leq k}\sigma(S)$.

We first show that the global influence function after fixing a partial seed set S and any possible feedback of S is still submodular.

Proposition 5.11. *Given a triggering model $I_{G,D}$, any $S\subseteq V$, any feedback model (either full-adoption or myopic) and any partial realization φ that is a valid feedback of S (i.e., $\exists\phi:\varphi=\Phi^{\mathfrak{f}}_{G,D,\phi}(S)$ or $\exists\phi:\varphi=\Phi^{\mathfrak{m}}_{G,D,\phi}(S)$, depending on the feedback model considered), the function $\mathcal{T}:\{0,1\}^{|V|}\to\mathbb{R}_{\geq 0}$ defined as $\mathcal{T}(X)=\mathbb{E}_{\phi\simeq\varphi}[|I^{\phi}_{G,D}(S\cup X)|]$ is submodular.*

Proof. Fix a feedback model, $S\subseteq V$ and φ that is a valid feedback of S. Let $\overline{S}$ be the set of infected vertices indicated by the feedback of S. Formally, $\overline{S}$ is the set of all vertices that are reachable from S by only using edges e with $\varphi(e)=\texttt{L}$.

We consider a new triggering model $I_{G',F'}$ defined as follows:

- G' shares the same vertex set with G;
- The edge set of G' is obtained by removing all edges e in G with $\varphi(e)\neq\texttt{U}$;
- The distribution $\mathcal{F}'_v$ is normalized from $\mathcal{F}_v$. Specifically, for each $\mathrm{Trig}_v\subseteq\Gamma(v)$, let $p(\mathrm{Trig}_v)$ be the probability that Trig_v is chosen as the triggering set under $\mathcal{F}_v$. Let $\Gamma'(v)$ be the set of v's in-neighbors in G', and we have $\Gamma'(v)\subseteq\Gamma(v)$ by our construction. Then, $\mathcal{F}'_v$ is defined such that $\mathrm{Trig}_v\subseteq\Gamma'(v)$ is chosen as the triggering set with probability $p(\mathrm{Trig}_v)/\sum_{\mathrm{Trig}'_v\subseteq\Gamma'(v)}p(\mathrm{Trig}'_v)$.

A simple coupling argument reveals that

$$\mathcal{T}(X)=\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup X)\right|\right]=\sigma_{G',F'}(\overline{S}\cup X). \tag{5.10}$$

We define a coupling of a realization ϕ of G with $\phi \simeq \varphi$ to a realization ϕ' of G' in a natural way: $\phi(e) = \phi'(e)$ for all edges e in G'. From our construction of $F' = \{\mathcal{F}'_v\}$, it is easy to see that, when ϕ is coupled with ϕ', the probability that ϕ is sampled under $I_{G,D}$ conditioning on $\phi \simeq \varphi$ equals to the probability that ϕ' is sampled under $I_{G',F'}$. Under this coupling, it is easy to see that u is reachable from S by live edges under ϕ if and only if it is reachable from $\overline{S}$ by live edges under ϕ'. This proves Eqn. (5.10).

Finally, since $\sigma_{G',F'}(\cdot)$ is submodular, for any two vertex sets A, B with $A \subsetneq B$ and any $u \notin B$,

$$\mathcal{T}(A \cup \{u\}) - \mathcal{T}(A) = \sigma_{G',F'}(\overline{S} \cup A \cup \{u\}) - \sigma_{G',F'}(\overline{S} \cup A)$$

is weakly larger than

$$\mathcal{T}(B \cup \{u\}) - \mathcal{T}(B) = \sigma_{G',F'}(\overline{S} \cup B \cup \{u\}) - \sigma_{G',F'}(\overline{S} \cup B)$$

if $u \notin \overline{S}$, and

$$\mathcal{T}(A \cup \{u\}) - \mathcal{T}(A) = \mathcal{T}(B \cup \{u\}) - \mathcal{T}(B) = 0$$

if $u \in \overline{S}$. In both case, the submodularity of $\mathcal{T}(\cdot)$ holds. □

Next, we show that the marginal gain to the global influence function after selecting one more seed according to π^g is at least $1/k$ fraction of the marginal gain of including all the vertices in S^* as seeds.

Proposition 5.12. *Given a triggering model $I_{G,D}$, any $S \subseteq V$, any feedback model and any partial realization φ that is a valid feedback of S, let $s = \pi^g(S, \varphi)$ be the next seed chosen by the greedy policy. We have*

$$\mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S \cup \{s\})\right|\right] - \mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right] \geq \frac{1}{k}\left(\mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S \cup S^*)\right|\right] - \mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\right).$$

Proof. Let $S^* = \{s^*_1, \ldots, s^*_k\}$. By the greedy nature of π^g, we have

$$\forall v : \mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S \cup \{s\})\right|\right] \geq \mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S \cup \{v\})\right|\right],$$

and this holds for v being any of $s^*_1, \ldots, s^*_k$ in particular.

Let $S^*_i = \{s^*_1, \ldots, s^*_i\}$ for each $i = 1, \ldots, k$ and $S^*_0 = \emptyset$, the proposition concludes from the following calculations

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup\{s\})\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\\
\geq&\frac{1}{k}\sum_{i=1}^{k}\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup\{s^*_i\})\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\right)\\
\geq&\frac{1}{k}\sum_{i=1}^{k}\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup S^*_{i-1}\cup\{s^*_i\})\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup S^*_{i-1})\right|\right]\right)\\
&\hfill\text{(Proposition 5.11)}\\
=&\frac{1}{k}\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup S^*)\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\right),
\end{aligned}
$$

where the last equality is by a telescoping sum, by noticing that $S^*_i=S^*_{i-1}\cup\{s^*_i\}$ and $S^*=S^*_k$. □

Finally, we prove the following proposition which is a more general statement than Theorem 5.10.

Proposition 5.13. *For a triggering model $I_{G,D}$ and any $\ell\in\mathbb{Z}^+$, we have $\sigma^{\mathfrak{f}}(\pi^g,\ell)\geq(1-(1-1/k)^{\ell})\sigma(S^*)$, and the same holds for the myopic feedback model.*

Proof. We will only consider the full-adoption feedback model, as the proof for the myopic feedback model is identical. We prove this by induction on ℓ. The base step for $\ell=1$ holds trivially by Proposition 5.12 by considering $S=\emptyset$ in the proposition.

Suppose the inequality holds for $\ell=\ell_0$. We investigate the expected marginal gain to the global influence function by selecting the (ℓ_0+1)-th seed. For a seed set $S\subseteq V$ with $|S|=\ell_0$ and a partial realization φ, let $P(S,\varphi)$ be the probability that the policy π^g chooses S as the first ℓ_0 seeds and φ is the feedback. That is, $P(S,\varphi)=\Pr_{\phi\sim F}\left(\mathcal{S}^{\mathfrak{f}}(\pi^g,\phi,\ell_0)=S\wedge\Phi^{\mathfrak{f}}_{G,D,\phi}(S)=\varphi\right)$. The mentioned expected marginal gain is

$$\begin{aligned}
&\sigma^{\mathfrak{f}}\left(\pi^g, \ell_0+1\right) - \sigma^{\mathfrak{f}}\left(\pi^g, \ell_0\right) \\
=&\sum_{S,\varphi:|S|=\ell_0} P(S,\varphi)\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup\{\pi^g(S,\varphi)\})\right|\right] - \mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\right) \\
\geq&\sum_{S,\varphi:|S|=\ell_0} P(S,\varphi)\cdot\frac{1}{k}\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S\cup S^*)\right|\right] - \mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\right) \quad \text{(Proposition 5.12)} \\
\geq&\sum_{S,\varphi:|S|=\ell_0} P(S,\varphi)\cdot\frac{1}{k}\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S^*)\right|\right] - \mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,D}(S)\right|\right]\right) \\
=&\frac{1}{k}\sigma(S^*) - \frac{1}{k}\sigma^{\mathfrak{f}}(\pi^g,\ell_0),
\end{aligned}$$

where the last equality follows from the law of total probability.

By rearranging the above inequality and the induction hypothesis,

$$\begin{aligned}
\sigma^{\mathfrak{f}}\left(\pi^g, \ell_0+1\right) &\geq \frac{1}{k}\sigma(S^*) + \frac{k-1}{k}\sigma^{\mathfrak{f}}\left(\pi^g, \ell_0\right) \\
&\geq \left(\frac{1}{k} + \frac{k-1}{k}\left(1-\left(1-\frac{1}{k}\right)^{\ell_0}\right)\right)\sigma(S^*) \\
&= \left(1-\left(1-\frac{1}{k}\right)^{\ell_0+1}\right)\sigma(S^*),
\end{aligned}$$

which concludes the inductive step. □

By taking $\ell = k$ and noticing that $1-(1-1/k)^k > 1-1/e$, it is easy to see that Proposition 5.13 implies Theorem 5.10.

Finally, putting Theorem 5.10, Lemma 5.8 and Lemma 5.9 together, Theorem 5.6 can be concluded easily.

Proof of Theorem 5.6. Since `ICM` and `LTM` are special cases of triggering models, we have

$$\inf_{G,D,k:\ I_{G,D}\text{ is }\texttt{ICM}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))} \geq \inf_{G,D,k} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}$$

and

$$\inf_{G,D,k:\ I_{G,D}\text{ is }\texttt{LTM}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))} \geq \inf_{G,D,k} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}.$$

Lemma 5.8 and Lemma 5.9 show that both

$$\inf_{G,D,k:\ I_{G,D}\text{ is }\texttt{ICM}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))} \qquad \text{and} \qquad \inf_{G,D,k:\ I_{G,D}\text{ is }\texttt{LTM}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}$$

are at most $1 - 1/e$. On the other hand, Theorem 5.10 implies

$$\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S^g(k))} \geq \frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S^*)} \geq 1 - \frac{1}{e}$$

for any triggering model $I_{G,D}$ and any k, where S^*, as usual, denotes the optimal seeds in the non-adaptive setting.

Putting together, Theorem 5.6 concludes for the full-adoption feedback model. Since all those inequalities hold for the myopic feedback model as well, Theorem 5.6 concludes for all feedback models. □

5.4 Supremum of Greedy Adaptivity Gap

In this section, we show that, for the full-adoption feedback model, both the adaptivity gap and the supremum of the greedy adaptivity gap are unbounded. As a result, in some cases, the adaptive version of the greedy algorithm can perform significantly better than its non-adaptive counterpart.

Theorem 5.14. *The greedy adaptivity gap with full-adoption feedback is unbounded: there exists a triggering model $I_{G,D}$ and k such that*

$$\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S^g(k))} = 2^{\Omega(\log\log|V|/\log\log\log|V|)}.$$

Theorem 5.15. *The adaptivity gap for the general triggering model with full-adoption feedback is infinity.*

In Sect. 5.4.1, we consider a variant of InfMax such that the seeds can only be chosen among a prescribed vertex set $\overline{V} \subseteq V$, where $\overline{V}$ is specified as an input to the InfMax instance. We show that, under this setting with `LTM`, both the adaptivity gap and the supremum of the greedy adaptivity gap with the full-adoption feedback model are unbounded (Lemma 5.18). Since it is common in practice that only a subset of nodes in a network is visible or accessible to the seed-picker, Lemma 5.18 is also interesting on its own. In Sect. 5.4.2, we show that how Lemma 5.18 can be used to prove Theorem 5.14 and Theorem 5.15. Notice that Theorem 5.14 and Theorem 5.15 hold for the standard InfMax setting without a prescribed set of seed candidates, but we do not know if they hold for `LTM` (instead, they are for the more general triggering model).

We first present the following lemma revealing a special additive property for `LTM`, which will be used later.

Lemma 5.16. *Suppose $I_{G,D}$ is `LTM`. If $U_1, U_2 \subseteq V$ with $U_1 \cap U_2 = \emptyset$ satisfy that there is no path from any vertices in U_1 to any vertices in U_2 and vice versa, then $\sigma(U_1) + \sigma(U_2) = \sigma(U_1 \cup U_2)$.*

Proof. For any seed set $S \subseteq V$, $\sigma(S)$ can be written as follows:

$$\sigma(S) = \sum_{\phi} \Pr(\phi \text{ is sampled}) \cdot \left| I_{G,D}^{\phi}(S) \right|. \tag{5.11}$$

For U_1 and U_2 in the lemma statement, since each vertex can only have at most one incoming live edge (in Definition 2.11, each T_v has size at most 1), under any realization ϕ, each vertex $v \in V \setminus (U_1 \cup U_2)$ that is reachable from vertices in $U_1 \cup U_2$ is reachable from either vertices in U_1 or vertices in U_2, but not both. Therefore, $|I_{G,D}^{\phi}(U_1)| + |I_{G,D}^{\phi}(U_2)| = |I_{G,D}^{\phi}(U_1 \cup U_2)|$ for any ϕ, and the lemma follows from considering the decomposition of $\sigma(U_1)$ and $\sigma(U_2)$ according to (5.11). □

5.4.1 On Linear Threshold Model with Prescribed Seed Candidates

Definition 5.17. The *influence maximization problem with prescribed seed candidates* is an optimization problem which takes as inputs $G = (V, E)$, D, $k \in \mathbb{Z}^+$, and $\overline{V} \subseteq V$, and outputs a seed set $S \subseteq \overline{V}$ that maximizes the expected total number of infections: $S \in \operatorname{argmax}_{S \subseteq \overline{V}: |S| \leq k} \sigma(S)$. The *adaptive influence maximization problem with prescribed seed candidates* has the same definition as it is in Definition 5.3, with the exception that the range of the function π is now $\overline{V}$, and Π is the set of all such policies.

Lemma 5.18. *For* InfMax *with prescribed seed candidates with `LTM` and the full-adoption feedback, the adaptivity gap is infinity, and the greedy adaptivity gap is $2^{\Omega(\log |V| / \log\log |V|)}$.*

Proof. For $d, W \in \mathbb{Z}^+$ with d being sufficiently large and $W \gg d^{d+1}$, we construct the following (adaptive) InfMax instance with prescribed seed candidates:

- the edge-weighted graph $G = (V, E, w)$ consists of an $(d+1)$-level directed full d-ary tree with the root node being the sink (i.e., an in-arborescence) and W vertices each of which is connected *from* the root node of the tree; the weight of each edge in the tree is $1/d$, and the weight of each edge connecting from the root to those W vertices is 1;

- the number of seeds is given by $k = 2(\frac{d+1}{2})^d$;
- the prescribed set for seed candidates $\overline{V}$ is the set of all the leaves in the tree.

Since the leaves are not reachable from one to another, Lemma 5.16 indicates that choosing any k vertices among $\overline{V}$, i.e., the leaves, infects the same number of vertices in expectation. It is easy to see that a single seed among the leaves will infect the root node with probability $1/d^d$, and those W vertices will be infected with probability 1 if the root of the tree is infected. Thus, for any seed set $S \subseteq \overline{V}$, by assuming all vertices in the tree are infected (in the sake of finding an upper bound for $\sigma(S)$), we have $\sigma(S) \leq \frac{1}{d^d} \cdot |S| \cdot W + \sum_{i=0}^{d} d^i < \frac{|S|W}{d^d} + d^{d+1}$. This gives an upper bound for the performance of both the non-adaptive greedy algorithm and the non-adaptive optimal seed set.

Now, we analyze the seeds chosen by the greedy adaptive policy. At a particular iteration when executing the greedy adaptive policy, we classify the internal tree nodes (i.e., the nodes that are neither leaves nor the root) into the following three types:

- Unexplored: the subtree rooted at this internal node contains no seed.
- Explored: the subtree rooted at this internal node contains seeds, and no edge in the path connecting this internal node to the root is known to be blocked (i.e., all edges in the path have statuses either `L` or `U`).
- Dead: if an edge in the path connecting this internal node to the root is known to be blocked.

Here we give some intuitions for the behavior of the greedy adaptive policy. Our objective is to infect the root, which will infect those W vertices that constitute most vertices of the graph. Before the root is infected, once a internal node is known to be "dead", the policy should never choose any seed from the leaves that are descendants of this node, as those seeds will never have a chance to infect the root (this explains our naming). Moreover, as we will see soon, the greedy adaptive policy will keep "exploring" an explored node before starting to "exploring" an unexplored node, until this explored node becomes dead.

We will show that, *if the root node is not infected yet, at any iteration of the greedy adaptive policy, each internal level of the tree can contain at most one explored* *node.* This is a formal statement describing what we meant just now by saying that we should keep exploring an explored node.

Firstly, since only one seed can be chosen at a single iteration, among all the nodes at a particular level of the tree, at most one of them can change the status from "unexplored" to "explored". Suppose for the sake of contradiction that, at a particular iteration of the greedy adaptive policy, an internal node v' which is previously unexplored become explored, while there is already another explored node v at the same level of v'. Suppose this is the first iteration we see two explored nodes at the same level. Let u be the least common ancestor of v and v'. Let ℓ_u be the level containing u. It is easy to see that all the nodes on the path from v to the root, which includes u, are explored (they cannot be unexplored, as the descendants of each of those nodes contains the descendants of v, which contain seeds; they cannot be dead, for otherwise v is dead). Since v and v' are the first pair of explored nodes at the same level, before the iteration where v' is explored, all nodes on the path between v' and u are unexplored (excluding u). Let d_u be the number of u's children that are not dead. Given the feedback from previous iterations, since all the descendants of v' and all the nodes on the path between v' and u (excluding u) are unexplored, the probability that a seed from a leaf that is a descendant of v' infects u is $\frac{1}{d^{\ell_u-1}\cdot d_u}$. On the other hand, if at this same iteration we pick a seed from a leaf which is a descendant of v and the path from this leaf to v contains no blocked edge, the probability that this seed infects u is at least $\frac{1}{d^{\ell_u-2}(d-1)d_u}$. This is because there is at least one dead node that is a descendant of v (we know that all the nodes on the path between v and the root are explored and uninfected, and we know that seeds have been chosen among the leaves on the subtree rooted at v; the only reason that those seeds have not made the root infected is that there are dead nodes that "block the cascade", and we know there is no dead node on the path between v and the root). Since the only way that a seed corresponding to either v or v' can infect the root is to first infect u and we have $\frac{1}{d^{\ell_u-2}(d-1)d_u} > \frac{1}{d^{\ell_u-1}\cdot d_u}$, the marginal influence of a seed corresponding to v' is smaller than the marginal influence of a seed corresponding to v. In other words, "exploring" v' provide less marginal influence than "exploring" v, which leads to the desired contradiction.

Next, we evaluate the expected number of seeds required to infect the root, under the greedy adaptive policy. Suppose the tree only has two levels (i.e., a star). The number of seeds among the leaves required to infect the root is a random variable with uniform distribution on $\{1, \ldots, d\}$, with expectation $\frac{d+1}{2}$. We will show that, by induction on the number of levels of the tree, with a d-level tree as it is in our case, the expected number of seeds required to infect the root is $(\frac{d+1}{2})^d$, which equals to $\frac{k}{2}$. Let $x_1, \ldots, x_d$ be the d children of the root node. By the claim we showed just now, at most one of $x_1, \ldots, x_d$ can be "explored" at any iteration. The greedy adaptive policy will do the following: it first explores one of $x_1, \ldots, x_d$, say, x_1; it will continue exploring x_1 until x_1 is dead or until the root is infected. The only situation that x_1 is dead is that x_1 is infected but the edge between x_1 and the root is blocked. Therefore, the greedy adaptive policy will attempt to infect $x_1, x_2, x_3, \ldots$ one by one, until one of those children infects the root. By the induction hypothesis, the expected number of seeds required to infect each of $x_1, \ldots, x_d$ is $(\frac{d+1}{2})^{d-1}$. Let X be the random variable indicating the smallest d' such that $x_{d'}$ is in the triggering set of the root (this means that the greedy adaptive policy will need to infect $x_1, \ldots, x_{d'}$ in order to infect the root). Then the expect number of seeds required to infect the root is

$$\sum_{d'=1}^{d} \left(\Pr(X = d') \cdot d' \left(\frac{d+1}{2} \right)^{d-1} \right) = \left(\frac{d+1}{2} \right)^{d},$$

where $d'(\frac{d+1}{2})^{d-1}$ is the expected number of seeds required to infected all of $x_1, \ldots, x_{d'}$ by the linearity of expectation.

After proving that the expected number of seeds required to infect the root is $(\frac{d+1}{2})^d = \frac{k}{2}$, by Markov's inequality, the k seeds chosen according to the greedy adaptive policy will infect the root with probability at least 1/2. Therefore, $\sigma^{\mathfrak{f}}(\pi^g, k) \geq \frac{1}{2}W$, and the optimal adaptive policy can only be better: $\max_{\pi \in \Pi} \sigma^{\mathfrak{f}}(\pi, k) \geq \sigma^{\mathfrak{f}}(\pi^g, k) \geq \frac{1}{2}W$.

Putting together, both the adaptivity gap and the supremum of the greedy adaptivity gap is at least

$$\frac{\frac{1}{2}W}{\frac{kW}{d^d} + d^{d+1}} = \frac{\frac{1}{2}W}{\frac{1}{2^{d-1}}(1 + \frac{1}{d})^d W + d^{d+1}.} = \Omega\left(2^d\right),$$

if setting $W = d^{d+10} \gg d^{d+1}$. The lemma concludes by noticing $d = \Omega(\frac{\log |V|}{\log\log |V|})$ (in particular, $|V| = W + o(W) = d^{d+10} + o(d^{d+10})$, so $\log |V| = d \log d + o(d \log d)$, $\log\log |V| = \log d + o(\log d)$, and $d = \Omega(\frac{\log |V|}{\log\log |V|})$). □

5.4.2 Proof of Theorem 5.14, 5.15

To prove Theorem 5.14 and Theorem 5.15, we construct an InfMax instance with a special triggering model $I_{G,D}$ which is a combination of `ICM` and `LTM`.

Definition 5.19. The *mixture of* `ICM` *and* `LTM` is a triggering model $I_{G,D}$ where $G = (V, E, w)$ is an edge-weighted graph with $w(u, v) \in (0, 1]$ for each $(u, v) \in E$ and each vertex v is labelled either **IC** or **LT** such that T_v is sampled according to $\mathcal{F}_v$ described in Definition 2.7 if v is labelled **IC** and T_v is sampled according to $\mathcal{F}_v$ described in Definition 2.11 if v is labelled **LT**. In addition, each vertex v labelled `L` satisfies $\sum_{u\in\Gamma(v)} w(u, v) \leq 1$.

To conclude Theorem 5.14 and Theorem 5.15, we construct an edge-weighted graph $G = (V, E, w)$ on which the greedy adaptive policy significantly outperforms the non-adaptive greedy algorithm. Let $M \gg d^{d+1}$ be a large integer. We reuse the graph with a tree and W vertices in the proof of Lemma 5.18. We create M such graphs and name them $T_1, \ldots, T_M$. Let $L = d^d$ be the number of leaves in each T_i. Let $\mathbb{Z}_L = \{1, \ldots, L\}$ and $\mathbb{Z}_L^M$ be the set of all M-dimensional vectors whose entries are from $\mathbb{Z}_L$. For each $\mathbf{z} = (z_1, \ldots, z_M) \in \mathbb{Z}_L^M$, create a vertex $a_{\mathbf{z}}$ and create a directed edge from $a_{\mathbf{z}}$ to the z_i-th leaf of the tree T_i for each $i = 1, \ldots, M$. The weight of each such edge is 1. Let $A = \{a_{\mathbf{z}} \mid \mathbf{z} \in \mathbb{Z}_L^M\}$. Notice that $|A| = L^M$ and each $a_{\mathbf{z}} \in A$ is connected to M vertices from $T_1, \ldots, T_M$ respectively. The leaves of $T_1, \ldots, T_M$ are labelled as **IC**, and the remaining vertices are labelled as **LT**. Finally, set $k = 2(\frac{d+1}{2})^d$ as before.

Due to that M is large, it is more beneficial to seed a vertex in A than a vertex elsewhere. In particular, seeding a root in certain T_i infects W vertices, while seeding a vertex in A will infects $M \cdot (\frac{1}{d})^d W \gg W$ vertices in expectation.

It is easy to see that, in the non-adaptive setting, the optimal seeding strategy is to choose k seeds from A such that they do not share any out-neighbors, in which case the k chosen seeds will cause the infection of exactly k leaves in each T_i. This is also what the non-adaptive greedy algorithm will do. As before, to find an upper bound for any seed set S with $|S| = k$, we assume that all vertices in each T_i are infected, and we have $\sigma(S) \leq M\left(k \cdot \frac{1}{d^d} W + \sum_{i=0}^d d^i\right)$.

By the same analysis in the proof of Lemma 5.18, by choosing k seeds among A as described above, which is equivalent as choosing k leaves in each of $T_1, \ldots, T_M$ simultaneously, the root in each T_i is infected with probability at least $\frac{1}{2}$. Therefore, the expected total number of infected vertices is at least $M \cdot \frac{1}{2} W$.

It may seem problematic that the greedy adaptive policy may start to seed the roots among $T_1, \ldots, T_M$ when it sees that there are already a lot of infected roots (so seeding a root is better than seed a vertex in A). However, since $M \gg d^{d+1}$, by simple calculations, this can only happen when there are already $(1 - o(1))M$ trees with infected roots, in which case the number of infected vertices is already much more than $M \cdot \frac{1}{2}W$.

Putting together as before, both the adaptivity gap and the supremum of the greedy adaptivity gap is at least

$$\frac{M \cdot \frac{1}{2}W}{M(\frac{kW}{d^d} + d^{d+1})} = \frac{\frac{1}{2}W}{\frac{1}{2^{d-1}}(1+\frac{1}{d})^d W + d^{d+1}.} = \Omega\left(2^d\right),$$

if fixing $W = d^{d+10} \gg d^{d+1}$. Theorem 5.15 concludes by letting $d \to \infty$. To conclude Theorem 5.14, we need to show that $d = \Omega(\log\log|V|/\log\log\log|V|)$. To see this, we set $M = d^{d+10}$ which is sufficiently large for our purpose. Since we have $L = d^d$, we have $|V| = L^M + o(L^M) = d^{d^{d+11}} + o(d^{d^{d+11}})$, which implies $d = \Omega(\log\log|V|/\log\log\log|V|)$.

5.5 Greedy Algorithms in Practice and Robustness of Our Results

Recall that, in the greedy algorithm, we find a vertex s that maximizes the marginal gain of the influence $\sigma(S \cup \{s\}) - \sigma(S)$ in each iteration. However, in practice, the greedy algorithm is implemented with $\sigma(\cdot)$ estimated by Monte-Carlo method, reverse reachable sets coverage (see Sect. 2.2.2 for details), or other *randomized approximation algorithms*. As a result, in reality, when implementing the greedy algorithm, a vertex s that *approximately* maximizes the marginal gain of the influence is found in each iteration *with high probability*. In this section, we discuss the applicability of all our results in previous sections under this approximation setting. In Sect. 5.5.1, We first define the (ε, δ)-*greedy algorithm* where in each iteration a vertex s that approximately maximizes the marginal gain $\sigma(S \cup \{s\}) - \sigma(S)$ within factor $(1-\varepsilon)$ is found with probability at least $(1-\delta)$, which captures the practical implementations of greedy algorithms. In Sect. 5.5.2, we discuss the robustness of our results by studying under what ε and δ our results hold.

5.5.1 (ε, δ)-Greedy Algorithms

Definition 5.20. An (ε, δ)-*greedy algorithm* is a randomized iterative algorithm that satisfies the following:

1. the algorithm initializes $S = \emptyset$;

2. for each of the k iterations, with probability at least $1-\delta$, the algorithm finds $s \in V$ such that

$$\sigma(S \cup \{s\}) - \sigma(S) \geq (1-\varepsilon) \max_{s' \in V} \left(\sigma(S \cup \{s'\}) - \sigma(S)\right),$$

and update $S \leftarrow S \cup \{s\}$;

3. the algorithm outputs S.

Since an (ε, δ)-greedy algorithm is an approximation version of the "exact" greedy algorithm, it achieves an approximation ratio that is close to $(1-1/e)$. The proof is standard, and we include it here for completeness.

Theorem 5.21. *For any $\varepsilon \leq \frac{1}{k}$, an $(\varepsilon, \delta/k)$-greedy algorithm gives a $(1-1/e-\varepsilon)$-approximation for submodular* InfMax *with probability $(1-\delta)$.*

Proof. Let $S^* = \{s_1^*, \ldots, s_k^*\}$ be an optimal solution which maximizes $\sigma(\cdot)$, and let $S = \{s_1, \ldots, s_k\}$ be the seed set output by any $(\varepsilon, \delta/k)$-greedy algorithm with $\varepsilon \leq 1/k$. Let $S_i^* = \{s_1^*, \ldots, s_i^*\}$ and $S_i = \{s_1, \ldots, s_i\}$. In particular, let $S_0^* = S_0 = \emptyset$. Similar to Proposition 5.12, we will show that, for each $i = 0, 1, \ldots, k-1$,

$$\sigma(S_{i+1}) - \sigma(S_i) \geq \frac{1-\varepsilon}{k}(\sigma(S_i \cup S^*) - \sigma(S_i)) \qquad \text{with probability at least } 1 - \frac{\delta}{k}. \tag{5.12}$$

This is because

$$\begin{aligned}
\sigma(S_{i+1}) - \sigma(S_i) &\geq (1-\varepsilon)\left(\max_{s \in V}\left(\sigma(S_i \cup \{s\}) - \sigma(S_i)\right)\right) \\
&\qquad\qquad\qquad\qquad \text{(by definition of } (\varepsilon,\delta)\text{-greedy)} \\
&\geq (1-\varepsilon)\frac{1}{k}\sum_{j=1}^{k}\left(\sigma(S_i \cup \{s_j^*\}) - \sigma(S_i)\right) \quad \text{(since } s \text{ is the maximizer)} \\
&\geq \frac{1-\varepsilon}{k}\sum_{j=1}^{k}\left(\sigma(S_i \cup S_j^*) - \sigma(S_i \cup S_{j-1}^*)\right) \quad \text{(submodularity of } \sigma(\cdot)\text{)} \\
&= \frac{1-\varepsilon}{k}\left(\sigma(S_i \cup S^*) - \sigma(S_i)\right). \qquad \text{(telescoping sum)}
\end{aligned}$$

Next, similar to Proposition 5.13, we can prove by induction that for each $i = 1, \ldots, k$

$$\sigma(S_i) \geq \left(1 - \left(1 - \frac{1}{k}\right)^i - \varepsilon\right)\sigma(S^*) \qquad \text{with probability at least } 1 - \frac{i\delta}{k}. \tag{5.13}$$

The base step for $i = 1$ follows from Eqn. (5.12):

$$\sigma(S_1) = \sigma(S_1) - \sigma(S_0) \geq \frac{1-\varepsilon}{k}(\sigma(S_0 \cup S^*) - \sigma(S_0)) > \left(\frac{1}{k} - \varepsilon\right)\sigma(S^*).$$

For the inductive step, by Eqn. (5.12) again, we have, with probability at least $(1 - \delta/k)$,

$$\sigma(S_{i+1}) - \sigma(S_i) \geq \frac{1-\varepsilon}{k}(\sigma(S_i \cup S^*) - \sigma(S_i)) \geq \frac{1-\varepsilon}{k}(\sigma(S^*) - \sigma(S_i)),$$

which implies

$$\sigma(S_{i+1}) \geq \frac{1-\varepsilon}{k}\sigma(S^*) + \frac{k-1+\varepsilon}{k}\sigma(S_i).$$

By the induction hypothesis, with probability at least $(1 - \frac{i\delta}{k})$, we have

$$\sigma(S_i) \geq \left(1 - \left(1 - \frac{1}{k}\right)^i - \varepsilon\right)\sigma(S^*).$$

Putting together by a union bound, with probability at least $1 - \frac{(i+1)\delta}{k}$, we have

$$\begin{aligned}
\sigma(S_{i+1}) &\geq \frac{1-\varepsilon}{k}\sigma(S^*) + \frac{k-1+\varepsilon}{k}\left(1 - \left(1 - \frac{1}{k}\right)^i - \varepsilon\right)\sigma(S^*) \\
&= \left(1 - \left(1 - \frac{1}{k}\right)^{i+1} - \varepsilon + \frac{\varepsilon}{k}\left(1 - \varepsilon - \left(1 - \frac{1}{k}\right)^i\right)\right)\sigma(S^*) \\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{(elementary calculations)} \\
&\geq \left(1 - \left(1 - \frac{1}{k}\right)^{i+1} - \varepsilon\right)\sigma(S^*), \qquad \text{(since } \varepsilon \leq \tfrac{1}{k} \text{ and } i \geq 1\text{)}
\end{aligned}$$

which concludes the inductive step.

The theorem concludes by taking $i = k$ in Eqn. (5.13) and noticing that $1 - (1 - 1/k)^k > 1 - 1/e$. □

We can define the (ε, δ)-*greedy adaptive policy* similarly.

Definition 5.22. An adaptive policy π is a (ε, δ)-*greedy adaptive policy* if, for any $S \subseteq V$ and any partial realization φ, with probability at least $1-\delta$, we have $\pi(S, \varphi) = v$ for v such that

$$\mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S \cup \{v\})\right| - \left|I^{\phi}_{G,D}(S)\right|\right] \geq (1-\varepsilon)\max_{s \in V}\mathop{\mathbb{E}}_{\phi \simeq \varphi}\left[\left|I^{\phi}_{G,D}(S \cup \{s\})\right| - \left|I^{\phi}_{G,D}(S)\right|\right].$$

5.5.2 Greedy Adaptivity Gap for (ε, δ)-Greedy Algorithms

We re-exam all the theorems and lemmas in Sect. 5.3 and Sect. 5.4. Throughout this section, we use $\mathcal{A}^g_{\varepsilon,\delta}$ to denote the set of all (ε, δ)-greedy algorithms and $\Pi^g_{\varepsilon,\delta}$ to denote the set of all (ε, δ)-greedy adaptive policies. Since the greedy adaptive policy we are studying now is randomized, for any (ε, δ)-greedy adaptive policy $\pi^g \in \Pi^g_{\varepsilon,\delta}$, the values $\sigma^{\mathfrak{f}}(\pi^g, k)$ and $\sigma^{\mathrm{m}}(\pi^g, k)$ are the expected numbers of infected vertices under the full-adoption feedback setting and the myopic feedback setting respectively, where the expectation is taken over *both* the sampling of a realization *and the randomness when implementing* π^g. Correspondingly, for a randomized non-adaptive (ε, δ)-greedy algorithm $A^g \in \mathcal{A}^g_{\varepsilon,\delta}$, we will slightly abuse the notation and use $\sigma(A^g, k)$ to denote the expected number of infected vertices when k seeds are chosen based on algorithm A^g, where the expectation is again taken over both the sampling of a realization and the randomness when implementing A^g.

The argument behind all the proofs in this section is the same, which we summarize as follows. To show that the greedy adaptivity gap remains the same in the (ε, δ)-greedy setting, we find an ε that is small enough such that, in the InfMax instance we constructed, requiring the marginal influence being at least a $(1 - \varepsilon)$ fraction of the maximum marginal influence is the same as requiring the maximum marginal influence. This is done by setting ε to be small enough such that the only seed that produces the marginal influence within $(1 - \varepsilon)$ of the maximum marginal influence is the seed that produce the maximum marginal influence. By definition, the (ε, δ)-greedy algorithm/policy will behave exactly the same as their exact deterministic counterpart with probability at least $1 - \delta$. By setting $\delta = o(1/k)$ and taking a union bound over all the k iterations, with probability at least $1 - o(1)$, the (ε, δ)-greedy algorithm/policy will behave the same way as the exact deterministic greedy algorithm/policy.

5.5.2.1 Infimum of Greedy Adaptivity Gap for (ε, δ)-Greedy Algorithms

In this section, we show that, for $\varepsilon = o(1/k)$ and $\delta = o(1/k)$, the infimum of the greedy adaptivity gap for (ε, δ)-greedy algorithms is between $1 - 1/e - \varepsilon$ and $1 - 1/e$. We will formally state what exactly we mean by this, and we will prove this by showing that Lemma 5.8, Lemma 5.9 and Theorem 5.10 can be adapted in the (ε, δ)-greedy setting.

The following lemma extends Lemma 5.8 to the (ε, δ)-greedy setting.

Lemma 5.23. *Given any two functions $\varepsilon : \mathbb{Z}^+ \to \mathbb{R}^+$ and $\delta : \mathbb{Z}^+ \to \mathbb{R}^+$ satisfying $\varepsilon(k) = o(1/k)$ and $\delta(k) = o(1/k)$, for any $\tau > 0$, there exists G, D, k such that $I_{G,D}$ is an* `ICM` *and, for any adaptive policy $\pi^g \in \Pi^g_{\varepsilon(k),\delta(k)}$ and any non-adaptive algorithm $A^g \in \mathcal{A}^g_{\varepsilon(k),\delta(k)}$, we have*

$$\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(A^g, k)} \leq 1 - \frac{1}{e} + \tau \qquad \textit{and} \qquad \frac{\sigma^{\mathfrak{m}}(\pi^g, k)}{\sigma(A^g, k)} \leq 1 - \frac{1}{e} + \tau.$$

Proof. We construct the same InfMax instance $(G = (V, E, w), k + 1)$ as it is in the proof of Lemma 5.8, with only one change: set $\Upsilon = \varepsilon(k+1) \cdot W$ instead of the previous setting $\Upsilon = W/k^2$.[1] Notice that $\varepsilon(k + 1)$ means the value of the function $\varepsilon(\cdot)$ with input $k + 1$, not ε times $(k + 1)$. To avoid possible confusion, we write $\varepsilon := \varepsilon(k + 1)$ and $\delta := \delta(k + 1)$ for this proof. The remaining part of the proof is an adaption of the proof of Lemma 5.8 to the (ε, δ) setting.

We first show that, for any $A^g \in \mathcal{A}^g_{\varepsilon,\delta}$, with probability at least $1 - (k + 1) \cdot \delta = 1 - o(1)$, A^g will output $\{s, u_1, \ldots, u_k\}$.

From Eqn. (5.1), (5.2), (5.3) and (5.4), by the definition of (ε, δ)-greedy, with probability at least $1 - \delta$, the first seed chosen must have expected influence at least $(1 - \varepsilon)\sigma(\{s\}) \geq (1 - o(1/k)) \cdot 2W$. Since any other vertex does not have an influence which is even close to $2W$, the first seed chosen by A^g is s with probability at least $1 - \delta$.

Next, we show that, if A^g has chosen s and i vertices from $\{u_1, \ldots, u_k\}$ after $i + 1$ iterations, A^g will choose the next seed from $\{u_1, \ldots, u_k\}$ with probability $1 - \delta$. Let $U_i = \{s, u_1, \ldots, u_i\}$ and $U_0 = \{s\}$ as before. Without loss of generality, we only need to show that, supposing A^g has chosen U_i as the first $(i + 1)$ seeds, with probability at least $1 - \delta$, A^g will choose a vertex from $\{u_{i+1}, \ldots, u_k\}$ as the next seed. By our calculation in Eqn. (5.7), (5.8) and (5.9), with probability at least $1 - \delta$, A^g will choose a seed x such that

$$\sigma(U_i \cup \{x\}) - \sigma(U_i) \geq (1 - \varepsilon)(\sigma(U_i \cup \{u_{i+1}\}) - \sigma(U_i))$$

$$> W + (4k - 2)\Upsilon - 1.1\varepsilon W = W + (4k - 3.1)\Upsilon,$$

where the last inequality is due to $\varepsilon(W + (4k-2)\Upsilon) = \varepsilon(W + o(W)) < 1.1\varepsilon W = 1.1\Upsilon$. On the other hand, from Eqn. (5.8) and (5.9), choosing v_1 or v_2 as the next seed does

[1] If $\varepsilon(k+1) \cdot W$ is not an integer, we can always make W large enough and find a positive rational number $\varepsilon' < \varepsilon(k + 1)$ such that $\varepsilon' W \in \mathbb{Z}^+$. The remaining part of the proof will not be invalidated by this change.

not provide enough marginal gain to $\sigma(\cdot)$:

$$\begin{aligned}\sigma(U_i \cup \{v_1\}) - \sigma(U_i) &\leq 1 + W + (4k-4)\Upsilon < (1-\varepsilon)(\sigma(U_i \cup \{u_{i+1}\}) - \sigma(U_i)),\\ \sigma(U_i \cup \{v_2\}) - \sigma(U_i) &\leq 1 + W - \frac{W}{k} + (4k+5)\Upsilon = W + 4k\Upsilon - \omega(\Upsilon)\\ &\qquad\qquad\qquad\qquad\qquad \left(\text{since } \Upsilon = \varepsilon W = o\left(\tfrac{W}{k}\right)\right)\\ &\qquad\qquad < (1-\varepsilon)(\sigma(U_i \cup \{u_{i+1}\}) - \sigma(U_i)),\end{aligned}$$

and the marginal influence of the remaining vertices other than v_1 and v_2 are even smaller. Therefore, we conclude that, with probability at least $1-\delta$, A^g will choose a vertex from $\{u_{i+1}, \ldots, u_k\}$ as the next seed.

Putting together, by a union bound, with probability $1-(k+1)\delta = 1-o(1)$, A^g will choose $\{s, u_1, \ldots, u_k\}$, which will infected $(k+2)W + o(W)$ vertices in expectation, as calculated in the proof of Lemma 5.8. Therefore,

$$\sigma(A^g, k+1) \geq (1-(k+1)\delta)((k+2)W + o(W)) + (k+1)\delta \cdot 0 = kW - o(kW).$$

Second, we show that, for any greedy adaptive policy $\pi^g \in \Pi^g_{\varepsilon,\delta}$, π^g will choose $\{s, v_1, \ldots, v_k\}$ with probability at least $1 - 1/k - (k+1)\delta = 1 - o(1)$. By the same analysis in the non-adaptive case, with probability $(1-\delta)$, the first seed chosen by π^g is s. We assume that s fails to infect t which happens with probability $1 - 1/k$, and this is given as the feedback to π^g. The marginal influence of v_1 is then $\overline{w}(t) + \overline{w}(v_1) + \sum_{i=1}^k \overline{w}(w_{1i}) = 4k\Upsilon + 1 + W$. With probability at least $1-\delta$, π^g will choose a second seed with marginal influence at least $(1-\varepsilon)(4k\Upsilon + 1 + W) > W + 4k\Upsilon - \varepsilon \cdot 1.1W = W + (4k - 1.1)\Upsilon$. It is easy to see that v_1 is the only vertex that can provide enough marginal influence. In particular, the marginal influence of each of $u_1, \ldots, u_k$ is $1 + W + (4k-2)\Upsilon$, which is less than $W + (4k-1.1)\Upsilon$, the marginal influence of v_2 is $1 + (1 - \frac{1}{k})W + k \cdot \frac{4k-2}{k-1}\Upsilon = W + 4k\Upsilon - \omega(\Upsilon)$ (notice that $k \cdot \frac{4k-2}{k-1}\Upsilon = 4k\Upsilon + \Theta(\Upsilon)$ and $W/k = \omega(\Upsilon)$), which is less than $W + (4k-1.1)\Upsilon$, and the marginal influence of $v_3, \ldots, v_k$ are even smaller than that of v_2.

We have shown that the first two seeds are s and v_1 with probability at least $1 - 1/k - 2\delta$. Next, we show that, for each $i = 1, \ldots, k-1$, if π^g has chosen $s, v_1, \ldots, v_i$, with probability $(1-\delta)$, π^g will choose v_{i+1} as the next seed. Suppose π^g has chosen $s, v_1, \ldots, v_i$. From the proof of Lemma 5.8, we have seen that v_{i+1} has the highest marginal influence, which is $1 + (1-1/k)^i W + k \cdot \frac{4k-2}{k-1}\Upsilon$. With probability at least $1-\delta$, π^g will choose a seed with marginal influence at least $(1-\varepsilon)$ fraction of this value:

$$\begin{aligned}
&(1-\varepsilon)\left(1+\left(1-\frac{1}{k}\right)^i W + k\frac{4k-2}{k-1}\Upsilon\right)\\
&> \left(1-\frac{1}{k}\right)^i W + k\frac{4k-2}{k-1}\Upsilon - \varepsilon\left(\left(1-\frac{1}{k}\right)^i W + k\frac{4k-2}{k-1}\Upsilon\right)\\
&\geq \left(1-\frac{1}{k}\right)^i W + k\frac{4k-2}{k-1}\Upsilon - \varepsilon\left(\left(1-\frac{1}{k}\right)^1 W + k\frac{4k-2}{k-1}\Upsilon\right)\\
&\geq \left(1-\frac{1}{k}\right)^i W + k\frac{4k-2}{k-1}\Upsilon - 1.1\Upsilon.\\
&\quad\text{(since } \varepsilon(1-\tfrac{1}{k})^1 W < \varepsilon W = \Upsilon \text{ and } \varepsilon k\tfrac{4k-2}{k-1}\Upsilon = \Theta(4k\varepsilon\Upsilon) = o(\Upsilon) < 0.1\Upsilon)
\end{aligned}$$

The marginal influence of v_{i+2} is $(1-1/k)^{i+1}W + k\frac{4k-2}{k-1}\Upsilon = (1-1/k)^i W + k\frac{4k-2}{k-1}\Upsilon - \frac{1}{k}(1-1/k)^i W < (1-1/k)^i W + k\frac{4k-2}{k-1}\Upsilon - 0.63\frac{W}{k}$, which is less than the value above (as $\Upsilon = o(W/k)$). The marginal influence of $v_{i+3}, \ldots, v_k$ are even smaller, and we do not need to consider them. The marginal influence of u_1 is $\sum_{j=i+1}^{k} \overline{w}(w_{j1}) = (1-1/k)^i W + (k-i)\cdot\frac{4k-2}{k-1}\Upsilon \leq (1-1/k)^i W + (k-1)\cdot\frac{4k-2}{k-1}\Upsilon < (1-1/k)^i W + k\frac{4k-2}{k-1}\Upsilon - 2\Upsilon$, which is again less than the value above.

Putting together, by a union bound, with probability $(1-1/k)(1-(k+1)\delta) = o(1)$, π^g will choose $\{s, v_1, \ldots, v_k\}$, which can infect, as computed in the proof of Lemma 5.8, $k(1-(1-1/k)^k)W + o(kW)$ vertices. Therefore,

$$\sigma^{\mathfrak{f}}(\pi^g, k) = \sigma^{\mathfrak{m}}(\pi^g, k) \leq (1-o(1))\left(k\left(1-\left(1-\frac{1}{k}\right)^k\right)W + o(kW)\right) + o(1)\cdot|V|$$

$$= k\left(1-\left(1-\frac{1}{k}\right)^k\right)W + o(kW).$$

The theorem concludes by taking the ratio of the computed upper-bound of $\sigma^{\mathfrak{f}}(\pi^g, k) = \sigma^{\mathfrak{m}}(\pi^g, k)$ and the computed lower-bound of $\sigma(A^g, k)$, and then taking the limits $k \to \infty$ and $W \to \infty$. □

Remark 5.24. Lemma 5.23 provides an upper bound for each of $\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}$ and $\frac{\sigma^{\mathfrak{m}}(\pi^g,k)}{\sigma(A^g,k)}$, where the numerator and the denominator in each ratio represent the number of infected vertices (in the non-adaptive setting and the adaptive setting respectively) *in expectation*. The same proof for Lemma 5.23 can be used to show the following stronger version of Lemma 5.23. Let $I_{G,D}^{\phi}(\pi^g, k)$ be the set of infected vertices when the adaptive policy π^g is used and the underlying live-edge realization is ϕ. Let

$I^{\phi}_{G,D}(A^g, k)$ have similar meaning corresponding to non-adaptive algorithm A^g. The same proof for Lemma 5.23 implies that, under the same setting in Lemma 5.23, for both full-adoption and myopic feedback models, we have

$$\Pr_{\phi\sim F}\left(\frac{|I^{\phi}_{G,D}(\pi^g,k)|}{|I^{\phi}_{G,D}(A^g,k)|}\leq 1-\frac{1}{e}+\tau\right)\geq 1-2(k+1)\delta-\frac{1}{k}=1-o(1), \tag{5.14}$$

where we have taken a union bound of the "bad" events that the "correct" seed is not chosen in each of the $(k+1)$ iterations in both π^g and A^g, as well as the "bad" event that s infects t (with probability $1/k$).

The following lemma extends Lemma 5.9 to the (ε, δ)-greedy setting.

Lemma 5.25. *Given any two functions $\varepsilon : \mathbb{Z}^+ \to \mathbb{R}^+$ and $\delta : \mathbb{Z}^+ \to \mathbb{R}^+$ satisfying $\varepsilon(k) = o(1/k)$ and $\delta(k) = o(1/k)$, for any $\tau > 0$, there exists G, D, k such that $I_{G,D}$ is an `LTM` and, for any adaptive policy $\pi^g \in \Pi^g_{\varepsilon(k),\delta(k)}$ and any non-adaptive algorithm $A^g \in \mathcal{A}^g_{\varepsilon(k),\delta(k)}$, we have*

$$\frac{\sigma^{\mathsf{f}}(\pi^g,k)}{\sigma(A^g,k)}\leq 1-\frac{1}{e}+\tau \qquad \text{and} \qquad \frac{\sigma^{\mathsf{m}}(\pi^g,k)}{\sigma(A^g,k)}\leq 1-\frac{1}{e}+\tau.$$

Proof. We construct the same InfMax instance $(G = (V, E, w), k + 1)$ as it is in the proof of Lemma 5.9, with only one change: set $\Upsilon = \varepsilon \cdot W$ instead of the previous setting $\Upsilon = W/k^2$. The remaining part of the proof is exactly the same as how we have adapted the proof of Lemma 5.8 to Lemma 5.23. We omit the details here. □

Remark 5.26. Similarly, we can prove a stronger version of Lemma 5.25 given by exactly the same equation (5.14).

Next, it is easy to show that Theorem 5.10 holds for the (ε, δ)-greedy setting.

Theorem 5.27. *For a triggering model $I_{G,D}$, any $\varepsilon \in (0, 1/k]$, any function $\delta : \mathbb{Z}^+ \to \mathbb{R}^+$ such that $\delta(k) = o(1/k)$, and any $\pi^g \in \Pi^g_{\varepsilon,\delta(k)}$, we have*

$$\sigma^{\mathsf{f}}(\pi^g,k)\geq\left(1-\frac{1}{e}-\varepsilon\right)\max_{S\subseteq V,|S|\leq k}\sigma(S) \text{ and } \sigma^{\mathsf{m}}(\pi^g,k)\geq\left(1-\frac{1}{e}-\varepsilon\right)\max_{S\subseteq V,|S|\leq k}\sigma(S).$$

Proof. Let $S^* \in \operatorname{argmax}_{S\subseteq V,|S|\leq k}\sigma(S)$ be an optimal non-adaptive seed set. For any $S \subseteq V$, any partial realization ϕ that is a valid feedback of S under any feedback model (either full-adoption or myopic), letting $s^* \in \operatorname{argmax}_{s'} \mathbb{E}_{\phi\simeq\varphi}[|I^{\phi}_{G,D}(S\cup\{s'\})|]$ be the vertex which maximizes the expected marginal influence given φ, with probability at least $1-\delta$, π^g will pick the next seed $s=\pi^g(S,\varphi)$ such that

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I_{G,D}^{\phi}(S\cup\{s\})\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I_{G,D}^{\phi}(S)\right|\right]\\
\geq&(1-\varepsilon)\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I_{G,D}^{\phi}(S\cup\{s^*\})\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I_{G,D}^{\phi}(S)\right|\right]\right) && \text{(Definition 5.22)}\\
\geq&\frac{1-\varepsilon}{k}\left(\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I_{G,D}^{\phi}(S\cup S^*)\right|\right]-\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I_{G,D}^{\phi}(S)\right|\right]\right). && \text{(Proposition 5.12)}
\end{aligned}$$

We can then prove that, for any $\ell\in\mathbb{Z}^+$, both $\sigma^{\mathfrak{f}}(\pi^g,\ell)$ and $\sigma^{\mathfrak{m}}(\pi^g,\ell)$ are no less than $(1-(1-1/k)^{\ell}-\varepsilon)\sigma(S^*)$, by using the same arguments in the proof of Proposition 5.13. The theorem concludes by taking $\ell=k$. □

Finally, we formally state and prove that, when $\varepsilon=o(1/k)$ and $\delta=o(1/k)$, the infimum of the adaptivity gap under the (ε,δ)-greedy setting is between $1-1/e-\varepsilon$ and $1-1/e$, which adapts Theorem 5.6 to the (ε,δ)-greedy setting.

Theorem 5.28. *Given any two functions $\varepsilon:\mathbb{Z}^+\to\mathbb{R}^+$ and $\delta:\mathbb{Z}^+\to\mathbb{R}^+$ such that $\varepsilon(k)=o(1/k)$ and $\delta(k)=o(1/k)$, we have*

$$\begin{aligned}
\inf_{G,D,k:I_{G,D}\textit{ is }\texttt{ICM}}\left(\inf_{\pi^g\in\Pi^g_{\varepsilon(k),\delta(k)},A^g\in\mathcal{A}^g_{\varepsilon(k),\delta(k)}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}\right)&\in\left(1-\frac{1}{e}-\varepsilon(k),1-\frac{1}{e}\right),\\
\inf_{G,D,k:I_{G,D}\textit{ is }\texttt{ICM}}\left(\sup_{\pi^g\in\Pi^g_{\varepsilon(k),\delta(k)},A^g\in\mathcal{A}^g_{\varepsilon(k),\delta(k)}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}\right)&\in\left(1-\frac{1}{e}-\varepsilon(k),1-\frac{1}{e}\right),\\
\inf_{G,D,k:I_{G,D}\textit{ is }\texttt{LTM}}\left(\inf_{\pi^g\in\Pi^g_{\varepsilon(k),\delta(k)},A^g\in\mathcal{A}^g_{\varepsilon(k),\delta(k)}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}\right)&\in\left(1-\frac{1}{e}-\varepsilon(k),1-\frac{1}{e}\right),\\
\inf_{G,D,k:I_{G,D}\textit{ is }\texttt{LTM}}\left(\sup_{\pi^g\in\Pi^g_{\varepsilon(k),\delta(k)},A^g\in\mathcal{A}^g_{\varepsilon(k),\delta(k)}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}\right)&\in\left(1-\frac{1}{e}-\varepsilon(k),1-\frac{1}{e}\right),\\
\inf_{G,D,k}\left(\inf_{\pi^g\in\Pi^g_{\varepsilon(k),\delta(k)},A^g\in\mathcal{A}^g_{\varepsilon(k),\delta(k)}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}\right)&\in\left(1-\frac{1}{e}-\varepsilon(k),1-\frac{1}{e}\right),\\
\inf_{G,D,k}\left(\sup_{\pi^g\in\Pi^g_{\varepsilon(k),\delta(k)},A^g\in\mathcal{A}^g_{\varepsilon(k),\delta(k)}}\frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(A^g,k)}\right)&\in\left(1-\frac{1}{e}-\varepsilon(k),1-\frac{1}{e}\right).
\end{aligned}$$

All the six statements above also hold for the myopic feedback model.

Proof. Since $\sigma(A^g,k)\leq\max_{S\subseteq V,|S|\leq k}\sigma(S)$ always hold, the lower bound $1-\frac{1}{e}-\varepsilon$ holds for each of the six statements according to Theorem 5.27. Then, Lemma 5.23 implies the first two statements, Lemma 5.25 implies the third and the fourth. Finally, since both ICM and LTM are special cases of the triggering model, the left-hand side of the fifth statement is at most the left-hand side of the first (or the third), and the left-hand side of the sixth statement is at most the left-hand side of the second (or the fourth). Thus, the first four statements imply the last two. □

5.5.2.2 Supremum of Greedy Adaptivity Gap for (ε, δ)-Greedy Algorithm

We first show that Lemma 5.18 holds for the (ε, δ)-greedy setting for very mild restrictions on ε and δ. the following lemma shows that, under the (ε, δ)-greedy setting where $\varepsilon = O(\log\log k/\log k)$ and $\delta = o(1/k)$, the greedy adaptivity gap for InfMax with prescribed seed candidates with LTM and full-adoption feedback is $2^{\Omega(\log|V|/\log\log|V|)}$.[2] Notice that Lemma 5.29 below is a stronger claim: it says that the adaptive greedy policy significantly outperforms any non-adaptive InfMax algorithm, including the non-adaptive greedy algorithm and even the optimal non-adaptive algorithm.

Lemma 5.29. *There exists a constant $c > 0$ such that, given any two functions $\varepsilon : \mathbb{Z}^+ \to \mathbb{R}^+$ and $\delta : \mathbb{Z}^+ \to \mathbb{R}^+$ such that $\varepsilon(k) \leq \frac{c\log\log k}{\log k}$ and $\delta(k) = o(1/k)$, for* InfMax *with prescribed seed candidates with LTM, there exists k such that, for any valid seed set S (i.e., S is a subset of the candidate set $\overline{V}$ and $|S| \leq k$) and any $\pi^g \in \Pi^g_{\varepsilon(k),\delta(k)}$, we have*

$$\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S)} \geq 2^{c\log(|V|)/\log\log(|V|)}.$$

Proof. The sketch of the proof of this lemma follows the proof of Lemma 5.18. We construct the same InfMax instance with the same $k, \overline{V}, d, W$ as given in the proof of Lemma 5.18. Again, by Lemma 5.16, choosing any k vertices among $\overline{V}$ infects the same number of vertices in expectation. We can reach the same conclusion that $\sigma(S) < \frac{|S|W}{d^d} + d^{d+1}$ by the same arguments. It then remains to analyze the greedy adaptive policy.

Consider an arbitrary greedy adaptive policy $\pi^g \in \Pi^g_{\varepsilon(k),\delta(k)}$. Let the three status "unexplored", "explored" and "dead" have the same meanings as they are in the proof of Lemma 5.16. Correspondingly, we will show that, *with probability at least* $1 - k\delta(k) = 1 - o(1)$, *if the root node is not infected yet, at any iteration of the greedy adaptive policy, each internal level of the tree can contain at most one explored node.*

[2] Lemma 5.18 also says that the adaptivity gap under the same setting is infinity. Since the adaptivity gap is about *optimal* algorithm/policy which is irrelevant to the greedy algorithm, this result holds as always, and it makes no sense to "adapt" it into the setting in this section. Similarly, Theorem 5.15 is also irrelevant here, and it holds as always.

Let v, v', u, ℓ_u, d_u have the same meaning as in the proof of Lemma 5.18. We have already seen that, at the current iteration, choosing v' is suboptimal, and choosing v' yields a marginal influence which is at most a fraction

$$\frac{1}{d^{\ell_u - 1} d_u} \Big/ \frac{1}{d^{\ell_u - 2}(d-1) d_u} = 1 - \frac{1}{d}$$

of the marginal influence of v. Therefore, if we set ε such that $\varepsilon < \frac{1}{d}$, the next seed chosen by the policy π^g will not be a leaf that is a descendent of an unexplored node with probability at least $1 - \delta$. By a union bound, with probability at least $1 - k\delta = 1 - o(1)$, if the root node is not infected yet, it will never happen that, at an iteration, there are more than one explored node in the same level.

The remaining part of the proof is almost the same. Since this crucial claim holds with probability at least $1 - o(1)$, the same induction argument shows that $\sigma^{\mathfrak{f}}(\pi^g, k) \geq \frac{1}{2} W$, and we have $\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S)} = \Omega(2^d) = 2^{\Omega(\log |V| / \log\log |V|)}$ as long as $\varepsilon < \frac{1}{d}$. Noticing that $d = \Omega(\frac{\log k}{\log\log k})$ (in particular, since $k = 2(\frac{d+1}{2})^d$, $\log k = d \log d + O(d)$ and $\log\log k = \log d + o(\log d)$, we have $d = \Omega(\frac{\log k}{\log\log k})$), implying $\frac{1}{d} = O(\frac{\log\log k}{\log k})$. The lemma holds with a sufficiently small c. □

Finally, we extend Theorem 5.14 to the (ε, δ)-greedy setting.

Theorem 5.30. *For any constant $c > 0$, given any two functions $\varepsilon : \mathbb{Z}^+ \to \mathbb{R}^+$ and $\delta : \mathbb{Z}^+ \to \mathbb{R}^+$ such that $\varepsilon(k) = O(\frac{1}{k^{2+c}})$ and $\delta(k) = o(1/k)$, there exists a triggering model $I_{G,D}$ and k such that, for any valid seed set S (i.e., S is a subset of the candidate set $\overline{V}$ and $|S| \leq k$) and any $\pi^g \in \Pi^g_{\varepsilon(k),\delta(k)}$, we have*

$$\frac{\sigma^{\mathfrak{f}}(\pi^g, k)}{\sigma(S)} \geq 2^{c' \log(|V|) / \log\log(|V|)},$$

where $c' > 0$ is a universal constant.

Proof. The sketch of the proof follows from Sect. 5.4.2. We construct the same InfMax instance with the same triggering model that is a mixture of `ICM` and `LTM`. The analysis for the non-adaptive algorithms is the same. We have $\sigma(S) \leq M(k \cdot \frac{1}{d^d} W + d^{d+1})$ for any S with $|S| \leq k$.

Consider any $\pi^g \in \Pi^g_{\varepsilon(k),\delta(k)}$. Consider an arbitrary iteration. Let $a_{\mathbf{z}} \in A$ be the seed that maximizes the marginal influence, which will be the one picked by the exact greedy adaptive policy. Recall that a vertex in A corresponds to the selection of a seed among the leaves in each of $T_1, \ldots, T_M$. Naturally, $a_{\mathbf{z}}$ makes the optimal selection in all the M trees. From the argument in the proof of Lemma 5.18, in each tree T_i and in each iteration, as long as the seed selected in T_i satisfies that there is at most one explored node at each level of T_i, we will have the greedy adaptivity gap being $2^{\Omega(\log|V|/\log\log|V|)}$ on the subgraph T_i, and the same argument in Sect. 5.4.2 shows that the greedy adaptivity gap overall is $2^{\Omega(\log\log|V|/\log\log\log|V|)}$. To conclude the proof of this theorem, we will show that $\varepsilon = O(\frac{1}{k^{2+c}})$ is sufficient to make sure that this will happen for all $T_1, \ldots, T_M$.

To show this, we consider a suboptimal $a'_{\mathbf{z}} \in A$ such that, at some tree T_i, there are more than one explored node at some level of T_i, and we find a lower bound of the difference between the marginal influence of $a_{\mathbf{z}}$ and the marginal influence of $a'_{\mathbf{z}}$. Let v, v', u, ℓ_u, d_u have the same meaning as in the proof of Lemma 5.18. Let p_u be the probability that, given the feedback at the current iteration, the path connecting from u to the root contains only live edges (i.e., u will infect the root). Then the marginal influence of v' is at most

$$\frac{1}{d^{\ell_u-1}d_u} \cdot p_u \cdot W + d + 1$$

(where the second term d is the number of nodes on the path from v' to the root, which can potentially be infected), and the marginal influence of v is at least

$$\frac{1}{d^{\ell_u-2}(d-1)d_u} \cdot p_u \cdot W.$$

The difference is at least

$$Wp_u \cdot \frac{1}{d^{\ell_u-2}d_u}\left(\frac{1}{d-1} - \frac{1}{d}\right) - d - 1 \geq \frac{W}{d^d(d-1)} - d - 1,$$

where we used the fact that $d_u \leq d$ and $p_u \geq \frac{1}{d^{d-\ell_u}}$.

On the other hand, the marginal influence of $a_{\mathbf{z}}$ is at most $M(W+d+1)$ (we have assumed the root is infected at this iteration for each of $T_1, \ldots, T_M$). It suffices to find an ε such that

$$\frac{W}{d^d(d-1)} - d - 1 > \varepsilon M(W+d+1).$$

Since we have set $W = M = d^{d+10}$, this is equivalent to

$$\frac{d^{10}}{d-1} - d - 1 > \varepsilon d^{d+10}\left(d^{d+10} + d + 1\right),$$

which implies

$$\varepsilon = O\left(\frac{1}{d^{2d+11}}\right).$$

Finally, recalling that $k = 2(\frac{d+1}{2})^d$, elementary calculations shows that $\varepsilon = O(\frac{1}{k^{2+c}})$ is a sufficient condition to the above:

$$\frac{1}{k^{2+c}} = \frac{1}{2^{2+c}}\left(\frac{2}{d+1}\right)^{2d+cd} < \left(\frac{1}{d}\right)^{2d+cd} \cdot 2^{2d+cd} = \frac{1}{d^{2d+11}} \cdot \frac{2^{2d+cd}}{2^{(cd-11)\log d}},$$

and the second term in the product above tends to 0 as $d \to \infty$. □

5.6 A Variant of Greedy Adaptive Policy

Although we have seen that the adaptive version of the greedy algorithm can perform worse than its non-adaptive counterpart, in general, we would still recommend the use of it as long as it is feasible, as it can also perform significantly better than the non-adaptive greedy algorithm (Theorem 5.14) while never being too bad (Theorem 5.10). As we remarked, the adaptivity may be harmful because exploiting the feedback may make the seed-picker too myopic. In this section, we propose a less aggressive risk-free version of the greedy adaptive policy, π^{g-}, in that it balances between the exploitation of the feedback and the focus on the average in the conventional non-adaptive greedy algorithm.

First, we apply the non-adaptive greedy algorithm with $|V|$ seeds to obtain an order $\mathcal{L}$ on all vertices. Then for any $S \subseteq V$ and any partial realization φ, $\pi^{g-}(S, \varphi)$ is defined to be the first vertex v in $\mathcal{L}$ that is not known to be infected. Formally, v is the first vertex in $\mathcal{L}$ that are not reachable from S when removing all edges e with $\varphi(e) \in \{\mathtt{B}, \mathtt{U}\}$. This finishes the description of the policy.

This adaptive policy is always no worse than the non-adaptive greedy algorithm, as it is easy to see that those seeds chosen by π^g are either seeded or infected by previously selected seeds in π^{g-}.

However, π^{g-} can sometimes be conservative. It is possible that π^{g-} has the same performance as the non-adaptive greedy algorithm, but π^g is much better. Especially, when there is no path between any two vertices among the first k vertices in $\mathcal{L}$, π^{g-} will make the same choice as the non-adaptive greedy algorithm. The InfMax instance in Sect. 5.4.2 is an example of this.

We have seen that π^{g-} sometimes performs better than π^g (e.g., in those instances constructed in the proofs of Lemma 5.8 and Lemma 5.9) and sometimes performs worse than the π^g (e.g., in the instance constructed in Sect. 5.4.2). Therefore, given a *particular* InfMax instance, for deciding which of π^{g-} and π^g to be used (we should never consider the non-adaptive greedy algorithm if adaptivity is available, as it is always weakly worse than π^{g-}), we recommend a comparison of the two policies by simulations. Notice that the seed-picker can randomly sample a realization ϕ and simulate the feedback the policy will receive. Thus, given $I_{G,D}$, both π^{g-} and π^g can be estimated by taking an average over the numbers of infected vertices in a large number of simulations. In the next section, we evaluate the three algorithms—the non-adaptive greedy algorithm, the greedy adaptive policy π^g and the conservative greedy adaptive policy π^{g-}—empirically by experiments on social networks in our real life.

5.7 Empirical Experiments

In this section, we compare the three algorithms—the non-adaptive greedy algorithm, the greedy adaptive policy π^g and the conservative greedy adaptive policy π^{g-}—empirically by experiments on the social networks in our real life. As a quick result obtained from our experiments, we have observed the followings.

1. The greedy adaptive policy π^g outperforms the conservative greedy adaptive policy π^{g-} and the non-adaptive greedy algorithm in most scenarios.

2. The conservative greedy adaptive policy π^{g-} always outperforms the non-adaptive greedy algorithm.

3. Occasionally, the greedy adaptive policy π^g is outperformed by the conservative adaptive policy π^{g-}, or even the non-adaptive greedy algorithm.

Notice that our results in Sect. 5.3 supports the third observation, and the second observation follows easily from our definition of π^{g-} in the last section.

5.7.1 Experiments Setup

We implement the experiments on four undirected graphs, Nethept, CA-HepPh, DBLP and com-YouTube, which are parts of Table 3.2. We implement the three algorithms with $k = 200$ seeds.

For the diffusion model, we implement both `ICM` and `LTM`. For `ICM`, we use `UICM` with $p = 0.01$. For `LTM`, we consider `ULTM`. For each dataset, we sample three realizations ϕ_1, ϕ_2, ϕ_3 as the “ground-truth”. Therefore, a total of six experiments are performed for each dataset: the two models ICM and LTM for each of the three realizations. For each of those six experiments, when a seed s is chosen, all vertices that are reachable from s in the ground-truth realization are considered infected, and given as the feedback. In particular, we consider the full-adoption feedback in our experiments.

To implement the three algorithms, we sample 1,000,000 reverse reachable sets, and perform the greedy maximum coverage algorithm (described in Sect 2.2.2) which iteratively selects the seed that maximizes the number of extra reverse reachable sets covered by this seed. We iteratively select seeds in this way until a sufficient number of seeds is selected (we decided to select 10,000 seeds, which turns out to be sufficient), and we ordered them in a list. Naturally, the non-adaptive greedy algorithm choose the first $k = 200$ seeds in this list. The conservative adaptive greedy policy iteratively select the first not-yet-selected seed in the list that is not known to be infected, as described in Sect. 5.6.

As for the adaptive greedy policy, the first seed is the same as the one for non-adaptive greedy algorithm and the conservative adaptive greedy policy. In each future iteration, the vertices that are infected (given as the feedback) are removed from the graph, and 1,000,000 new reverse reachable sets are sampled on the remainder graph. Notice that, for LTM, the degrees of the vertices in the remainder graph may decrease, which increase the weights of the incoming edges of these vertices. Then, a seed that covers most reverse reachable sets is selected as the next seed.

We remark that removing infected vertices from the graph and sampling reverse reachable sets on the remainder graph is the correct way to implement the algorithm. Since we are considering the full-adoption feedback, we know that there is no directed live edge from an infected vertex to an uninfected vertex, for otherwise the uninfected vertex should have been infected. When sampling the reverse reachable set, the triggering set of any uninfected vertex should not intersect with any infected vertex. Given an arbitrary uninfected vertex v and letting X be the set of all infected vertices, v should include each vertex in $\Gamma(v) \setminus X$ to its triggering set with probability 0.01 independently under ICM, and v should include exactly one vertex chosen uniformly at random in $\Gamma(v) \setminus X$ to its triggering set under LTM. Consequently, for both ICM and LTM, we can and we should remove those infected vertices from the graph and sample reverse reachable sets in the remainder graph.

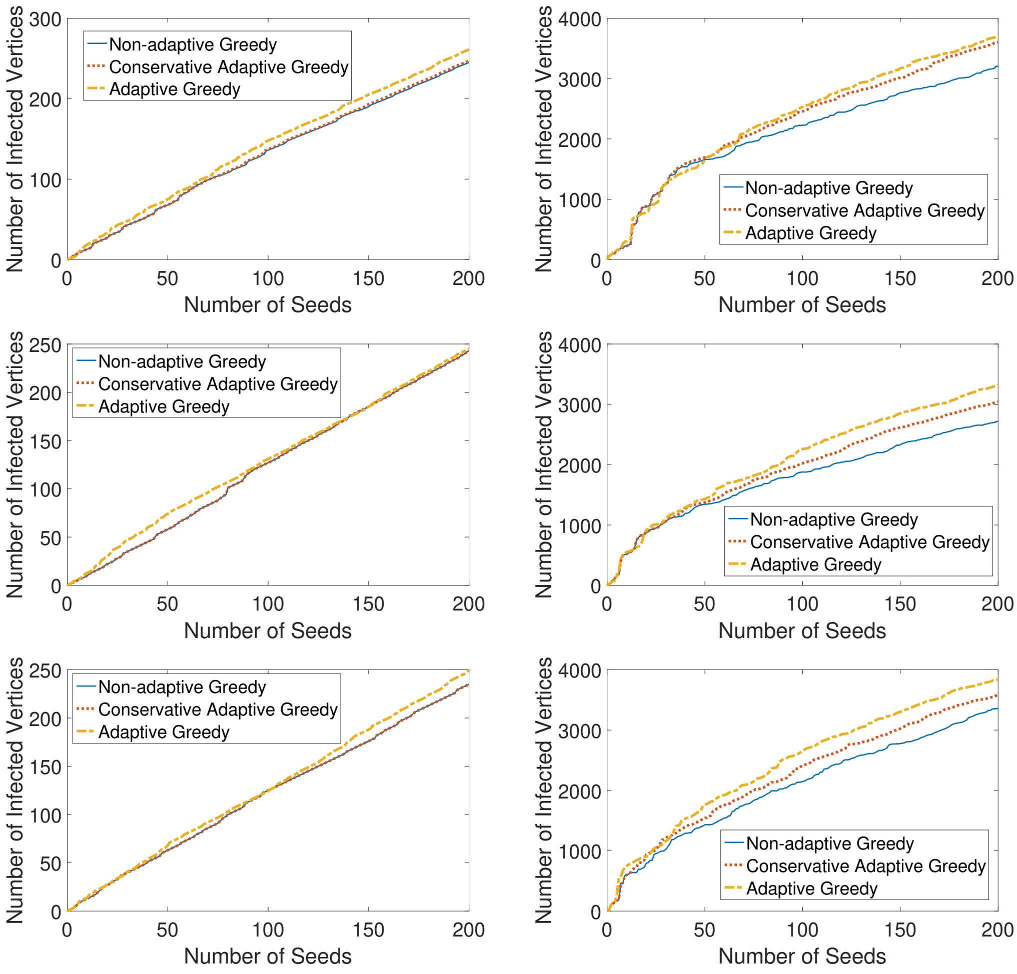


Figure 5.1: The results for the dataset Nethept. The three rows correspond to the three realizations ϕ_1, ϕ_2, ϕ_3, the left column is for `ICM`, and the right column is for `LTM`.

5.7.2 Results

As we mentioned, for each dataset, we have six figures corresponding to `ICM` and `LTM` for each of the three realizations ϕ_1, ϕ_2, ϕ_3. In each figure, the x-axis is the number of seeds, and the y-axis is the number of infected vertices in the realization. The three curves correspond to the outcomes of the three algorithms. Figure 5.1, 5.2, 5.3 and 5.4 correspond to the datasets Nethept, CA-HepPh, DBLP and com-YouTube respectively. The three observations mentioned at the beginning of this section can be easily observed from the figures.

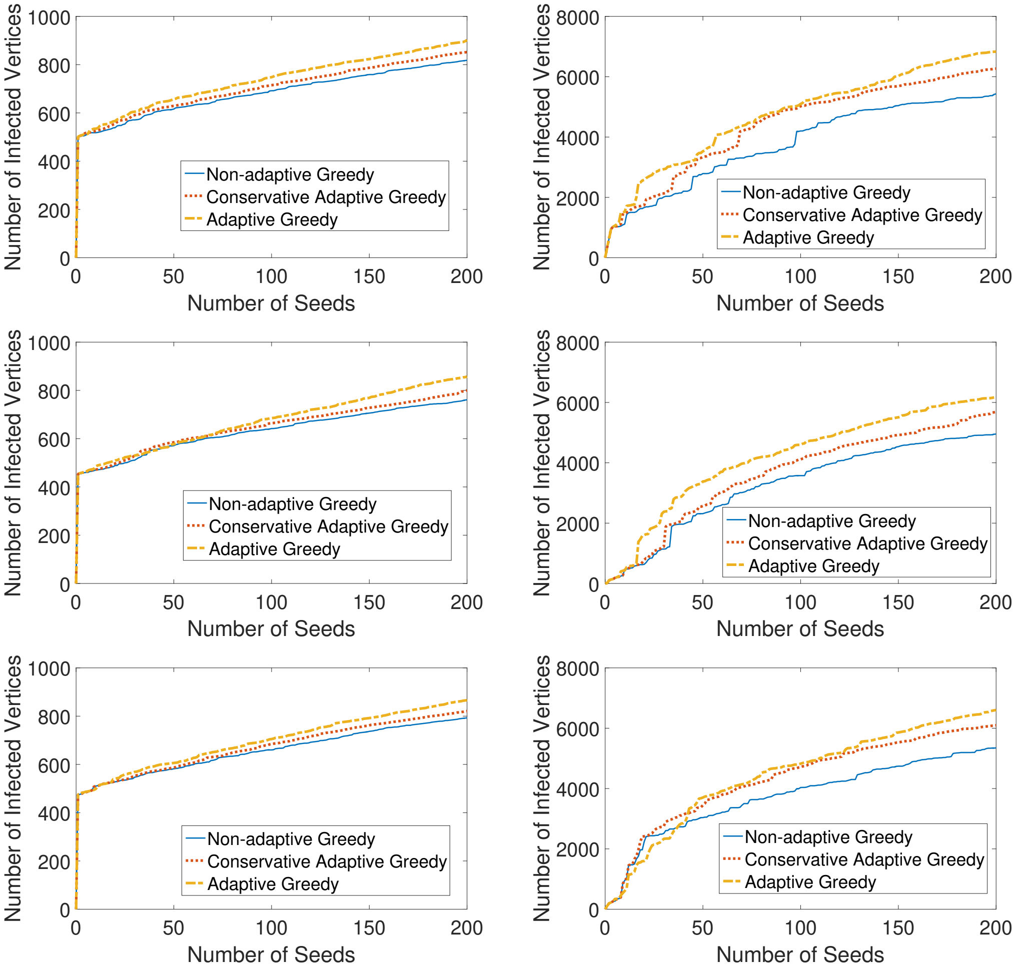


Figure 5.2: The results for the dataset CA-HepPh. The three rows correspond to the three realizations ϕ_1, ϕ_2, ϕ_3, the left column is for `ICM`, and the right column is for `LTM`.

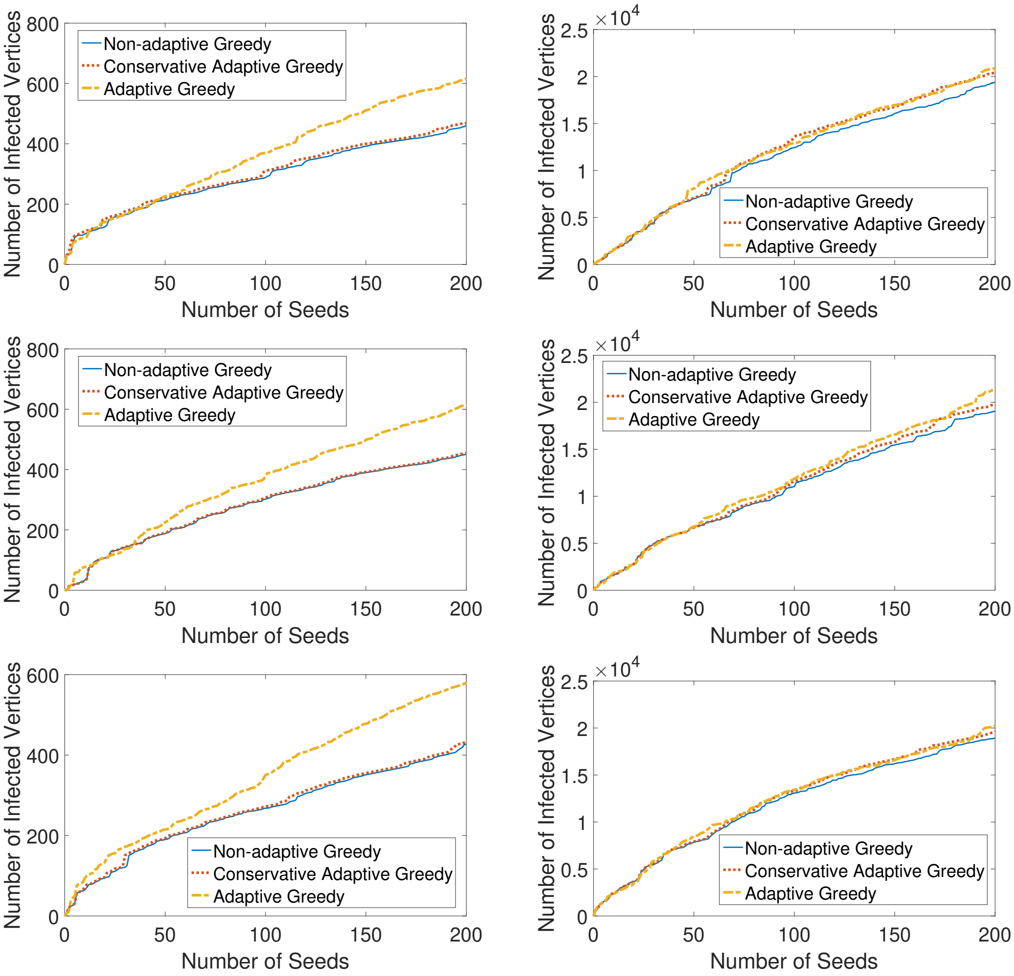


Figure 5.3: The results for the dataset DBLP. The three rows correspond to the three realizations ϕ_1, ϕ_2, ϕ_3, the left column is for `ICM`, and the right column is for `LTM`.

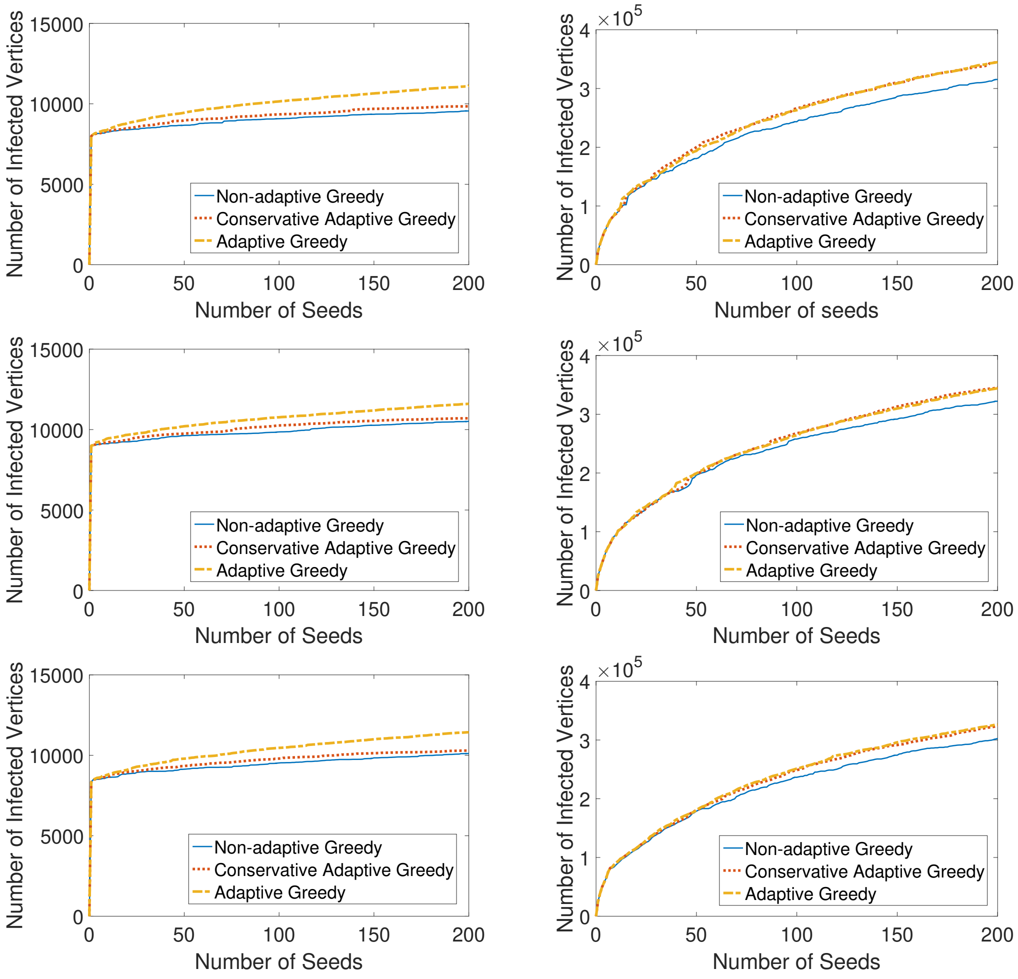


Figure 5.4: The results for the dataset com-YouTube. The three rows correspond to the three realizations ϕ_1, ϕ_2, ϕ_3, the left column is for `ICM`, and the right column is for `LTM`.

5.8 Conclusion and Open Problems

We have seen that the infimum of the greedy adaptivity gap is exactly $(1 - 1/e)$ for ICM, LTM, and general triggering models with both the full-adoption feedback model and the myopic feedback model. We have also seen that the supremum of this gap is infinity for the full-adoption feedback model. One natural open problem is to find the supremum of the greedy adaptivity gap for the myopic feedback model. Another natural open problem is to find the supremum of the greedy adaptivity gap for the more specific ICM and LTM.

The greedy adaptivity gap studied in this chapter is closely related to the adaptivity gap studied in the past. Since the non-adaptive greedy algorithm is always a $(1 - 1/e)$-approximation of the non-adaptive optimal solution, a constant adaptivity gap implies a constant greedy adaptivity gap. For example, the adaptivity gap for ICM with myopic feedback is at most 4 [61], so the greedy adaptivity gap in the same setting is at most $\frac{4}{1-1/e}$. In addition, the greedy adaptive policy is known to achieve a $(1-1/e)$-approximation to the adaptive optimal solution for ICM with full-adoption feedback [36], so the adaptivity gap and the greedy adaptivity gap could either be both constant or both unbounded for ICM with full-adoption feedback model, but it remains open which case is true. The adaptivity gap for ICM with full-adoption feedback, as well as the adaptivity gap for LTM with both feedback models, are all important open problems. We believe these problems can be studied together with the greedy adaptivity gap.

Part II

Nonsubmodular Influence Maximization

CHAPTER 6

2-Quasi Submodular Diffusion Model

We have mentioned that InfMax admits a $(1-1/e)$-factor approximation algorithm if the diffusion model is submodular. Otherwise, in the worst case, the problem is NP-hard to approximate to within a factor of $n^{1-\varepsilon}$. This chapter studies whether this worst-case hardness result can be circumvented by making assumptions about the diffusion model.

We propose a new diffusion model that is a special case of the general threshold model (see Sect. 2.1.1) where the local influence functions are *2-quasi-submodular*, which is almost submodular except that the second infected neighbor has more marginal influence to a vertex than the first. This model is motivated by the corresponding observation from many empirical and sociological studies [4, 50, 63, 75].

We present strong inapproximability results for this model. Our inapproximability results hold even for any 2-quasi-submodular local influence function f fixed in advance and the graphs are undirected. This result also indicates that the "threshold" between submodularity and nonsubmodularity is sharp, regarding the approximability of influence maximization.

6.1 Introduction

InfMax becomes qualitatively different in nonsubmodular settings. In the submodular case, seeds erode each other's effectiveness, and so should generally not be put too close together. However, in the nonsubmodular case, it may be advantageous to place the initial seeds close together to create synergy and yield more infections. The intuition that it is better to saturate one market first, and then expand implicitly assumes nonsubmodular influence in the cascades. For general nonsubmodular cascades, it is NP-hard even to approximation InfMax to within an $n^{1-\varepsilon}$ factor of optimal [45], and the inapproximability results have been extended to several more restrictive nonsubmodular models [20, 53]. Unfortunately, empirical research shows that many cascades are indeed not submodular [4, 50, 63, 75, 32].

Key Question: Can this worst-case inapproximability result of $n^{1-\epsilon}$ for nonsubmodular InfMax be circumvented by making realistic assumptions about the diffusion model?

The same research showing that cascades are often not submodular empirically also shows that the local submodularity often fails in one particular way—the second infected neighbor of an agent is, on average, more influential than the first. When Leskovec et al. [50] studied the probability a person buys a book versus the number of incoming recommendations he receives, they observed a peak in the marginal probability of buying at 2 incoming recommendations and then a slow drop. While this work presents observational evidence, it suggests that if a person does not buy a book after the first recommendation, but receives another, he is more likely to be persuaded by the second recommendation. But thereafter, they are less likely to respond to additional recommendation.

Backstrom et al. [4] made the same observation when they calculated the probability a person joins a community (e.g., LiveJournal and DBLP) as a function of the number t of his friends already in the community. Romero et al. [63] studied hashtag adoption in the Twitter network, and considered the fraction of users who adopt a hashtag after having t neighbors' adoptions. They coalesced their study's observations into a model where the marginal influence increases linearly from zero to two adopting neighbors and then linearly decreases thereafter.

These empirical studies motivate our study of the *2-quasi-submodular* diffusion model (more accurately, the general threshold model with 2-quasi-submodular local influence functions) where the marginal effect of the second infected neighbor is greater than the first, but after that the marginal effect decreases.

Our result we present an inapproximability result for InfMax with the 2-quasi-submodular model. In particular, for *any* 2-quasi-submodular function f, and even for undirected graphs, we show that it is NP-hard to approximate InfMax within a factor of n^τ when each agent has f as its local influence function, where $\tau > 0$ is a constant depending on f. This can be seen as a threshold result for approximability of InfMax, because if f is submodular, then InfMax can be approximated to within a $(1 - 1/e)$-factor, but if f is just barely nonsubmodular InfMax can no longer be approximated to within any constant factor. To further strengthen our inapproximability result, we also show that, for any $\gamma \in (0, 1)$, when only n^γ agents have the fixed 2-quasi-submodular f as their local influence functions and the remaining agents' local influence functions are submodular (or even identical to a fixed submodular function), InfMax is still NP-hard to approximate to within a factor of n^τ, where $\tau > 0$ is a constant depending on f and γ.

6.2 Additional Related Work

The work most related to the present chapter are several inapproximability results for InfMax. If no assumption is made for the influence function, InfMax is NP-hard to approximate to within a factor of $n^{1-\varepsilon}$ for any $\varepsilon > 0$ [44].

Chen [12] found inapproximability results on a similar optimization problem: instead of maximizing the total number of infected vertices given k initial targets, he considered the problem of finding a minimum-sized set of initial seeds such that all vertices will eventually be infected. This work studied restrictions of this problem to various threshold models.

An important difference between our hardness result in Sect. 6.4 and all the previous results is that our result holds for *any* 2-quasi-submodular functions. In particular, in this work, f is fixed in advance before the NP-hardness reduction, while in previous work, specific influence functions were constructed within the reductions.

The notion of "near submodularity" was also proposed and studied in [73]. Our definition differs from the one in [73] in that a 2-quasi-submodular function can be, intuitively, very far from being submodular (for example, the 2-threshold cascade model). However, our reduction in Sect. 6.4 works for all 2-quasi-submodular functions, and 2-quasi-submodular functions can be arbitrarily close to submodular functions.

Similar to our inapproximability result for the 2-quasi-submodular model in Sect. 6.4 but independent to our work, Li et al. [53] studies InfMax with almost submodular local influence functions, and shows that, for any $\gamma, \varepsilon \in (0, 1)$, InfMax is hard to approximate to within factor $1/n^{\frac{\gamma}{c}}$ even if the graph only contains n^γ vertices that admit nonsubmodular local influence functions that are ε-almost submodular (and the remaining vertices admit submodular local influence functions), where $c = 3+3/\log \frac{2}{2-\varepsilon}$. When the number of vertices admitting ε-almost submodular local influence functions is a constant, Li et al. [53] provides a constant-factor approximation algorithm for InfMax.

Our result in Sect. 6.4 can be seen as a generalization to the inapproximability result in Li et al.: their results construct a 2-quasi-submodular influence function f

(although ε can be arbitrarily small and fixed in advance, making f arbitrarily close to a submodular function); in contrast, our result holds for any f that is fixed in advance and universal for all vertices. In addition, our inapproximability result holds even for undirected graphs, while the graph constructed in the reduction in Li et al. is directed.

At a high level, the techniques of the two approaches are similar. However, the gadgets used in our more general result require additional ideas.

In Sect 6.5, we show that our results seamlessly extend to the setting of Li et al. where only a sublinear fraction of vertices (e.g., n^{γ}) admit nonsubmodular local influence functions.

6.3 Preliminaries

Due to the heavy use of notations, in this particular chapter, we will use $N = |V|$ to denote the total number of vertices in the input graph, instead of n as in the other chapters.

To eventually define our 2-quasi-submodular setting, we first define some special classes of the general threshold model.

We say that the general threshold model $I_{G,F}$ is *symmetric*, if $f_v(IN_v)$ only depends on $|IN_v|$ for each $v \in V$. That is, the local influence function f_v only depends on the *number* of v's infected neighbors so that each of v's infected neighbors is of equal importance. In this case, f_v can be viewed as a function $f_v : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{\geq 0}$ which takes an integer as input, rather than a set of vertices. Note that $f_v(0) = 0$, as we have assumed $f_v(\emptyset) = 0$ in Definition 2.1. As examples, `UICM`, `WICM` and `ULTM` studied in previous chapters are all special cases of the symmetric general threshold models.

We say that the general threshold model $I_{G,F}$ is *universal* if it is symmetric and, in addition, all f_v's are identical. Notice that `UICM` is universal (with $f(t) = 1-(1-p)^t$), while `WICM` and `ULTM` are not (f_v in both models depends on $\deg(v)$). In this chapter, we will only consider the universal general threshold model, and let $f : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be the common local influence function. We will denote the model by $I_{G,f}$. Since the main result in this chapter is a hardness result, stronger assumptions on the model make the result stronger.

We will consider the universal general threshold model with *2-quasi-submodular* local influence functions f, which is "almost" submodular such that the submodularity is only violated for the first two inputs of f. In particular, we fail to have the submodular constraint $f(1) - f(0) \geq f(2) - f(1)$, and instead we have $f(1) - f(0) <$

notation	meaning
N	total number of vertices $\lvert V\rvert$
$f:\mathbb{Z}_{\geq 0}\to\mathbb{R}_{\geq 0}$	symmetric local influence function that only depends on the number of infected vertices
$I_{G,f}$	universal general threshold model where local influence function for each vertex is f
$a_0,a_1,a_2,\ldots$	denote $f(0),f(1),f(2),\ldots$ respectively

Table 6.1: New notations used in Chapter 6.

$f(2)-f(1)$, which is just $f(2)>2f(1)$ as $f(0)=0$. (Notice that, for a set function f such that $f(A)$ only depends on the cardinality of A, submodular constraint $\forall A\subsetneq B, v\notin B: f(A\cup v)-f(A)\geq f(B\cup v)-f(B)$ becomes $\forall i\geq 0: f(i+1)-f(i)\geq f(i+2)-f(i+1)$.)

Definition 6.1. $f:\mathbb{Z}_{\geq 0}\to[0,1]$ is *2-quasi-submodular* if $f(2)>2f(1)$ and $f(i)-f(i-1)$ is non-increasing in i for $i\geq 2$.

In general, for any non-zero submodular function f, if we sufficiently decrease $f(1)$, f becomes 2-quasi-submodular. Thus, from any non-zero submodular function, we can obtain a 2-quasi-submodular function.

We note that the 2-complex contagion (Definition 2.16) can be viewed as the universal general threshold model with a 2-quasi-submodular f (with $f(0)=f(1)=0$ and $f(i)=1$ for $i\geq 2$).

Clearly, f can be encoded by an increasing sequence of positive real numbers $a_0,a_1,a_2,\ldots$ so that $f(i)=a_i$. We will use $a_0,a_1,a_2,\ldots$ to denote $f(0),f(1),f(2),\ldots$, and notice that these numbers are constant if we consider InfMax instances with f fixed in advance (instead of being a part of inputs). All the new notations used in this particular are summarized in Table 6.1.

6.4 Hardness of Approximation for 2-Quasi-Submodular Influence Maximization

We prove the following theorem in this section which says that, for any fixed 2-quasi-submodular f, there exists a constant τ depending on f such that InfMax with the universal general threshold model $I_{G,f}$ is NP-hard to approximate to within factor N^τ, where recall that N is the number of vertices of the graph.

Theorem 6.2. *Consider the* InfMax *problem with the universal general threshold model $I_{G,f}$ for any fixed 2-quasi-submodular f. There exists a constant $\tau > 0$ depending on f such that it is NP-hard to distinguish between the following two cases:*

- YES*: there exists a seed set S with $|S| = k$ such that $\sigma(S) = \Theta(N)$;*
- NO*: for any seed set S with $|S| = k$, we have $\sigma(S) = O(N^\tau)$,*

even if G is an undirected graph.

Since 2-complex contagion is a special case of $I_{G,f}$ with 2-quasi-submodular f, we immediately have the following corollary.

Corollary 6.3. *There exists a constant $\tau > 0$ such that* InfMax *with r-complex contagion is NP-hard to approximate to within factor N^τ, even if $r = 2$ and the graph is undirected.*

The sequence notation $(a_i)_{i=0,1,2...}$ is used to represent f in this section. Because f is 2-quasi-submodular, we have $a_0 = 0$ and $a_2 > 2a_1$. We denote $p^* = \lim_{i\to\infty} a_i$, which exists because (a_i) is increasing and bounded by 1 (see Definition 6.1). We consider two cases: $a_1 > 0$ and $a_1 = 0$. We note that we have $a_2 > 0$ by the 2-quasi-submodular assumption. In the case $a_1 > 0$, we will first assume the graph is directed, and later we will show that this assumption is not essential.

The remaining part of this section is organized as follows: Sect. 6.4.1 provides a sketch of the proof of Theorem 6.2 for the case $a_1 > 0$, with arguments presented in an intuitive level, Sect. 6.4.2 to Sect. 6.4.7 prove the theorem rigorously for the case $a_1 > 0$, and Sect. 6.4.8 prove the theorem rigorously for the case $a_1 = 0$. Finally, in a similar style to the result in [53], we prove a variant of Theorem 6.2 in Sect. 6.5 saying that the inapproximability also holds if only N^γ (for some fixed $\gamma \in (0, 1)$) vertices admit the fixed 2-quasi-submodular function f while the remaining vertices admit certain fixed non-zero submodular function g.

6.4.1 Proof Sketch of Theorem 6.2 for $a_1 > 0$

We prove the theorem by a reduction from the SetCover problem.

Definition 6.4. Given a universe U of n elements, a set of K subsets $A = \{A_i \mid A_i \subseteq U\}$, and a positive integer k, the SetCover problem asks if we can choose k subsets $\{A_{i_1}, \ldots, A_{i_k}\} \subseteq A$ such that $A_{i_1} \cup \cdots \cup A_{i_k} = U$.

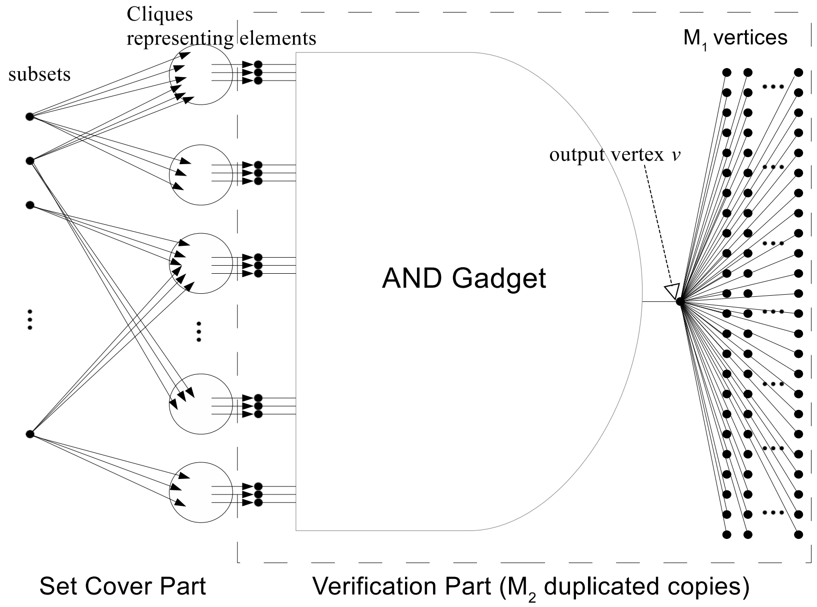


Figure 6.1: The high-level structure of the reduction for the proof of Theorem 6.2

We construct a graph G which consists of two parts: the set cover part and the verification part, where the set cover part simulates SetCover and the verification part verifies if all the elements in the SetCover instance are covered. The construction is shown in Fig. 6.1. We first assume that the graph G is directed, and then we show that this assumption is not essential by constructing a *directed edge gadget* to simulate directed edges.

Given a SetCover instance, in the set cover part, we use a single vertex to represent a subset A_i and a clique of size m to represent each element in U. If an element is in a subset, we create m directed edges from the vertex representing the subset to each the m vertices in the clique representing the element. If a vertex representing a subset is picked, then all vertices in the cliques corresponding to the elements contained in this subset will be infected with probability close to p^*, by choosing m large enough. We call such cliques as being activated. In a YES instance of SetCover, we can choose k seeds such that all cliques are activated.

In the verification part, we construct a *AND gadget*, simulating the logical AND operation, to verify if all the cliques are activated. The AND gadget takes n inputs, each of which is a set of vertices from each of the n cliques. The output of the AND gadget is a vertex v, such that it will only be infected with a positive constant probability if all the n cliques are activated.

We connect the output vertex v of this AND gadget to a huge bundle of M_1 vertices, such that a constant fraction of those M_1 vertices will be infected only if all the cliques are activated (which corresponds to the case the SetCover is a YES instance). By making M_1 large enough, we can achieve a hardness of approximation ratio N^τ. To avoid the seed-picker bypassing the set cover game by directed seeding the output vertex v, we duplicate the verification part by M_2 times for some sufficiently large M_2.

Finally, we replace all directed edges in Fig. 6.1 by directed edge gadgets, including those connecting the vertices representing subsets and the cliques representing elements, and those connecting the set cover part and the verification part. To complete the proof of Theorem 6.2, we present the construction of the AND gadget and the directed edge gadget in the next few subsections.

6.4.1.1 The Probability Filter Gadget

In this section, we present the construction of a gadget called *probability filter gadget*, which is the key component in the constructions of both AND gadget and directed edge gadgets mentioned above.

Given a set of vertices that will be infected with a same probability x, the probability filter gadget tests if x is larger than certain threshold p_1. It outputs a vertex infected with probability almost 0 if $x < p_1$, and with certain non-negligible probability p_2 if $x > p_1$.

The probability scaling down gadget Firstly, we need to construct the *probability scaling down gadget* which takes a vertex u with infection probability p_u as input, and output a vertex v such that v is infected with probability $p_v = \alpha p_u$, where $\alpha \leq p^*$ is an adjustable parameter. The construction of this gadget is shown in Fig. 6.2: we add many paths of different lengths from u to v, and we can achieve $p_v = \alpha p_u$ by adjusting the number of paths and the length of each path.

The probability separation block Next, we construct a *probability separation block*, which is the building block to the probability filter gadget. The probability separation block takes h vertices as input and outputs one vertex such that

1. if each input is infected independently with a same probability that is greater than certain threshold p_1, then the output vertex will be infected with a slightly higher probability;

2. if each input is infected independently with a same probability that is less than p_1, then the output vertex will be infected with a slightly lower probability.

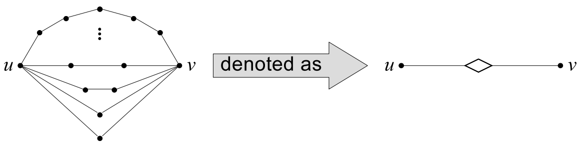


Figure 6.2: The probability scaling down gadget

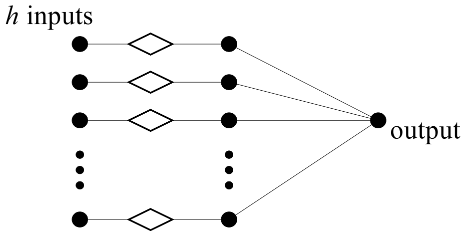


Figure 6.3: The probability separation block

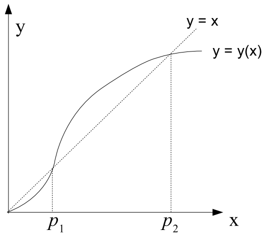


Figure 6.4: The output probability y versus the input probability x

The construction of the probability separation block is shown in Fig. 6.3, in which the h inputs' infection probabilities are scaled down by a certain factor α by the probability scaling down gadgets, and then they are connected to the output vertex. It is exactly the 2-quasi-submodularity of f which enables us to adjust the two parameters h and α such that (1) and (2) above hold.

Suppose each of the h vertices in the input are infected with probability x, and let $y = y(x)$ be the probability that the output vertex is infected. We claim that we can tune the values of α and h such that the graph of $y(x)$ looks like Fig. 6.4.

By considering the number of infected neighbors of the output vertex, we have

$$y = \sum_{i=1}^{h} \binom{h}{i} a_i (\alpha x)^i (1 - \alpha x)^{h-i},$$

which is $y = h\alpha a_1 x + \frac{h(h-1)}{2}\alpha^2(a_2 - 2a_1)x^2 + o(x^2)$ for sufficiently small x. Choosing a sufficiently small constant $\delta > 0$ and choosing α, h to satisfy $h\alpha a_1 = 1 - \delta$, we have

$$y - x = -\delta x + \frac{h(h-1)}{2}\alpha^2(a_2 - 2a_1)x^2 + o(x^2).$$

Since $y - x = -\delta x + o(x)$, we can see that $y < x$ for small enough x.

On the other hand, for sufficiently large h and sufficiently small δ (and adjusting α such that $h\alpha a_1 = 1 - \delta$ still holds), the second order derivative of $y - x$, which is

$$\frac{d^2(y - x)}{dx^2} = h(h-1)\alpha^2(a_2 - 2a_1) + o(1) \approx \frac{1}{a_1^2}(a_2 - 2a_1) > 0,$$

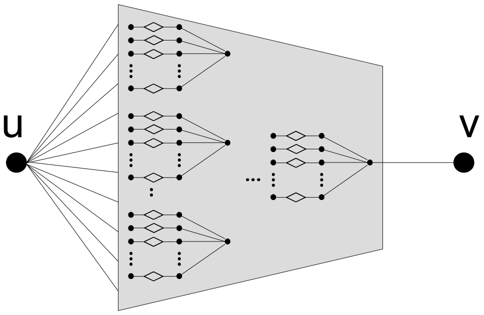


Figure 6.5: The directed edge gadget $\langle u, v\rangle$

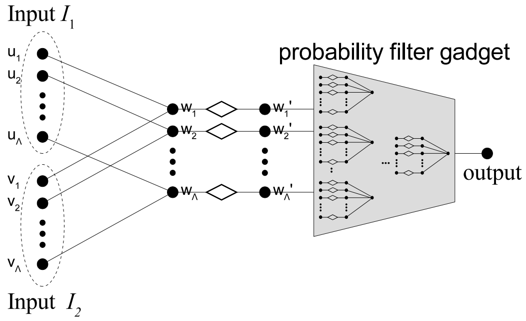


Figure 6.6: The AND gadget with two inputs

can be considerably more significant than its first order derivative $-\delta$. Therefore, $y - x$, starting from 0 at $x = 0$ and being negative for very small x, will soon become positive after x increases. This proves our claim. Notice that the 2-quasi-submodularity of f makes sure $a_2 - 2a_1 > 0$.

The probability filter gadget The probability filter gadget consists of ℓ layers such that the i-th layer consists of $h^{\ell-i}$ probability separation blocks, where the output vertices of every h probability separation blocks in the i-th layer are the inputs of a single probability separation block in the $(i+1)$-th layer. Because there are $h^{\ell-1}$ probability separation blocks in the first layer, the probability filter gadget takes $\Lambda = h^\ell$ vertices as input. The probability filter gadget outputs a single vertex after ℓ layers.

From Fig. 6.4, if we make ℓ large enough, we conclude that the probability filter gadget does the following job, which tests if the input vertices are infected with a probability larger than the threshold value p_1.

1. if each vertex in the Λ inputs is infected independently with a same probability less than p_1, then the vertex on the output end will be infected with a probability close to 0;

2. if each vertex in the Λ inputs is infected independently with a same probability in $(p_1, p_2]$, then the vertex on the output end will be infected with a probability close to p_2.

6.4.1.2 The AND Gadget and the Directed Edge Gadget

Both the AND gadget and the directed edge gadget can be constructed by using a single probability filter gadget as the core.

The directed edge gadget The construction of the directed edge gadget $\langle u, v\rangle$ is shown in Fig. 6.5. It uses a single probability filter gadget, whose input vertices are connected to u, and whose output vertex is connected to v. By adjusting h and α making p_1 small enough[1], we can make the output v infected with noticeable probability (almost p_2) if u is infected. On the other hand, if v is infected, then the expected number of infected vertices among those h vertices on the input end of the top layer probability separation block is $h\alpha a_1 = 1 - \delta < 1$, which suggests that the cascade process will die out after a few layers from right to left. In particular, the influence of v cannot be passed to u.

The AND gadget The AND gadget in Fig. 6.1 takes n sets of vertices as input. It tests if all cliques are activated, that is, if each vertex in each input set is infected with probability almost p^*.

Here, we first construct a smaller AND gadget which only takes two input sets. Let $I_1 = \{u_1, u_2, \ldots, u_\Lambda\}$ and $I_2 = \{v_1, v_2, \ldots, v_\Lambda\}$ be the two input sets. The AND gadget should do the following:

1. if each vertex in I_1 and I_2 is infected with probability p^*, the AND gadget outputs a vertex which is infected with a notable probability;

2. if all vertices in at least one of I_1, I_2 are infected with probability 0, the AND gadget outputs a vertex which is infected with a negligible probability.

We create Λ vertices $w_1, w_2, \ldots, w_\Lambda$ and create two edges $(u_i, w_i), (v_i, w_i)$ for each $i = 1, 2, \ldots, \Lambda$. In case (1), each w_i will be infected with probability $q_1 = a_2(p^*)^2 + 2a_1p^*(1-p^*)$; in case (2), each w_i will be infected with probability at most $q_2 = a_1p^*$. Obviously, $q_1 > q_2$, and the AND gadget needs to "amplify" the gap between q_1 and q_2.

This naturally reminds us the probability filter gadget. In particular, if the threshold p_1 of the probability filter gadget is in between: $q_1 > p_1 > q_2$, we can just make $\{w_1, \ldots, w_\Lambda\}$ the inputs of the probability filter gadget, and we are done. However, by our discussion about probability separation block in the last subsection, p_1 is only guaranteed to exist, which may not be in (q_2, q_1). To settle this, we use probability scaling down gadgets to rescale the infection probability of w_i such that p_1 will be in

[1] if we further check the calculations in the subsection where we construct the probability separation block, we can see that p_1 can be made arbitrarily small, by choosing small enough $\delta = 1 - h\alpha a_1$. Detailed calculations and justifications are in the later sections.

between after rescaling q_1, q_2.[2] Fig. 6.6 shows the construction of this AND gadget.

To construct the AND gadget allowing n input sets, we can use this AND gadget as a building block and construct an *AND circuit* with $\log_2 n$ levels of AND gadgets. The last level contains a single AND gadget, whose output is connected to the M_1 vertices on the right-hand side of Fig. 6.1. For each AND gadget in Level i, its output become one input of a certain AND gadget in Level $i+1$. The inputs of the AND gadgets in Level 1 are exactly those associated to the n cliques representing elements of the VertexCover instance.

We conclude the proof sketch here. In the remaining sections, we present the full proof of Theorem 6.2 which realizes the intuitions and ideas in this section.

6.4.2 Proof of Theorem 6.2 for $a_1 > 0$ with Directed Graphs

We first define the following AND gadget which simulates the logical AND operation. The construction of this AND gadget is deferred to Section 6.4.4—6.4.6. We note that the nonsubmodularity property $a_2 > 2a_1$ plays an important role in the construction of the AND gadget. In particular, the construction of the AND gadget uses a smaller gadget called the "probability filter gadget" as a building block (see Figure 6.8), and 2-quasi-submodularity is essential for constructing the probability filter gadget (refer to Section 6.4.4.2 for details).

Definition 6.5. An $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-*AND gadget* takes I sets which each contains Λ vertices as input, and outputs one vertex such that

1. if all the vertices in all I sets are infected independently with probabilities less than $\frac{11}{10}p_0$, and moreover the infection probabilities of the vertices in at least one input set are less than $\frac{1}{2}p_0$, then the output vertex will be infected with probability less than ε_1;

2. if all the vertices in all I sets are infected independently with probabilities in the interval $(p_0, \frac{11}{10}p_0)$, the output vertex will be infected with probability in $(p_2 - \varepsilon_2, p_2]$,

We remark that the choices for both factors of p_0 in 1 of the above definition, $\frac{11}{10}$ and $\frac{1}{2}$, are only required to be close enough to 1 and 0 respectively. We aim to simulate the case where at least one of the inputs is not "active" (being far from

[2] It seems worrying that q_1 and q_2 may be both less than p_1, in which case the construction fails as we can only scale probabilities "down". However, as we have remarked, we can make p_1 arbitrarily small such that $p_1 \ll q_2 < q_1$

the threshold p_0) and the other ones are not "too active" (being at most somewhere around the threshold p_0), in which case the AND gadget outputs "false" (such that the output vertex is infected with negligible probability ε_1).

With the choice of the seven parameters satisfying the relation in the below lemma, we can construct the AND gadget.

Lemma 6.6. *Given any 2-quasi-submodular function f with $a_1 > 0$, any constant threshold $p_0 > 0$ and any $I = 2^\ell$ that is an integer power of 2, there exists a constant $p_2 > 0$ depending on p_0 and f such that for any $\varepsilon_1 > 0$ and any constant $\varepsilon_2 > 0$, we can construct an $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget with $\Lambda = O\left((1/\varepsilon_1)^{c_1} I^{c_2}\right)$, and the numbers of vertices and edges in this AND gadget are both $O\left((1/\varepsilon_1)^{c_1} I^{c_2+1}\right)$, where c_1 and c_2 are two constants.*

The following lemma is needed in the next section for the proof of Theorem 6.2 for undirected graphs.

Lemma 6.7. *Given any 2-quasi-submodular function f with $a_1 > 0$ and any $I = 2^\ell$ that is an integer power of 2, there exists $p_2 > 0$ such that for any $\varepsilon_1 > 0$ and any constant $\varepsilon_2 > 0$, we can construct an $(I, \Lambda, p^*(p_2 - \varepsilon_2), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget. We have $\Lambda = O\left((1/\varepsilon_1)^{c_1} I^{c_2}\right)$ and the AND gadget contains $O\left((1/\varepsilon_1)^{c_1} I^{c_2+1}\right)$ vertices and $O\left((1/\varepsilon_1)^{c_1} I^{c_2+1}\right)$ edges, where c_1 and c_2 are two constants.*

Notice that Lemma 6.6 does not imply Lemma 6.7: in Lemma 6.6, we first fix the third parameter p_0, and the existence of the fourth parameter p_2 relies on the third; in Lemma 6.7, we simultaneously fix the third and the fourth parameters.

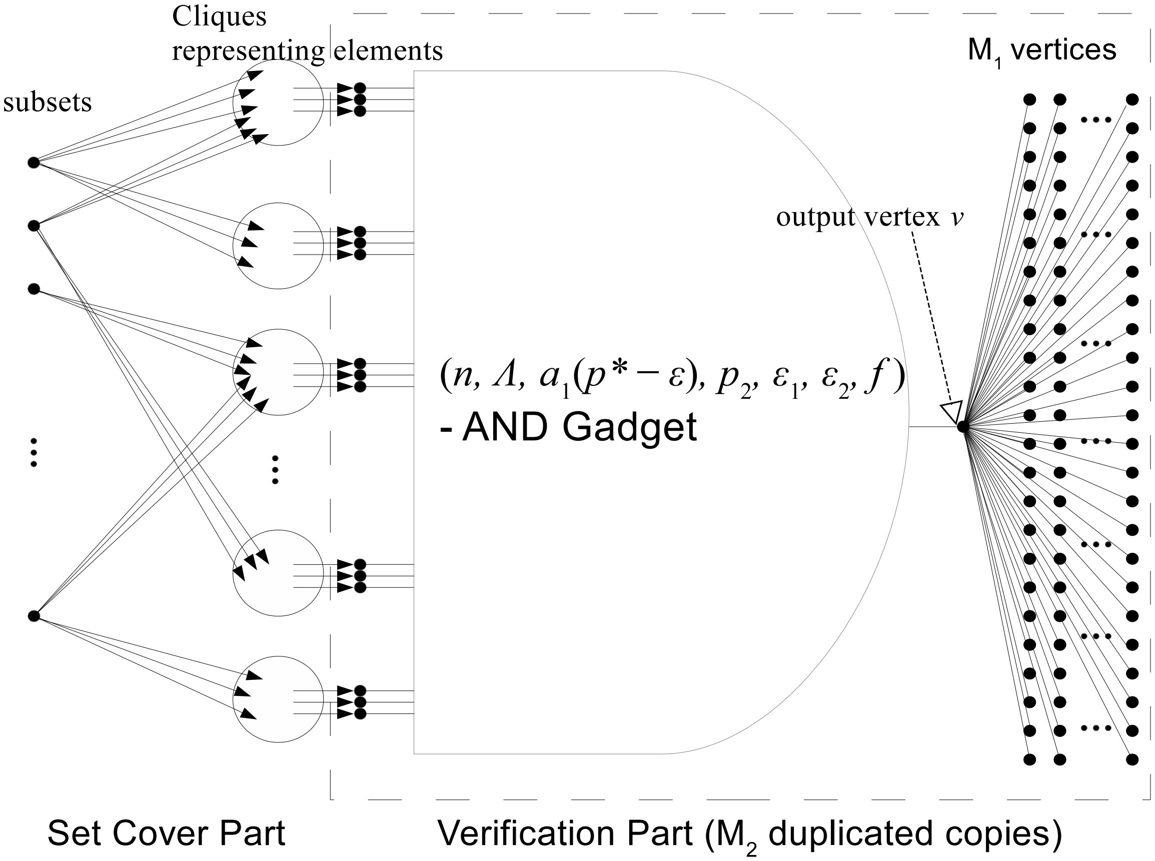
The construction of the AND gadget and the proof of Lemma 6.6 and Lemma 6.7 are deferred to Section 6.4.6. In this section, we aim to prove Theorem 6.2 for $a_1 > 0$ with directed graphs and assuming Lemma 6.6, while we do not need Lemma 6.7 at this moment. We remark that the construction of AND gadget requires no directed edges, although we consider directed graph in this section.

6.4.2.1 A Reduction from SetCover

We prove the theorem by a reduction from SetCover.

Without loss of generality, we will assume $K = O(n)$.[3] We will also assume that each element in U is covered by at least one subset A_i in SetCover (otherwise we

[3] One way to justify this assumption is to consider VertexCover, which can be viewed as a special case of SetCover by viewing vertices as subsets and edges as elements. In a connected graph, the number of vertices K never exceeds $O(n)$, if n is the number of edges.



Cliques representing elements
subsets
M$_1$ vertices
output vertex v
$(n, \Lambda, a_1(p^*-\varepsilon), p_2, \varepsilon_1, \varepsilon_2, f)$ - AND Gadget
Set Cover Part
Verification Part (M$_2$ duplicated copies)

Figure 6.7: The high-level structure of the reduction

know for sure the instance is a NO instance). In addition, we assume the number of elements $n = |U|$ is an integer power of 2, as we can add elements into U and let these elements be included in all sets A_i in the case n is not an integer power of 2.

We construct a graph G with N vertices which consists of two parts: the set cover part and the verification part, where the set cover part simulates the SetCover instance and the verification part verifies if all the elements in the SetCover instance are covered. The construction is shown in Figure 6.7.

Define $\varepsilon = 2\left(p^* - a_{\lfloor a_1 n \rfloor}\right)$ which approaches to 0 as $n \to \infty$ if $a_1 > 0$. According to Lemma 6.6, for $p_0 = a_1(p^* - \varepsilon)$ and $I = n$, there exists a constant $p_2 > 0$, such that if we set $\varepsilon_1 = \frac{1}{n}$ and $\varepsilon_2 = \frac{1}{100}p_2$, we can construct an $(n, \Lambda, p^* - \varepsilon, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget, where $\Lambda = O\left((1/\varepsilon_1)^{c_1} n^{c_2}\right) = O\left(n^{c_1+c_2}\right)$. We will use this AND gadget later. Define $M_1 = n^{c_1+c_2+10}$, $M_2 = n^2$, and $m = M_2\Lambda$.

The set cover part Given a SetCover instance, we use a single vertex to represent a subset A_i and a clique of size m to represent each element in U. If an element is in a subset, we create m directed edges from the vertex representing the subset to each the m vertices in the clique representing the element.

The verification part We construct the $(n, \Lambda, a_1(p^* - \varepsilon), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget mentioned. We associate each of the n cliques to one of the n inputs of this AND gadget, such that a matching is formed between the n cliques and the n inputs. For each of the n cliques and its associated input, we choose Λ vertices from the clique, and connect them to the Λ vertices of the associated input by Λ directed edges. We create M_1 vertices and let the output vertex v of the AND gadget be connected to these M_1 vertices with undirected edges. Then, we duplicate the AND gadget and the attached M_1 vertices to a total of M_2 copies such that the vertices at the input ends of the AND gadgets in all these M_2 copies are connected from the different vertices in the n cliques as inputs. This, in particular, justifies our choice of clique size $m = M_2\Lambda$.

The size of the construction To show that the reduction is in polynomial time, it is enough to show that the number of vertices N in the graph G we constructed is a polynomial of n. According to Lemma 6.6, the AND gadget has $O\left((1/\varepsilon_1)^{c_1} n^{c_2+1}\right) = O\left(n^{c_1+c_2+1}\right)$ vertices. We have

$$N = K + mn + M_2\left(O\left(n^{c_1+c_2+1}\right) + M_1\right) = K + mn + \Theta\left(n^{c_1+c_2+12}\right) = \Theta\left(n^{c_1+c_2+12}\right),$$

where $K + mn$ is the size for the set cover part and $M_2\left(O\left(n^{c_1+c_2+1}\right) + M_1\right)$ is the size for the verification part.

Finally, noticing that $N = \Theta\left(n^{c_1+c_2+12}\right)$ and letting $\tau = \frac{1}{c_1+c_2+12}$ (which depends on c_1, c_2, and c_1, c_2 depends only on f), the lemma below immediately concludes Theorem 6.2 for the case $a_1 > 0$ with directed edges.

Lemma 6.8. *If the* SetCover *instance is a* YES *instance, by choosing k seeds appropriately, we can infect at least $\Theta\left(n^{c_1+c_2+12}\right)$ vertices in expectation in the graph G we have constructed; if it is a* NO *instance, we can infect at most $O\left(n^{c_1+c_2+11}\right)$ vertices in expectation for any choice of k seeds.*

Proof. If the SetCover instance is a YES instance, we are able to choose k subsets $\{A_{i_1}, \ldots, A_{i_k}\} \subseteq A$ such that $A_{i_1} \cup \cdots \cup A_{i_k} = U$. We choose the k vertices corresponding to these k subsets as seeds.

We say that a clique representing an element is *activated* if all its m vertices are infected with probabilities more than $p^* - \varepsilon$. If a vertex representing a subset is seeded, for each clique representing the element it covers, each of the m vertices in this clique will be infected with probability a_1. Thus, ma_1 vertices will be infected in expectation. According to Chernoff-Hoeffding inequality, with probability at least $1-\exp\left(-\frac{1}{8}a_1^2m\right)$, there are more than $\frac{1}{2}a_1m$ infected vertices in the clique. If this happens, in the next cascade iteration, each vertex in the clique has more than $\frac{1}{2}a_1m$ infected neighbors, so it will be infected with probability at least $a_{\lfloor\frac{1}{2}a_1m\rfloor} \geq a_{\lfloor a_1n\rfloor} > p^* - \varepsilon$ (notice that $\frac{1}{2}m = \Theta(n^{c_1+c_2+2}) \gg n$). Therefore, if a vertex representing a subset is seeded and a clique representing an element is in this subset, then this clique is activated with probability at least $1-\exp\left(-\frac{1}{8}a_1^2n\right)$.

By our choice of k seeds, each of the clique is activated with probability at least $1-\exp\left(-\frac{1}{8}a_1^2n\right)$. By a union bound, all the n cliques will be activated with probability at least

$$p_{\text{activated}} = 1 - Kn\exp\left(-\frac{1}{8}a_1^2n\right) = \Theta(1).$$

In the highly likely case where all the n cliques are activated, all the vertices at the input ends of all the AND gadgets will be infected with probability more than $a_1(p^*-\varepsilon)$. Since the parameter $p_0 = a_1(p^*-\varepsilon)$ is set for the AND gadget, the output vertex v falls into case (2) in Definition 6.5, which means it will be infected with probability more than $p_2 - \varepsilon_2$. Therefore, all the M_1 vertices connected to v in each of the M_2 copies will be infected with probability at least $a_1(p_2-\varepsilon_2)$, so the expected total number of infected vertices is at least $p_{\text{activated}} \cdot a_1(p_2-\varepsilon_2)M_1M_2 = \Theta\left(n^{c_1+c_2+12}\right)$.

On the other hand, if the SetCover instance is a NO instance, consider any choice of k seeds with k_1 of them in the K vertices representing subsets, k_2 of them in the n cliques, and the remaining $k_3 = k - k_1 - k_2$ of them in the verification part. We first show that at least one clique will not be activated.

The k_3 vertices in the verification part play no role in activating the cliques, as the n cliques are connected to the verification part by directed edges. As for the k_2 vertices in the cliques, since we assume each element in U is in at least one subset, infecting any vertex in any clique is at most as good as infecting the vertex representing the subset covering the element that the clique represents. Therefore, when analyzing the activation of cliques, we can reason as if these k_2 seeds are among the K subsets. Since the SetCover instance is a NO instance, and we have picked $k_1 + k_2 \leq k$ subsets, at least one clique will not be activated.

Among the M_2 AND gadgets, at most k_2 of them take the input vertices which are connected from the k_2 seeds in the cliques. Since these k_2 seeds are infected with probability 1 making these input vertices infect with probability a_1 which may be larger than $\frac{11}{10}a_1(p^*-\varepsilon)$, the outputs of these k_2 AND gadgets are unknown as it falls into neither case (1) nor case (2). We have also assumed k_3 seeds are selected in the verification parts, so we also do not know the outputs of another (at most) k_3 AND gadgets.

For the remaining $M_2 - k_2 - k_3$ AND gadgets, they fall into case (1) by the fact that at least one clique is not activated and our setting $p_0 = a_1(p^* - \varepsilon)$ for the AND gadget. Since we have set the AND gadget parameter $\varepsilon_1 = \frac{1}{n}$, the output vertex v will be infected with probability less than $\frac{1}{n}$, which will infect at most $a_1 \frac{M_1}{n}$ vertices in expectation among the M_1 vertices on the right-hand side of Figure 6.7. Notice that each AND gadget has $O(n^{c_1+c_2+1})$ vertices by Lemma 6.6, and the set cover part has $K + nm$ vertices. In this case, even if all the $K + nm + (M_2 - k_2 - k_3) \cdot O\left(n^{c_1+c_2+1}\right) = O\left(n^{c_1+c_2+3}\right)$ vertices in the set cover part and the $(M_2 - k_2 - k_3)$ AND gadgets are infected, the total number of infected vertices cannot exceed $O\left(n^{c_1+c_2+3}\right) + M_2 \cdot a_1 \frac{M_1}{n} = O\left(n^{c_1+c_2+11}\right)$.

Finally, for those remaining $k_2 + k_3$ AND gadgets whose outputs are unknown, even if all vertices in these $k_2 + k_3$ copies of AND gadgets and their attached M_1 vertices are infected, this total number is still $(k_2 + k_3) \cdot (O\left(n^{c_1+c_2+1}\right) + M_1) = (k_2 + k_3) \cdot O\left(n^{c_1+c_2+10}\right) = O\left(n^{c_1+c_2+11}\right)$. Therefore, if the SetCover instance is a NO instance, we can infect at most $O\left(n^{c_1+c_2+11}\right)$ vertices in G. □

6.4.3 Proof of Theorem 6.2 for $a_1 > 0$ with Undirected Graphs

To prove Theorem 6.2 for undirected graphs, we will need the following *directed edge gadget* which simulates directed edges, and the construction of this gadget also requires the property $a_2 > 2a_1$. This is because the directed edge gadget also uses probability filter gadgets as building blocks.

Definition 6.9. A $(\Upsilon, \epsilon, b, f)$-*directed edge gadget* $\langle u, v\rangle$ takes one vertex u as input and output one vertex v such that the following properties hold.

1. *directed property:* If u is connected to each of the Υ vertices $v_1, \ldots, v_\Upsilon$ by a directed edge gadget $\langle u, v_i\rangle$, and $v_1, \ldots, v_\Upsilon$ are already infected, then u will be infected with probability less than ϵ.

2. If the input u is infected, then the output v will be infected with probability b. Moreover, $b > 0$.

The size of a directed edge gadget is given by the following lemma.

Lemma 6.10. *For any 2-quasi-submodular function f with $a_1 > 0$, any positive integer Υ and any $\epsilon > 0$, there exists $b \in (0,1)$ such that we can construct a $(\Upsilon, \epsilon, b, f)$-directed edge gadget with $\Theta\left(\Upsilon^d (1/\epsilon)^d\right)$ vertices and $\Theta\left(\Upsilon^d (1/\epsilon)^d\right)$ edges, where $d > 1$ is a constant depending only on f.*

We also need the following lemma.

Lemma 6.11. *Given an $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget, for any Υ and ϵ, we can construct a $(\Upsilon, \epsilon, b, f)$-directed edge gadget with $b \in \left(p_2 - \frac{1}{2}\varepsilon_2, p_2\right]$ using $\Theta\left(\Upsilon^d (1/\epsilon)^d\right)$ vertices and $\Theta\left(\Upsilon^d (1/\epsilon)^d\right)$ edges, where $d > 1$ is a constant depending only on f.*

The construction of the directed edge gadget and the proofs of Lemma 6.10 and Lemma 6.11 are deferred to Section 6.4.7.

6.4.3.1 A Reduction from SetCover

According to Lemma 6.7, for $I = n$, there exists a constant $p_2 > 0$, such that if we set $\varepsilon_1 = \frac{1}{n}$ and $\varepsilon_2 = \frac{1}{100} p_2$, we can construct an $(n, \Lambda, p^*(p_2 - \varepsilon_2), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget, where $\Lambda = O\left((1/\varepsilon_1)^{c_1} n^{c_2}\right) = O\left(n^{c_1 + c_2}\right)$. Define $M_2 = n^2$ and $m = M_2 \Lambda$ as before, and we will define M_1 later.

Applying Lemma 6.11, we can construct a (mn, m^{-2}, b, f)-directed edge gadget such that $b \in \left(p_2 - \frac{1}{2}\varepsilon_2, p_2\right]$. The numbers of vertices and edges in this directed edge gadget are both

$$\Theta\left((mn)^d \left(1/m^{-2}\right)^d\right) = \Theta\left(m^{3d} n^d\right) = \Theta\left(n^{(3c_1 + 3c_2 + 7)d}\right).$$

We will use this directed edge gadget exclusively in the construction.

Finally, define $M_1 = n^{(30c_1 + 30c_2 + 70)d}$.

We will construct an undirected graph G similar to the one in the last section, with some modifications. We make the following two modifications:

1. We replace all directed edges in Figure 6.7 by (mn, m^{-2}, b, f)-directed edge gadgets. These consist of 1) the directed edges connecting between the K vertices representing subsets and the n cliques representing elements and 2) the directed edges connecting between the set cover part and the verification part.

2. We use the $(n, \Lambda, p^*(p_2 - \varepsilon_2), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadgets in the verification part instead of the $(n, \Lambda, a_1(p^* - \varepsilon), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadgets.

For the remaining parts of the construction, all the edges, including the ones in the clique, the ones in the AND gadget, and the ones connected to the M_1 vertices on the right-hand side of Figure 6.7, are undirected edges. In particular, we recall that the edges in the AND gadget are undirected.

The size of the construction We show that the total number of vertices in G is still of polynomial size. In the set cover part, we have created at most Kmn directed edge gadgets between the K vertices and the mn vertices in the n cliques. The total size of the set cover part is at most $K + mn + Kmn \cdot \Theta\left(n^{(3c_1+3c_2+7)d}\right) = O\left(n^{(3c_1+3c_2+7)d+c_1+c_2+4}\right)$.

In the verification part, the AND gadget contains $O\left(n^{c_1+c_2+1}\right)$ vertices by Lemma 6.7, the total number of vertices is $M_2(O(n^{c_1+c_2+1}) + M_1) = \Theta\left(n^{(30c_1+30c_2+70)d+2}\right)$.

Since $d > 1$ by Lemma 6.7, N is dominated by the number of vertices in the verification part: $N = \Theta\left(n^{(30c_1+30c_2+70)d+2}\right)$.

Finally, with $\tau = \frac{1}{(30c_1+30c_2+70)d+2}$, Theorem 6.2 for undirected graphs follows immediately from the following lemma.

Lemma 6.12. *If the* SetCover *instance is a* YES *instance, by choosing k seeds appropriately, we can infect $\Theta\left(n^{(30c_1+30c_2+70)d+2}\right)$ vertices in expectation in the graph G; if it is a* NO *instance, we can infect at most $O\left(n^{(30c_1+30c_2+70)d+1}\right)$ vertices in expectation for any choice of k seeds.*

Proof. If the SetCover instance is a YES instance, we are able to choose k subsets $\{A_{i_1}, \ldots, A_{i_k}\} \subseteq A$ such that $A_{i_1} \cup \cdots \cup A_{i_k} = U$. We choose the k vertices corresponding to these k subsets as the seeds. Since the SetCover instance is a YES instance, for each clique, each vertex is connected from a seed by a directed edge gadget, which will be infected with probability b. In each clique, bm vertices will be infected in expectation, and the remaining vertices in the clique will be infected with probability at least $a_{\lfloor bm \rfloor}$, which has limit p^* as $n \to \infty$.

By the same analysis in the proof of Lemma 6.8, with a high probability $p_{\text{activated}} = \Theta(1)$, all the n cliques will be activated such that all vertices in the clique will be infected with probability $p^* - \varepsilon$ for certain $\varepsilon = o(1)$. By our construction and Lemma 6.11, each of the mn vertices that are passed into the input of the AND gadget will be infected with probability

$$(p^* - \varepsilon)b \in \left((p^* - \varepsilon)\left(p_2 - \frac{1}{2}\varepsilon_2\right), (p^* - \varepsilon)p_2\right] \subseteq \left(p^*(p_2 - \varepsilon_2), \frac{11}{10}p^*(p_2 - \varepsilon_2)\right),$$

by noticing that $\varepsilon = o(1)$ and $\varepsilon_2 = \frac{1}{100}p_2 < \frac{1}{10}p_2$ is a constant. Thus, the AND gadget falls into case (2) of Definition 6.5, so the output vertex v of the AND gadget will be infected with probability more than $p_2 - \varepsilon_2$. Therefore, each of the M_1 vertices will be infected with probability $p_{\text{activated}} a_1 (p_2 - \varepsilon_2)$, and the expected total number of infected vertices in those M_2 copies of M_1 vertices is already $p_{\text{activated}} a_1 (p_2 - \varepsilon_2) M_1 M_2 = \Theta\left(n^{(30c_1+30c_2+70)d+2}\right)$.

If the SetCover instance is a NO instance, consider any choice of the k seeds with k_1 seeds in the K vertices representing subsets, k_2 seeds in the directed edge gadgets connecting the K vertices and nm vertices in the n cliques, k_3 seeds in the n cliques, k_4 seeds in the directed edge gadgets between the n cliques in the set cover part and the inputs of the AND gadget in the verification part, and the remaining $k_5 = k - k_1 - k_2 - k_3 - k_4$ seeds in the verification parts. We first aim to show that at least one clique will not be activated with high probability.

When analyzing cliques' activation, it is easy to see that putting k_2 seeds on the directed edge gadgets is at most as good as putting them on the corresponding vertices representing the subsets. Similarly, putting k_4 seeds on the directed edge gadgets connecting the set cover part and the verification part is at most as good as putting them on the corresponding vertices in the cliques, and having $k_3 + k_4$ seeds in the cliques is at most as good as having them in the K vertices representing the subsets covering the elements that those cliques represent. Thus, we can reason as if we have selected $k_1 + k_2 + k_3 + k_4$ subsets in the SetCover problem. Since the SetCover instance is a NO instance, those $k_1 + k_2 + k_3 + k_4 \leq k$ seeds cannot cover all the cliques. As for the k_5 seeds in the verification part, their influences on each vertex in the n cliques is at most m^{-2} based on Definition 6.9, which has remote effect to the cliques, and we will discuss it later.

To show that at least one clique is not activated, it remains to show that the clique not covered by those $k_1 + k_2 + k_3 + k_4$ vertices cannot be activated. For each of those vertices representing subsets that are not picked, since it is connected to at most mn vertices (m vertices in each of the n cliques) by the (mn, m^{-2}, b, f)-directed edge gadgets, it will be infected with probability at most m^{-2} by Definition 6.9. For each vertex in each uncovered clique, it may only be infected due to 1) the influence from one of the K vertices which is not seeded and which is infected with probability at most m^{-2}, or 2) the influence from the k_5 seeds from the verification parts. In particular, it will be infected due to (1) with probability bm^{-2}, and it will be infected due to (2) with probability m^{-2}. By a union bound, the probability that there exist infected vertices in an uncovered clique is at most $m \cdot (bm^{-2} + m^{-2}) = O(m^{-1})$. Since there can be at most n uncovered cliques, the probability that all uncovered cliques contain no infected vertex is at least $p_{\text{no}} = 1 - n \cdot O\left(m^{-1}\right) > 1 - O\left(\frac{1}{n}\right)$. Therefore, with the probability above, there exists at least one clique which is not activated.

In the case that not all cliques are activated, since all the vertices in a not activated clique are infected with probability 0, the corresponding input vertices to the AND gadget are also infected with probability $0b = 0$. The output vertices v in the AND gadgets therefore fall into case (1) in at least $M_2 - k_3 - k_4 - k_5$ copies. Thus, in each of the corresponding $M_2 - k_3 - k_4 - k_5$ copies of the M_1 vertices bundle (on the rightmost of Figure 6.7), the expected number of infected vertices is at most $\varepsilon_1 \cdot M_1 = O\left(n^{(30c_1+30c_2+70)d-1}\right)$. In this case, even if all the vertices in the entire set cover part, the mn directed edge gadgets connecting the two parts, all the M_2 AND-gadgets, and the remaining $k_3 + k_4 + k_5$ copies of the M_1 vertices bundles, the total number of infected vertices is at most

$$\begin{aligned} & Kmn \cdot \Theta\left(n^{(3c_1+3c_2+7)d}\right) + mn \cdot \Theta\left(n^{(3c_1+3c_2+7)d}\right) + M_2 \cdot O\left(n^{c_1+c_2+1}\right) \\ & \qquad + (k_3+k_4+k_5)\left(O\left(n^{c_1+c_2+1}\right) + M_1\right) + (M_2 - k_3 - k_4 - k_5) O\left(n^{(30c_1+30c_2+70)d-1}\right) \\ = & O\left(n^{(3c_1+3c_2+7)d+c_1+c_2+4}\right) + \Theta\left(n^{(3c_1+3c_2+7)d+3}\right) + O\left(n^{c_1+c_2+3}\right) \\ & \qquad + O\left(n^{(30c_1+30c_2+70)d+1}\right) + O\left(n^{(30c_1+30c_2+70)d+1}\right) \\ = & O\left(n^{(30c_1+30c_2+70)d+1}\right). \end{aligned} \tag{6.1}$$

Finally, even assuming all vertices in G are infected in the case that all cliques are activated (which happens with probability $1 - p_{\text{no}} < O\left(\frac{1}{n}\right)$), the expected number of infected vertices is at most

$$p_{\text{no}} \cdot O\left(n^{(30c_1+30c_2+70)d+1}\right) + (1 - p_{\text{no}})N = O\left(n^{(30c_1+30c_2+70)d+1}\right),$$

which concludes the lemma. □

6.4.4 Constructions of Some Other Required Gadgets

Before constructing the AND gadget and the directed edge gadget, we need some other gadgets. In this section and the next two sections, graph with undirected edges are considered.

We will construct the *probability scaling down gadget* and the *probability filter gadget*, which are used to construct the AND gadget and the directed edge gadget. The relation of these gadgets are shown in Figure 6.8.

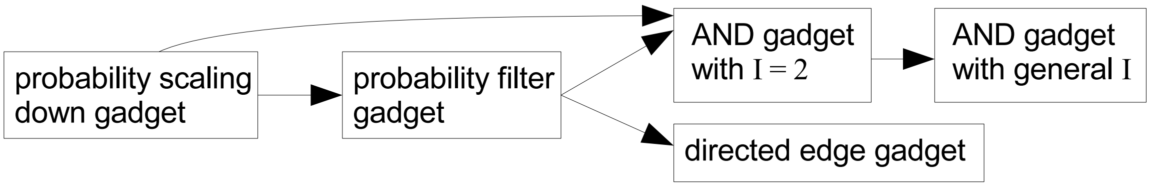

probability scaling down gadget
probability filter gadget
AND gadget with I = 2
AND gadget with general I
directed edge gadget

Figure 6.8: The relation of all the gadgets defined

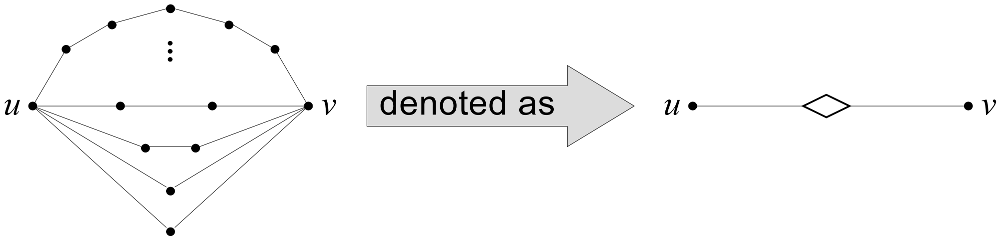


Figure 6.9: The probability scaling down gadget

6.4.4.1 Probability Scaling Down Gadget

We first define and construct the following *probability scaling down gadget* which is an essential component of both the AND gadget and the directed edge gadget.

Definition 6.13. The (α, ε, f)*-probability scaling down gadget* takes one vertex u as input and output a vertex v such that

- if u is infected with probability p_u, v will be infected with probability $p_v \in (\alpha p_u - \varepsilon, \alpha p_u]$.

Lemma 6.14. *For any 2-quasi-submodular function f with $a_1 > 0$, any constant $\varepsilon > 0$ and any α with $0 < \alpha \leq p^*$, there exists an (α, ε, f)-probability scaling down gadget with constant numbers of vertices and edges.*

Proof. To construct this gadget, we iteratively add paths from u to v, where a path of length ℓ consists of $\ell - 1$ vertices $w_1, \ldots, w_{\ell-1}$ and ℓ edges $(u, w_1), (w_1, w_2), \ldots, (w_{\ell-1}, v)$. Given p_u, by repeatedly adding paths from u to v, we are increasing p_v. In each iteration i, we add a path of length ℓ_i from u to v, where ℓ_i is the minimum length to maintain $p_v \leq \alpha p_u$. That is, either it is true that $p_v > \alpha p_u$ if a path of length $\ell_i - 1$ was added, or $\ell_i = 2$ which is already the minimum length a path can ever be. The iterative process ends if $p_v \in (\alpha p_u - \varepsilon, \alpha p_u]$, and it is straightforward to check that such process will end as long as $\alpha \in (0, p^*]$. Figure 6.9 illustrates the probability scaling down gadget.

The size of the probability scaling down gadget depends on the influence function

f and the small constant ε. Since f is fixed in advance, the size of this gadget is constant. □

Remark 6.15. The probability scaling down gadget is symmetric. Given $p_v = \alpha p_u$, then $p_u = \alpha p_v$ if v becomes the input and u becomes the output.

6.4.4.2 Probability Filter Gadget

Based on the probability scaling down gadget, we can construct the following *probability filter gadget.*

Definition 6.16. A $(\Lambda, p_1, p_2, \varepsilon_1, \varepsilon_2, f)$*-probability filter gadget* takes Λ vertices as input, and outputs a vertex such that

1. if each vertex in the Λ inputs is infected independently with a same probability less than p_1, then the vertex on the output end will be infected with a probability less than ε_1;

2. if each vertex in the Λ inputs is infected independently with a same probability in $(p_1, p_2]$, then the vertex on the output end will be infected with a probability in $(p_2 - \varepsilon_2, p_2]$.

We aim to show the following lemma in this subsection.

Lemma 6.17. *Given any 2-quasi-submodular influence function f with $a_1 > 0$, any constant $\varepsilon_2 > 0$, any $\varepsilon_1 > 0$, and any ratio $r > 0$, we can construct a $(\Lambda, p_1, p_2, \varepsilon_1, \varepsilon_2, f)$-probability filter gadget with $p_2/p_1 > r$ and $\Lambda = O((1/\varepsilon_1)^c)$, and this probability filter gadget contains $O((1/\varepsilon_1)^c)$ vertices and $O((1/\varepsilon_1)^c)$ edges, where c is a constant.*

To construct the probability filter gadget, we first construct the gadget shown in Figure 6.10, which is the building block of this gadget. We will call this building block *probability separation block.* As shown in the figure, this building block takes h vertices as input and outputs one vertex. Particularly, we apply h probability scaling down gadgets to “scale down” the probabilities of all input vertices' infection by a factor of α, and then connect those vertices to the output vertex.

The probability filter gadget consists of ℓ layers such that the i-th layer consists of $h^{\ell-i}$ such probability separation blocks, where the output vertices of every h probability separation blocks in the i-th layer are the input of a probability separation block in the $(i+1)$-th layer. Because there are $h^{\ell-1}$ probability separation blocks in the first layer, the probability filter gadget takes $\Lambda = h^{\ell}$ vertices as input. The probability

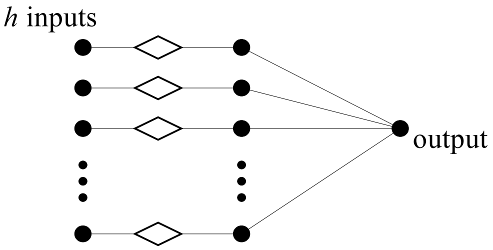


Figure 6.10: The probability separation block

filter gadget outputs a single vertex after ℓ layers. We will tune the value of α, h and ℓ such that the two properties in Definition 6.16 hold for certain thresholds p_1 and p_2.

For each probability separation block, suppose each of the h vertices in the input are infected with probability x independently, and let $y = y(x)$ be the probability that the output vertex is infected. We aim to tune the value of α and h such that the graph of $y(x)$ looks like Figure 6.11.

By considering the number of infected neighbors of the output vertex, it is straightforward to see that

$$y = \sum_{i=1}^{h} \binom{h}{i} a_i (\alpha x)^i (1 - \alpha x)^{h-i}. \tag{6.2}$$

For sufficiently small x, we have

$$y = h\alpha a_1 x + \frac{h(h-1)}{2}\alpha^2(a_2 - 2a_1)x^2 + o(x^2).$$

Choosing a sufficiently small constant $\delta > 0$ and choosing α (h will be set in the future) to satisfy $h\alpha a_1 = 1 - \delta$, we have

$$y - x = -\delta x + \frac{h(h-1)}{2}\alpha^2(a_2 - 2a_1)x^2 + o(x^2).$$

Since $y - x = -\delta x + o(x)$, we can see that $y < x$ for small enough x. On the other hand, for sufficiently large h and sufficiently small δ (and adjusting α such that

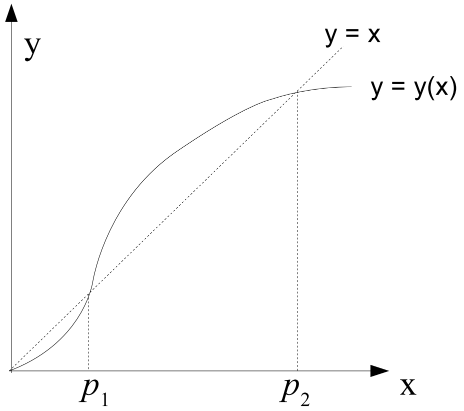


Figure 6.11: The output probability y versus the input probability x

$h\alpha a_1 = 1 - \delta$ still holds[4]), we have

$$\frac{h(h-1)}{2}\alpha^2 = \frac{1}{2}h^2\alpha^2 - \frac{h}{2}\alpha^2 = \frac{(1-\delta)^2}{2a_1^2} - \frac{(1-\delta)^2}{2ha_1^2} > \frac{1}{3a_1^2}.$$

We can see from the following that $y > x$ after a while as x increases.

$$x_1 = \frac{6a_1^2}{a_2 - 2a_1}\delta \qquad \Longrightarrow \qquad \begin{aligned} y(x_1) - x_1 &> -\delta x_1 + \frac{a_2 - 2a_1}{3a_1^2}x_1^2 + o(x_1^2) \\ &= \frac{6a_1^2}{a_2 - 2a_1}\delta^2 + o(\delta^2) \\ &> 0. \end{aligned}$$

Notice that the 2-quasi-submodularity of f makes sure $a_2 > 2a_1$ such that x_1 is positive.

We have seen that $y < x$ for small enough x, and $y > x$ after x increases. There must be a threshold p_1 such that $y = x$ at $x = p_1$ by the Intermediate Value Theorem. On the other hand, y is upper bounded by p^* while x can be as large as 1, so $y \leq x$ for sufficiently large x. The Intermediate Value Theorem suggests there exists another threshold $x = p_2 > p_1$ such that $y = x$. Consequently, Figure 6.11 indeed represents the graph of $y(x)$ for the proper choices of α and h.

Finally, from the graph in Figure 6.11, we can see that the infection probability of the output vertices in the i-th layer increases as i increases, if all the $\Lambda = h^\ell$ input

[4] According to Definition 6.13 and Lemma 6.14, given the scale α^* for which we want to adjust to, we can construct a probability scaling down gadget such that the actual scale α is arbitrarily close to α^*. Although we cannot make the adjustment exact, a close enough approximation would still satisfy our purpose here, as all we want is δ to be small enough, or $h\alpha a_1$ to be close enough to 1.

vertices are infected with an independent probability larger than p_1. In contrast, the infection probability of the output vertices in the i-th layer decreases as i increase, if all the $\Lambda = h^\ell$ input vertices are infected with an independent probability less than p_1. By setting ℓ large enough, we can make both (1) and (2) in Definition 6.16 hold.

Before we move on, we show some properties of the thresholds p_1 and p_2, and our objective is to show the following proposition which is a part of Lemma 6.17.

Proposition 6.18. *For any large ratio $r > 0$, we can find h and α such that $p_2/p_1 > r$.*

By the calculation above, the proposition below follows immediately.

Proposition 6.19. $p_1 < \frac{6a_1^2}{a_2 - 2a_1}\delta$.

We also have the following lower bound for p_2.

Proposition 6.20. *By choosing h sufficiently large and δ sufficiently small, we have $p_2 > a_1\gamma$ for any γ such that*

$$a_2(1 - e^{-\gamma} - \gamma e^{-\gamma}) - a_1(\gamma - \gamma e^{-\gamma}) > 0.$$

Proof. By replacing all $a_3, a_4, \dots, a_h$ to a_2 in Equation (6.2), we have

$$\begin{aligned}
y &\geq \sum_{i=1}^{h} \binom{h}{i} a_2 (\alpha x)^i (1 - \alpha x)^{h-i} - \binom{h}{1} (a_2 - a_1)\alpha x (1 - \alpha x)^{h-1} \\
&= a_2 \left(\sum_{i=0}^{h} \binom{h}{i} (\alpha x)^i (1 - \alpha x)^{h-i} - (1 - \alpha x)^h \right) - h(a_2 - a_1)\alpha x (1 - \alpha x)^{h-1} \\
&= a_2 - a_2 (1 - \alpha x)^h - h(a_2 - a_1)\alpha x (1 - \alpha x)^{h-1} \\
&= a_2 - a_2 \exp(h \ln(1 - \alpha x)) - h(a_2 - a_1)\alpha x \exp((h-1)\ln(1 - \alpha x)) \\
&\geq a_2 - a_2 \exp(-h\alpha x) - h(a_2 - a_1)\alpha x \exp(-\alpha x (h-1)). \\
&\hspace{20em} \text{(concavity of ln function)}
\end{aligned}$$

Letting $x = a_1\gamma$, we have

$$\begin{aligned}
y - x &\geq a_2 - a_2 \exp(-\gamma(1-\delta)) - (a_2 - a_1)(1-\delta)\gamma \exp\left(\gamma(1-\delta)\left(\frac{1}{h} - 1\right)\right) - a_1\gamma \\
&\hspace{16em} \text{(since } x = a_1\gamma \text{ and } h\alpha a_1 = 1 - \delta\text{)} \\
&> a_2(1 - e^{-\gamma} - \gamma e^{-\gamma}) - a_1(\gamma - \gamma e^{-\gamma}) - \epsilon,
\end{aligned}$$

where in the last step, for any $\epsilon > 0$, we can find small enough δ and large enough h to make the inequality holds. Rigorously, we have $1-\delta \to 1$ and $\frac{1}{h} \to 0$ for $\delta \to 0$ and $h \to \infty$. The expression in the second last step is a continuous function, which has limit $a_2(1-e^{-\gamma}-\gamma e^{-\gamma})-a_1(\gamma-\gamma e^{-\gamma})$, and the last step is obtained by the definition of limit.

Therefore, $y-x>0$ for any $x>p_1$ with $x=a_1\gamma$, where γ satisfies

$$a_2(1-e^{-\gamma}-\gamma e^{-\gamma})-a_1(\gamma-\gamma e^{-\gamma})>0,$$

which implies the proposition. □

We remark that there always exists γ satisfying the inequality in Proposition 6.20. To see this, we show that $\Phi(\gamma) := a_2(1-e^{-\gamma}-\gamma e^{-\gamma})-a_1(\gamma-\gamma e^{-\gamma})>0$ when γ is sufficiently small. By straightforward calculations, we have $\Phi(0)=\Phi'(0)=0$ and $\Phi''(0)=a_2-2a_1>0$, which means $\Phi(0)=0$ and Φ is increasing on $[0,\gamma_0)$ for some small γ_0, which further implies that Φ is positive on $[0,\gamma_0)$.

Proposition 6.19 implies that we can construct the probability filter gadget with arbitrarily small p_1 by setting δ small. On the other hand, Proposition 6.20 implies that p_2 can be made larger than some number depending only on a_1 and a_2, which in particular can be considerably larger than p_1, which yields Proposition 6.18.

Finally, we are ready to show Lemma 6.17.

Proof of Lemma 6.17. The possibility of this construction is straightforward, as the construction is already made explicit in this section. It remains to show that the gadget contains $O((1/\varepsilon_1)^c)$ vertices and $O((1/\varepsilon_1)^c)$ edges, and $\Lambda = O((1/\varepsilon_1)^c)$.

Since ε_2 is a constant, we only need constantly many layers such that the input probability x increases to more than $p_2-\varepsilon_2$, if x is initially larger than p_1.

To investigate how many layers are needed to make x decreases to less than ε_1 in the case x is initially smaller than p_1, recall that in each layer of the probability filter gadget, the input probability x is updated to y such that $y-x=-\delta x+o(x)$ for sufficiently small x, so each time x is decreased by a factor of $(1-\delta)$. After a constant number of layers, x will be sufficiently small such that the term $o(x)$ is negligible, and after another $\frac{\log(1/\varepsilon_1)}{\log(1/(1-\delta))}$ layers, x will decrease by a factor of $(1-\delta)^{\frac{\log(1/\varepsilon_1)}{\log(1/(1-\delta))}}=\varepsilon_1$, which makes the value of x much smaller than ε_1. Therefore, we need at most $\ell = O(\log(1/\varepsilon_1))$ layers. Let χ_v, χ_e be the number of vertices and edges respectively in a probability separation block shown in Figure 6.10, and they are both constants according to Lemma 6.14. The total number of vertices in a probability filter gadget is

$$\sum_{i=1}^{\ell} \chi_v \cdot h^{\ell-i} = \chi_v \frac{h^\ell - 1}{h-1} = \Theta\left(h^\ell\right) = O((1/\varepsilon_1)^c),$$

and the total number of edges has the same asymptotic bound by the same calculation above, with χ_v changed to χ_e. Thus, we conclude that the gadget contains $O((1/\varepsilon_1)^c)$ vertices and $O((1/\varepsilon_1)^c)$ edges.

For Λ, we have $\Lambda = h^\ell = O((1/\varepsilon_1)^c)$ by our construction, which concludes the last part of the lemma. □

6.4.5 Construction of the AND Gadget with $I = 2$

In this section, we construct the AND gadget with parameter $I = 2$. The AND gadget makes use of a single probability filter gadget with the same choices of parameters Λ, p_2, ε_1, ε_2 and f. The AND gadget takes two sets I_1, I_2 of vertices as inputs, and each set has $\Lambda = h^\ell$ vertices. Let $I_1 = \{u_1, u_2, \ldots, u_\Lambda\}$ and $I_2 = \{v_1, v_2, \ldots, v_\Lambda\}$. We create Λ vertices $w_1, w_2, \ldots, w_\Lambda$ and create two edges $(u_i, w_i), (v_i, w_i)$ for each $i = 1, 2, \ldots, \Lambda$. We apply the probability scaling down gadgets to create another Λ vertices $w'_1, w'_2, \ldots, w'_\Lambda$ such that $p(w'_i) = \beta p(w_i)$ for each $i = 1, 2, \ldots, \Lambda$, where β is set to the value such that

$$\beta\varphi_T^+(p_0) < p_2, \qquad \beta\varphi_T^-(p_0) > p_1, \qquad \text{and} \qquad \beta\varphi_F^+(p_0) < p_1,$$

where

$$\begin{aligned}
\varphi_T^+(p_0) &= (a_2 - 2a_1)\left(\frac{11}{10}p_0\right)^2 + 2a_1\left(\frac{11}{10}p_0\right), \\
\varphi_T^-(p_0) &= (a_2 - 2a_1)p_0^2 + 2a_1 p_0, \\
\varphi_F^+(p_0) &= \frac{11}{20}(a_2 - 2a_1)p_0^2 + \frac{16}{10}a_1 p_0.
\end{aligned}$$

The construction is shown in Figure 6.12.

Notice that if all u_i and v_i are infected with an independent probability in the interval $(p_0, \frac{11}{10}p_0)$, that is, the inputs I_1, I_2 fall into case (2) in Definition 6.5, w_i will be infected with probability

$$\begin{aligned}
p(w_i) &= a_2 p(u_i)p(v_i) + a_1 p(u_i)(1 - p(v_i)) + a_1 p(v_i)(1 - p(u_i)) \\
&= (a_2 - 2a_1)p(u_i)p(v_i) + a_1 p(u_i) + a_1 p(v_i),
\end{aligned}$$

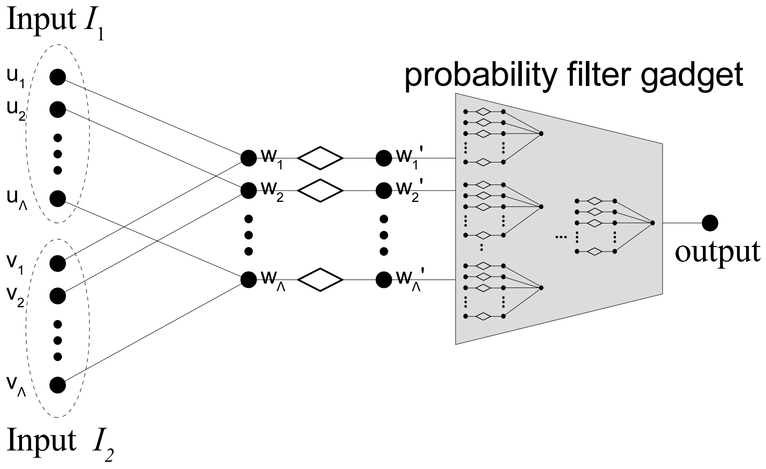


Figure 6.12: AND gadget with $I = 2$

which is in the interval $\big(\varphi_T^-(p_0), \varphi_T^+(p_0)\big)$.

On the other hand, if one of u_i and v_i is infected with probability less than $\frac{1}{2}p_0$ and the other one is infected with probability less than $\frac{11}{10}p_0$, that is, the inputs I_1, I_2 fall into case (1) in Definition 6.5, w_i will be infected with probability

$$\begin{aligned} p(w_i) &= (a_2 - 2a_1)p(u_i)p(v_i) + a_1 p(u_i) + a_1 p(v_i) \\ &< \frac{11}{20}(a_2 - 2a_1)p_0^2 + \frac{16}{10}a_1 p_0 \\ &= \varphi_F^+(p_0). \end{aligned}$$

Given that $\big(\beta\varphi_T^-(p_0), \beta\varphi_T^+(p_0)\big) \subseteq (p_1, p_2)$ and $\beta\varphi_F^+(p_0) < p_1$, it is now straightforward to check that the two properties (1) and (2) in Definition 6.5 hold for $I = 2$, since the probability filter gadget will "filter" the two probabilities such that one goes to a value less than ε_1 and the other goes into $(p_2 - \varepsilon_2, p_2]$.

By our construction of probability scaling down gadget, the factor must satisfy $\beta \le p^*$. It seems worrying that $\big(\varphi_T^-(p_0), \varphi_T^+(p_0)\big)$ and $\varphi_F^+(p_0)$ will be both scaled down to smaller than p_1 even if we take maximum $\beta = p^*$. Indeed, Proposition 6.18 and Proposition 6.19 ensure that this cannot happen, as we can always make p_1 small enough by making δ small enough. We remark here that the choice of δ depends on p_0 and p^* (it needs to be considerably smaller than some polynomial of p_0 such that $\big(\varphi_T^-(p_0), \varphi_T^+(p_0)\big)$ and $\varphi_F^+(p_0)$ can be scaled down to different sides of p_1), where p^* depends only on f.

Now we prove the following lemma, which is a special case of Lemma 6.6 with $I = 2$.

Lemma 6.21. *Given any 2-quasi-submodular function f with $a_1 > 0$ and any constant threshold $p_0 > 0$, there exists a constant $p_2 > 0$ depending on p_0 and f such that for any constant $\varepsilon_2 > 0$ and any $\varepsilon_1 > 0$, we can construct a $(2, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget with $\Lambda = O\left((1/\varepsilon_1)^{c_1}\right)$, and the number of vertices and edges in this AND gadget are both $O\left((1/\varepsilon_1)^{c_1}\right)$, where c_1 is a constant.*

Proof. The existence of this AND gadget is shown by the explicit construction in this section.

To show that p_2 only depends on p_0 and f, notice that it depends on h, δ and f (in particular, a_1 and a_2 only) according to Proposition 6.20. Additionally, h, α are selected such that $\delta = 1 - h\alpha a_1$ is small enough, and we have remarked just now that δ depends on p_0 and f. Therefore, p_2 only depends on p_0 and f, as the graph $y = y(x)$ determines the value of p_2.

For the size of this AND gadget and the input size Λ, the size of this AND gadget is the size of a probability filter gadget plus 3Λ for those u_i, v_i, w_i, and the size of each of both input sets is Λ. Therefore, Lemma 6.17 implies the second part of this lemma. □

Remark 6.22 (Remark of Lemma 6.21)**.** Lemma 6.21 shows that when constructing a $(2, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget, we are free to set up the parameter p_0, and the parameter p_2 will be determined. *After p_2 is determined*, we are still free to choose $\varepsilon_1, \varepsilon_2$, and Λ will be then determined. In fact, the two parameters $\varepsilon_1, \varepsilon_2$ decides the number of layers needed in the probability filter gadget, and we can achieve (1) and (2) in Definition 6.5 for *any* valid function $y(x)$ with two intersections to the line $y = x$ as it is in Figure 6.11. That is the reason why we can choose $\varepsilon_1, \varepsilon_2$ after p_2 is determined. In particular, for the same function $y(x)$ but different $\varepsilon_1, \varepsilon_2$, we just need the AND gadgets with different numbers of layers in their inner probability filter gadgets. We will make use of this observation to construct AND gadgets with the same parameters p_0, p_2, f but different $\varepsilon_1, \varepsilon_2$ in the next section.

To conclude this section, we show that we can also construct a $(2, \Lambda, p_2 - \varepsilon_2, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget and a $(2, \Lambda, p^*(p_2 - \varepsilon_2), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget which will be used in the next section. Notice that Lemma 6.21 does not imply the possibility of constructing this AND gadget, as p_2's existence is supposed to depend on the third parameter, which now become $p_2 - \varepsilon_2$ and $p^*(p_2 - \varepsilon_2)$, two constants related to p_2.

Lemma 6.23. *Given any 2-quasi-submodular influence function f and any constant threshold $p_0 > 0$, we can construct a $(2, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget, a $(2, \Lambda, p_2 -$*

$\varepsilon_2, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget and a $(2, \Lambda, p^*(p_2 - \varepsilon_2), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget with the same parameters $\Lambda, p_2, \varepsilon_1, \varepsilon_2$.

Proof. The three AND gadgets are only different at the third parameter, which is the input threshold determining which of the two cases (1) and (2) in Definition 6.5 the inputs fall into. By our construction, we can use the same structure for the three AND gadgets, except that we use three different scaling down factors $\beta_1, \beta_2, \beta_3$ for the different thresholds p_0, $p_2 - \varepsilon_2$ and $p^*(p_2 - \varepsilon_2)$. In particular, the three probability filter gadgets inside the three AND gadgets can be exactly the same, provided that the "gap" p_2/p_1 is large enough such that

- $\big(\beta_1\varphi_T^-(p_0), \beta_1\varphi_T^+(p_0)\big)$ and $\beta_1\varphi_F^+(p_0)$ are on the different sides of p_1,
- $\big(\beta_2\varphi_T^-(p_2 - \varepsilon_2), \beta_2\varphi_T^+(p_2 - \varepsilon_2)\big)$ and $\beta_2\varphi_F^+(p_2 - \varepsilon_2)$ are on the different sides of p_1, and
- $\big(\beta_3\varphi_T^-(p^*(p_2 - \varepsilon_2)), \beta_3\varphi_T^+(p^*(p_2 - \varepsilon_2))\big)$ and $\beta_3\varphi_F^+(p^*(p_2 - \varepsilon_2))$ are on the different sides of p_1.

We know that this is always possible by Proposition 6.18.

As the same probability filter gadget is used in the two AND gadgets, the four parameters $\Lambda, p_2, \varepsilon_1, \varepsilon_2$, which are inherited from the probability filter gadget by our construction, are identical for the three AND gadgets. □

6.4.6 Construction of the AND Gadget with General I of an Integer Power of 2

In this section, we construct the AND gadget in Definition 6.5 with general I that is an integer power of 2.

An $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget is a $(\log_2 I)$-level AND circuit using 2-set-input AND gadgets constructed in the previous section as building block. We will use three different types of 2-set-input AND gadgets.

- Type A: $(2, \Lambda_0, p_0, p_2, \frac{1}{3}(p_2 - \varepsilon_2), \varepsilon_2, f)$-AND gadget.
- Type B: $(2, \Lambda_0, p_2 - \varepsilon_2, p_2, \frac{1}{3}(p_2 - \varepsilon_2), \varepsilon_2, f)$-AND gadget.
- Type C: $(2, \Lambda_C, p_2 - \varepsilon_2, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget.

Lemma 6.23 indicates that we can construct A and B, and by Lemma 6.21 Λ_0 is a constant since $\frac{1}{3}(p_2 - \varepsilon_2)$ is a constant. By Lemma 6.21 and its remark, we can construct C based on B by adjusting the number of layers in the inner probability filter gadget, and $\Lambda_C = O\left((1/\varepsilon_1)^{c_1}\right)$ for some constant c_1.

Figure 6.13 shows the construction of this AND gadget. The type and the number of AND gadgets in each of the $\log_2 I$ levels are set as follows:

- Level ($\log_2 I$): A single AND gadget of Type C is constructed.
- Level ($\log_2 I - 1$): 2 groups of Λ_C Type B AND gadgets are constructed, and the output vertices in each group are connected to each of the input ends I_1, I_2 of the AND gadget in Level ($\log_2 I$).
- Level ($\log_2 I - 2$): 2^2 groups of $\Lambda_0\Lambda_C$ Type B AND gadgets are constructed, and the output vertices in each group are connected to each of the input ends I_1, I_2 of the AND gadgets in each of the 2 groups in Level ($\log_2 I - 1$).
- Level ($\log_2 I - 3$): 2^3 groups of $\Lambda_0^2\Lambda_C$ Type B AND gadgets are constructed, and the output vertices in each group are connected to each of the input ends I_1, I_2 of the AND gadgets in each of the 2^2 groups in Level ($\log_2 I - 2$).
- $\cdots$
- Level 2: $2^{\log_2 I - 2}$ groups of $\Lambda_0^{\log_2 I - 3}\Lambda_C$ Type B AND gadgets are constructed, and the output vertices in each group are connected to each of the input ends I_1, I_2 of the AND gadgets in each of the $2^{\log_2 I - 3}$ groups in Level 3.
- Level 1: $2^{\log_2 I - 1}$ groups of $\Lambda_0^{\log_2 I - 2}\Lambda_C$ Type A AND gadgets are constructed, and the output vertices in each group are connected to each of the input ends I_1, I_2 of the AND gadgets in each of the $2^{\log_2 I - 2}$ groups in Level 2.

Finally, the two input sets I_1, I_2 in each of the $2^{\log_2 I - 1} = \frac{I}{2}$ AND gadget groups in Level 1 form two of the I input sets for the $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget we are constructing, and the output vertex of the Type C AND gadget in Level ($\log_2 I$) is the output of the $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget.

We now show that (1) and (2) in Definition 6.5 hold.

1. If all the vertices in all I input sets are infected with independent probabilities less than $\frac{11}{10}p_0$, and the infection probabilities of the vertices in at least one set are less than $\frac{1}{2}p_0$, then the Type A AND gadgets in at least one group in

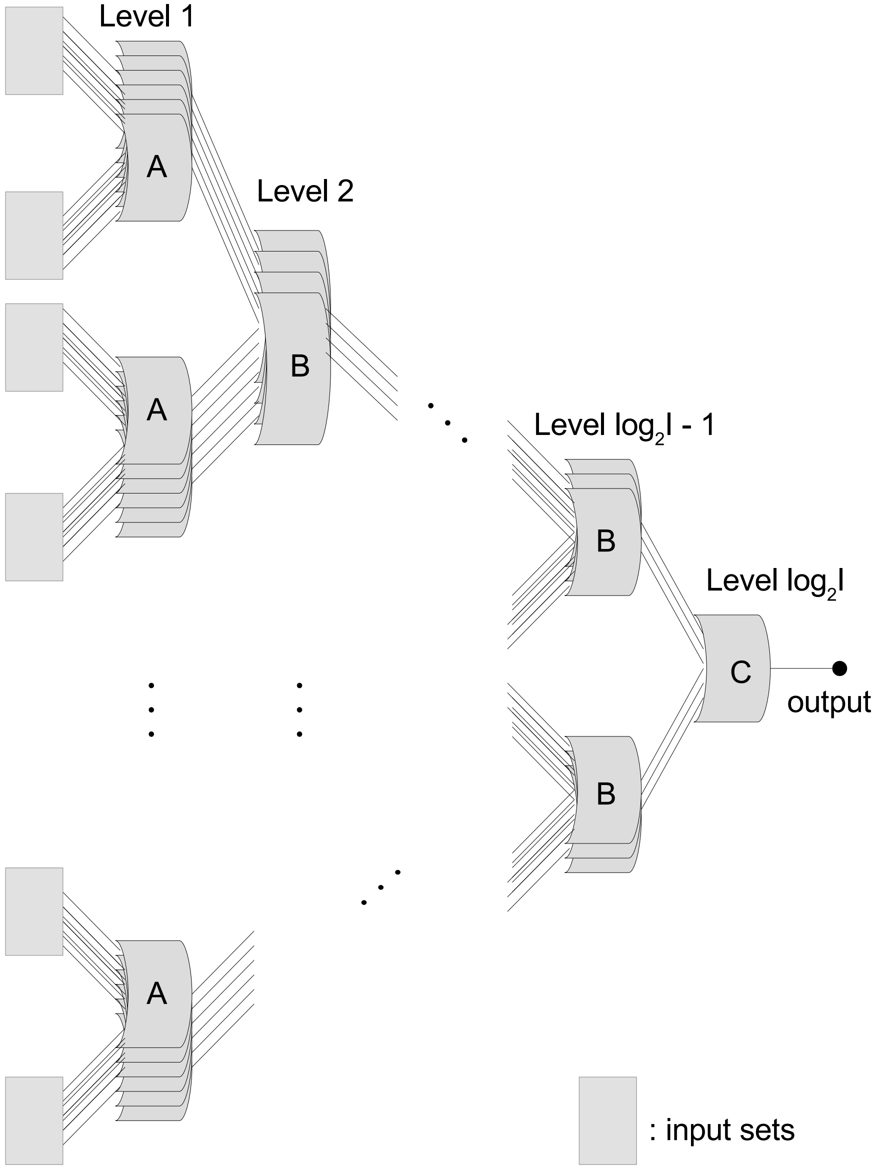


Figure 6.13: The $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget

Level 1 will output vertices with infection probabilities less than $\frac{1}{3}(p_2 - \varepsilon_2)$. Since the threshold (the third parameter) of Type B AND gadgets is set to $(p_2 - \varepsilon_2)$ and $\frac{1}{3}(p_2 - \varepsilon_2) < \frac{1}{2}(p_2 - \varepsilon_2)$, the Type B AND gadgets in at least one group in each of Level $2, 3, \ldots, \log_2 I - 1$ will output vertices with infection probabilities less than $\frac{1}{3}(p_2 - \varepsilon_2)$. Finally, at least one of the two input sets for the Type C AND gadget in Level $(\log_2 I)$ will be infected with probabilities less than $\frac{1}{3}(p_2 - \varepsilon_2)$, which is less than $\frac{1}{2}(p_2 - \varepsilon_2)$. Thus, the output of the entire $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget is a vertex with infection probabilities less than ε_1, which implies (1) in Definition 6.5.

2. If all the vertices in all I input sets are infected with independent probabilities in $(p_0, \frac{11}{10}p_0)$, all the Type A AND gadgets in Level 1 will output vertices with infection probabilities in $(p_2 - \varepsilon_2, p_2]$. Since $(p_2 - \varepsilon_2, p_2] \subseteq \left(p_2 - \varepsilon_2, \frac{11}{10}(p_2 - \varepsilon_2)\right)$ for small enough ε_2,[5] all the Type B AND gadgets in each of Level $2, 3, \ldots, \log_2 I - 1$ will output vertices with infection probabilities in $(p_2 - \varepsilon_2, p_2]$. Finally, the Type C AND gadget in Level $(\log_2 I)$ will output a vertex with infection probability in $(p_2 - \varepsilon_2, p_2]$.

Finally, we prove Lemma 6.6 and Lemma 6.7 in Section 6.4.2.

Proof of Lemma 6.6. The existence of the $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget is proved by the explicit construction above. It remains to show that the number of vertices and edges in this AND gadget is $O\left((1/\varepsilon_1)^{c_1} I^{c_2+1}\right)$, and the input size is $\Lambda = O\left((1/\varepsilon_1)^{c_1} I^{c_2}\right)$.

By Lemma 6.21, the number of vertices and edges in the Type A and B AND gadgets are constants, since the parameter $\frac{1}{3}(p_2 - \varepsilon_2)$ is a constant. Let χ be a constant upper bound for these. As for Type C AND gadget, it has $O\left((1/\varepsilon)^{c_1}\right)$ vertices and edges by Lemma 6.21. Since there are $2^{\log_2 I - i} \Lambda_0^{\log_2 I - i - 1} \Lambda_C$ AND gadgets in Level i and $\Lambda_C = O\left((1/\varepsilon)^{c_1}\right)$ as mentioned, the total number of vertices and edges have the following bound.

$$O\left((1/\varepsilon)^{c_1}\right) + \sum_{i=1}^{\log_2 I - 1} \chi \cdot 2^{\log_2 I - i} \Lambda_0^{\log_2 I - i - 1} \Lambda_C < \chi \Lambda_C \cdot (2\Lambda_0)^{\log_2 I} = O\left((1/\varepsilon_1)^{c_1} I^{c_2+1}\right),$$

where $c_2 = \log_2 \Lambda_0$ is a constant.

[5] If the parameter ε_2 in the $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget we are constructing is not small enough to satisfy this, we can replace ε_2 with another smaller ε_2' and instead construct a $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2', f)$-AND gadget. Notice that the description 2 of Definition 6.5 implies that a $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2', f)$-AND gadget is also a valid $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget for $\varepsilon_2' < \varepsilon_2$.

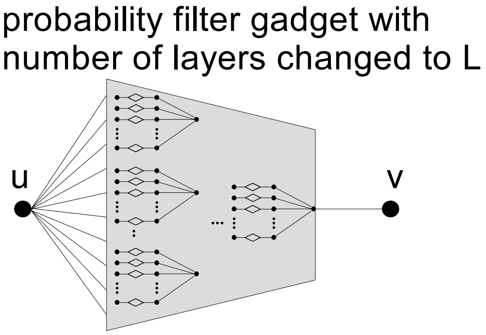


Figure 6.14: The directed edge gadget $\langle u, v\rangle$

As for Λ, there are $\Lambda_0^{\log_2 I-2}\Lambda_C$ AND gadgets in each of the $\frac{I}{2}$ groups in Level 1, and each of these AND gadgets takes Λ_0 vertices as one of the two inputs. Therefore, we have

$$\Lambda = \Lambda_0 \cdot \Lambda_0^{\log_2 I-2}\Lambda_C = O\left((1/\varepsilon_1)^{c_1} I^{c_2}\right),$$

which concludes the last part of the lemma. □

Proof of Lemma 6.7. Based on Lemma 6.23, by changing all the Type A $(2, \Lambda_0, p_0, p_2, \frac{1}{3}(p_2-\varepsilon_2), \varepsilon_2, f)$-AND gadgets in Level 1 to the Type A' $(2, \Lambda_0, p^*(p_2-\varepsilon_2), p_2, \frac{1}{3}(p_2-\varepsilon_2), \varepsilon_2, f)$-AND gadgets, we obtain an $(I, \Lambda, p^*(p_2-\varepsilon_2), p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget.

The size of the AND gadget only changes by a constant, as the only difference between the two AND gadgets are the different probability scaling down gadgets used for different β for A and A'. Since the probability scaling down gadget has a constant size, we conclude the second half of the lemma. □

6.4.7 Construction of Directed Edge Gadget

The $(\Upsilon, \epsilon, b, f)$-directed edge gadget in Definition 6.9 can be constructed by modifying the number of layers in the $(\Lambda, p_1, p_2, \varepsilon_1, \varepsilon_2, f)$-probability filter gadget in Definition 6.16. While still keeping the parameter h and α such that $a_1 h\alpha = 1-\delta$ in the probability separation block of the probability filter gadget, we modify the number of layers in the circuit to $L = \frac{\log(\Upsilon/\epsilon)}{\log(1/(1-\delta))} + 1$.

To construct a directed edge gadget $\langle u, v\rangle$, we connect u to all the h^L inputs to the circuit, and let v be the output. The construction of directed edge gadget is shown in Figure 6.14.

To show property (1) in Definition 6.9, suppose u is connected to Υ infected vertices $v_1, v_2, \ldots, v_\Upsilon$ by the directed edge gadgets. If the vertices in the i-th layer are infected with probability x_i, then the vertices in the $(i-1)$-th layer will be infected with probability $x_{i-1} = a_1 \alpha x_i$, which can be easily seen from Figure 6.10 and by noticing the symmetric property of probability scaling down gadgets mentioned in Remark 6.15. Therefore, each vertices in the first level that are adjacent to u will be infected with probability $(a_1\alpha)^L$. Since there are h^L vertices in the first level and u is assumed to be connected to Υ vertices by the directed edge gadgets, the expected number of u's infected neighbors is

$$\mathbb{E}[\text{num of infected neighbors}] = \Upsilon h^L (a_1\alpha)^L = \Upsilon(1-\delta)^L = \epsilon(1-\delta) < \epsilon,$$

where recall that we have set

$$L = \frac{\log(\frac{\Upsilon}{\epsilon})}{\log \frac{1}{1-\delta}} + 1.$$

Therefore, by Markov's inequality, the probability that u has infected neighbor(s) is less than ϵ, which means u will be infected with probability less than ϵ.

For (2), suppose u is connected to v by a directed edge gadget $\langle u, v\rangle$ and u is already infected. Then all the h^L inputs of the inner probability filter gadget will be infected with probability a_1 independently, and v will be infected with probability in $(p_2 - \varepsilon_2, p_2]$ if δ is set small enough such that a_1 passes the threshold p_1. In particular, $b > 0$.

Lastly, we prove Lemma 6.10 and Lemma 6.11.

Proof of Lemma 6.10. The possibility of the construction is already made explicit.

Let λ be the upper bound of the number of vertices and edges in a probability separation block in the probability filter gadget (which is a constant), the total number of vertices in a directed edge gadget is

$$\sum_{i=0}^{L-1} \lambda h^i = \lambda \frac{h^L - 1}{h-1} = \Theta\left(h^L\right) = \Theta\left(h^{\frac{\log \Upsilon}{\log \frac{1}{1-\delta}} + \frac{\log(\frac{1}{\epsilon})}{\log \frac{1}{1-\delta}} + 1}\right) = \Theta\left(\Upsilon^d (1/\epsilon)^d\right),$$

and the total number of edges is

$$\underbrace{h^L}_{\text{number of edges from } u \text{ to the probability scaling down gadget}} + \sum_{i=0}^{L-1} \lambda h^i = \Theta\left(h^L\right) = \Theta\left(\Upsilon^d (1/\epsilon)^d\right).$$

where $d = \frac{\log h}{\log \frac{1}{1-\delta}}$.

To show that d depends only on f, it is enough to notice that we only need to set up the values of h and δ such that $p_1 < a_1$ as mentioned. □

Proof of Lemma 6.11. Given an $(I, \Lambda, p_0, p_2, \varepsilon_1, \varepsilon_2, f)$-AND gadget which consists of many 2-set-input AND gadgets (see Figure 6.13), we can obtain a $(\Lambda, p_1, p_2, \varepsilon_1, \varepsilon_2, f)$-probability filter gadget which is the core of an arbitrary 2-set-input AND gadget. We construct the $(\Upsilon, \epsilon, b, f)$-directed edge gadget by increasing the number of layers in this probability filter gadget, just as what we did earlier. By our analysis above, we already have $b \in (p_2 - \varepsilon_2, p_2]$. Moreover, by Figure 6.11, increasing the number of layers makes b closer to p_2. Therefore, we can have $b \in \left(p_2 - \frac{1}{2}\varepsilon_2, p_2\right]$ by just increasing the number of layers, which proves the possibility of the construction.

By our discussion in Section 6.4.4.2, we only need a constant number of layers to have $b \in \left(p_2 - \frac{1}{2}\varepsilon_2, p_2\right]$, as $\frac{1}{2}\varepsilon_2$ is a constant. Thus, requiring $b \in \left(p_2 - \frac{1}{2}\varepsilon_2, p_2\right]$ does not change the number of layers asymptotically. Following the proof of Lemma 6.10, we conclude the second half of the lemma. □

6.4.8 Proof of Theorem 6.2 for $a_1 = 0$

In the case $a_1 = 0$, the constructions of both the AND gadget and the directed edge gadget fail. Modifications of the structure in Figure 6.7 as well as the structure of the AND gadget are required. We will discuss these modifications in this section, and the remaining details are left to the readers.

Modification to the AND gadget The AND gadget for the case $a_1 = 0$ is much simpler. The input $\varepsilon_1, \varepsilon_2$ is no longer needed, and both p_0, p_2 in the original AND gadget are set to $\frac{1}{2}a_2$. The definition of the modified AND gadget is shown below.

Definition 6.24. A (I, Λ, f)-*AND gadget* takes I sets of Λ vertices each as inputs, and output a vertex such that

1. if the vertices in at least one input set are infected with probability 0, then the output vertex will be infected with probability 0;
2. if the vertices in all input sets are infected with independent probability at least $\frac{1}{2}a_2$, then the output vertex will be infected with probability at least $\frac{1}{2}a_2$,

The construction of a $(2, \Lambda, f)$-AND gadget is shown in Figure 6.15. It is easy to see that the infection of the output vertex will not affect any other vertices in this circuit due to $a_1 = 0$. Due to the same reason, property (1) above is trivial for the case $I = 2$ here. Let x be the probability that each vertex in the two input sets is infected, and let y be the probability the output is infected. Then,

$$y = \sum_{i=2}^{\Lambda} \binom{\Lambda}{i} a_i (a_2 x)^i (1 - a_2 x)^{\Lambda - i}.$$

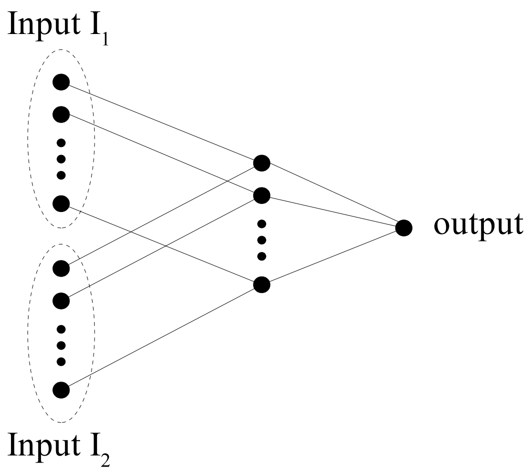


Figure 6.15: The modified AND gadget with parameter $(2, \Lambda, f)$

To satisfy (2), we only need to choose Λ large enough such that $y(\frac{1}{2}a_2) \geq \frac{1}{2}a_2$. This is always possible, as we have $y(\frac{1}{2}a_2) \to p^* > \frac{1}{2}a_2$ as $\Lambda \to \infty$ (the expected number of infected neighbors of the output vertex is $\frac{1}{2}a_2\Lambda$ which goes to infinity).

Lemma 6.25. *For any f with $a_2 > a_1 = 0$, we can construct a $(2, \Lambda_0, f)$-AND gadget with constant size, and Λ_0 is a constant depending on f.*

Proof. The construction above shows the existence of the gadget, and Λ_0 is a constant that is large enough to make $y(\frac{1}{2}a_2) \geq \frac{1}{2}a_2$ true, which depends only on f.

From Figure 6.15, it is clear that the gadget has $3\Lambda_0 + 1$ vertices and $3\Lambda_0$ edges, which are both constants. □

To construct a (I, Λ, f)-AND gadget with I being an integer power of 2, we use the same "tower structure" in Figure 6.13. Specifically, all the AND gadgets in all $\log_2 I$ levels are identically the $(2, \Lambda_0, f)$-AND gadget in Figure 6.15, and the output vertices of $2^{\log_2 I - i}$ groups of $\Lambda_0^{\log_2 I - i}$ $(2, \Lambda_0, f)$-AND gadgets in Level i are connected to the input ends of $2^{\log_2 I - i - 1}$ groups of $\Lambda_0^{\log_2 I - i - 1}$ $(2, \Lambda_0, f)$-AND gadgets in Level $(i + 1)$. It is straightforward to check that (1) and (2) in Definition 6.24 hold for this construction.

Lemma 6.26. *For any f with $a_2 > a_1 = 0$ and any I that is an integer power of 2, we can construct a (I, Λ, f)-AND gadget with $O\left(I^{c+1}\right)$ vertices and $O\left(I^{c+1}\right)$ edges, and $\Lambda = I^c$, where c is a constant depending on f.*

Proof. The existence of this AND gadget is shown by the explicit construction.

The numbers of vertices and edges are both

$$\sum_{i=1}^{\log_2 I} 3\Lambda_0 \cdot 2^{\log_2 I - i} \Lambda_0^{\log_2 I - i} < 3\Lambda_0 \cdot (2\Lambda_0)^{\log_2 I} = O\left(I^{c+1}\right),$$

where $c = \log_2 \Lambda_0$ is a constant, and it depends only on f as Λ_0 depends only on f according to Lemma 6.25. Notice that the number of vertices in a $(2, \Lambda_0, f)$-AND gadget is counted as $3\Lambda_0$ other than $3\Lambda_0 + 1$ in Lemma 6.25, because the output vertex of each $(2, \Lambda_0, f)$-AND gadget is counted as one of the input vertices in one of the $(2, \Lambda_0, f)$-AND gadgets in the next level.

Finally, since there are $\Lambda_0^{\log_2 I - 1}$ $(2, \Lambda_0, f)$-AND gadgets in each group in Level 1, we have

$$\Lambda = \Lambda_0 \cdot \Lambda_0^{\log_2 I - 1} = I^c,$$

which concludes the lemma. □

Modification to the set cover part We will use a pair of vertices to represent a subset in the SetCover problem and use a pair of cliques to represent an element in U. The pair of vertices are connected to each vertex of the two cliques by a specially designed gadget shown in the bottom of Figure 6.16.

If the two vertices representing a subset are both infected, it is straightforward to check that each vertex at the output end of the gadget at the bottom of Figure 6.16 will be infected with probability a_2^7. Given there are m vertices in a clique, the expected number of infected vertices in a clique is $a_2^7 m$. By choosing m large enough (but still a constant) such that $a_{\lfloor a_2^7 m \rfloor} > p^* - \varepsilon$, each vertex in the clique will be infected with probability at least $p^* - \varepsilon$. Therefore, if a subset is picked such that the two vertices representing it are chosen as seeds, all pairs of cliques representing its elements will be activated. Naturally, given the SetCover instance in which we are choosing k subsets, we are asked to choose $2k$ seeds in the InfluenceMaximization instance.

On the other hand, since $a_1 = 0$, an activated clique will not be able to infect the pair of vertices representing a subset, so the connection between the pair of vertices to each vertex in the clique is like a directed edge. Moreover, it is easy to see that we still need two seeds to pick a subset even if some cliques representing elements in this subset are activated. Although we have the option to choose the two seeds "on the gadget", we still need to pick at least two seeds to "choose a subset". Thus, it does not matter if any of these seeds is not exactly in the pair of vertices representing the

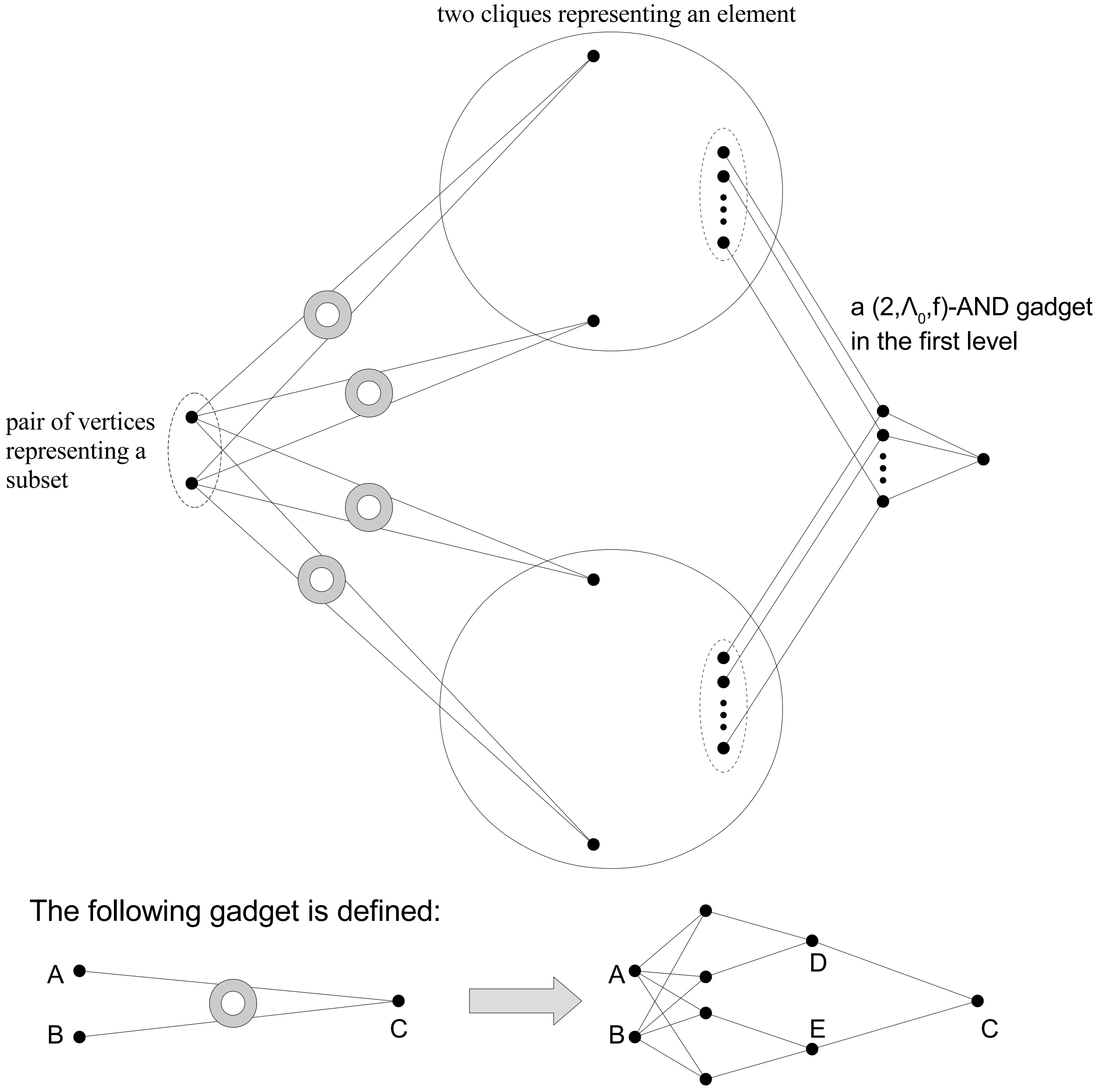


Figure 6.16: Connection between a pair of vertices representing a subset and vertices in the two cliques representing an element, and a $(2, \Lambda_0, f)$-AND gadget in the first level of the $(2n, (2n)^c, f)$-AND gadget.

subset.

The M_2 $(n, \Lambda, p^* - \varepsilon, p_2, 1/n, \varepsilon_2, f)$-AND gadgets in Figure 6.7 is changed to M_2 $(2n, (2n)^c, f)$-AND gadgets here. Moreover, each of the n groups of the $(2, \Lambda_0, f)$-AND gadgets in Level 1 of the $(2n, (2n)^c, f)$-AND gadget corresponds to the vertices in *the two cliques representing the same element* in U. A single $(2, \Lambda_0, f)$-AND gadget is illustrated on the right-hand side of Figure 6.16.

Modification to the connection to the M_1 vertices In Figure 6.7, the output vertex v is connected to the M_1 vertices by M_1 edges. Since $a_1 = 0$, such construction will fail to satisfy our purpose here. To fix this, we can use $2M_2$ $(2n, (2n)^c, f)$-AND gadgets such that the outputs of every two AND gadgets are connected to each of the M_1 vertices.[6]

In addition, we also update the value of M_1 to $M_1 = n^{c+10}$.

Modification to the clique size m Since there are $(2n)^c$ vertices in each of the n inputs for each of the $2M_2$ $(2n, (2n)^c, f)$-AND gadgets, to furnish enough inputs, we update the clique size to $m = 2M_2 \cdot (2n)^c = 2^{1+c}n^{2+c} = O\left(n^{c+2}\right)$.

Modification to Lemma 6.8 To conclude this section, we have the following lemma corresponding to Lemma 6.8 in Section 6.4.2.

Lemma 6.27. *If the* SetCover *instance is a* YES *instance, by choosing $2k$ seeds appropriately, we can infect at least $\frac{1}{4}a_2^3 \cdot n^{c+12}$ vertices in expectation in the graph G we have constructed; if it is a* NO *instance, we can infect at most $O\left(kn^{c+10}\right)$ vertices in expectation for any choice of $2k$ seeds.*

Proof. If the SetCover instance is a YES instance, we choose the $2k$ seeds representing the k subsets, and all the $2n$ cliques will be activated such that each vertex in all these clique will be infected with probability $p^* - \varepsilon$. Since $p^* \geq a_2$, we have $p^* - \varepsilon > \frac{1}{2}a_2$ as ε is sufficiently small due to large size of m. All the $2M_2$ $(2n, (2n)^c, f)$-AND gadgets fall into case (2), so that the output vertices are infected with probabilities at least $\frac{1}{2}a_2$. The M_1 vertices in each of the $2M_2$ copies of the verification part are connected to two vertices with infection probabilities at least $\frac{1}{2}a_2$, so the total expected number

[6] Another way to fix this is to reduce the number of levels by 1 in the $(2n, (2n)^c, f)$-AND gadget, such that we have two output vertices of the AND gadget instead of only one output in Definition 6.24.

of infected vertices in G is at least

$$M_2 \times \underbrace{\left(\frac{1}{2}a_2\right)^2}_{\text{probability both output vertices are infected}} \times \underbrace{(a_2 M_1)}_{\text{expected num of infections in } M_1 \text{ vertices}} = \frac{1}{4}a_2^3 \cdot n^{c+12}.$$

If the SETCOVER instance is a NO instance, consider any choice of $2k$ seeds with k_1 seeds in the vertices representing subsets, k_2 seeds in the connection gadgets between vertices representing subsets and vertices in the cliques, k_3 seeds in the $2n$ cliques, k_4 seeds in those $(2, \Lambda_0, f)$-AND gadgets at Level 1 of the $(2n, (2n)^c, f)$-AND gadgets, and $k_5 = 2k - k_1 - k_2 - k_3 - k_4$ seeds in the remaining part of the verification parts (the $(2, \Lambda_0, f)$-AND gadgets at the remaining levels and the M_1 vertices connecting to the $(2n, (2n)^c, f)$-AND gadgets). Again, we first prove that at least one clique will not be activated such that all its vertices are infected with probability 0.

First of all, those k_5 seeds cannot have effect in activating cliques. This is because their influence cannot pass through the $(2, \Lambda_0, f)$-AND gadgets in the first level, as the infection of the output vertex in each $(2, \Lambda_0, f)$-AND gadget cannot further infect the input vertices due to $a_1 = 0$.

Secondly, for those k_1 and k_2 seeds, they are at the vertex-pairs representing the subsets and the gadgets connected to those pairs respectively. We call the vertices in those gadgets connecting to a pair *the vertices around the pair*. It is easy to see that we need to choose at least 2 seeds in or around a pair to pick a subset. To see this, even if vertex C in the gadget (at the bottom of Figure 6.16) is already infected (which is possible as C belongs to a clique which may have been activated already) such that D and E already have one infected neighbor, we still cannot make both A and B infected by picking only 1 seed in or around the pair (A, B). Thus, we assume without loss of generality that all k_2 seeds are on the pairs representing the subsets, as we need at least 2 seeds in or around a pair (A, B) in which case we can assume the seeds are just at A and B.

For those k_3 seeds on the cliques and k_4 seeds on the $(2, \Lambda_0, f)$-AND gadgets in the first level, since each AND gadget in the first level takes two sets of vertices from two cliques *representing the same element in* U, we need at least 3 seeds to activate two cliques representing the same element in U: one in the middle of the AND gadget, and one in each of the two cliques (such that the two vertices connecting to the seed in the middle of the AND gadget have two infected neighbors, and stand a chance to activate the two cliques). In contrast, we only need 2 seeds to activate these two cliques, by choosing the pair of vertices representing the subset covering the element that these two cliques represent. Therefore, we can assume that those k_3 and k_4 seeds are also on those pairs representing subsets.

Since $k_1+k_2+k_3+k_4 \leq 2k$ and the SetCover instance is a NO instance, by the fact that we need 2 seeds to pick a subset, we conclude that at least one clique will not be activated, and the vertices in this clique are infected with probability 0.

By the effect of the $(2n,(2n)^c,f)$-AND gadget, except for those (at most) k_4+k_5 AND gadgets containing seeds, the output vertices of the remaining $2M_2-k_4-k_5$ AND gadgets will be infected with probability 0, which have no effect on those M_1 vertices. Therefore, even if all the vertices in the set cover part, the k_4+k_5 copies of the verification parts, and the $2M_2$ $(2n,(2n)^c,f)$-AND gadgets are infected, the total number of infected vertices cannot exceed

$$\underbrace{2K+6K(2n)m+2nm}_{\text{size of the set cover part}}+2M_2\underbrace{(2n)^{c+1}}_{\text{size of an AND gadget}}+(k_4+k_5)\underbrace{\left((2n)^{c+1}+M_1\right)}_{\text{size of a verification part}}$$
$$=O\left(kn^{c+10}\right),$$

which concludes the lemma. □

Noticing that the total number of vertices in G is

$$N=2K+6K(2n)m+2nm+M_2\left((2n)^{c+1}+M_1\right)=\Theta\left(n^{c+12}\right),$$

and $kn^{c+10}=O(n^{c+11})$. we conclude Theorem 6.2 in the case $a_1=0$ by setting $\tau=\frac{1}{c+12}$.

6.5 A Variant of Theorem 6.2

Li et al. [53] considered a model where there is only a sublinear fraction of vertices admitting nonsubmodular local influence functions that are almost submodular. They showed that, even though this appears to make the diffusion model globally closer to submodularity, InfMax is still NP-hard to approximate to within N^τ for certain constant τ. In this section, we adapt Theorem 6.2 to a variant that is of a similar style of this.

Theorem 6.28. *Consider the* InfMax *problem with general threshold model $I_{G,F}$. For any fixed 2-quasi-submodular f, any fixed submodular function $g:\mathbb{Z}_{\geq 0}\mapsto[0,1]$ with $g(1)>g(0)=0$, and any $\gamma\in(0,1)$, there exists a constant τ depending on*

f and γ such that, even if $f_v \in F$ is symmetric with either $f_v = f$ or $f_v = g$, and $|\{v \in V : f_v = f\}| \leq N^\gamma$, it is NP-hard to distinguish between the following two cases:

- YES*: there exists a seed set S with $|S| = k$ such that $\sigma(S) = \Theta(N)$;*
- NO*: for any seed set S with $|S| = k$, we have $\sigma(S) = O(N^{1-\tau})$.*

Proof. We again discuss two different cases: $a_1 = f(1) > 0$ and $a_1 = f(1) = 0$.

For the first case, the reduction in Sect. 6.4.3 can be modified to prove this theorem (if we only need to prove this theorem for directed graphs, the much simpler reduction in Sect. 6.4.2 can be used), with the following modifications.

- Except for those M_1 vertices on the right-hand side of Fig. 6.7 in each of the M_2 copies of the verification part, the remaining vertices are equipped with f. Those M_1 vertices in each of the M_2 copies are equipped with g.
- Change $M_1 = n^{(30c_1+30c_2+70)d}$ (as it is in Sect. 6.4.3) to $M_1 = n^{\frac{1}{\gamma}(30c_1+30c_2+70)d}$, where n is the number of elements in the SetCover instance and c_1, c_2, d are the constants in Lemma 6.6 and Lemma 6.10.

Recall from Sect. 6.4.3 that the set cover part has $O(n^{(3c_1+3c_2+7)d+c_1+c_2+4})$ vertices and the AND gadget has $O(n^{c_1+c_2+1})$ vertices, the total number of vertices in G is

$$N = O\left(n^{(3c_1+3c_2+7)d+c_1+c_2+4}\right) + M_2\left(O\left(n^{c_1+c_2+1}\right) + M_1\right) = \Theta\left(n^{\frac{1}{\gamma}(30c_1+30c_2+70)d+2}\right),$$

which is of polynomial size. Moreover, the total number of vertices that are equipped with f is

$$O\left(n^{(3c_1+3c_2+7)d+c_1+c_2+4}\right) + M_2 O\left(n^{c_1+c_2+1}\right) = o\left(n^{(30c_1+30c_2+70)d}\right) \ll N^\gamma.$$

The remaining part of the proof is almost identical to the proof of Lemma 6.12. The only difference is that, if the output of the AND gadget, the vertex v in Fig. 6.7, is infected, then each of those M_1 vertices is now infected with probability $g(1)$, instead of $a_1 = f(1)$ before. With this change, when the SetCover instance is a YES instance, the total number of infected vertices (for properly choosing seeds corresponding to the subsets) become

$$p_{\text{activated}} g(1)(p_2 - \varepsilon_2) M_1 M_2 = \Theta\left(n^{\frac{1}{\gamma}(30c_1+30c_2+70)d+2}\right) = \Theta(N),$$

where $p_{\text{activated}}$ is the same as it is in the proof of Lemma 6.12, p_2, ε_2 are the parameters for the AND gadget which are the same as defined in Sect. 6.4.3. When the

SetCover instance is a NO instance, following the same analysis, the upper bound for the total number of infected vertices can also be computed by Equation (6.1), with M_1 replaced by the modified value $n^{\frac{1}{\gamma}(30c_1+30c_2+70)d}$ here. In particular, the first three terms in (6.1) are dominated, the fourth and the fifth terms are both at most $O(n^{\frac{1}{\gamma}(30c_1+30c_2+70)d+1})$. Therefore, we conclude the theorem for the case $a_1 > 0$ by setting $\tau = \frac{1}{\frac{1}{\gamma}(30c_1+30c_2+70)d+2}$.

For the second case $a_1 = f(1) = 0$, the reduction is almost the same as it is in Sect. 6.4.8, except for the following changes.

- Those M_1 vertices in each of the M_2 copies are equipped with g, while the remaining vertices are equipped with f.
- Change the value of M_1 from n^{c+10} (as it is in Sect. 6.4.8) to $n^{\frac{1}{\gamma}(c+10)}$.

Following the same analysis in Sect. 6.4.8, we can see that the graph has $N = n^{\frac{1}{\gamma}(c+10)+2}$ vertices, and there are only $O(n^{c+4}) \ll N^\gamma$ vertices that have f as their local influence functions. Corresponding to Lemma 6.27, we can show that the expected number of infections is at least $\frac{1}{4}a_2^2 g(2) n^{\frac{1}{\gamma}(c+10)+2} = \Theta(N)$ when appropriately choosing $2k$ seeds for a YES instance, and the expected number of infections can be at most $O(kn^{\frac{1}{\gamma}(c+10)})$ for a NO instance. By noticing $kn^{\frac{1}{\gamma}(c+10)} = O(n^{\frac{1}{\gamma}(c+10)+1})$ and taking $\tau = \frac{1}{\frac{1}{\gamma}(c+10)+2}$, we conclude the theorem for the case $a_1 = 0$. □

We remark that Theorem 6.28 can be viewed as a generalization of the inapproximability result in [53] in the following two directions.

1. Our result holds for any f that is fixed in advance, while f is set to $f(1) = \frac{1-\varepsilon}{2}$ and $f(2) = 1$ in [53] (where ε is an arbitrary constant fixed in advance).
2. Our result holds for undirected graphs, while it is unknown if the proof in [53] can be adapted to show the same inapproximability result for undirected graphs (notice that an undirected graph can be viewed as a special case of a directed graph with anti-parallel edges, so an inapproximability result for a more special case is stronger).

CHAPTER 7

Bootstrap Percolation on Graphs with Hierarchical Communities

We have seen in the previous chapter that even strong assumptions on diffusion models fail to make nonsubmodular InfMax approximable. Can we circumvent the inapproximability result by making natural assumptions on the network topology instead?

In this chapter, we present strong inapproximability results for a very restricted class of networks called the *(stochastic) hierarchical blockmodel*, a special case of the well-studied *(stochastic) blockmodel* in which relationships between blocks admit a tree structure. We also provide a dynamic-programming-based polynomial time algorithm which optimally computes a directed variant of the influence maximization problem on hierarchical blockmodel networks. Our algorithm indicates that the inapproximability result is due to the bidirectionality of influence between agent-blocks.

7.1 Introduction

We know a lot about what social networks look like, and previous hardness reductions make no attempt to capture realistic features of networks. It is very plausible that by restricting the space of networks we might regain tractability.

In this chapter, we consider two natural network topologies: the *hierarchical blockmodel* and the *stochastic hierarchical blockmodel*. Each is a natural restriction on the classic *(stochastic) blockmodel* [24, 41, 74] network structure. In (stochastic) blockmodels, agents are partitioned into ℓ blocks. The weight (or likelihood in the stochastic setting) of an edge between two vertices is based solely on blocks to which the vertices belong. The weights (or probabilities) of edges between two blocks can be represented by an $\ell \times \ell$ matrix. In the (stochastic) hierarchical blockmodel, the structure of the $\ell \times \ell$ matrix is severely restricted to be "tree-like".[1]

Our (stochastic) hierarchical blockmodel describes the hierarchical structure of the communities, in which a community is divided into many sub-communities, and each sub-community is further divided, etc. Typical examples include the structure of a country, which is divided into many provinces, and each province can be divided into cities. Our model captures the natural observation that people in the same sub-community in the lower hierarchy tend to have tighter (or more numerous) bonds among each other [23]. Such a highly abstracted model necessarily fails to capture all features of social networks. However, when we use this model as a lower bound, that is a strength as it shows that the problem is hard even in the case that communities structure can be represented by a tree. Additionally, this is a very natural model which captures salient features of real-world networks, so our upper bounds in this model are still interesting.

Our results We present inapproximability results for InfMax with both the hierarchical blockmodel and the stochastic hierarchical blockmodel. We show that InfMax is NP-hard to approximate to within a factor of $N^{1-\varepsilon}$ for an arbitrary $\varepsilon > 0$. Moreover, this result holds in the hierarchical blockmodel even if we assume all agents have unit threshold $r_v = 1$. We also extend this hardness result to the stochastic hierarchical blockmodel.

Moreover, for the hierarchical blockmodel, we present a dynamic-programming-based polynomial time algorithm for InfMax when we additionally assume the influence from one block to another is "one-way". This provides insights to the above intractability result: the difficulty comes from the bidirectionality of influence between agent-blocks.

7.2 Preliminaries

Same as in the previous chapter, we use N instead of n to denote the total number of vertices.

We consider bootstrap percolation $I_{G,R}$ (Definition 2.15) for this chapter.

We consider two graph models—the *hierarchical blockmodel* and the *stochastic hierarchical blockmodel*, which are the special case of the well studied *blockmodel* [74] and *stochastic blockmodel* [41] respectively.

[1] Previous work on community detection in networks [57] defines a different, but related stochastic hierarchical blockmodel, where the hierarchy is restricted to two levels.

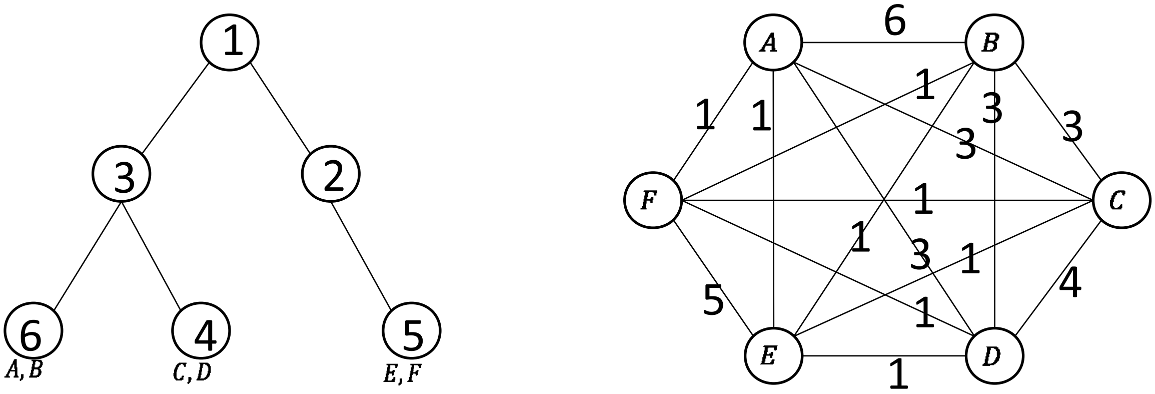


Figure 7.1: An example of a hierarchy tree with its corresponding graph. The number on each node of the hierarchy tree on the left-hand side indicates the weight of the node, which reflects the weight of the corresponding edges on the hierarchical block graph on the right-hand side in the above-mentioned way.

7.2.1 Hierarchical Blockmodel

Definition 7.1. A *hierarchical blockmodel* is an undirected *edge-weighted* graph $G = (V, T)$, where V is the set of all vertices of the graph G, and $T = (V_T, E_T, w_T)$ is a node-weighted binary tree T called a *hierarchy tree.* In addition, w_T satisfies $w_T(t_1) \leq w_T(t_2)$ for any $t_1, t_2 \in V_T$ such that t_1 is an ancestor of t_2.[2] Each leaf node $t \in V_T$ corresponds to a subset of vertices $V(t) \subseteq V$, and the $V(t)$ sets partition the vertices of V. In general, if t is not a leaf, we denote $V(t) = \cup_{t'\text{: a leaf, and an offspring of } t} V(t')$.

For $u, v \in V$, the weight of the edge (u, v) in G is just the weight of the least common ancestor of u and v in T. That is $w(u, v) = \max_{t: u,v \in V(t)} w(t)$. If this weight is 0, then we say that the edge does not exist.

To avoid possible confusion, in this chapter, we use the words *node* and *vertex* to refer to the vertices in T and G respectively.

Figure 7.1 provides an example of how a hierarchy tree defines the weights of edges in the corresponding graph.

Additionally, we can assume without loss of generality that the hierarchy tree is a *full* binary tree, as a node in T having only one child plays no role at deciding the weights of edges in G. For example, in Figure 7.1, the node having weight 2 does not affect the weight configuration on the right-hand side. We can delete this node and promote the node with weight 5 to be a child of the root node. We will keep the full binary tree assumption from now on.

[2] Since, as it will be seen later, each node in the hierarchy tree represents a community and its children represent its sub-communities, naturally, the relation between two persons is stronger if they are in a same sub-community in a lower level.

Since hierarchical blockmodel is essentially an edge-weighted graph, we should consider the weighted version of the bootstrap percolation mentioned in the paragraph immediately following Definition 2.16.

7.2.2 Stochastic Hierarchical Blockmodel

The *stochastic hierarchical blockmodel* is similar to the hierarchical blockmodel defined in the last section, in the sense that the structure of the graph is determined by a hierarchy tree. Instead of assigning weights to different edges measuring the strength of relationships, here we assign a probability with which the edge between each pair of vertices appears. Technically speaking, a stochastic hierarchical blockmodel is a distribution of unweighted undirected graphs, where each edge is sampled with a certain probability.

Definition 7.2. A *stochastic hierarchical blockmodel* is a distribution $\mathcal{G} = (V, T)$ of unweighted undirected graphs where V, T are the same as they are in Definition 7.1 with the additional restriction that the node weights in T belong to the interval $[0, 1]$. Let H be the weighted graph defined by the hierarchical blockmodel $H = (V, T)$, and let $w(e)$ denote the weight of edge e in H. Then $G = (V, E)$ is sampled by independently including each edge e with probability $w(e)$.

When it comes to the choices of S, the InfMax problem can be defined in two different ways, regarding whether we allow the seed-picker to see the sampling $G \sim \mathcal{G}$ *before* choosing the seed set S.

Definition 7.3. *Pre-sampling stochastic hierarchical blockmodel* InfMax is an optimization problem which takes as inputs $\mathcal{G}$, R, and an integer k and outputs

$$\operatorname*{argmax}_{S \subseteq V : |S| = k} \mathop{\mathbb{E}}_{G \sim \mathcal{G}} \left[\sigma_{G,R}(S) \right],$$

a seed set of size k that maximizes the expected global influence.

Definition 7.4. *Post-sampling stochastic hierarchical blockmodel* InfMax is an average case version of InfMax which takes as input $\mathcal{G}$, R, and an integer k, and outputs the solution of the InfMax instance (G, R, k) after sampling G from $\mathcal{G}$.

7.3 Hardness of Approximation for Hierarchical Blockmodel

In this section, we provide a strong inapproximability result for InfMax problem for the hierarchical blockmodel with bootstrap percolation even when all vertices have the same threshold 1. Specifically, we will show that it is NP-hard to approximate optimal $\sigma(S)$ within a factor of $N^{1-\varepsilon}$ for any $\varepsilon > 0$ (recall that $N = |V|$ is the total number of vertices in the graph).

Theorem 7.5. *Consider the* InfMax *problem with bootstrap percolation $I_{G,R}$. For any constant $\varepsilon > 0$, even if G is a hierarchical blockmodel and $r_v = 1$ for all $v \in V$, it is NP-hard to distinguish between the following two cases:*

- YES*: there exists a seed set S with $|S| = k$ such that $\sigma(S) = \Theta(N)$;*
- NO*: for any seed set S with $|S| = k$, we have $\sigma(S) = O(N^{\varepsilon})$.*

We will prove this by a reduction from the VertexCover problem, a well-known NP-complete problem.

Definition 7.6. Given an undirected graph $\overline{G} = (\overline{V}, \overline{E})$ and a positive integer $\overline{k}$, the VertexCover problem $(\overline{G}, \overline{k})$ asks if we can choose a subset of vertices $\overline{S} \subseteq \overline{V}$ such that $|\overline{S}| = \overline{k}$ and such that each edge is incident to at least one vertex in $\overline{S}$.

The reduction Given a VertexCover instance $(\overline{G}, \overline{k})$, let $n = |\overline{V}|$ and $m = |\overline{E}|$. We use $A_1, \ldots, A_n$ to denote the n vertices and $e_1, \ldots, e_m$ to denote the m edges.[3] We make the assumptions $n > \overline{k}$ is an integer power of 2 and $m > n + \overline{k}$.[4] Let $W = nm$, $M = (n(2W + m) - 1)^{\frac{1}{\varepsilon}}$, and $\delta > 0$ be a sufficiently small real number.

We will construct the graph $G = (V, E, w)$ by constructing a hierarchy tree T which uniquely determines G (see Definition 7.1 in Section 7.2.1). The construction of T is shown in Figure 7.2. The first $\log_2 n$ levels of T is a full balanced binary subtree with n leaves, and the weight of the nodes in all these levels is δ. Each of those n leaves is the root of a subtree corresponding to each vertex A_i in the VertexCover instance.

[3] We use the letter A to denote the vertices in a VertexCover instance instead of commonly used v, while v is used for the vertices in an InfMax instance. Since VertexCover can be viewed as a special case of SetCover with vertices corresponding to subsets and edges corresponding to elements, the letter A, commonly used for subsets, is used here.

[4] For the assumption that n is an integer power of 2, we can just add isolated vertices to $\overline{G}$. For the assumption $m > n + \overline{k}$, notice that allowing the graph $\overline{G}$ to be a multi-graph does not change the nature of VertexCover, we can ensure m to be sufficiently large by just duplicating edges.

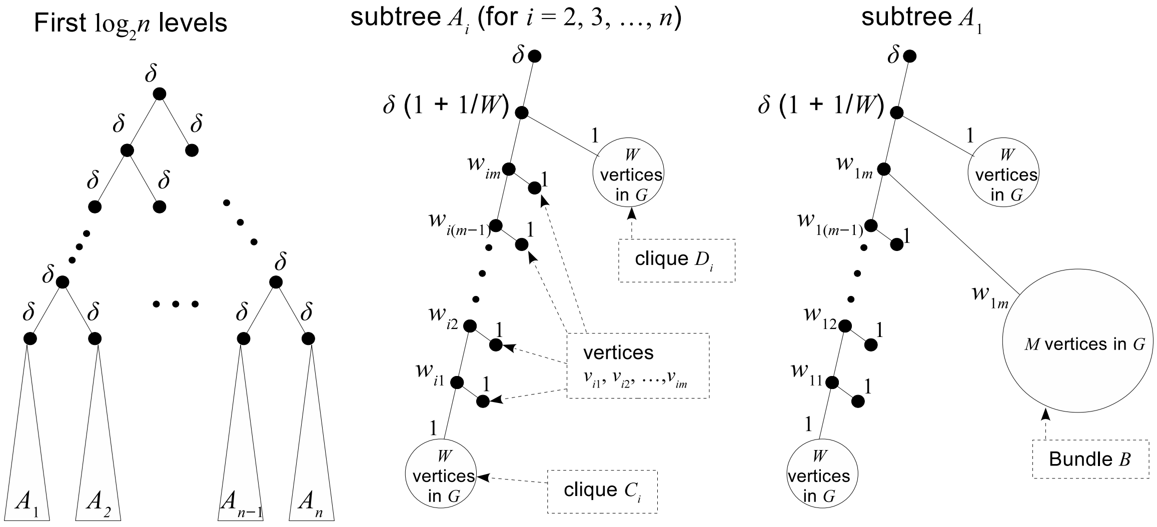


Figure 7.2: The construction of the hierarchy tree T for proving Theorem 7.5.

The structure of the subtrees corresponding to $A_2, \ldots, A_n$ and A_1 are shown on the right-hand side of Figure 7.2. The numbers on the tree nodes indicate the weights, and in particular

$$w_{ij} = \begin{cases} \frac{[1-(n+\bar{k}-1)W\delta-(n-1)(j-1)\delta-2\delta]+\delta}{W-1+j} & \text{if edge } e_j \text{ is incident to } A_i \\ \frac{1-(n+\bar{k}-1)W\delta-(n-1)(j-1)\delta-2\delta}{W-1+j} & \text{otherwise} \end{cases}, \tag{7.1}$$

for each $i = 1, \ldots, n$ and $j = 1, \ldots, m$.

The leaves of each subtree A_i are the leaves of T, which, as we recall from Definition 7.1 correspond to subsets of vertices in $G = (V, E, w)$. Among all the leaves shown on the right-hand side of Figure 7.2, each solid dot corresponds to a subset of V containing only one vertex, and each hollow circle corresponds to a subset of V containing many vertices with the corresponding number of vertices shown.

For each subtree A_i with $i = 2, \ldots, n$, we have constructed $m + 2$ leaves corresponding to $2W + m$ vertices in G. They are, in up-to-down order, a clique D_i of W vertices, vertices $v_{im}, v_{i(m-1)}, \ldots, v_{i1}$, and a clique C_i of W vertices. As each vertex has threshold 1 and the leaf nodes corresponding to C_i, D_i both have weight 1, infecting any vertex in C_i or D_i will cause the infection of all W vertices (which justifies the name "clique").

The construction of A_1 is similar. The only difference is that, instead of connecting to a node corresponding to the vertex v_{1m}, the node with weight w_{1m} is now connected to another node with the same weight and corresponding to a bundle B in G with M vertices. We shall not call this large bundle B a "clique", as the weight of the edge between each pair of these M vertices is $w_{1m} \ll 1$, which is much weaker.

It is easy to calculate the total number of vertices in the construction: $N = M + M^{\varepsilon}$.

We present a toy example illustrating the construction of T in Fig. 7.3, where the explicit construction of T corresponding to a small graph with 4 vertices and 4 edges is given. In the toy example, $n = m = 4$, $W = mn = 16$, $\bar{k} = 2$, $M = (n(2W+m)-1)^{1/\varepsilon} = 143^{1/\varepsilon}$, and the values of ε and δ are set sufficiently small and unassigned for clarity. The w_{ij}s, defined according to (7.1), are as follows: corresponding to the edge $e_1 = (A_1, A_2)$, we have $w_{11} = w_{21} = \frac{1-81\delta}{16}$ (shown by larger dots) and $w_{31} = w_{41} = \frac{1-82\delta}{16}$ (shown by smaller dots); corresponding to the edge $e_2 = (A_1, A_3)$, we have $w_{12} = w_{32} = \frac{1-84\delta}{17}$ (shown by larger dots) and $w_{22} = w_{42} = \frac{1-85\delta}{17}$ (shown by smaller dots); corresponding to the edge $e_3 = (A_1, A_4)$, we have $w_{13} = w_{43} = \frac{1-87\delta}{18}$ (shown by larger dots) and $w_{23} = w_{33} = \frac{1-88\delta}{18}$ (shown by smaller dots); corresponding to the edge $e_4 = (A_3, A_4)$, we have $w_{34} = w_{44} = \frac{1-90\delta}{18}$ (shown by larger dots) and $w_{14} = w_{24} = \frac{1-91\delta}{18}$ (shown by smaller dots). In this example, $\overline{G}$ has a 2-vertex cover $\{A_1, A_3\}$. Corresponding to this, the $k = m + \overline{k} = 6$ seeds should be put at $C_1, C_2, C_3, C_4, D_1, D_3$ respectively, so that the vertices in the large bundle B (the one containing $143^{1/\varepsilon}$ vertices) will be eventually infected: firstly, all the vertices in $C_1, C_2, C_3, C_4, D_1, D_3$ will be infected; in the next step, the influence of these infected vertices is just enough to infect v_{11}, as $|C_1|w_{11} + |D_1| \cdot (1 + \frac{1}{16})\delta + (|C_2| + |C_3| + |C_4| + |D_3|)\delta = 1$; we can also check by calculation that the additional infection of v_{11} will further infect v_{21}, and the additional infection of v_{21} will further infect v_{31}, v_{41}, making all the four vertices corresponding to e_1 infected; finally, it is easy to check that the cascade will carry on level-by-level and eventually reach the bundle B. In general, each level i corresponding to the edge $e_i = (A_{j_1}, A_{j_2})$ contains n vertices $v_{i1}, \ldots, v_{in}$, and two of them, v_{ij_1}, v_{ij_2}, are connected to the tree by a weight heavier than that of the remaining $n-2$ vertices. If the vertices in at least one of D_{j_1}, D_{j_2} are infected (corresponding to the case the vertex A_{j_1} or A_{j_2} is included in the vertex cover), after the infection of the vertices in the first $i-1$ levels, the corresponding vertex in v_{ij_1}, v_{ij_2} will be infected, which will further infect all the remaining $n-1$ vertices in the i-th level. On the other hand, if none of the vertices in D_{j_1}, D_{j_2} is infected, even if the cascade reaches the $(i-1)$-th level, no vertex in the i-th level will be infected and the cascade will end here without reaching the bundle B which contains most vertices of G.

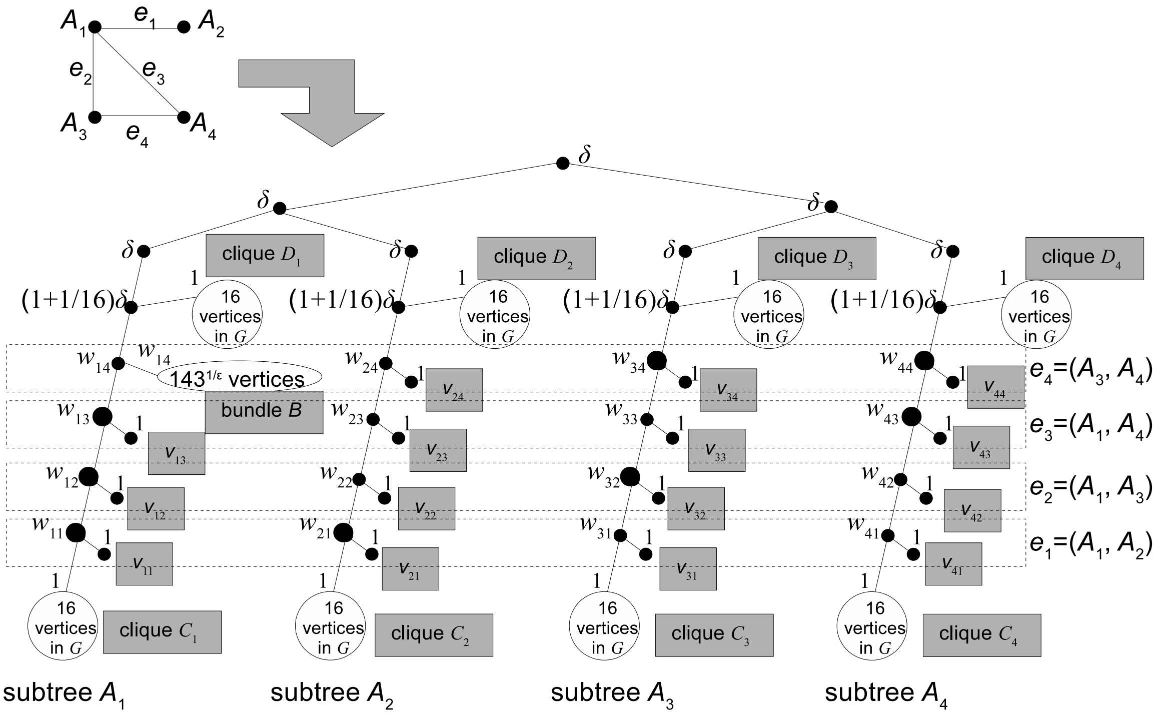

A_1
e_1
A_2
e_2
e_3
A_3
e_4
A_4
δ
clique D_1
clique D_2
clique D_3
clique D_4
$(1+1/16)\delta$
1
16 vertices in G
w_{14}
$143^{1/\varepsilon}$ vertices
bundle B
w_{13}, w_{12}, w_{11}
w_{24}, w_{23}, w_{22}, w_{21}
w_{34}, w_{33}, w_{32}, w_{31}
w_{44}, w_{43}, w_{42}, w_{41}
v_{14}, v_{13}, v_{12}, v_{11}
v_{24}, v_{23}, v_{22}, v_{21}
v_{34}, v_{33}, v_{32}, v_{31}
v_{44}, v_{43}, v_{42}, v_{41}
$e_4=(A_3, A_4)$
$e_3=(A_1, A_4)$
$e_2=(A_1, A_3)$
$e_1=(A_1, A_2)$
clique C_1
clique C_2
clique C_3
clique C_4
subtree A_1
subtree A_2
subtree A_3
subtree A_4

Figure 7.3: A toy example illustrating the construction of T for proving Theorem 7.5.

The reduction correctness For a VertexCover instance $(\overline{G}, \overline{k})$, consider the InfMax instance (G, R, k) with $k = n + \overline{k}$. We aim to show that,

1. If the VertexCover instance $(\overline{G}, \overline{k})$ is a YES instance, then there exists $S \subseteq V$ with $|S| = k$ such that $\sigma(S) \geq M$;

2. If the VertexCover instance $(\overline{G}, \overline{k})$ is a NO instance, then for any $S \subseteq V$ with $|S| = k$ we have $\sigma(S) \leq M^\varepsilon = n(2W + m) - 1$.

Proof of (1). Suppose we have a YES VertexCover instance $(\overline{G}, \overline{k})$ with $\overline{S} \subseteq \overline{V}$ covering all edges in $\overline{E}$. In the InfMax instance, we aim to show that at least M vertices will be infected if we choose those $k = n + \overline{k}$ seeds in the following way:

- choose an arbitrary seed in each of the cliques $C_1, \ldots, C_n$ (a total of n seeds are chosen);

- for each $A_i \in \overline{S}$, choose an arbitrary seed in the clique D_i (a total of $\overline{k}$ seeds are chosen).

By such a choice, in the first round of the cascade, all the W vertices in each of $C_1, \ldots, C_n$ and each of those $\overline{k}$ (D_i)'s are infected. We aim to show that all vertices in B will be infected after at most $3m$ cascade rounds. We call the set of n vertices

$\{v_{1j}, \ldots, v_{nj}\}$ *the j-th level*, and we will show that the cascade carries on level by level. In particular, we will first show that all vertices in the first level will be infected in at most 3 rounds. Next, given that all vertices in the first j levels are infected, by similar calculations, we can show that all vertices in the $(j+1)$-th level will be infected.

Consider the first level $\{v_{11}, \ldots, v_{n1}\}$. Let $e_1 = (A_{i_1}, A_{i'_1}) \in \overline{E}$. Since the VERTEXCOVER instance is a YES instance, either $A_{i_1} \in \overline{S}$ or $A_{i'_1} \in \overline{S}$, or both. Assume $A_{i_1} \in \overline{S}$ without loss of generality, then all vertices in D_{i_1} are already infected. In the coming round, the vertex $v_{i_1 1} \in V$ will be infected, as

$$\begin{aligned} f_{v_{i_1 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i \in \overline{S}} D_i\right) &= \delta\left|\bigcup_{i \neq i_1} C_i \cup \bigcup_{i \neq i_1, A_i \in \overline{S}} D_i\right| + w_{i_1 1}|C_{i_1}| + \delta\left(1 + \frac{1}{W}\right)|D_{i_1}| \\ &= \delta((n-1) + (\overline{k}-1))W + \frac{1 - (n + \overline{k} - 1)W\delta - \delta}{W} \cdot W \\ &\quad + \delta\left(1 + \frac{1}{W}\right)W \\ &= 1. \end{aligned}$$

If $A_{i'_1} \in \overline{S}$ as well, then $v_{i'_1 1} \in V$ will also be infected in the this round, due to the same calculation. On the other hand, if $A_{i'_1} \notin \overline{S}$, $v_{i'_1 1}$ will be infected in the next round, as

$$\begin{aligned} f_{v_{i'_1 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i \in \overline{S}} D_i \cup \{v_{i_1 1}\}\right) &= \delta\left|\bigcup_{i \neq i'_1} C_i \cup \bigcup_{A_i \in \overline{S}} D_i \cup \{v_{i_1 1}\}\right| + w_{i'_1 1}|C_{i'_1}| \\ &= \delta((n - 1 + \overline{k})W + 1) + \frac{1 - (n + \overline{k} - 1)W\delta - \delta}{W} \cdot W \\ &= 1. \end{aligned}$$

Therefore, both $v_{i_1 1}$ and $v_{i'_1 1}$ will be infected in both cases.

In the next round, the remaining $n-2$ vertices $\{v_{i_0 1}\}_{i_0 \notin \{i_1, i'_1\}; 1 \leq i_0 \leq n}$ will be infected, as we have

$$\begin{aligned}
f_{v_{i_0 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i \in \overline{S}} D_i \cup \{v_{i_1 1}, v_{i'_1 1}\}\right) &= \delta \left|\bigcup_{i \neq i_0} C_i \cup \bigcup_{A_i \in \overline{S}} D_i \cup \{v_{i_1 1}, v_{i'_1 1}\}\right| + w_{i_0 1}|C_{i_0}| \\
&= \delta((n-1+\overline{k})W+2)+ \\
&\qquad \frac{1-(n+\overline{k}-1)W\delta-2\delta}{W}\cdot W \\
&= 1,
\end{aligned}$$

in the case $A_{i_0} \notin \overline{S}$ (such that no vertex in D_{i_0} is infected at this moment), and

$$\begin{aligned}
& f_{v_{i_0 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{i \neq i_0, A_i \in \overline{S}} D_i \cup \{v_{i_1 1}, v_{i'_1 1}\}\right) \\
=& \delta \left|\bigcup_{i \neq i_0} C_i \cup \bigcup_{i \neq i_0, A_i \in \overline{S}} D_i \cup \{v_{i_1 1}, v_{i'_1 1}\}\right| + w_{i_0 1}|C_{i_0}| + \delta\left(1+\frac{1}{W}\right)|D_{i_0}| \\
=& \delta((n-1+\overline{k}-1)W+2) + \frac{1-(n+\overline{k}-1)W\delta-2\delta}{W}\cdot W + \delta\left(1+\frac{1}{W}\right)W \\
=& 1+\delta > 1,
\end{aligned}$$

in the case $A_{i_0} \in \overline{S}$ (such that all vertices in D_{i_0} are infected at the first round). In conclusion, all the n vertices $\{v_{i1}\}_{1 \leq i \leq n}$ will be eventually infected in at most 3 rounds.

The analysis of the second level is similar. For $e_2 = (A_{i_2}, A_{i'_2}) \in \overline{E}$, we have either $A_{i_2} \in \overline{S}$ or $A_{i'_2} \in \overline{S}$ (or both), making one of $v_{i_2 2}, v_{i'_2 2}$ infected (or both), which further makes both $v_{i_2 2}, v_{i'_2 2}$ infected (if one of them is not infected previously), and which eventually makes all the n vertices $\{v_{i2}\}_{1 \leq i \leq n}$ infected.

For each $j = 1, \ldots, m$ with $e_j = (A_{i_j}, A_{i'_j})$, we have either $A_{i_j} \in \overline{S}$ or $A_{i'_j} \in \overline{S}$ (or both). Similar as above, after either two or three rounds, all the vertices in $\{v_{ij}\}_{1 \leq i \leq n}$ will be infected, if all the vertices in $\{v_{i1}\}_{1 \leq i \leq n}, \ldots, \{v_{i(j-1)}\}_{1 \leq i \leq n}$ are already infected.

Therefore, we can see that the cascade after the first round carries on in the following order:

$$v_{i_1 1} \to v_{i'_1 1} \to \{v_{i1}\}_{i \neq i_1, i'_1} \to v_{i_2 2} \to v_{i'_2 2} \to \{v_{i2}\}_{i \neq i_2, i'_2} \to \cdots$$

$$\to v_{i_m m} \to v_{i'_m m} \to \{v_{im}\}_{i \neq i_m, i'_m} \to B.$$

Therefore, we conclude 1 as we already have M infected vertices by just counting those in the bundle B. □

For the proof of (2), we present a general proof idea before the formal proof.

To show (2) by contradiction, we assume that we can choose a seed set $S \subseteq V$ such that $|S| = k = n + \overline{k}$ and $\sigma(S) > M^{\varepsilon}$. By a careful analysis, we can conclude that the only possible way to choose S is as follows.

- an arbitrary vertex from each of $C_1, \ldots, C_n$ (a total of n vertices are chosen);
- an arbitrary vertex from each of $D_{\pi_1}, \ldots, D_{\pi_{\overline{k}}}$ for certain $\{\pi_1, \ldots, \pi_k\} \subseteq \{1, \ldots, n\}$ (a total of $\overline{k}$ vertices are chosen).

The intuitive reason for this is the following: firstly, choosing k seeds among the $2n$ cliques $C_1, \ldots, C_n, D_1, \ldots, D_n$ is considerably more beneficial, as a seed would cause the infection of W vertices; secondly, if we cannot choose both C_i and D_i, it is always better to choose C_i because the weights $w_{i1}, \ldots, w_{im}$ are considerably larger than $\delta(1 + 1/W)$, if δ is set sufficiently small.

Since the VERTEXCOVER instance is a NO instance, there exists an edge $e_j = (A_{i_j}, A_{i'_j})$ such that no vertex in D_{i_j} and $D_{i'_j}$ is chosen as seed. By following similar analysis as in the proof of 1, we can see that the cascade would stop at the level $\{v_{ij}\}_{i=1,\ldots,n}$, which concludes (2).

Proof of (2). Assume that we can choose seed set $S \subseteq V$ such that $|S| = k = n + \overline{k}$ and $\sigma(S) > M^{\varepsilon}$. First notice that choosing any seeds from B is at most as good as choosing seeds from $C_1 \cup \{v_{1j}\}_{1 \leq j \leq m-1}$. By our assumption $m > n + \overline{k} = k$, we can assume without loss of generality that no seed is chosen in B. With this assumption, we will prove that none of these M vertices will be infected in the cascade. Since the graph G has a total of $N = M + M^{\varepsilon}$ vertices, this contradicts that $\sigma(S) > M^{\varepsilon}$.

Suppose, for the sake of contradiction, a vertex $u \in B$ is infected in round t of the cascade, and there is no infected vertex in B in the first $t-1$ rounds. Let I_u be the set of infected vertices before round t. Since u is infected in round t, we have $f_u(I_u) \geq 1$, which, by Definition 7.1, implies

$$\sum_{v \in I_u} w(u, v) \geq 1.$$

We analyze the constituents of I_u.

We set δ to be sufficiently (but still polynomially) small such that

$$(n-1)(2W+m)\delta+\delta\left(1+\frac{1}{W}\right)\ll w_{1m}.^{5}$$

Then the infection of each vertex in $C_1\cup\{v_{1j}\}_{1\leq j\leq m-1}$ has contribution w_{1m} to $f_u(I_u)$, while the net contribution from the infections of all vertices in $V\setminus\{C_1\cup\{v_{1j}\}_{1\leq j\leq m-1}\cup B\}$ is much less than w_{1m}. On the other hand, even if all the $W+m-1$ vertices in $C_1\cup\{v_{1j}\}_{1\leq j\leq m-1}$ are included in I_u, the contribution to $f_u(I_u)$ is

$$(W+m-1)w_{1m}\leq 1-(n+\overline{k}-1)W\delta-(n-1)(m-1)\delta-\delta<1,$$

which is still not enough. Thus, we conclude that $C_1\cup\{v_{1j}\}_{1\leq j\leq m-1}\subseteq I_u$, and the vertices from $V\setminus\{C_1\cup\{v_{1j}\}_{1\leq j\leq m-1}\cup B\}$ should contribute at least $(n+\overline{k}-1)W\delta+(n-1)(m-1)\delta+\delta$ to $f_u(I_u)$. From the term $(n+\overline{k}-1)W\delta$, we can see that at least $n+\overline{k}-1$ cliques from the $2n-1$ cliques $C_2,\ldots,C_n,D_1,\ldots,D_n$ must be included in I_u. Coupled with the observation $C_1\subseteq I_u$, we need at least $n+\overline{k}$ infected cliques from $C_1,\ldots,C_n,D_1,\ldots,D_n$.

On the other hand, the only way to infect a clique C_i or D_i is to seed one of its vertices. To see this for each D_i, it is enough to notice that the weight $\delta(1+1/W)$ is extremely small. To see this for each C_i, notice that only $v_{i1},\ldots,v_{im}$ have non-negligible influence to C_i, and

$$\sum_{j=1}^{m}w_{ij}<\sum_{j=1}^{m}\frac{1}{W-1+j}<m\times\frac{1}{W}=\frac{1}{n}\ll 1.$$

Therefore, to have $u\in B$ infected in round t, the only possible way is to choose $k=n+\overline{k}$ seeds from $n+\overline{k}$ cliques, among all the $2n$ cliques $C_1,\ldots,C_n,D_1,\ldots,D_n$. Lastly, it is straightforward to check that the infection of an vertex in D_i is less influential than the infection of an vertex in the corresponding C_i (both C_i and D_i contain the same number of vertices, so their influences to the outside subtrees are the same; however, C_i is connected to $v_{i1},\ldots,v_{im}$ with higher weights than D_i). Thus, we can assume without loss of generality that S consists of

- an arbitrary vertex from each of $C_1,\ldots,C_n$ (a total of n vertices are chosen);
- an arbitrary vertex from each of $D_{\pi_1},\ldots,D_{\pi_{\overline{k}}}$ for certain $\{\pi_1,\ldots,\pi_k\}\subseteq\{1,\ldots,n\}$

[5] This is always possible: when $\delta\to 0$, the left-hand side approaches to 0, while we have $\lim_{\delta\to 0}w_{1m}=\frac{1}{W+m-1}$ for the right-hand side.

(a total of $\overline{k}$ vertices are chosen).

Since the VertexCover instance is a NO instance, for the choice $\overline{S} = \{A_{\pi_1}, \dots, A_{\pi_{\overline{k}}}\}$, there exists edge e_j that is not covered by $\overline{S}$. Let j^* be the smallest j such that e_j is not covered by $\overline{S}$.

We first deal with the case $j^* = 1$. The case where $j^* > 1$ is dealt with subsequently.

If $j^* = 1$, for $e_1 = (A_{i_1}, A_{i'_1})$, we have $A_{i_1}, A_{i'_1} \notin \overline{S}$. In this case, $v_{i_1 1}$ will not be infected, as

$$\begin{aligned} f_{v_{i_1 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i \in \overline{S}} D_i\right) &= \delta \left|\bigcup_{i \neq i_1} C_i \cup \bigcup_{A_i \in \overline{S}} D_i\right| + w_{i_1 1}|C_{i_1}| \\ &= \delta(n-1+\overline{k})W + \frac{1-(n+\overline{k}-1)W\delta - \delta}{W} \cdot W \\ &= 1-\delta < 1, \end{aligned}$$

and $v_{i'_1 1}$ will not be infected for the same reason. For $i_0 \neq i_1, i'_1$, $v_{i_0 1}$ will not be infected either, as we have

$$\begin{aligned} f_{v_{i_0 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i \in \overline{S}} D_i\right) &= \delta \left|\bigcup_{i \neq i_0} C_i \cup \bigcup_{A_i \in \overline{S}} D_i\right| + w_{i_0 1}|C_{i_0}| \\ &= \delta(n-1+\overline{k})W + \frac{1-(n+\overline{k}-1)W\delta - 2\delta}{W} \cdot W \\ &= 1-2\delta < 1, \end{aligned}$$

in the case $A_{i_0} \notin \overline{S}$, and

$$\begin{aligned} f_{v_{i_0 1}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i \in \overline{S}} D_i\right) &= \delta \left|\bigcup_{i \neq i_0} C_i \cup \bigcup_{i \neq i_0, A_i \in \overline{S}} D_i\right| + w_{i_0 1}|C_{i_0}| + \delta\left(1+\frac{1}{W}\right)|D_{i_0}| \\ &= \delta(n-1+\overline{k}-1)W + \frac{1-(n+\overline{k}-1)W\delta - 2\delta}{W} \cdot W \\ &\quad + \delta\left(1+\frac{1}{W}\right)W \\ &= 1-\delta < 1, \end{aligned}$$

in the case $A_{i_0} \in \overline{S}$. Thus, none of $\{v_{i1}\}_{1 \le i \le n}$ will be infected. Since $w_{ij_1} > w_{ij_2}$ whenever $j_1 < j_2$ for any i (easy to see by observing $w_{ij} \approx \frac{1}{W-1+j}$), none of $\{v_{ij}\}_{1 \le i \le n; 2 \le j \le m}$

will be infected. In particular, no vertex in B can be infected, which leads to the desired contradiction.

If $j^* > 1$, by the similar analysis in the proof of 1 for the YES instance case, after many cascade rounds, all vertices in $\{v_{ij}\}_{1\leq i\leq n;1\leq j\leq j^*-1}$ will be infected. For $e_{j^*} = (A_{i_{j^*}}, A_{i'_{j^*}})$, we have $A_{i_{j^*}}, A_{i'_{j^*}} \notin \overline{S}$. In this case, $v_{i_{j^*}j^*}$ will not be infected, as

$$\begin{aligned}
&f_{v_{i_{j^*}j^*}}\left(\bigcup_{i=1}^{n} C_i \cup \bigcup_{A_i\in\overline{S}} D_i \cup \{v_{ij}\}_{1\leq i\leq n;1\leq j\leq j^*-1}\right)\\
=&\delta\left|\bigcup_{i\neq i_{j^*}} C_i \cup \bigcup_{A_i\in\overline{S}} D_i\right| + w_{i_{j^*}j^*}\left|C_{i_{j^*}} \cup \{v_{i_{j^*}j}\}_{1\leq j\leq j^*-1}\right| + \delta\left|\{v_{ij}\}_{i\neq i_{j^*},1\leq j\leq j^*-1}\right|\\
=&\delta(n-1+\overline{k})W\\
&\qquad + \frac{1-(n+\overline{k}-1)W\delta-(n-1)(j^*-1)\delta-\delta}{W-1+j^*}\cdot(W+j^*-1)+\delta(n-1)(j^*-1)\\
=&1-\delta<1,
\end{aligned}$$

and $v_{i'_{j^*}j^*}$ will not be infected for the same reason. Following similar analysis, $v_{i_0j^*}$ will not be infected for $i_0 \neq i_{j^*}, i'_{j^*}$, and none of $\{v_{ij^*}\}_{1\leq i\leq n}$ will be infected. By the same observation $w_{ij_1} > w_{ij_2}$ whenever $j_1 < j_2$, none of $\{v_{ij}\}_{1\leq i\leq n;j^*\leq j\leq m}$ will be infected. In particular, no vertex in B can be infected, which again leads to the desired contradiction. We conclude (2) here. □

Since $M = \Theta(N)$, (1) and (2) imply Theorem 7.5.

7.4 Hardness of Approximation for Stochastic Hierarchical Blockmodel

In this section, we will present strong inapproximability results for both pre-sampling and post-sampling versions of stochastic hierarchical blockmodel InfMax. A major difference between the results in Section 7.3 and this section is that the strong inapproximability result no longer holds if we assume $r_v = 1$ for all $v \in V$ in the stochastic hierarchical blockmodel. In fact, if all the thresholds are fixed to be 1, $\sigma(\cdot)$ in both Definition 7.3 and Definition 7.4 become submodular, in which case we can have a simple greedy $(1-1/e)$-approximation algorithm [44, 59]. In particular, assuming $r_v = 1$ for all $v \in V$ makes post-sampling InfMax trivial: as an infected seed will eventually infect a whole connected component of G, the optimal way of choosing S

is to choose k seeds from the first k largest connected components, after seeing the sampling $G \sim \mathcal{G}$. For pre-sampling InfMax, the model becomes exactly `ICM`, which is submodular.

The following two theorems are the same, except that Theorem 7.7 corresponds to the hardness for pre-sampling model (see Definition 7.3), while Theorem 7.8 show the same hardness result for the post-sampling model (see Definition 7.4) via a randomized Karp's reduction.

Theorem 7.7. *Consider the pre-sampling stochastic hierarchical blockmodel* InfMax *problem* $(\mathcal{G}, R, k)$. *For any* $\varepsilon > 0$, *it is NP-hard to distinguish between the following two cases:*

- YES*: there exists a seed set* S *with* $|S| = k$ *such that* $\mathop{\mathbb{E}}_{G\sim\mathcal{G}}[\sigma(S)] = \Theta(N)$;
- NO*: for any seed set* S *with* $|S| = k$, *we have* $\mathop{\mathbb{E}}_{G\sim\mathcal{G}}[\sigma(S)] = O(N^{\varepsilon})$.

Theorem 7.8. *Consider the post-sampling stochastic hierarchical blockmodel* InfMax *problem* $(\mathcal{G}, R, k)$. *For any* $\varepsilon > 0$ *and* $c > 0$, *it is NP-hard to distinguish between the following two cases with probability at least* N^{-c} *(where the probability is taken over* $G \sim \mathcal{G}$*):*

- YES*: there exists a seed set* S *with* $|S| = k$ *such that* $\sigma(S) = \Theta(N)$;
- NO*: for any seed set* S *with* $|S| = k$, *we have* $\sigma(S) = O(N^{\varepsilon})$.

As a remark to Theorem 7.8, the theorem says that if we have an oracle that outputs a solution which approximates $\max_{S\subseteq V,|S|\leq k}\sigma(S)$ within a factor of $N^{1-\varepsilon}$ for certain samples $G \sim \mathcal{G}$, and with probability at least N^{-c} we receive a sample G in the set of graphs for which the oracle outputs valid solutions, then we can use this oracle to solve any NP-complete problem as long as we have randomness to sample $G \sim \mathcal{G}$.

We will prove both Theorem 7.7 and Theorem 7.8 by a reduction from VertexCover. Given a VertexCover instance $(\overline{G} = (\overline{V}, \overline{E}), \overline{k})$, we will construct a hierarchy tree T which determines $\mathcal{G}$ for both proofs.

The reduction Let $n = |\overline{V}|$ and $m = \overline{E}$ as usual. Assume $m > n > \overline{k}^2 + 2$, and $\log_2 n$ is an integer.[6] In addition, we assume that $A_1 \in \overline{S}$ whenever the

[6] Notice that we can assume $n \gg \overline{k}$ is an integer power of 2 by adding isolated vertices to $\overline{G}$ which are never picked, and we can assume $m > n$ by duplicate each edge (which makes $\overline{G}$ a multi-graph).

VERTEXCOVER instance is a YES instance.[7]

We define the following variables used in this section.

$$\delta = \frac{1}{10mn^2\overline{k}}, \qquad \text{and} \qquad \Delta = mn^2\delta = \frac{1}{10\overline{k}}, \qquad W = m^{10}n^{10}.$$

Let M be an extremely large number whose value will be decided later.

The construction of T is shown in Figure 7.4. T is a full balanced binary tree with $\log_2 n$ levels and n leaves. The weight of all non-leaf nodes is $1/W$, and the weight of all leaves is 1. The i-th leaf corresponds to $A_i \in \overline{V}$ in the VERTEXCOVER instance. Recall from Definition 7.2 that $\mathcal{G} = (V, T)$ is determined by T, and in particular each leaf of T corresponds to a subset of V. As the weight of each leaf is 1, meaning each edge appear with probability 1, its corresponding subset of vertices forms a clique in all $G \sim \mathcal{G}$. We will call the clique corresponding to the i-th leaf *the i-th clique* in the remaining part of this section. For each clique i, we will first describe the vertices we have constructed in Figure 7.4, and then define their thresholds.

For positive integers x, y, denote by $B(x, y)$ a bundle of x vertices with threshold y. For each $i = 1, \ldots, n$, we construct the following vertices for the i-th clique:

- a bundle of $\overline{k}W^2$ vertices: $B_i := B\left(\overline{k}W^2, \infty\right)$, and
- $m(n-2)$ bundles of W^3 vertices: $B_{ij\imath} := B\left(W^3, r_{ij\imath}\right)$ for $j = 1, \ldots, m$ and $\imath = 1, \ldots, n-2$.

For $i = 1$, we add an extra bundle $C := \left(M, r_{1(m+1)}\right)$. The thresholds $\{r_{ij\imath}\}$ and $r_{1(m+1)}$ of those constructed vertices will be defined later.

By our construction, the 1-st clique has $M + \overline{k}W^2 + m(n-2)W^3$ vertices, which is much more than the number of vertices $\overline{k}W^2 + m(n-2)W^3$ in each of the remaining cliques. As a remark, we have constructed $N = M + nm(n-2)W^3 + n\overline{k}W^2$ vertices for G. Moreover, for M whose value we have not decided yet, we can make it arbitrarily close to N.

Denote by $B_{\cdot j\imath} := \{B_{ij\imath}\}_{i=1,\ldots,n}$ the n bundles in a horizontal level in Figure 7.4 (for example, in Figure 7.4, after the top-level $\{B_1, \ldots, B_n\}$, there come levels $B_{\cdot 11}, B_{\cdot 12}, \ldots$). We will call $B_{\cdot j\imath}$ a *level* and abuse the word "level" to refer to the vertices in $B_{\cdot j\imath}$.

The correspondence between the VERTEXCOVER instance and the graph we constructed is as follows. Recall that each vertex $A_i \in \overline{V}$ corresponds to the i-th clique.

[7] This assumption can be made without loss of generality because we can add two extra vertices named A_1, A_2 and one extra edge (A_1, A_2) such that one of A_1, A_2 much be chosen to cover this edge, and we can assume A_1 is chosen.

Now, for each edge $e_j \in \overline{E}$, we have constructed $n-2$ levels $B_{\cdot j1}, \ldots, B_{\cdot j(n-2)}$, which are $n(n-2)$ bundles of W^3 vertices. For example, in Figure 7.4, we have illustrated the $n-2$ levels corresponding to e_1 and the $n-2$ levels corresponding to e_m, while the levels corresponding to the remaining edges in $\overline{E}$ are omitted.

For each $j = 1, \ldots, m$ and $\imath = 1, \ldots, n-2$, we denote by $B_{\prec j\imath}$ the union of the first $(j-1)(n-2)+\imath-1$ levels (where the levels are ordered from up to down in Figure 7.4):

$$\begin{aligned} B_{\prec j\imath} :=& \bigcup_{(n-2)j'+\imath'<(n-2)j+\imath} B_{\cdot j'\imath'} \\ =& B_{\cdot 11} \cup B_{\cdot 12} \cup \ldots \cup B_{\cdot 1(\imath-1)} \cup B_{\cdot 1\imath} \cup \ldots \cup B_{\cdot 1(n-3)} \cup B_{\cdot 1(n-2)} \cup \\ & B_{\cdot 21} \cup B_{\cdot 22} \cup \ldots \cup B_{\cdot 2(\imath-1)} \cup B_{\cdot 2\imath} \cup \ldots \cup B_{\cdot 2(n-3)} \cup B_{\cdot 2(n-2)} \cup \\ & \cdots \\ & B_{\cdot j1} \cup B_{\cdot j2} \cup \ldots \cup B_{\cdot j(\imath-1)}. \end{aligned}$$

Next, we define the thresholds $\{r_{ij\imath}\}$ and $r_{1(m+1)}$. Denote

$$\omega_{j\imath} := \left((j-1)(n-2)+(\imath-1)\right) W^3 + (n-1)\left((j-1)(n-2)+(\imath-1)\right) W^2,$$

which is the expected number of neighbors of each $b_{ij\imath} \in B_{ij\imath}$ in $B_{\prec j\imath}$. For each fixed j, denote by i_j, i'_j the two indices such that $e_j = (A_{i_j}, A_{i'_j})$ with $i_j < i'_j$, and all $r_{ij\imath}$'s are defined as follows.

$$\begin{bmatrix} r_{1j1} & r_{2j1} & \cdots & r_{nj1} \\ r_{1j2} & r_{2j2} & \cdots & r_{nj2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{1jn} & r_{2jn} & \cdots & r_{njn} \end{bmatrix} := \begin{bmatrix} \omega_{j1}+(1-\Delta)W^2 & \omega_{j1}+(1-\Delta)W^2 & \cdots & \omega_{j1}+(1-\Delta)W^2 \\ \omega_{j2}+(1-\Delta)W^2 & \omega_{j2}+(1-\Delta)W^2 & \cdots & \omega_{j2}+(1-\Delta)W^2 \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{jn}+(1-\Delta)W^2 & \omega_{jn}+(1-\Delta)W^2 & \cdots & \omega_{jn}+(1-\Delta)W^2 \end{bmatrix} +$$

$$\left[\begin{array}{ccccccccccc} & & & & \text{Column } i_j & & \text{Column } i'_j & & & & \\ 1W^2 & 2W^2 & 3W^2 & \cdots & 0 & \cdots & 0 & \cdots & (n-3)W^2 & (n-2)W^2 \\ (n-2)W^2 & 1W^2 & 2W^2 & \cdots & 0 & \cdots & 0 & \cdots & (n-4)W^2 & (n-3)W^2 \\ (n-3)W^2 & (n-2)W^2 & 1W^2 & \cdots & 0 & \cdots & 0 & \cdots & (n-5)W^2 & (n-4)W^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 2W^2 & 3W^2 & 4W^2 & \cdots & 0 & \cdots & 0 & \cdots & (n-2)W^2 & 1W^2 \end{array}\right]$$

Notice that for different $\imath_1, \imath_2 \in \{1, \ldots, n{-}2\}$, $(r_{1j\imath_1} - \omega_{j\imath_1}, r_{2j\imath_1} - \omega_{j\imath_1}, \ldots, w_{nj\imath_1} - \omega_{j\imath_1})$

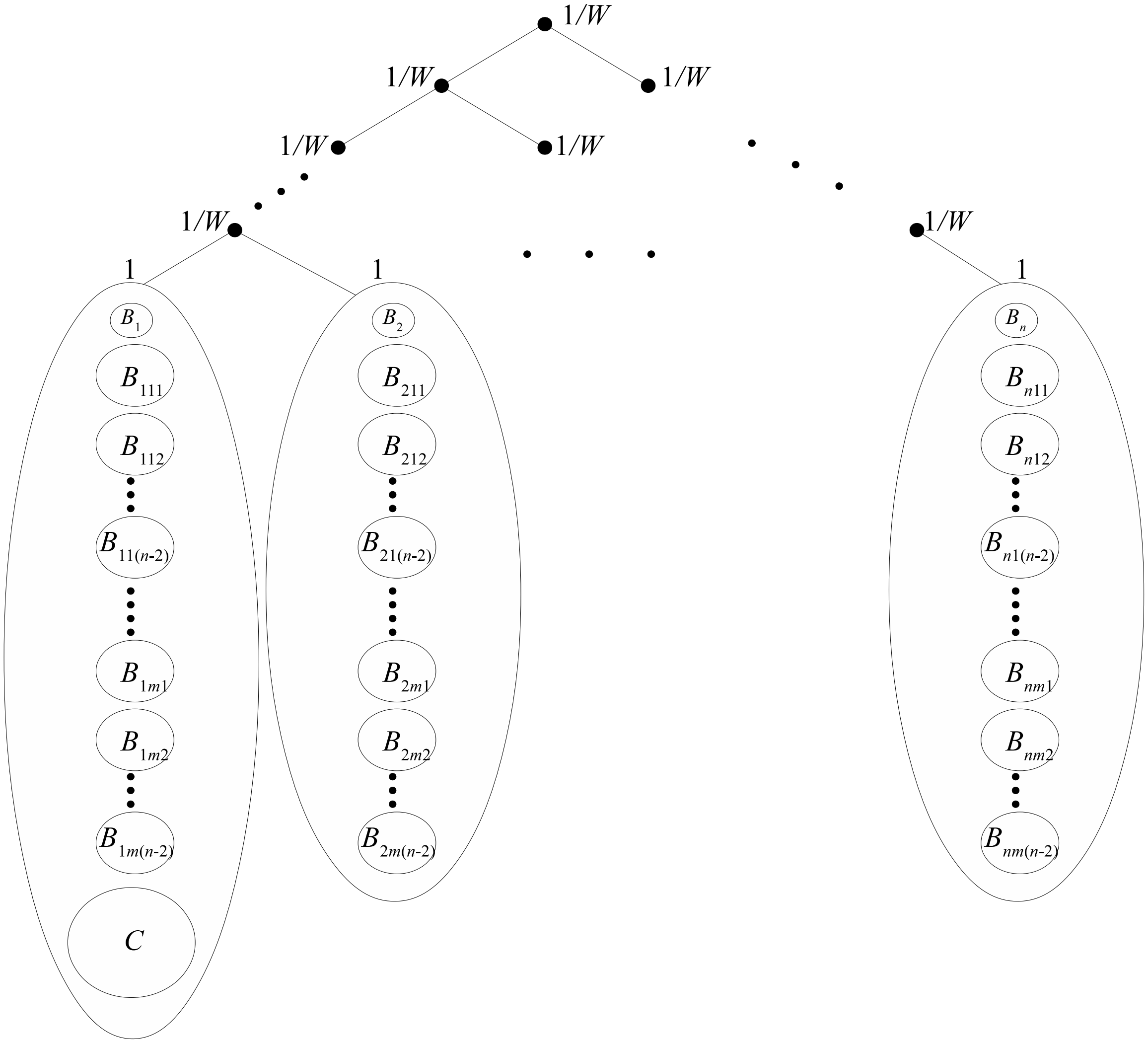


Figure 7.4: The construction of the hierarchy tree T for proving Theorem 7.7 and 7.8.

is a permutation of $(w_{1j\imath_2} - \omega_{j\imath_2}, w_{2j\imath_2} - \omega_{j\imath_2}, \dots, w_{nj\imath_2} - \omega_{j\imath_2})$. Specifically, for the second matrix above, excluding the i_j-th and the i'_j-th columns, the first row is an arithmetic progression $1W^2, 2W^2, (n-2)W^2$, and the $(\imath+1)$-th row is obtained by cyclically shifting the $\imath$-th row to the right by 1 unit.

Finally, for the threshold $r_{1(m+1)}$ of each vertex in the bundle C. We define

$$r_{1(m+1)} := m(n-2)W^3 + (n-1)m(n-2)W^2 + (1-\Delta)W^2.$$

As we will see later, $r_{1(m+1)}$ is slightly less than the expected number of neighbors of each $c \in C$ in $V \setminus C$, by an amount of $\Theta(\Delta W^2)$.

The high-level ideas Before presenting rigorous arguments, we provide high level ideas of the reduction in this subsection.

We have constructed the hierarchy tree T, which corresponds to a graph distribution $\mathcal{G}$ (refer to Definition 7.2). In the next subsection, we will show that a sample $G \sim \mathcal{G}$ can simulate the corresponding VERTEXCOVER instance with high probability. In particular, we will say such samples are "good" samples, which we will define rigorously, and we will prove that a sample is good with probability $1 - o(1)$.

Given a VERTEXCOVER instance $(\overline{G}, \overline{k})$, we consider the INFMAX instance (G, R, k), where G is a good sample and $k = \overline{k}W^2$.

Suppose we have a good sample G. If the VERTEXCOVER instance is a YES instance, we can find $\overline{S} \subseteq \overline{V}$ with $|\overline{S}| = \overline{k}$ such that $\overline{S}$ covers all edges in $\overline{E}$. For each $A_i \in \overline{S}$, we choose W^2 seeds from the bundle B_i, so a total of $\overline{k}W^2 = k$ seeds are chosen.

Similar to what happens in Section 7.3, the cascade will flow level-by-level. In particular, for the first edge $e_1 \in \overline{E}$ and i_1, i_1' such that $e_1 = (A_{i_1}, A_{i_1'})$, the vertices in the bundles $B_{i_1 11}$ and $B_{i_1' 11}$ have the lowest threshold in the level $B_{\cdot 11}$. On the other hand, by our choice of k seeds, we have chosen W^2 seeds from one (or both) of B_i and $B_{i'}$. Calculations show that these seeds are just enough to infect all vertices in $B_{i_1 11}$ and $B_{i_1' 11}$. The infection of these vertices will eventually infected the entire level $B_{\cdot 11}$, and similar analysis shows that the levels $B_{\cdot 12}, B_{\cdot 13}, ...$ will be infected one-by-one. Finally, the cascade can reach the huge bundle C, and most vertices in G will be infected.

If the VERTEXCOVER instance is a NO instance, we can assume all seeds are chosen from $\{B_1, \ldots, B_n\}$, as it is always a better idea to choose seeds from vertices having higher thresholds in a clique.[8] We say that the i-th clique is activated if we have chosen almost W^2 seeds from B_i, or more than this number. We can draw an analogy between activating the i-th clique in INFMAX and picking the set A_i in VERTEXCOVER.

Since the VERTEXCOVER instance is a NO instance, certain element e_{j^*} is not covered, and we will show that the cascade will stop at one of the $n-2$ levels $B_{\cdot j^* 1}, \ldots, B_{\cdot j^*(n-2)}$. Intuitively, the thresholds of vertices in these levels shift cyclically by our construction, and there exists a level whose vertices' thresholds are shifted to the position such that the cascade fails on all leaves. In particular, even if we put

[8] Rigorously, this may not be true in the post-sampling case, where the seed-picker can see the sample G. The vertices not in $\{B_1, \ldots, B_n\}$ may happen to have more neighbors across cliques, and the seed-picker can take advantage of this. We will reason about this later. However, for now, we assume all seeds are chosen from $\{B_1, \ldots, B_n\}$.

all $k = \overline{k}W^2$ seeds in a single bundle B_i, there exists a level $\imath$ such that $r_{ij^*\imath}$ is large enough, making the cascade still fail on leaf i. On the other hand, there are only two leaves i_{j^*}, i'_{j^*} having lowest $r_{ij^*\imath}$ in all levels $\imath = 1, \ldots, n-2$, which are exactly those i_{j^*}, i'_{j^*} with $e_{j^*} = (A_{i_{j^*}}, A_{i'_{j^*}})$. However, we have very few seeds (considerably fewer than W^2) on the i_{j^*}-th and the i'_{j^*}-th cliques, by our assumption that e_{j^*} is not covered.

Since the cascade will fail on a certain intermediate level, it cannot reach the huge bundle C. By making C contain most vertices in G (i.e., making M large enough), we can see that the number of infected vertices corresponding to a YES VertexCover instance is significantly higher, which implies both Theorem 7.7 and Theorem 7.8.

In the next two subsections, we will rigorously prove the correctness of our reduction.

Good samplings In this subsection, we define "good" samplings $G \sim \mathcal{G}$ which are useful in the reduction from VertexCover, in the sense that G successfully simulates the VertexCover instance, and we show that a sample $G \sim \mathcal{G}$ is good with a high probability.

Firstly, consider a W^3 sized bundle $B_{ij\imath}$, and an arbitrary vertex v not in the i-th clique. Over all the samplings $G \sim \mathcal{G}$, v's expected number of neighbors in $B_{ij\imath}$ is

$$\mathop{\mathbb{E}}_{G\sim\mathcal{G}}\left[|\Gamma(v) \cap B_{ij\imath}|\right] = \frac{1}{W} \cdot W^3 = W^2.$$

Secondly, consider a set D_i of δW^2 vertices in the i-th clique, and a set D_{-i} of $(\overline{k}+1)W^2$ vertices that are not in the i-th clique, the expected total number of edges between D_i and D_{-i} is

$$\mathop{\mathbb{E}}_{G\sim\mathcal{G}}\left[|\{(u,v) : u \in D_i, v \in D_{-i}\}|\right] = \frac{1}{W} \cdot \delta W^2 \cdot (\overline{k}+1)W^2 = \delta(\overline{k}+1)W^3.$$

We define a sampling $G \sim \mathcal{G}$ to be "good" if the above two numbers roughly concentrate on their expectations.

Definition 7.9. A sampling $G \sim \mathcal{G}$ is *good* if the following holds.

1. For all $i = 1, \ldots, n$, $j = 1, \ldots, m$ and $\imath = 1, \ldots, n-2$, and any vertex v not in the i-th clique,
$$(1-\delta)W^2 < |\Gamma(v) \cap B_{ij\imath}| < (1+\delta)W^2.$$
2. For any set D_i of δW^2 vertices in the i-th clique, and any set D_{-i} of $(\overline{k}+1)W^2$

vertices that are not in the i-th clique, the number of edges between D_i and D_{-i} is less than $W^{3.6}$:

$$|\{(u,v) : u \in D_i, v \in D_{-i}\}| < W^{3.6}.$$

The following lemma shows that a sampling $G \sim \mathcal{G}$ is good with high probability.

Lemma 7.10. *A sampling $G \sim \mathcal{G}$ is good with probability more than $1 - e^{-\sqrt{W}}$.*

Proof. We apply Chernoff-Hoeffding inequality and union bounds to show this lemma. In a random sample $G \sim \mathcal{G}$, for each $i = 1, \ldots, n$; $j = 1, \ldots, m$; $\imath = 1, \ldots, n-2$ and v, requirement 1 in Definition 7.9 fails with probability

$$\Pr\left[\left|W^2 - |\Gamma(v) \cap B_{ij\imath}|\right| \geq \delta W^2\right] \leq 2\exp\left(-\frac{1}{2}\left(\delta W^2\right)^2 \frac{1}{W^3}\right) < e^{-W^{0.6}},$$

where the last inequality is due to $(\delta W^2)^2 = \frac{1}{\overline{k}^2} m^{38} n^{36} > W^{3.6}$.

For each D_i and D_{-i}, requirement 2 in Definition 7.9 fails with probability

$$\Pr\left[|\{(u,v) : u \in D_i, v \in D_{-i}\}| \geq W^{3.6}\right] \leq \exp\left(-\frac{1}{2}\frac{\left(W^{3.6} - \delta(\overline{k}+1)W^3\right)^2}{\delta W^2 \cdot (\overline{k}+1)W^2}\right) < e^{-W^3}.$$

By a union bound, the probability that a sample $G \sim \mathcal{G}$ is not good is

$$\begin{aligned}
\Pr[\text{not good}] &< nm(n-2)Ne^{-W^{0.6}} + \binom{N}{\delta W^2}\binom{N}{\overline{k}W^2}e^{-W^3} \\
&< N^2 e^{-W^{0.6}} + N^{\delta W^2 + \overline{k}W^2} e^{-W^3} \\
&= e^{2\log N} e^{-W^{0.6}} + e^{(\delta W^2 + \overline{k}W^2)\log N} e^{-W^3} \\
&< e^{-\sqrt{W}},
\end{aligned}$$

(as $N = \text{poly}(W)$, which implies $\log N = o(W^c)$ for arbitrary $c > 0$)

which immediately implies the lemma. □

The reduction correctness In this section, we show that InfMax on a good sample $G \sim \mathcal{G}$ simulates the VertexCover problem.

Lemma 7.11. *Consider* InfMax *with $k = \overline{k}W^2$ seeds. For any good sample $G \sim \mathcal{G}$,*

1. *if the* VertexCover *instance is a* YES *instance, a total of $\overline{k}W^2 + nm(n-2)W^3 + M$ vertices can be infected by properly choosing the k seeds;*

2. *if the* VertexCover *instance is a* NO *instance, at most $N - M$ vertices can be infected for any choices of the k seeds.*

Proof of (1). Suppose the VertexCover instance is a YES instance. Let $\overline{S}$ be the choice of $\overline{k}$ vertices in VertexCover instance that covers all edges in $\overline{E}$. As mentioned earlier, we can assume $A_1 \in \overline{S}$. For each $A_i \in \overline{S}$, we choose W^2 seeds from the bundle B_i, so a total of $\overline{k}W^2 = k$ seeds are chosen.

We show that all vertices in the level $B_{\cdot 11}$ will be infected. Consider $e_1 = (A_{i_1}, A_{i'_1})$ with $i_1 < i'_1$. By the fact the VertexCover instance is a YES instance and the way we choose the seeds, W^2 vertices in either B_{i_1} or $B_{i'_1}$, or both, are seeded. Assume without loss of generality that W^2 vertices from B_{i_1} are seeded, then all the vertices in the bundle $B_{i_1 11}$, having threshold $r_{i_1 11} = \omega_{11} + (1-\Delta)W^2 + 0 = (1-\Delta)W^2 < W^2$ will be infected. As for the vertices in $B_{i'_1 11}$, they will be infected in the same way if W^2 vertices from $B_{i'_1}$ are also seeded. On the other hand, if no vertex in $B_{i'_1}$ is seeded, all vertices in $B_{i'_1 11}$ will be infected due to the influence of $B_{i_1 11}$. This is because 1) each vertex in $B_{i'_1 11}$ has more than $(1-\delta)W^2$ infected neighbors in $B_{i_1 11}$ by requirement 1 of Definition 7.9, and 2) each vertex in $B_{i'_1 11}$ has threshold $(1-\Delta)W^2 < (1-\delta)W^2$. In the next $n-2$ iterations, by a careful calculation and based on requirement 1 of Definition 7.9, all vertices in the remaining $n-2$ bundles $\{B_{i11}\}_{i \neq i_1, i'_1}$ will be infected in the following order:

$$B_{111} \to B_{211} \to \cdots B_{(i_1-1)11} \to B_{(i_1+1)11} \to \cdots B_{(i'_1-1)11} \to B_{(i'_1+1)11} \to \cdots \to B_{n11}. \tag{7.2}$$

Therefore, the entire level $B_{\cdot 11}$ will be infected.

By similar analysis, we will show that the next level $B_{\cdot 12}$ will be infected after the previous level $B_{\cdot 11}$. Again, assume without loss of generality that W^2 seeds in B_{i_1} are chosen. (Remember that the first $n-2$ levels are for edge $e_1 \in \overline{E}$, so we are still working on e_1.) Each vertex in $B_{i_1 12}$ has $(W^2 + W^3)$ infected neighbor in the i_1-th clique, and has more than $(n-1)(1-\delta)W^2$ infected neighbors in $\{B_{i11}\}_{i \neq i_1}$, which is a total of more than $W^3 + nW^2 - (n-1)\delta W^2$ neighbors. Moreover, each vertex in $B_{i_1 12}$ has threshold $r_{i_1 12} = \omega_{12} + (1-\Delta)W^2 + 0 = W^3 + (n-1)W^2 + (1-\Delta)W^2 = W^3 + nW^2 - \Delta W^2$ which is less than the number of infected neighbors, as $-\Delta < -(n-1)\delta$. Therefore, all vertices in $B_{i_1 12}$ will be infected. As for the vertices in $B_{i'_1 12}$, following the analysis in the last paragraph, they will be infected at the same iteration if W^2 vertices in $B_{i'_1}$ are seeded, and they will be infected at the next iteration due to the extra influence from $B_{i_1 12}$ if not. Finally, the remaining $n-2$ bundles $\{B_{i12}\}_{i \neq i_1, i'_1}$

will be infected in the following order:

$$B_{212} \to B_{312} \to \cdots B_{(i_1-1)12} \to B_{(i_1+1)12} \to \cdots B_{(i'_1-1)12} \to B_{(i'_1+1)12} \to \cdots B_{n12} \to B_{112}, \tag{7.3}$$

which is similar to (7.2), but is cyclically shifted to the left by 1 unit, due to our cyclic construction of the thresholds. Thus, we have shown that the level $B_{\cdot 12}$ will be infected after the previous level $B_{\cdot 11}$.

Following the same analysis, we can conclude that all levels will be infected in the following order:

$$\begin{gathered} B_{\cdot 11} \to B_{\cdot 12} \to \cdots \to B_{\cdot 1(n-2)} \to \\ B_{\cdot 21} \to B_{\cdot 22} \to \cdots \to B_{\cdot 2(n-2)} \to \\ \cdots \\ B_{\cdot m1} \to B_{\cdot m2} \to \cdots \to B_{\cdot m(n-2)}. \end{gathered}$$

Lastly, each vertex $c \in C$ has $W^2 + m(n-2)W^3$ infected neighbors in the 1-st clique (notice that we assume $A_1 \in \overline{S}$, which implies W^2 vertices in B_1 are seeded, which contributes W^2 infected neighbors), and more than $(n-1) \cdot m(n-2)(1-\delta)W^2$ infected neighbors from the other $n-1$ cliques, which is a total of $m(n-2)W^3 + (n-1)m(n-2)W^2 + W^2 - (n-1)m(n-2)\delta W^2$ neighbors. In addition, c has threshold $r_{1(m+1)} = m(n-2)W^3 + (n-1)m(n-2)W^2 + (1-\Delta)W^2$, which is less than the number of infected neighbors, as we have $-\Delta = -mn^2\delta < -(n-1)m(n-2)\delta$. Consequently, all vertices in C will be infected. By summing up the total number of infected vertices, we conclude the first part of this lemma. □

Proof of (2). Suppose the VERTEXCOVER instance is a NO instance. For those $n\overline{k}W^2$ vertices in $\{B_i\}_{i=1,\ldots,n}$ having threshold ∞, they will not be infected unless being seeded, which means at least $(n-1)\overline{k}W^2$ of them will not be infected. To show that the total number of infected vertices cannot exceed $N - M$, it is enough to show that at most $(n-1)\overline{k}W^2$ vertices can be infected in the bundle C of M vertices. We will show the following stronger claim.

Proposition 7.12. *If the* VERTEXCOVER *instance is a* NO *instance, all vertices in C will not be infected unless being seeded.*

To show Proposition 7.12, we show that the cascade will stop at an intermediate level. We will first identify this level, and then show this claim in Proportion 7.13.

Consider an arbitrary seed set S (with $|S| = k$). Let S_i be the seeds chosen from the i-th clique, and $k_i = |S_i|$ so that $\sum_{i=1}^n k_i = k$. We say that the i-th clique is activated if $k_i \geq (1-9\Delta)W^2$. Since $(\overline{k}+1)(1-9\Delta)W^2 = \overline{k}W^2 + \left(1 - \frac{9}{10} - \frac{9}{10\overline{k}}\right) W^2 > k$, at most $\overline{k}$ cliques can be activated.

If we draw an analogy between activating a clique and picking a vertex in VertexCover, by the fact that the VertexCover instance is a NO instance, there exists j^* where $e_{j^*} = (A_{i_{j^*}}, A_{i'_{j^*}})$ such that both i_{j^*}-th and i'_{j^*}-th cliques are not activated. For the ease of illustration, assume without loss of generality that $i_{j^*} = n-1$ and $i'_{j^*} = n$. Since we have assumed $n > \overline{k}^2 + 2$, there exists $\imath^* \leq n - 1 - \overline{k}$ such that the $\imath^*$-th, the $(\imath^*+1)$-th, ..., and the $(\imath^* + \overline{k} - 1)$-th cliques are not activated. (If we have an activated clique within any $\overline{k}$ consecutive cliques in the first $n-2$ cliques, the total number of activated cliques is at least $\frac{n-2}{\overline{k}} > \overline{k}$, which is a contradiction.) We will show that the cascade stops at the level $B_{\cdot j^* \imath^*}$. That is, there are only $o(W^3)$ infected vertices in

$$\left(\bigcup_{(n-2)j+\imath \geq (n-2)j^* + \imath^*} B_{\cdot j \imath} \right) \cup C = V \setminus (B_1 \cup \cdots \cup B_n \cup B_{\prec j^* \imath^*}) .$$

We will show that this is true even in the case that all vertices in the previous $(n-2)(j^*-1) + \imath^* - 1$ levels (i.e., those in $B_{\prec j^* \imath^*}$) are infected.

Proposition 7.13. *There are only $o(W^3)$ infected vertices in the level $B_{\cdot j^* \imath^*}$, given that all vertices in $B_{\prec j^* \imath^*}$ and at most $\overline{k}W^2$ vertices elsewhere (i.e., in $V \setminus B_{\prec j^* \imath^*}$) are infected.*

The "$\overline{k}W^2$ vertices elsewhere" mentioned in Proposition 7.13 refer to the $k = \overline{k}W^2$ seeds. Notice that the seed-picker may choose the seeds outside $B_{\prec j^* \imath^*}$, and Proposition 7.13 holds even if all vertices in $B_{\prec j^* \imath^*}$ are infected and the k seeds are all outside $B_{\prec j^* \imath^*}$.

Before proving Proposition 7.13, we remark that Proposition 7.13 immediately implies Proposition 7.12: the vertices in the later levels $B_{\cdot j^*(\imath^*+1)}, B_{\cdot j^*(\imath^*+2)}, \ldots, B_{\cdot m(n-2)}$ have thresholds even higher than the thresholds of vertices in $B_{\cdot j^* \imath^*}$, and the thresholds increase by $\Theta(W^3)$ for each next level.

Proposition 7.13 can be proved by just a sequence of calculations.

Proof of Proposition 7.13. Suppose all vertices in $B_{\prec j^* \imath^*}$ and at most $\overline{k}W^2$ vertices elsewhere are infected after a certain cascade iteration t. We will first show that less than δW^2 not-seeded vertices can be infected in each bundle $B_{i j^* \imath^*}$ for $i = 1, \ldots, n$ in the next cascade iteration $t+1$. Specifically, we will show this separately for (i) the

2 bundles $B_{nj^*\imath^*}$ and $B_{(n-1)j^*\imath^*}$, (ii) the $\overline{k}$ bundles $B_{\imath^*j^*\imath^*}, B_{(\imath^*+1)j^*\imath^*}, \ldots, B_{(\imath^*+\overline{k}-1)j^*\imath^*}$, and (iii) the remaining $n-2-\overline{k}$ bundles. Then, we will show the same claim for later iterations.

(i) For each vertex in the bundle $B_{nj^*\imath^*}$, by requirement 1 of Definition 7.9, the number of infected neighbors among the vertices in $B_{\prec j^*\imath^*}$ is less than

$$\underbrace{((n-2)(j^*-1)+\imath^*-1)W^3}_{\text{from the } n\text{-th clique}}+\underbrace{(n-1)\cdot((n-2)(j^*-1)+\imath^*-1)\cdot(1+\delta)W^2}_{\text{from the other } n-1 \text{ cliques}}<\omega_{j^*\imath^*}+\Delta W^2. \tag{7.4}$$

For each vertex in the bundle $B_{nj^*\imath^*}$, we have already counted the number of infected neighbors in $B_{\prec j^*\imath^*}$. Next, we consider the infected neighbors in $V \setminus B_{\prec j^*\imath^*}$. There are at most $\overline{k}W^2$ of them by our assumption, and they are the seeds $S = \bigcup_{i=1}^{n} S_n$.

The number of infected neighbors among seed set S_n contributes at most $k_n < (1-9\Delta)W^2$, as we have assumed the n-th clique is not activated. Summing up this and (7.4), the total number of infected neighbors in $B_{\prec j^*\imath^*} \cup S_n$ is at most $\omega_{j^*\imath^*} + (1-8\Delta)W^2$. Since by our construction $r_{nj^*\imath^*} = \omega_{j^*\imath^*} + (1-\Delta)W^2 + 0$, to have δW^2 not-seeded vertices infected, the number of edges between each of these δW^2 vertices and $\bigcup_{i=1}^{n-1} S_i$ should be more than

$$7\Delta W^2, 7\Delta W^2-1, 7\Delta W^2-2, \ldots, 7\Delta W^2-\delta W^2+1$$

respectively. This requires a total of

$$\sum_{t=0}^{\delta W^2-1}(7\Delta W^2-t) > \delta W^2(7\Delta W^2-\delta W^2+1) > W^{3.6}$$

edges, where the last inequality is based on the fact $\delta W^2 = \frac{1}{10\overline{k}}m^{19}n^{18} \gg W^{1.6}$. Since $\sum_{i=1}^{n-1} k_i < (\overline{k}+1)W^2$, this is a contradiction to requirement 2 of Definition 7.9.

For exactly the same reason, we can only have less than δW^2 not-seeded vertices infected in the bundle $B_{(n-1)j^*\imath^*}$, as $r_{(n-1)j^*\imath^*} = r_{nj^*\imath^*}$.

(ii) Next, we consider these $\overline{k}$ bundles: $B_{\imath^*j^*\imath^*}, B_{(\imath^*+1)j^*\imath^*}, \ldots, B_{(\imath^*+\overline{k}-1)j^*\imath^*}$, whose corresponding cliques $\imath^*, \imath^*+1, \ldots, \imath^*+\overline{k}-1$ are not activated by our assumption. Based on our construction, the vertices in these bundles have thresholds

$$\omega_{j^*\imath^*}+(1-\Delta)W^2+1W^2, \omega_{j^*\imath^*}+(1-\Delta)W^2+2W^2, \ldots, \omega_{j^*\imath^*}+(1-\Delta)W^2+\overline{k}W^2$$

respectively, which are all more than $r_{nj^*\imath^*}$. By the same arguments, we can show that having δW^2 not-seeded vertices infected in any of these bundles requires even more edges, which contradicts requirement 2 of Definition 7.9.

(iii) For each of the remaining $n-2-\overline{k}$ bundles $B_{ij^*\imath^*}$ with $i \neq \imath, \imath+1, \ldots, \imath+\overline{k}-1, n-1, n$, although the corresponding i-th clique may be activated, the threshold $r_{ij^*\imath^*}$ is at least $\omega_{j^*\imath^*} + (1-\Delta)W^2 + (\overline{k}+1)W^2$. The number of seeds chosen in the i-th clique k_i cannot offset the term $(\overline{k}+1)W^2$. Therefore, applying the same arguments shows us that less than δW^2 not-seeded vertices can be infected in each of these bundles.

We have shown that less than δW^2 not-seeded vertices can be infected in each bundle $B_{ij^*\imath^*}$ in iteration $t+1$. To show this claim for future iterations, assume for the sake of contradiction that 1) at iteration $t^* > t+1$, less than δW^2 not-seeded vertices are infected in each bundle $B_{ij^*\imath^*}$, and 2) at iteration t^*+1, for certain i^* we have at least δW^2 not-seeded vertices infected in the bundle $B_{i^*j^*\imath^*}$. Denote by D_{-i^*} the set of those vertices outside the i-th clique which are infected during the iterations $t+1, t+2, \ldots, t^*$, and D_{i^*} be the set of those vertices in the i-th clique which are infected during the iterations $t+1, t+2, \ldots, t^*, t^*+1$. Following the same arguments, for some δW^2 vertices from D_{i^*}, the number of edges between each of these δW^2 vertices and $D_{-i^*} \cup S$ should be more than

$$7\Delta W^2, 7\Delta W^2 - 1, 7\Delta W^2 - 2, \ldots, 7\Delta W^2 - \delta W^2 + 1$$

respectively, whose summation is more than $W^{3.6}$. On the other hand, since $|D_{-i^*}| < (n-1)\cdot \delta W^2 < W^2$, we have $|D_{-i^*} \cup S| < (1+\overline{k})W^2$, which again contradicts to requirement 2 of Definition 7.9. Therefore, we conclude Proposition 7.13. □

As we have remarked that Proposition 7.13 implies Proposition 7.12, we conclude the second part of Lemma 7.11. □

Finally, by making M sufficiently large, both Theorem 7.7 and Theorem 7.8 follow from Lemma 7.10 and Lemma 7.11.

7.5 Hierarchical Blockmodel with One-Way Influence

In this section, we consider a variant to the hierarchical blockmodel in which the influence between any two vertex-blocks can only be "one-way". To each node in the hierarchy tree, a *sign* is assigned deciding the directions of the edges between the two vertex-blocks associated to its two children. For example, let t be a node in the hierarchy tree, and t_L, t_R be its left child and right child respectively. If t has a positive sign, then all edges between $V(t_L)$ and $V(t_R)$ are from $V(t_L)$ to $V(t_R)$; otherwise, these edges are from $V(t_R)$ to $V(t_L)$. In this manner, the influence between $V(t_L)$ and $V(t_R)$ is one-way.

In InfMax, the seed-picker needs to decide not only the choice of those k seeds, but also the sign at each tree node. That is, the algorithm to InfMax problem should also output the optimal directions of influence between each pair of vertex-blocks.

Our algorithm also works in the more restrictive, but, perhaps, more practical setting where the signs for all tree nodes are fixed as input and the seed-picker only needs to decide the choice of k seeds. The directed influence between two communities may be observed in our real life for multiple reasons. In some scenarios (e.g., Twitter), the network itself is directed. Status differences between members of different communities could create a uniform direction of influence. Another reason of directed influence may be government regulations. For example, in the cellphone market, many Chinese users adopt iPhone products due to the influence of American users, while Huawei cellphones, adopted by many Chinese users, are banned in the United States of America.

7.5.1 A Dynamic Programming Algorithm

We present a dynamic-programming-based algorithm for InfMax for this variant of the hierarchical blockmodel, when the thresholds of the vertices are deterministic. Our algorithm makes use of the following observation: *for a tree node t, the influence from the infected vertices in the vertex-block $V(t)$ to each vertex in $V \setminus V(t)$ only depends on the* ***number*** *of infected vertices in $V(t)$.* This is formally described in Definition 7.14 and Lemma 7.15 below.

Definition 7.14. Given a set $I \subseteq V$ of infected vertices and a vertex $v \in V \setminus I$, the *influence* from I to v is defined by $\sum_{u\in I} w(u, v)$, where $w(u, v)$ is the weight of the edge (u, v) which is the weight of the deepest node $t \in V_T$ such that $V(t)$ contains both u and v.

By our definition, if the influence from the set of all infected vertices to an uninfected vertex v exceeds r_v, v will be infected.

Lemma 7.15. *Consider an arbitrary node $t \in V_T$. The influence from a set of infected vertices $I_1 \subseteq V(t)$ in $V(t)$ to a vertex $u \in V \setminus V(t)$ only depends on $|I_1|$. Moreover, for any $v_1, v_2 \in V(t)$ and an arbitrary set of infected vertices outside $V(t)$, $I_2 \subseteq V \setminus V(t)$, the influences from I_2 to v_1 and v_2 are the same.*

Proof. For any $v_1, v_2 \in V(t)$ and $u \in V \setminus V(t)$, let t_{v_1}, t_{v_2}, t_u be the leaves such that $v_1 \in V(t_{v_1})$, $v_2 \in V(t_{v_2})$ and $u \in V(t_u)$. The least common ancestor of t_{v_1} and t_u is the same as the least common ancestor of t_{v_2} and t_u, which is the least common ancestor of t and t_u. This implies that the edges (v_1, u) and (v_2, u) have the same weight, and the lemma follows easily from this observation. □

For each tree node $t \in V_T$, each $i = 1, \ldots, k$, and each $\nu = 0, 1, \ldots, |V|$, define $H[t, i, \nu]$ be the smallest positive real number γ satisfying the following:

- given that the threshold of each vertex is updated to $r_v \leftarrow r_v - \gamma$, where we assume the vertex with $r_v - \gamma \leq 0$ is infected immediately, we can choose i seeds in $V(t)$ such that at least ν vertices in $V(t)$ will be infected (due to the influence of these i seeds).

Intuitively, this means we can infect ν vertices by i seeds, given that the influence from infected vertices outside $V(t)$ is $H[t, i, \nu]$. Correspondingly, let $\Sigma[t, i, \nu] \subseteq V(t)$ store the seeding strategy that allocate i seeds in $V(t)$ such that, given that the influence from certain set of infected vertices in $V \setminus V(t)$ to each vertex in $V(t)$ is $H[t, i, \nu]$, those i seeds infect at least ν vertices in $V(t)$.

If t is a leaf, the subgraph induced by $V(t)$ is a clique in which all the $|V(t)|(|V(t)|-1)$ edges have the equal weight. Obviously, the optimal strategy is to place the i seeds on those vertices with the highest thresholds. We propose Algorithm 7.1 to calculate $\Sigma[t, i, \nu]$ and $H[t, i, \nu]$ for each leaf t.

```
Input: vertex set $V(t)$, weight of each edge $w(t)$, threshold set $\{r_v\}_{v \in V(t)}$, integers
       $i, \nu$
Output: $\Sigma[t, i, \nu]$ and $H[t, i, \nu]$ for leaf $t$
set $\Sigma[t, i, v]$ be the $i$ vertices in $V(t)$ having the highest thresholds (set $\Sigma[t, i, v] = V(t)$
  if $i \geq |V(t)|$)
for each vertex $v \in V(t)$ do
    update $r_v \leftarrow r_v - i \cdot w(t)$
end
if $\nu \leq |\{r_v : r_v \leq 0\}| + i$ then
    set $H[t, i, \nu] = 0$
else
    set $H[t, i, \nu]$ be the $(\nu - i)$-th smallest threshold in $\{r_v\}_{v \in V(t)}$
end
return $\Sigma[t, i, \nu]$ and $H[t, i, \nu]$
```

Algorithm 7.1: Initialization for a Leaf t

If t is not a leaf, we aim to find a recurrence relation between $H[t,i,\nu]$ and $H[t_L,i_L,\nu_L], H[t_R,i_R,\nu_R]$. Suppose the sign of t is positive, and there are ν_L infected vertices in $V(t_L)$. Their influence to $V(t_R)$ is $\nu_L \cdot w(t)$ where $w(t)$ is the weight of t reflecting the weight of all edges from $V(t_L)$ to $V(t_R)$. We have a similar observation in the case that the sign of t is negative.

By considering all decompositions $i = i_L + i_R$ and $\nu = \nu_L + \nu_R$, if the sign of t is positive, we have

$$H^+[t,i,\nu] = \min_{i_L=0,\ldots,i;\quad \nu_L=0,\ldots,\nu} \Big\{ \max\big(H[t_L,i_L,\nu_L], H[t_R,i-i_L,\nu-\nu_L] - \nu_L \cdot w(t)\big)\Big\}; \tag{7.5}$$

if the sign of t is negative, we have

$$H^-[t,i,\nu] = \min_{i_R=0,\ldots,i;\quad \nu_R=0,\ldots,\nu} \Big\{ \max\big(H[t_L,i-i_R,\nu-\nu_R] - \nu_R \cdot w(t), H[t_R,i_R,\nu_R]\big)\Big\}, \tag{7.6}$$

where we set $H[t,i,\nu] = \infty$ if $\nu > |V(t)|$. Finally, we decide the sign of t:

$$H[t,i,\nu] = \min\big(H^+[t,i,\nu], H^-[t,i,\nu]\big). \tag{7.7}$$

The recurrence between $\Sigma[t,i,v]$ and $\Sigma[t_L,i_L,\nu_L], \Sigma[t_R,i_R,\nu_R]$ can be obtained in a natural way. The sign of t, $\text{sign}(t) \in \{+,-\}$, is defined naturally by (7.7). If $\text{sign}(t) = +$, we have $\Sigma[t,i,\nu] = \Sigma[t_L,i_L^*,\nu_L^*] \cup \Sigma[t_R,i-i_L^*,\nu-\nu_L^*]$, where (i_L^*,ν_L^*) is the minimizer for (7.5); if $\text{sign}(t) = -$, we have $\Sigma[t,i,\nu] = \Sigma[t_L,i-i_R^*,\nu-\nu_R^*] \cup \Sigma[t_R,i_R^*,\nu_R^*]$, where (i_R^*,ν_R^*) is the minimizer for (7.6).

Define the *height* of $t \in V_T$ be the length of the path to t's deepest descendant. The following Algorithm 7.2 solves InfMax for the hierarchical blockmodel with one-way influence. It is straightforward to check that Algorithm 7.2 runs in time $O(N^3k^2)$.

Remark 7.16. Algorithm 7.2 can be easily adapted to the variant of the InfMax problem where each sign(t) is fixed as input (instead of being a part of the output). Instead of computing both $H^+[t,i,\nu]$ and $H^-[t,i,\nu]$, and setting $H[t,i,\nu] = \min\big(H^+[t,i,\nu], H^-[t,i,\nu]\big)$, we only need to have $H[t,i,\nu] = H^{\text{sign}(t)}[t,i,\nu]$. Corresponding, we have either $\Sigma[t,i,\nu] = \Sigma[t_L,i_L^*,\nu_L^*] \cup \Sigma[t_R,i-i_L^*,\nu-\nu_L^*]$ or $\Sigma[t,i,\nu] = \Sigma[t_L,i-i_R^*,\nu-\nu_R^*] \cup \Sigma[t_R,i_R^*,\nu_R^*]$ depending on sign(t) which is now given by the input.

```
Input: hierarchical blockmodel $G = (V, T)$, threshold set $\{r_v\}_{v \in V}$, integer $k$
Output: 1) $S \subseteq V$ such that $|S| = k$ and $S$ maximizes $\sigma(S)$, and 2) the sign of each
        internal node $t$: $\text{sign}(t)$
for each height $i = 0, 1, \ldots, h$ do
    for each node $t \in V_T$ with height $i$ do
        if $t$ is a leaf then
            initialize $\Sigma[t, i, \nu]$ and $H[t, i, \nu]$ by Algorithm 7.1 for all $i = 0, 1, \ldots, k$ and
             $\nu = 0, 1 \ldots, N$
        else
            for for each $i = 0, 1, \ldots, k$ and $\nu = 0, 1 \ldots, N$ do
                $H^+[t, i, \nu] = \min_{i_L = 0, \ldots, i; \nu_L = 0, \ldots, \nu} \Big\{ \max\big(H[t_L, i_L, \nu_L], H[t_R, i - i_L, \nu - \nu_L] - \nu_L \cdot w(t)\big) \Big\}$
                $H^-[t, i, \nu] = \min_{i_R = 0, \ldots, i; \nu_R = 0, \ldots, \nu} \Big\{ \max\big(H[t_L, i - i_R, \nu - \nu_R] - \nu_R \cdot w(t), H[t_R, i_R, \nu_R]\big) \Big\}$
                $H[t, i, \nu] = \min\big(H^+[t, i, \nu], H^-[t, i, \nu]\big)$
                set $\text{sign}(t) = \operatorname{argmin}_{s \in \{+, -\}} H^s[t, i, \nu]$
                if $\text{sign}(t) = +$ then
                    set $\Sigma[t, i, \nu] = \Sigma[t_L, i_L^*, \nu_L^*] \cup \Sigma[t_R, i - i_L^*, \nu - \nu_L^*]$, where $(i_L^*, \nu_L^*)$
                     minimizes $H^+[t, i, \nu]$
                else
                    set $\Sigma[t, i, \nu] = \Sigma[t_L, i - i_R^*, \nu - \nu_R^*] \cup \Sigma[t_R, i_R^*, \nu_R^*]$, where $(i_R^*, \nu_R^*)$
                     minimizes $H^-[t, i, \nu]$
                end
            end
        end
    end
end
set $\nu^*$ be the maximum $\nu$ such that $H[r, k, \nu] = 0$, where $r$ is the root of $T$
return $\Sigma[r, k, \nu^*]$ and $\text{sign}(t)$ for each internal node $t$
```

Algorithm 7.2: Dynamic Programming Algorithm for Hierarchical Blockmodel InfMax with One-Way Influence

7.5.2 Further Discussions

We have seen inapproximability results in Section 7.3 and Section 7.4 for InfMax on the (stochastic) hierarchical blockmodel. Our algorithm in this section reveals the intrinsic reason why these problems are difficult.

In the hard InfMax instances in Figure 7.2 and Figure 7.4, we constructed the hierarchy tree by creating n branches corresponding to the n vertices in VertexCover. In the case the VertexCover instance is a YES instance, the influence of the properly chosen seeds passes through these n branches "back-and-forth" frequently: the infected vertices in branch A_i make vertices in branch A_j infected, while these newly infected vertices in A_j may have backward influence to A_i, and cause more infected vertices in A_i. This bidirectional effect is not considered in Algorithm 7.2, and is exactly why InfMax is hard. On the other hand, when there is no such bidirectional effect, even if the algorithm needs to decide the optimal directions at all internal nodes (with exponentially many choices $2^{\Theta(|V_T|)}$), InfMax becomes easy on the hierarchial blockmodel, as our algorithm in this section suggests.

Angell and Schoenebeck [2] show that a generalization of this algorithm works well empirically. This perhaps indicates that the bidirectional influence is, in the average case, not often so important in realistic settings.

CHAPTER 8

r-Complex Contagion on Graphs with Hierarchical Communities

We have seen strong inapproximability results in both Chapter 6 and Chapter 7, which are regarding InfMax with strong assumptions on diffusion model and network topology respectively. In particular, we have seen in Theorem 7.7 that InfMax with bootstrap percolation and stochastic hierarchical blockmodel is still extremely hard to approximate. In this chapter, we show that, under some further mild technical assumptions, InfMax with r-complex contagion (i.e., bootstrap percolation such that all vertices have the same threshold r) and stochastic hierarchical blockmodel becomes tractable.

When the graph is not exceptionally sparse, in particular, when the weight of the root of the hierarchy tree T is $\omega\left(n^{-(1+1/r)}\right)$, under certain mild assumptions, we prove that the optimal seeding strategy is to put all the seeds in a single community. This matches the intuition that in a nonsubmodular diffusion model placing seeds near each other creates synergy. However, it sharply contrasts with the intuition for submodular diffusion models (e.g., `ICM` and `LTM`) in which nearby seeds tend to erode each others' effects. Our key technique is a novel time-asynchronized coupling of four cascade processes. By this coupling argument, we reveal a supermodular property of r-complex contagion on Erdős-Rényi graphs.

Finally, we show that this observation yields a polynomial time dynamic programming algorithm which outputs optimal seeds if each edge appears with a probability (or the weight of each node in T is) either in $\omega\left(n^{-(1+1/r)}\right)$ or in $o\left(n^{-2}\right)$.

8.1 Our Results

Result 1: We first prove that, for InfMax on the stochastic hierarchical blockmodel with r-complex contagion, under certain mild technical assumptions, the optimal seeding strategy is to put all the seeds in a single community, if, for each vertex-pair (u, v), the probability that the edge (u, v) is included satisfies $p_{uv} = \omega(n^{-(1+1/r)})$. Notice that the assumption $p_{uv} = \omega(n^{-(1+1/r)})$ captures many real life social networks. In fact, it is well-known that an Erdős-Rényi graph $\mathcal{G}(n, p)$ with $p = o(1/n)$ is globally disconnected: with probability $1-o(1)$, the graph consists of a union of tiny connected components, each of which has size $O(\log n)$.

The technical heart of this result is a novel coupling argument in Proposition 8.16. We simultaneously couple four cascade processes to compare two probabilities: 1) the probability of infection spreading throughout an Erdős-Rényi graph after the $(k+1)$-st seed, conditioned on not already being entirely infected after k seeds; 2) the probability of infection spreading throughout the same graph after the $(k+2)$-nd seed, conditioned on not already being entirely infected after $k+1$ seeds. This shows that the marginal rate of infection always goes up, revealing the "supermodular" nature of the r-complex contagion. The supermodular property revealed by Proposition 8.16 is a property for cascade behavior on Erdős-Rényi random graphs in general, so it is also interesting on its own.

Our result is in sharp contrast to Balkanski et al.'s observation. Balkanski et al. [5] studies the stochastic blockmodel with the well-studied submodular diffusion model ICM, and remarks that "when an influential node from a certain community is selected to initiate a cascade, the marginal contribution of adding another node from that same community is small, since the nodes in that community were likely already influenced."

Algorithmic aspects The stochastic hierarchical structure seems optimized for a dynamic programming approach: perform dynamic programming from the bottom to the root in the tree-like community structure. This intuition can be misleading: we have seen in the previous chapter that the $\Omega(n^{1-\epsilon})$ inapproximability results extend to the setting where the networks are stochastic hierarchical blockmodels.

Result 2: However, Result 1 (when the network is reasonably dense, putting all the seeds in a single community is optimal) can naturally be extended to a dynamic programming algorithm. We show that this algorithm is optimal if the probability p_{uv} that each edge appears does not fall into a narrow regime. Interestingly, a heuristic based on dynamic programming works fairly well in practice [2]. Our second result theoretically justifies the success of this approach, at least in the setting of r-complex contagions.

8.2 Preliminaries

In this section, we use K instead of k to denote the number of seeds. We use $\sigma_{r,G}(S)$ to denote the total number of infected vertices at the end of the cascade, and $\sigma_{r,\mathcal{G}}(S) = \mathbb{E}_{G\sim\mathcal{G}}\left[\sigma_{r,G}(S)\right]$ if the graph G is sampled from some distribution $\mathcal{G}$. Notice that the function $\sigma_{r,G}(\cdot)$ is deterministic once the graph G and r are fixed.

8.2.1 Stochastic Hierarchical Blockmodels

We study the *stochastic hierarchical blockmodel* in the last chapter. The definition is slightly rephrased as follows.

Definition 8.1. A *stochastic hierarchical blockmodel* is a distribution $\mathcal{G} = (V, T)$ of unweighted undirected graphs sharing the same vertex set V, and $T = (V_T, E_T, w)$ is a weighted tree T called a *hierarchy tree*. The third parameter is the weight function $w : V_T \to [0, 1]$ satisfying $w(t_1) < w(t_2)$ for any $t_1, t_2 \in V_T$ such that t_1 is an ancestor of t_2. Let $L_T \subseteq V_T$ be the set of leaves in T. Each leaf node $t \in L_T$ corresponds to a subset of vertices $V(t) \subseteq V$, and the $V(t)$ sets partition the vertices in V. In general, if $t \notin L_T$, we denote $V(t) = \bigcup_{t'\in L_T:t' \text{ is an offspring of } t} V(t')$.

The graph $G = (V, E)$ is sampled from $\mathcal{G}$ in the following way. The vertex set V is deterministic. For $u, v \in V$, the edge (u, v) appears in G with probability equal to the weight of the least common ancestor of u and v in T. That is $\Pr((u, v) \in E) = \max_{t:u,v\in V(t)} w(t)$.

In the rest of this chapter, we use the words "tree node" and "vertex" to refer to the vertices in V_T and V respectively. In Definition 8.1, the tree node $t \in V_T$ corresponds to community $V(t) \subseteq V$ in the social network. Moreover, if t is not a leaf and $t_1, t_2, \ldots$ are the children of t in V_T, then $V(t_1), V(t_2), \ldots$ partition $V(t)$ into sub-communities. Thus, our assumption that for any $t_1, t_2 \in V_T$ where t_1 is an ancestor of t_2 we have $w(t_1) < w(t_2)$ implies that the relation between two vertices is stronger if they are in a same sub-community in a lower level, which is natural.

To capture the scenario where the advertiser has the information on the high-level community structure but lacks the knowledge of the detailed connections inside the communities, when defining the influence maximization problem as an optimization problem, we would like to include T as a part of input, but not G. Rather than choosing which specific vertices are seeds, the seed-picker decides the number of seeds on each leaf and the graph $G \sim \mathcal{G}(n, T)$ is realized after seeds are chosen. Moreover, we are interested in large social networks with $n \to \infty$, so we would like that a single encoding of T is compatible with varying n. To enable this feature, we consider the following variant of the stochastic hierarchical block model.

Definition 8.2. A *succinct stochastic hierarchical blockmodel* is a distribution $\mathcal{G}(n,T)$ of unweighted undirected graphs sharing the same vertex set V with $|V| = n$, where n is an integer which is assumed to be extremely large. The hierarchy tree $T = (V_T, E_T, w, v)$ is the same as it is in Definition 8.1, except for the followings.

1. Instead of mapping a tree node t to a weight in $[0,1]$, the weight function $w : V_T \to \mathcal{F}$ maps each tree node to a function $f \in \mathcal{F} = \{f \mid f : \mathbb{Z}^+ \to [0,1]\}$ which maps an integer (denoting the number of vertices in the network) to a weight in $[0,1]$. The weight of t is then defined by $(w(t))(n)$. We assume $\mathcal{F}$ is the space of all functions that can be succinctly encoded.

2. The fourth parameter $v : V_T \to (0,1]$ maps each tree node $t \in V_T$ to the fraction of vertices in $V(t)$. That is: $v(t) = |V(t)|/n$. Naturally, we have $\sum_{t\in L_T} v(t) = 1$ and $\sum_{t':t' \text{ is a child of } t} v(t') = v(t)$.

We assume throughout that $\mathcal{G}(n,T)$ has the following properties.

Large communities For tree node $t \in V_T$, because $v(t)$ does not depend on n, $|V(t)| = v(t)n = \Theta(n)$. In particular, $|V(t)|$ goes to infinity as n does.

Proper separation $w(t_1) = o\left(w(t_2)\right)$ for any $t_1, t_2 \in V_T$ such that t_1 is an ancestor of t_2. That is, the connection between sub-community t_2 is asymptotically (with respect to n) denser than its super-community t_1.

Our definitions of w and v are designed so that we can fix a hierarchy tree $T = (V_T, E_T, w, v)$ and naturally define $\mathcal{G}(n,T)$ for any n. As we will see in the next subsection, this allows us to take T as input and then allow $n \to \infty$ when considering InfMax (to be defined soon). This enables us to consider graphs having exponentially many vertices.

Finally, we define the *density* of a tree node.

Definition 8.3. Given a hierarchy tree $T = (V_T, E_T, w, v)$ and a tree node $t \in V_T$, the *density* of the tree node is $\rho(t) = w(t) \cdot (v(t)n)^{1/r}$.

8.2.2 Succinct Stochastic Hierarchical Blockmodel

We study the r-complex contagion on the succinct stochastic hierarchical blockmodel. Roughly speaking, given hierarchy tree T and an integer K, we want to choose K seeds which maximize the expected total number of infected vertices, where the expectation is taken over the graph sampling $G \sim \mathcal{G}(n,T)$ as $n \to \infty$.

Definition 8.4. The *influence maximization problem* InfMax is an optimization problem which takes as inputs an integer r, a hierarchy tree $T = (V_T, E_T, w, v)$ as in Definition 8.2, and an integer K, and outputs $\boldsymbol{k} \in \mathbb{N}_{\geq 0}^{|L_T|}$—an allocation of K seeds into the leaves L_T with $\sum_{t \in L_T} k_t = K$ that maximizes

$$\Sigma_{r,T}(\boldsymbol{k}) := \lim_{n \to \infty} \frac{\mathbb{E}_{G \sim \mathcal{G}(n,T)}\left[\sigma_{r,G}(S_{\boldsymbol{k}})\right]}{n},[1]$$

the expected fraction of infected vertices in $\mathcal{G}(n,T)$ with the seeding strategy defined by $\boldsymbol{k}$, where $S_{\boldsymbol{k}}$ denotes the seed set in G generated according to $\boldsymbol{k}$.

Before we move on, the following remark is very important throughout the paper.

Remark 8.5. In Definition 8.4, n is not part of the inputs to the InfMax instance. Instead, the tree T is given as an input to the instance, and we take $n \to \infty$ to compute $\Sigma_{r,T}(\boldsymbol{k})$ *after* the seed allocation is determined. Therefore, asymptotically, all the input parameters to the instance, including K, r and the encoding size of T, are *constants* with respect to n. Thus, there are two different asymptotic scopes in this paper: *the asymptotic scope with respect to the input size* and *the asymptotic scope with respect to n*. Naturally, when we are analyzing the running time of an InfMax algorithm, we should use the asymptotic scope with respect to the input size, not of n. On the other hand, when we are analyzing the number of infected vertices after the cascade, we should use the asymptotic scope with respect to n.

In this paper, we use $O_I(\cdot), \Omega_I(\cdot), \Theta_I(\cdot), o_I(\cdot), \omega_I(\cdot)$ to refer to the asymptotic scope with respect to the input size, and we use $O(\cdot), \Omega(\cdot), \Theta(\cdot), o(\cdot), \omega(\cdot)$ to refer to the asymptotic scope with respect to n. For example, with respect to n we always have $r = \Theta(1)$, $K = \Theta(1)$ and $|V_T| = \Theta(1)$.

Lastly, we have assumed that $r \geq 2$, so that the contagion is nonsubmodular. When $r = 1$, the model becomes a special case of `ICM`. As mentioned, for submodular InfMax, a simple greedy algorithm is known to achieve a $(1 - 1/e)$-approximation to the optimal influence [44, 45, 58].

[1] We divide the expected number of infected vertices by n to avoid an infinite limit. However, as a result, our analysis naturally ignores lower order terms.

8.2.3 Complex Contagion on Erdős-Rényi Graphs

In this section, we consider the r-complex contagion on the Erdős-Rényi random graph $\mathcal{G}(n,p)$. We review some results from [42] which are used in our paper.

Definition 8.6. The *Erdős-Rényi random graph* $\mathcal{G}(n,p)$ is a distribution of graphs with the same vertex set V with $|V| = n$ and we include an edge $(u,v) \in E$ with probability p independently for each pair of vertices u, v.

The InfMax problem in Definition 8.4 on $\mathcal{G}(n,p)$ is trivial, as there is only one possible allocation of the K seeds: allocate all the seeds to the single leaf node of T, which is the root. Therefore, $\sigma_{r,T}(\cdot)$ in Definition 8.4 depends only on the *number* of seeds $K = |\boldsymbol{k}|$, not on the seed allocation $\boldsymbol{k}$ itself. In this section, we slightly abuse the notation σ such that it is a function mapping an *integer* to $\mathbb{R}_{\geq 0}$ (rather than mapping *an allocation of K seeds* to $\mathbb{R}_{\geq 0}$ as it is in Definition 8.4), and let $\sigma_{r,\mathcal{G}(n,p)}(k)$ be the expected number of infected vertices after the cascade given k seeds. Correspondingly, let $\sigma_{r,G}(k)$ be the actual number of infected vertices after the graph G is sampled from $\mathcal{G}(n,p)$.

Theorem 8.7 (A special case of Theorem 3.1 in [42])**.** *Suppose $r \geq 2$, $p = o(n^{-1/r})$ and $p = \omega(n^{-1})$. We have*

1. *if k is a constant, then $\sigma_{r,\mathcal{G}(n,p)}(k) \leq 2k$ with probability $1 - o(1)$;*
2. *if $k = \omega\left((1/np^r)^{1/(r-1)}\right)$, then $\sigma_{r,\mathcal{G}(n,p)}(k) = n - o(n)$ with probability $1 - o(1)$.*

Theorem 8.8 (Theorem 5.8 in [42])**.** *If $r \geq 2$, $p = \omega(n^{-1/r})$ and $k \geq r$, then $\Pr_{G \sim \mathcal{G}(n,p)}\left[\sigma_{r,G}(k) = n\right] = 1 - o(1)$.*

When $p = \Theta(n^{-1/r})$, the probability that k seeds infect all the n vertices is positive, but bounded away from 1. We use $\mathrm{Po}(\lambda)$ to denote the Poisson distribution with mean λ.

Theorem 8.9 (Theorem 5.6 and Remark 5.7 in [42])**.** *If $r \geq 2$, $p = cn^{-1/r} + o(n^{-1/r})$ for some constant $c > 0$, and $k \geq r$ is a constant, then*

$$\lim_{n\to\infty} \Pr\left(\sigma_{r,\mathcal{G}(n,p)}(k) = n\right) = \zeta(k,c),$$

for some $\zeta(k,c) \in (0,1)$. Furthermore, there exist numbers $\zeta(k,c,\ell) > 0$ for $\ell \geq k$ such that

$$\lim_{n\to\infty} \Pr\left(\sigma_{r,\mathcal{G}(n,p)}(k) = \ell\right) = \zeta(k,c,\ell)$$

for each $\ell \geq k$, and $\zeta(k,c) + \sum_{\ell=k}^{\infty} \zeta(k,c,\ell) = 1$.

Moreover, the numbers $\zeta(k,c,\ell)$'s and $\zeta(k,c)$ can be expressed as the hitting probabilities of the following inhomogeneous random walk. Let $\xi_\ell \sim \mathrm{Po}\left(\binom{\ell-1}{r-1}c^r\right)$, $\ell \geq 1$ be independent, and let $\tilde{S}_\ell := \sum_{j=1}^{\ell}(\xi_j - 1)$ and $\tilde{T} := \min\{\ell : k + \tilde{S}_\ell = 0\} \in \mathbb{N} \cup \{\infty\}$. Then

$$\zeta(k,c) = \Pr\left(\tilde{T} = \infty\right) = \Pr\left(k + \tilde{S}_\ell \geq 1 \text{ for all } \ell \geq 1\right) \tag{8.1}$$

and $\zeta(k,c,\ell) = \Pr(\tilde{T} = \ell)$.

We have the following corollary for Theorem 8.9, saying that when $p = \Theta(n^{-1/r})$, if not all vertices are infected, then the number of infected vertices is constant. As a consequence, if the cascade spreads to more than constantly many vertices, then all vertices will be infected.

Corollary 8.10 (Lemma 11.4 in [42])**.** *If $r \geq 2$, $p = cn^{-1/r} + o(n^{-1/r})$ for some constant $c > 0$, and $k \geq r$, then*

$$\lim_{n\to\infty} \Pr\left(\phi(n) \leq \sigma_{r,\mathcal{G}(n,p)}(k) < n\right) = 0$$

for any function $\phi : \mathbb{Z}^+ \to \mathbb{R}^+$ such that $\lim_{n\to\infty} \phi(n) = \infty$.

8.3 Our Main Result

Our main result is the following theorem, which states that the optimal seeding strategy is to put all the seeds in a community with the highest density, when the root has a weight in $\omega(1/n^{1+1/r})$.

Theorem 8.11. *Consider the* InfMax *problem with $r \geq 2$, $T = (V_T, E_T, w, v)$, $K > 0$ and the weight of the root node satisfying $w(root) = \omega(1/n^{1+1/r})$. Let $t^* \in \operatorname{argmax}_{t \in L_T} \rho(t)$ and $\boldsymbol{k}^*$ be the seeding strategy that puts all the K seeds on t^*. Then $\boldsymbol{k}^* \in \operatorname{argmax}_{\boldsymbol{k}} \Sigma_{r,T}(\boldsymbol{k})$.*

Notice that the assumption $w(\text{root}) = \omega(1/n^{1+1/r})$ captures many real life social networks. In fact, it is well-known that an Erdős-Rényi graph $\mathcal{G}(n,p)$ with $p = o(1/n)$ is globally disconnected: with probability $1 - o(1)$, the graph consists of a union of tiny connected components, each of which has size $O(\log n)$.

The remaining part of this section is dedicated to proving Theorem 8.11. We assume $w(\text{root}) = \omega(1/n^{1+1/r})$ in this section from now on. It is worth noting that, in many parts of this proof, and also in the proof of Theorem 8.23, we have used the fact that an infection of $o(n)$ vertices contributes 0 to the objective $\Sigma_{r,T}(\boldsymbol{k})$, as we have taken the limit $n \to \infty$ and divided the expected number of infections by n in Definition 8.4.

Definition 8.12. Given $T = (V_T, E_T, w, v)$, a tree node $t \in V_T$ is *supercritical* if $w(t) = \omega(1/n^{1/r})$, is *critical* if $w(t) = \Theta(1/n^{1/r})$, and is *subcritical* if $w(t) = o(1/n^{1/r})$.

From the results in Sect. 8.2.3, if we allocate $k \geq r$ seeds on a supercritical leaf $t \in L_T$, then with probability $1 - o(1)$ all vertices in $V(t)$ will be infected; if we allocate k seeds on a subcritical leaf $t \in L_T$, at most a negligible amount of vertices, $2k = \Theta(1)$, will be infected; if we allocate $k \geq r$ seeds on a critical leaf $t \in L_T$, the number of infected vertices in $V(t)$ follows Theorem 8.9.

We say a tree node $t \in V_T$ is *activated* in a cascade process if the number of infected vertices in $V(t)$ is $v(t)n - o(n)$, i.e., almost all vertices in $V(t)$ are infected. Given a seeding strategy $\boldsymbol{k}$, let $P_{\boldsymbol{k}}$ be the probability that at least one tree node is activated when $n \to \infty$. Notice that this is equivalent to at least one leaf being activated. The proof of Theorem 8.11 consists of two parts. We will first show that, $P_{\boldsymbol{k}}$ completely determines $\Sigma_{r,T}(\boldsymbol{k})$ (Lemma 8.13). Secondly, we show that placing all the seeds on a single leaf with the maximum density will maximize $P_{\boldsymbol{k}}$ (Lemma 8.14).

Lemma 8.13. *Given any two seeding strategies $\boldsymbol{k}_1, \boldsymbol{k}_2$, if $P_{\boldsymbol{k}_1} \leq P_{\boldsymbol{k}_2}$, then $\Sigma_{r,T}(\boldsymbol{k}_1) \leq \Sigma_{r,T}(\boldsymbol{k}_2)$.*

Lemma 8.14. *Let $\boldsymbol{k}$ be the seeding strategy that allocates all the K seeds on a leaf $t^* \in \underset{t \in L_T}{\operatorname{argmax}}(\rho(t))$. Then $\boldsymbol{k}$ maximizes $P_{\boldsymbol{k}}$.*

Lemma 8.13 and Lemma 8.14 imply Theorem 8.11.

8.3.1 Proof Sketch of Lemma 8.13

We sketch the proof. The full proof is in the appendix.

Proof (sketch). Let E be the event that at least one leaf (or tree node) is activated at the end of the cascade.

In the case that E does not happen, we show there are only $o(n)$ infected vertices in V, regardless of the seeding strategy. First, Theorem 8.8 and Corollary 8.10 imply that the number of infected vertices in a critical or supercritical leaf t can only be either a constant or $v(t)n$. Because E does not happen, it must be the former. Second, Theorem 8.7 indicates that a subcritical leaf with a constant number of seeds will not have $\omega(1)$ infected vertices. As there are only a constant number of infections in each of the critical or supercritical leaves, and we have only a constant number $K = \Theta(1)$ of seeds, this implies that there are also only a constant number of infections in subcritical leaves.

If E happens, we can show that the expected total number of infected vertices does not vary significantly for different seeding strategies. Consider two leaves t_1, t_2 with their least common ancestor t. If a leaf t_1 is activated, we find a lower bound of the probability that a vertex $v \in V(t_2)$ is infected due to the influence of $V(t_1)$. We assume without loss of generality that $w(t) = o(1/n)$, which can only further reduce v's infection probability from the case when $w(t)$ is in $\Omega(1/n)$. With this assumption, the probability that $v \in V(t_2)$ is infected by the vertices in $V(t_1)$ is

$$\binom{v(t_1)n}{r} w(t)^r (1 - w(t))^{v(t_1)n-r} = \omega\left(n^r \left(\frac{1}{n^{1+\frac{1}{r}}}\right)^r \cdot 1\right) = \omega\left(\frac{1}{n}\right),$$

where the first equality uses the assumption $w(t) = o(1/n)$ so that $(1-w(t))^{v(t_1)n-r} = \Omega(1)$. Thus, there are $\omega(1/n) \cdot \Theta(n) = \omega(1)$ infected vertices in $V(t_2)$. Theorem 8.8 and Corollary 8.10 show that t_2 will be activated if t_2 is critical or supercritical. Therefore, when E happens, all the critical and supercritical leaves will be activated. As for subcritical leaves, the number of infected vertices may vary, but Theorem 8.7 intuitively suggests that adding a constant number of seeds is insignificant (we handle this rigorously in the full proof). Therefore, the expected total number of infections equals to the number of vertices in all critical and supercritical leaves, plus the expected number of infected vertices in subcritical leaves which does not significantly depend on the seeding strategy $\boldsymbol{k}$.

In conclusion, the number of infected vertices only significantly depends on whether or not E happens. In particular, we have a fixed fraction of infected vertices whose size does not depend on $\boldsymbol{k}$ if E happens, and a negligible number of infected vertices if E does not happen. Therefore, $P_{\boldsymbol{k}}$ characterizes $\Sigma_{r,T}(\boldsymbol{k})$, and a larger $P_{\boldsymbol{k}}$ implies a larger $\Sigma_{r,T}(\boldsymbol{k})$. □

8.3.2 Proof of Lemma 8.14

We first handle some corner cases. If $K < r$, then the cascade will not even start, and any seeding strategy is considered optimal. If T contains a supercritical leaf, the leaf with the highest density is also supercritical. Putting all the $K \geq r$ seeds in this leaf, by Theorem 8.8, will activate the leaf with probability $1 - o(1)$. Therefore, this strategy makes $P_{\boldsymbol{k}} = 1$, which is clearly optimal. In the remaining part of this subsection, we shall only consider $K \geq r$ and all the leaves are either critical or subcritical. Notice that, by the proper separation assumption, all internal tree nodes of T are subcritical.

We split the cascade process into two phases. In Phase I, we restrict the cascade within the leaf blocks ($V(t)$ where $t \in L_T$), and temporarily assume there are no edges between two different leaf blocks (similar to if $w(t) = 0$ for all $t \notin L_T$). After Phase I, Phase II consists of the remaining cascade process.

Proposition 8.15 shows that maximizing $P_{\boldsymbol{k}}$ is equivalent to maximizing the probability that a leaf is activated in Phase I. Therefore, we can treat T such that all the leaves, each of which corresponds to a $\mathcal{G}(n, p)$ random graph, are isolated.

Proposition 8.15. *If no leaf is activated after Phase I, then with probability* $1 - o(1)$ *no vertex will be infected in Phase II, i.e., the cascade will end after Phase I.*

We sketch the proof here, and the full proof is available in the appendix.

Proof (sketch). Consider any critical leaf t and an arbitrary vertex $v \in V(t)$ that is not infected after Phase I. Let K_{in} be the number of infected vertices in $V(t)$ after Phase I, and K_{out} be the number of infected vertices in $V \setminus V(t)$. If no leaf is activated after Phase I, Theorem 8.7 and Corollary 8.10 suggest that $K_{in} = O(1)$ and $K_{out} = O(1)$. The probability that v is connected to any of the K_{in} infected vertices in $V(t)$ can only be less than $w(t) = \Theta(n^{-1/r})$ conditioning on the cascade inside $V(t)$ does not carry to v, so the probability that v has a infected neighbors in $V(t)$ is $O(n^{-a/r})$. On the other hand, the probability that v has $r - a$ neighbors among the K_{out} outside infected vertices is $o(n^{-(r-a)/r})$. Therefore, the probability that v is infected in the next iteration is $\sum_{a=0}^{r-1} O(n^{-a/r}) \cdot o(n^{-(r-a)/r}) = o(1/n)$, and the expected total number of vertices infected in the next iteration after Phase I is $o(1)$. The proposition follows from the Markov's inequality. □

Since Theorem 8.7 suggests that any constant number of seeds will not activate a subcritical leaf, we should only consider putting seeds in critical leaves. In Proposition 8.16, we show that in a critical leaf t, the probability that the $(i + 1)$-th seed will activate t conditioning on the first i seeds failing to do so is increasing as i increases. Intuitively, Proposition 8.16 reveals a super-modular nature of the r-complex contagion on a critical leaf, making it beneficial to put all seeds together so that the synergy effect is maximized, which intuitively implies Lemma 8.14. The proof of Proposition 8.16 is the most technical result of this paper, we will present it in Sect. 8.4.

Proposition 8.16 (log-concavity of $\lim_{n\to\infty}\Pr(E_k^n)$). *Consider an Erdős-Rényi random graph $\mathcal{G}(n,p)$ with $p = cn^{-1/r} + o(n^{-1/r})$, and assume an arbitrary order on the n vertices. Let E_k^n be the event that seeding the first k vertices does not make all the n vertices infected. We have $\lim_{n\to\infty}\Pr(E_{k+2}^n \mid E_{k+1}^n) < \lim_{n\to\infty}\Pr(E_{k+1}^n \mid E_k^n)$ for any $k \geq r-1$.*

Equipped with Proposition 8.16, to show Lemma 8.14, we show that the seeding strategy that allocates $K_1 > 0$ seeds on a critical leaf t_1 and $K_2 > 0$ seeds on a critical leaf t_2 cannot be optimal. Firstly, it is obvious that both K_1 and K_2 should be at least r, for otherwise those K_1 (K_2) seeds on t_1 (t_2) are simply wasted.

Let E_k^n be the event that the first k seeds on t_1 fail to activate t_1 and F_k^n be the event that the first k seeds on t_2 fail to activate t_2. By Proposition 8.16, we have $\lim_{n\to\infty}\Pr(E_{K_1+1}^n \mid E_{K_1}^n) < \lim_{n\to\infty}\Pr(E_{K_1}^n \mid E_{K_1-1}^n)$ and $\lim_{n\to\infty}\Pr(F_{K_2+1}^n \mid F_{K_2}^n) < \lim_{n\to\infty}\Pr(F_{K_2}^n \mid F_{K_2-1}^n)$, which implies

$$\begin{aligned}&\lim_{n\to\infty}\frac{\Pr(E_{K_1+1}^n)\Pr(F_{K_2-1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}\cdot\frac{\Pr(E_{K_1-1}^n)\Pr(F_{K_2+1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}\\ =&\lim_{n\to\infty}\frac{\Pr(E_{K_1+1}^n \mid E_{K_1}^n)\Pr(F_{K_2+1}^n \mid F_{K_2}^n)}{\Pr(E_{K_1}^n \mid E_{K_1-1}^n)\Pr(F_{K_2}^n \mid F_{K_2-1}^n)} < 1.\end{aligned}$$

Therefore, we have either $\lim_{n\to\infty}\frac{\Pr(E_{K_1+1}^n)\Pr(F_{K_2-1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}$ or $\lim_{n\to\infty}\frac{\Pr(E_{K_1-1}^n)\Pr(F_{K_2+1}^n)}{\Pr(E_{K_1}^n)\Pr(F_{K_2}^n)}$ is less than 1. This means either the strategy putting K_1+1 seeds on t_1 and K_2-1 seeds on t_2, or the strategy putting K_1-1 seeds on t_1 and K_2+1 seeds on t_2 makes it more likely that at least one of t_1 and t_2 is activated. Therefore, the strategy putting K_1 and K_2 seeds on t_1 and t_2 respectively cannot be optimal. This implies an optimal strategy should not allocate seeds on more than one leaf.

Finally, a critical leaf t with $v(t)n$ vertices and weight $w(t)$ can be viewed as an Erdős-Rényi random graph $\mathcal{G}(m,p)$ with $m = v(t)n$ and $p = w(t) = \rho(t)\cdot(v(t)n)^{-1/r} = \rho(t)m^{-1/r}$, where $\rho(t) = \Theta(1)$ when t is critical. Taking $c = \rho(t)$ in Theorem 8.9, we can see that ξ_ℓ has a larger Poisson mean if c is larger, making it more likely that the $\mathcal{G}(m,p)$ is fully infected (to see this more naturally, larger c means larger p if we fix m). Thus, given that we should put all the K seeds in a single leaf, we should put them on a leaf with the highest density. This concludes Lemma 8.14.

8.4 Proof for Proposition 8.16

Since the event E_{k+1}^n implies E_k^n, we have $\Pr(E_{k+1}^n | E_k^n) = \Pr(E_{k+1}^n)/\Pr(E_k^n)$. Therefore, the inequality we are proving is equivalent to

$$\lim_{n\to\infty} \Pr(E_{k+2}^n)/\Pr(E_{k+1}^n) < \lim_{n\to\infty} \Pr(E_{k+1}^n)/\Pr(E_k^n),$$

and it suffices to show that

$$\lim_{n\to\infty} \Pr(E_{k+2}^n) \lim_{n\to\infty} \Pr(E_k^n) < \lim_{n\to\infty} \Pr(E_{k+1}^n) \lim_{n\to\infty} \Pr(E_{k+1}^n). \tag{8.2}$$

Proposition 8.16 shows that the failure probability, $\lim\limits_{n\to\infty} \Pr(E_k^n)$, is logarithmically concave with respect to k.

The remaining part of the proof is split into four parts: In Sect. 8.4.1, we begin by translating Eqn. (8.2) in the language of inhomogeneous random walks. In Sect. 8.4.2, we present a coupling of two inhomogeneous random walks to prove Eqn. (8.2). In Sect. 8.4.3, we prove the validity of the coupling. in Sect. 8.4.4, we finally show the coupling implies Eqn. (8.2).

8.4.1 Inhomogeneous Random Walk Interpretation

We adopt the inhomogeneous random walk interpretation from Theorem 8.9, and view the event E_k^n as the following: the random walk starts at $x = k$; in the i-th iteration, x moves to the left by 1 unit, and moves to the right by $\alpha(i) \sim \mathrm{Po}\left(\binom{i-1}{r-1}c^r\right)$ units; Let $\mathcal{E}_k$ be the event that the random walk reaches $x = 0$. By Theorem 8.9, $\Pr(\mathcal{E}_k) = \lim\limits_{n\to\infty} \Pr(E_k^n)$. Thus, $\lim\limits_{n\to\infty} \Pr(E_{k+2}^n) \lim\limits_{n\to\infty} \Pr(E_k^n) = \Pr(\mathcal{E}_{k+2}) \Pr(\mathcal{E}_k)$. In this proof, we let $\lambda(i) = \binom{i-1}{r-1}c^r$, and in particular, $\lambda(0) = \lambda(1) = \cdots = \lambda(r-1) = 0$. Note that as i increases, the expected movement of the walk increases, and make it harder to reach 0. This observation is important for our proof.

To compute $\Pr(\mathcal{E}_{k+2}) \Pr(\mathcal{E}_k)$, we consider the following process. A random walk in $\mathbb{Z}^2$ starts at $(k+2, k)$. In each iteration i, the random walk moves from (x, y) to $(x-1+\alpha(i), y-1+\beta(i))$ where $\alpha(i)$ and $\beta(i)$ are sampled from $\mathrm{Po}(\lambda(i))$ independently. If the random walk hits the axis $y = 0$ after a certain iteration $\mathcal{T}$, then it is stuck to the axis, i.e., for any $i > \mathcal{T}$, the update in the i-th iteration is from $(x, 0)$ to $(x-1+\alpha(i), 0)$; similarly, after reaching the axis $x = 0$, the random walk is stuck to the axis $x = 0$ and updates to $(0, y-1+\beta(i))$. Then, $\Pr(\mathcal{E}_{k+2}) \Pr(\mathcal{E}_k)$ is the probability that the random walk starting from $(k+2, k)$ reaches $(0, 0)$.

To prove (8.2), we consider two random walks in $\mathbb{Z}^2$ defined above. Let A be the random walk starting from $(k+2,k)$, and let B be the random walk starting from $(k+1,k+1)$. Let H_A and H_B be the event that A and B reaches $(0,0)$ respectively. To prove (8.2), it is sufficient to show:

$$\Pr(H_A) < \Pr(H_B).$$

To formalize this idea, we define a coupling between A and B such that: 1) whenever A reaches $(0,0)$, B also reaches $(0,0)$, and 2) with a positive probability, B reaches $(0,0)$ but A never does.

In defining the coupling, we use the idea of splitting and merging of Poisson processes [6]. We reinterpret the random walk by breaking down each *iteration* i into $J(i)$ *steps* such that it is symmetric in the x- and y-directions (with respect to the line $y=x$) and the movement in each step is "small".

If at the beginning of iteration i the process is at (x,y) with $x>0$ and $y>0$:

- At step 0 of iteration i, we sample $J(i) \sim \mathrm{Po}(2\lambda(i))$, set $(\alpha(i,0),\beta(i,0)) = (-1,-1)$, and update $(x,y) \mapsto (x+\alpha(i,0), y+\beta(i,0))$;

- At each step j for $j=1,\dots,J(i)$, $(\alpha(i,j),\beta(i,j)) = (1,0)$ with probability 0.5, and $(\alpha(i,j),\beta(i,j)) = (0,1)$ otherwise. Update $(x,y) \mapsto (x+\alpha(i,j), y+\beta(i,j))$;[2]

On the other hand, if $x=0$ (or $y=0$) at the beginning of iteration:

- At step 0 of iteration i, we sample $J(i) \sim \mathrm{Po}(2\lambda(i))$, set $\big(\alpha(i,0),\beta(i,0)\big) = (0,-1)$ (or $(-1,0)$ if $y=0$), and update $(x,y) \mapsto \big(x+\alpha(i,0), y+\beta(i,0)\big)$;

- At each step j for $j=1,\dots,J(i)$, with probability 0.5 $\big(\alpha(i,j),\beta(i,j)\big) = (1,0)$, (or $\big(\alpha(i,j),\beta(i,j)\big) = (0,1)$) and $(\alpha(i,j),\beta(i,j)) = (0,0)$, otherwise. Update $(x,y) \mapsto \big(x+\alpha(i,j), y+\beta(i,j)\big)$;

If at the end of iteration i, $(x,y)=(0,0)$ we stop the process.

Notice that we only switch from one type of iteration to the other if $x=0$ (or $y=0$) at the *end* of an iteration i. Here way say the random walk is stuck to the axis $x=0$ (or the axis $y=0$). If this happens, it will be stuck to this axis forever. Also, notice that in each step we have at most 1 unit movement. Also, in steps $j=1,\dots,J(i)$ the walk can only move further away from both axes $y=0$ and $x=0$.

[2] Standard results from Poisson process indicate that, $\sum_{j=1}^{J(i)} \alpha(i,j) \sim \mathrm{Po}(\lambda(i))$, and $\sum_{j=1}^{J(i)} \beta(i,j) \sim \mathrm{Po}(\lambda(i))$ which are two independent Poisson random variables.

Let $\big(x(i,j), y(i,j)\big)$ be the position of the random walk after iteration i step j, and $\big(x(i), y(i)\big)$ be its position at the end of iteration i. Moreover, let $\alpha(i) = \sum_{j=1}^{J(i)} \alpha(i,j)$ be the net movement in x direction during iteration i excluding the movement in Step 0, and let $\bar{\alpha}(i) = \alpha(i) + \alpha(i,0)$ be the net movement including movement at step 0. Similarly define y-directional movements $\beta(i) = \sum_{j=1}^{J(i)} \beta(i,j)$ and $\bar{\beta}(i)$.

8.4.2 The Coupling

We want to show that the probability of A reaching the origin is less that of B. To this end, we create a coupling between the two walks, which we outline here. Fig. 8.1 and Fig. 8.2 illustrate most aspects of this coupling. In the description of the coupling, we will let B move "freely", and define how A is "coupled with" B.

Recall that A starts at $(k+2, k)$ and B starts at $(k+1, k+1)$. At the beginning, we set A's movement to be identical to B's. Before one of them hits the origin, either of the following two events must happen: A and B become symmetric to the line $x = y$ at some step, $\mathcal{E}_{\mathsf{symm}}$, or A reaches the axis $y = 0$ at the end of some iteration, $\mathcal{E}_{\mathsf{skew}}$. This is called Phase I and is further discussed in Sect. 8.4.2.1.

In the first case $\mathcal{E}_{\mathsf{symm}}$, the positions of A and B are symmetric. We set A's movement to mirror B's movement. Therefore, in this case, A and B will both hit the origin, or neither of them will. This is called Phase II Symm and is further discussed in Sect. 8.4.2.2.

For the latter case $\mathcal{E}_{\mathsf{skew}}$, A reaches the axis $y = 0$ at iteration $\mathcal{T}_{\mathsf{skew}}$. We call the process is in Phase II Skew and further discussed in Sect. 8.4.2.3. Because B starts one unit above A and one unit to the left of A, at iteration $\mathcal{T}_{\mathsf{skew}}$, B is at the axis $y = 1$ and one unit to the left of A. Next we couple A's movement in the x-direction to be identical to B's, so that B is always one unit to the left of A. This coupling continues unless B hits the axis $x = 0$. Denote this iteration $\mathcal{T}^*$. At time $\mathcal{T}^*$, A is one unit to the right of the axis $x = 0$. Recall that at iteration $\mathcal{T}_{\mathsf{skew}}$ when $\mathcal{E}_{\mathsf{skew}}$ happens, B is one unit above the axis so that $y = 1$. Therefore, we can couple the movement of A in the x-direction after iteration $\mathcal{T}^*$ with B's movement in the y-direction after iteration $\mathcal{T}_{\mathsf{skew}}$. Because $\lambda(i)$ increases with i, we can couple the walks in such a way as to ensure that A moves toward the origin at a strictly slower rate than B does. Therefore, A only reaches the y-axis $x = 0$ if B reaches the x-axis $y = 0$, and we have shown that A is less likely to reach the origin than B does.

Let $\big(x^A(i,j), y^A(i,j)\big)$, and $\big(x^B(i,j), y^B(i,j)\big)$ be the coordinates for A and B respectively after iteration i step j. Similarly, let $J^A(i)$ and $J^B(i)$ be the number of steps for A and B in iteration i. Let $\alpha^A(i,j)$ and $\alpha^B(i,j)$ be the x-direction movements of both walks in iteration i step j, and $\beta^A(i,j)$ and $\beta^B(i,j)$ be the corresponding y-direction movements.

8.4.2.1 Phase I

Starting with $\big(x^A(0), y^A(0)\big) = (k+2, k)$ and $\big(x^B(0), y^B(0)\big) = (k+1, k+1)$, A moves in exactly the same way as B, i.e., $J^A(i) = J^B(i)$, $\alpha^A(i,j) = \alpha^B(i,j)$ and $\beta^A(i,j) = \beta^B(i,j)$, until one of the following two events happens.

Event $\mathcal{E}_{\mathsf{symm}}$ The current position of A and B are symmetric with respect to the line $y = x$, i.e., $x^A(i,j) - x^B(i,j) = y^B(i,j) - y^A(i,j)$ and $x^A(i,j) + x^B(i,j) = y^A(i,j) + y^B(i,j)$. Notice that $\mathcal{E}_{\mathsf{symm}}$ may happen in some middle step j of an iteration i. When $\mathcal{E}_{\mathsf{symm}}$ happens, we move on to Phase II Symm.

Event $\mathcal{E}_{\mathsf{skew}}$ A hits the axis $y = 0$ *at the end of an iteration.* Notice that this means A is then stuck to the axis $y = 0$ forever. When $\mathcal{E}_{\mathsf{skew}}$ happens, we move on to Phase II Skew. Note that B is one unit away from the axis $y = 0$, $y^B = 1$. We remark that the in the third part we show, if event $\mathcal{E}_{\mathsf{skew}}$ happens, B has a higher chance to reach $(0,0)$ than A.

The following three claims will be useful.

Claim 8.17. *A is always below the line $y = x$ before $\mathcal{E}_{\mathsf{symm}}$ happens, so A will never hit the axis $x = 0$ in Phase I.*

Proof. To see this, A can only have four types of movements in each step: lower-left $(x,y) \mapsto (x-1, y-1)$, up $(x,y) \mapsto (x, y+1)$, and right $(x,y) \mapsto (x+1, y)$. It is easy to see that, 1) A will never step across the line $y = x$ in one step, and 2) if A ever reaches the line $y = x$ at (w,w) for some w, then A must be at $(w, w-1)$ in the previous step. However, when A is at $(w, w-1)$, B should be at $(w-1, w)$ according to the relative position of A, B. In this case event $\mathcal{E}_{\mathsf{symm}}$ already happens. □

Claim 8.18. *$\mathcal{E}_{\mathsf{symm}}$ and $\mathcal{E}_{\mathsf{skew}}$ cannot happen simultaneously.*

Proof. Suppose $\mathcal{E}_{\mathsf{symm}}$ and $\mathcal{E}_{\mathsf{skew}}$ happen at the same time, then it must be that A is at $(1,0)$ and B is at $(0,1)$, as the relative position of A and B is unchanged in Phase I, and this must be at the end of a certain *iteration.* In the previous iteration, A must be at $(2,1)$, since $\mathcal{E}_{\mathsf{skew}}$ did not happen yet and A is below the line $y = x$. However, B is at $(1,2)$ when A is at $(2,1)$, implying that case $\mathcal{E}_{\mathsf{symm}}$ has already happened in the previous iteration, which is a contradiction. □

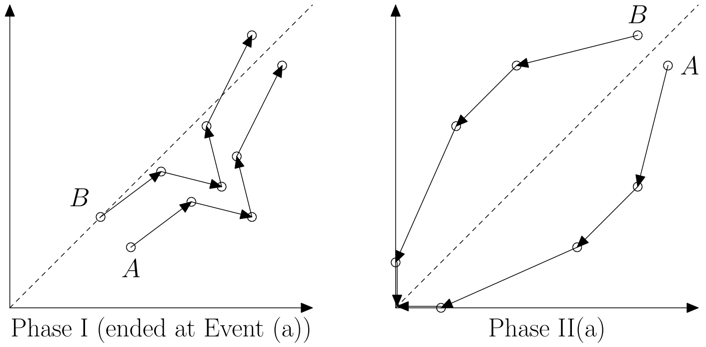


Figure 8.1: The coupling with Phase I ended at Event $\mathcal{E}_{\mathsf{symm}}$

Claim 8.19. *B cannot reach the axis $x = 0$ before either $\mathcal{E}_{\mathsf{symm}}$ or $\mathcal{E}_{\mathsf{skew}}$ happen.*

Proof. If $\mathcal{E}_{\mathsf{symm}}$ happens before $\mathcal{E}_{\mathsf{skew}}$, B cannot reach the axis $x = 0$ before $\mathcal{E}_{\mathsf{symm}}$ as A is always below the line $y = x$ and B is always on the upper-left diagonal of A. If $\mathcal{E}_{\mathsf{skew}}$ happens before $\mathcal{E}_{\mathsf{symm}}$, B cannot reach the axis $x = 0$ before $\mathcal{E}_{\mathsf{skew}}$, or even by the time $\mathcal{E}_{\mathsf{skew}}$ happens: by the time $\mathcal{E}_{\mathsf{skew}}$ happens, A can only at one of $(2, 0), (3, 0), (4, 0), \ldots$ (A cannot be at $(1, 0)$, for otherwise $\mathcal{E}_{\mathsf{symm}}$ and $\mathcal{E}_{\mathsf{skew}}$ happen simultaneously, which is impossible as shown just now), in which case B will not be at the axis $x = 0$. □

8.4.2.2 Phase II Symm

Let A move in a way that is symmetric to B with respect to the line $y = x$: $J^A(i) = J^B(j)$, $\alpha^A(i, j) = \beta^B(i, j)$ and $\beta^A(i, j) = \alpha^B(i, j)$. Notice that, in Phase II Symm, A may cross the line $y = x$, after which A is above the line $y = x$ while B is below.

8.4.2.3 Phase II Skew

If event $\mathcal{E}_{\mathsf{skew}}$ happens, we need a more complicated coupling. Suppose Phase II Skew starts after iteration $\mathcal{T}_{\mathsf{skew}}$. Here we use $\mathcal{T}^A_S$ (and $\mathcal{T}^B_S$) to denote the hitting time of A (and B) to a set of states S which is the first iteration of the process into the set S. For example $i = \mathcal{T}^B_{y=1}$ is the hitting time of B such that $y^B(i) = 1$. Here we list six relevant hitting times and their relationship.

$$\mathcal{T}_{\mathsf{skew}} = \mathcal{T}^B_{y=1} = \mathcal{T}^A_{y=0} < \mathcal{T}^B_{y=0}, \text{ and } \mathcal{T}_{\mathsf{skew}} < \mathcal{T}^B_{x=0} = \mathcal{T}^A_{x=1} < \mathcal{T}^A_{x=0}.$$

Back to the coupling, we first let the x-direction movement of A be the same with that of B. To be specific, in each iteration $\mathcal{T}_{\mathsf{skew}} < i \le \mathcal{T}^B_{x=0}$, set $J^A(i) = J^B(i)$. At step j, we set $\alpha^A(i, j) = \alpha^B(i, j)$ and $\beta^A(i, j) = 0$ ($\beta^A(i, j)$ is always 0 now, as A is stuck to the axis $y = 0$). Till now, the relative position of A and B in x-coordinate is preserved $x^A(i,j) = x^B(i,j) + 1$. Let $\mathcal{E}^*$ be the event that B reaches the axis $x = 0$, and let $\mathcal{E}^*$ happens at the end of iteration $\mathcal{T}^* = \mathcal{T}^B_{x=0}$. We further define $\Delta = \mathcal{T}^* - \mathcal{T}_{\mathsf{skew}}$ to be the additional time before $x^B = 0$ (if both stopping times exist), and $L = \mathcal{T}^B_{y=0} - \mathcal{T}_{\mathsf{skew}}$ to be the additional time before $y^B = 0$ (if both stopping times exist).

At the end of iteration $\mathcal{T}^*$, the positions for A is one unit to the right of the origin. That is $x^A(\mathcal{T}^*) = 1$ while $y^A(\mathcal{T}^*)) = 0$. Informally, we want to couple the movement of A from $(1,0)$ at $\mathcal{T}^*$ to the movement of B in the y-direction at $\mathcal{T}_{\mathsf{skew}}$ which is one unit above the axis at $y = 1$. Formally, starting at $(1,0)$, A is a 1-dimensional random walk on the axis $y = 0$, and we couple it to B in the following way.

- For each $t = 1, \ldots, L$, we couple A's movement in the x direction at iteration $\mathcal{T}^* + t$ with B's movement Δ steps earlier in the y direction at iteration $\mathcal{T}^* + t - \Delta = \mathcal{T}_{\mathsf{skew}} + t$ such that $\alpha^A(\mathcal{T}^* + t) \sim \mathrm{Po}(\lambda(\mathcal{T}^* + t))$ and $\alpha^A(\mathcal{T}^* + t) \geq \beta^B(\mathcal{T}_{\mathsf{skew}} + t)$. [3]
- We do not couple A to B for future iterations after $\mathcal{T}^* + L$.

A key property of this coupling is that the x-coordinate of A at $\mathcal{T}^* + t$ is always greater or equal to the y-coordinate of B at iteration $\mathcal{T}_{\mathsf{skew}} + t$.

Claim 8.20. *For all* $t = 1, \ldots, L$, $x^A(\mathcal{T}^* + t) \geq y^B(\mathcal{T}_{\mathsf{skew}} + t)$.

Proof. We use induction. For the base case, we have $1 = x^A(\mathcal{T}^*) = y^B(\mathcal{T}_{\mathsf{skew}})$ from the definitions of $\mathcal{T}_{\mathsf{skew}}$ and $\mathcal{T}^*$. For the inductive case, $\alpha^A(\mathcal{T}^* + t) \geq \beta^B(\mathcal{T}_{\mathsf{skew}} + t)$ due to our coupling. □

8.4.3 Validity of the Coupling

The coupling induces the correct marginal random walk process for B, as we have defined the coupling in a way that B is moving "freely" and A is being "coupled" with B. The only non-trivial part is to show that the coupling induces the correct

[3] Here is an example of such a coupling. Consider iteration $i = \mathcal{T}^* + t$ for A, and we want to couple it with B's movement at iteration $\iota = \mathcal{T}_{\mathsf{skew}} + t$. Let $J^B(\iota)$ be the number of steps of B in the iteration ι which is not necessary equal to the number of steps of A after iteration $\mathcal{T}^*$. At step 0, we sample a non-negative integer $d(i) \sim \mathrm{Po}(2(\lambda(i) - \lambda_\iota))$ independent to $J^B(\iota)$, and set the number of steps of A to be $J^A(i) = J^B(\iota) + d(i)$. Then set $\alpha^A(i,0) = -1$ and $\beta(i,0)^A = 0$. At each step $j = 1, \ldots, J^B(\iota)$, we set $(\alpha^A(i,j), \beta^A(i,j)) = (\beta^B_{\iota j}, 0)$. At the later steps $j = J^B(\iota) + 1, \ldots, J^A(i)$, we set $(\alpha^A(i,j), \beta^A(i,j)) = (1,0)$ with probability 0.5, or $(0,0)$ otherwise.

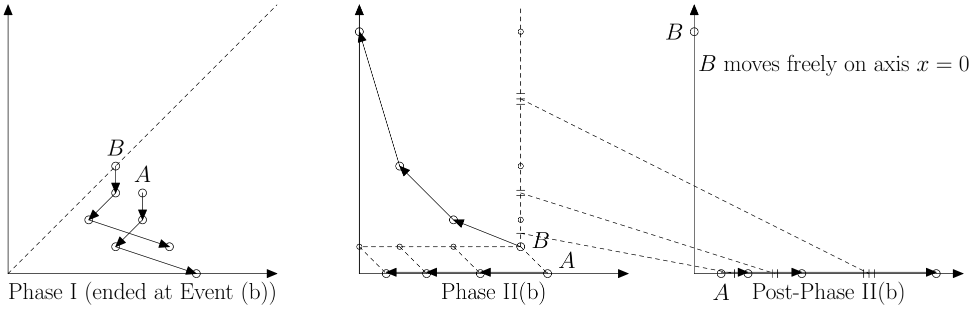


Figure 8.2: The coupling with Phase I ended at Event $\mathcal{E}_{\mathsf{skew}}$, if $\mathcal{E}^*$ happens

marginal random walk process for A. It is straightforward to check that the marginal probabilities are correct during Phase I, before the event $\mathcal{E}^*$ occurs, or if the event $\mathcal{E}^*$ does not occur. If the process enters Phase II Skew and B reaches the axis $x = 0$, the movement of A in the x direction is coupled with B's movement in y direction $\Delta = \mathcal{T}^* - \mathcal{T}_{\mathsf{skew}}$ iterations ago. We note that B's movements in the x direction and the y direction are independent and A does not contain two iterations that are coupled to a same iteration of B. Therefore, the movements of A in x direction after $\mathcal{T}^*$ are independent to its previous movement, so the marginal distribution is correct. Fig. 8.3 illustrates the coupling time line.

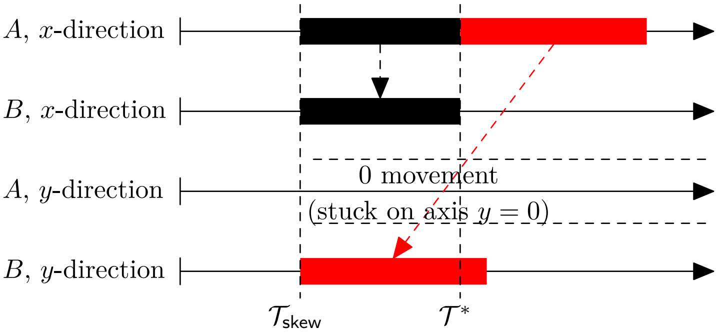


Figure 8.3: The time line for the coupling after event $\mathcal{E}_{\mathsf{skew}}$ happens.

Remark 8.21. The coupling of the two random walks A and B in $\mathbb{Z}^2$ in the proof above can be alternatively viewed as a coupling of four independent random walks in $\mathbb{Z}$ (this is why we have said that "we simultaneously couple four cascade processes" in the introduction), as the x-directional and y-directional movements for both A and B correspond to the four terms in inequality (8.2), which are intrinsically independent.

8.4.4 Proof of Inequality (8.2)

It suffices to show that in our coupling $H_A \subseteq H_B$ and $H_B \setminus H_A$ is not empty, because this implies inequality (8.2): $\Pr(H_A) = \Pr(H_B \cap H_A) < \Pr(H_B \cap H_A) + \Pr(H_B \setminus H_A) = \Pr(H_B)$. We aim to show that:

1. if the coupling never moves to Phase II, neither A nor B reaches $(0, 0)$;

2. if the coupling moves to Phase II Symm, A reaches $(0, 0)$ if and only if B reaches $(0, 0)$;

3. if the coupling moves to Phase II Skew, A reaches $(0, 0)$ implies that B also reaches $(0, 0)$;

4. there is an event with a positive probability such that B reaches $(0, 0)$ but A does not.

The first, second, and third show $H_A \subseteq H_B$. The last one shows $H_B \setminus H_A$ has a positive probability.

1 is trivial. 2 follows from symmetry.

To see 3, first notice that in Phase II Skew, $\mathcal{E}^*$ must happens if A ever reaches $(0, 0)$: because A can move to the left by at most 1 unit in each iteration, A must first reach $(1, 0)$, but at this point $x^B = 0$ and event $\mathcal{E}^*$ happens. Now consider the case that B never reaches the origin after event $\mathcal{E}^*$. Then the x movement of A remains coupled to the y-movement of B in such a way that $\bar{\alpha}^A(\mathcal{T}^* + t) \geq \bar{\beta}^B(\mathcal{T}_{\mathsf{skew}} + t)$. Walk A starts at $x^A = 1$, and walk B starts at $y^B = 1$. Therefore, A cannot reach the origin if B does not. In the case walk B meets the origin, the statement is vacuously true.

For 4, to show $\Pr(H_B \setminus H_A) > 0$, we define the following event which consists of four parts. i) For all $i = 1, \dots, k$, it happens that $\alpha^A(i) = \beta^A(i) = 0$, in which case the event $\mathcal{E}_{\mathsf{skew}}$ happens at $\mathcal{T}_{\mathsf{skew}} = k$ and A reaches $(2, 0)$. ii) For $i = k + 1$, it happens that $\alpha^A(i) = 0$ and $\beta^B(i) = 1$, in which case A reaches $(1, 0)$ and B reaches $(0, 1)$, and the process B reaches the axis $x = 0$ at iteration $\mathcal{T}^* = k + 1$. iii) In iteration $i = \mathcal{T}^* + 1$, it happens that $\beta^B(i) = 0$, so B reaches $(0, 0)$. On the other hand, by the coupling $\alpha^A(\mathcal{T}^* + 1) \geq \beta^B(\mathcal{T}_{\mathsf{skew}} + 1) = 1$, so A does not reach $(0, 0)$ at iteration $\mathcal{T}^* + 1 = k + 2$. iv) Finally, it happens that $\alpha^A(i) \geq 1$ for all $i > k + 2$. It is straightforward the i), ii), and iii) happen with positive probabilities. By direct computations, iv) happens with a positive probability as well.[4] Since the above event

[4] The event that $\alpha^A(i) \geq 1$ for all $i > k+2$ happens with probability $\prod_{i>k+2} \Pr(\mathrm{Po}(\lambda(i)) \geq 1) = \prod_{i>k+2}(1 - \exp(-\lambda(i))) \geq \prod_{i \geq r+1}(1 - \exp(-\binom{i-1}{r-1}c^r))$ which is a positive constant depending on r and c.

consisted of i), ii), iii) and iv) belongs to $H_B \setminus H_A$ and each of the four sub-events happens with a positive probability, 4 is implied.

From 2, 3, and 4, we learn that the probability that B reaches $(0, 0)$ is strictly larger than that of A, which implies inequality (8.2) and concludes the proof.

8.5 Optimal Seeds in Submodular Influence Maximization

We have seen that putting all the K seeds in a single leaf is optimal for r-complex contagion, when the root node has weight $\omega(1/n^{1+1/r})$. To demonstrate the sharp difference between r-complex contagion and a submodular cascade model, we present a submodular InfMax example where the optimal seeding strategy is to put no more than one seed in each leaf. The hierarchy tree T in our example meets all the assumptions we have made in the previous sections, including large communities, proper separation, and $w(\text{root}) = \omega(1/n^{1+1/r})$, where r is now an arbitrarily fixed integer with $r \geq 2$.

We consider `UICM` with $p = 1/n^{1-\frac{1}{4r}}$. The hierarchy tree T contains only two levels: a root and K leaves. The root has weight $1/n^{1+\frac{1}{2r}}$, and each leaf has weight 1. After $G \sim \mathcal{G}(n, T)$ is sampled and each edge in G is sampled with probability p, the probability that an edge appears between two vertices from different leaves is $(1/n^{1-\frac{1}{4r}}) \cdot (1/n^{1+\frac{1}{2r}}) = o(1/n^2)$, and the probability that an edge appears between two vertices from a same leaf is $1 \cdot (1/n^{1-\frac{1}{4r}}) = \omega(\log n/n)$. Therefore, with probability $1 - o(1)$, the resultant graph is a union of K connected components, each of which corresponds to a leaf of T. It is then straightforward to see that the optimal seeding strategy is to put a single seed in each leaf.

8.6 A Dynamic Programming Algorithm

In this section, we present an algorithm which finds an optimal seeding strategy when all $w(t)$'s fall into two regimes: $w(t) = \omega(1/n^{1+1/r})$ and $w(t) = o(1/n^2)$. We will assume this for $w(t)$'s throughout this section. Since a parent tree node always has less weight than its children (see Definition 8.1), we can decompose T into the *upper part* and the *lower part*, where the lower part consists of many subtrees whose roots have weights in $\omega(1/n^{1+1/r})$, and the upper part is a single tree containing only tree nodes with weights in $o(1/n^2)$ and whose leaves are the parents of those roots of the subtrees in the lower part. We call each subtree in the lower part a *maximal dense subtree* defined formally below.

Definition 8.22. Given a hierarchy tree $T = (V_T, E_T, w, v)$, a subtree rooted at $t \in V_T$ is a *maximal dense subtree* if $w(t) = \omega(1/n^{1+1/r})$, and either t is the root, or $w(t') = O(1/n^{1+1/r})$ where t' is the parent of t.

Since we have assumed either $w(t) = \omega(1/n^{1+1/r})$ or $w(t) = o(1/n^2)$, $w(t') = O(1/n^{1+1/r})$ in the definition above implies $w(t') = o(1/n^2)$.

The idea of our algorithm is the following: firstly, after the decomposition of T into the upper and lower parts, we will show that the weights of the tree nodes in the upper part, falling into $w(t) = o(1/n^2)$, are negligible so that we can treat the whole tree T as a forest with only those maximal dense subtrees in the lower part (that is, we can remove the entire upper part from T); secondly, Theorem 8.11 shows that after we have decide the number of seeds to be allocated to each maximal dense subtree, the optimal seeding strategy is to put all the seeds together in a single leaf that has the highest density defined in Definition 8.3; finally, we use a dynamic programming approach to allocate the K seeds among those maximal dense subtrees.

Now, we are ready to describe our algorithm, presented in Algorithm 8.1.

The correctness of Algorithm 8.1 follows immediately from Theorem 8.23 (below) and Theorem 8.11. Theorem 8.23 shows that we can ignore the upper part of T and treat T as the forest consisting of all the maximal dense subtrees of T when considering the InfMax problem. Recall Theorem 8.11 shows that for each subtree T_i and given the number of seeds, the optimal seeding strategy is to put all the seeds on the leaf with the highest density.

Theorem 8.23. *Given $T = (V_T, E_T, w, v)$, let $\{T_1, \ldots, T_m\}$ be the set of all T's maximal dense subtrees and let T^- be the forest consisting of $T_1, \ldots, T_m$. For any seeding strategy $\boldsymbol{k}$ and any $r \geq 2$, we have $\Sigma_{r,T}(\boldsymbol{k}) = \Sigma_{r,T^-}(\boldsymbol{k})$.*

Proof. Since the total number of possible edges between T^- and the rest of the tree is upper bounded by n^2 and each such edge appears with probability $o(1/n^2)$, the expected number of edges is $o(1)$. By Markov's inequality the probability there exists edges between T^- and the rest of the tree $o(1)$. Therefore, we have

$$\frac{\mathbb{E}_{G \sim \mathcal{G}(n,T)}[\sigma_{r,G}(\boldsymbol{k})]}{n} = \frac{o(1)O(n) + (1 - o(1))\mathbb{E}_{G \sim \mathcal{G}(n,T^-)}[\sigma_{r,G}(\boldsymbol{k})]}{n}.$$

Taking $n \to \infty$ we have concludes the proof. □

```
**Input:** $r \in \mathbb{Z}$ with $r \geq 2$, $T = (V_T, E_T, w, v)$, and $K \in \mathbb{Z}^+$
Find all maximal dense subtrees $T_1, \dots, T_m$, and let $r_1, \dots, r_m$ be their roots (Definition 8.22).
For each $T_i$ and each $k = 0, 1, \dots, K$, let $\boldsymbol{s}_i^*(k)$ be the seeding strategy that puts $k$ seeds in the leaf $t \in L_{T_i}$ with the highest density, and let
    $$h(T_i, k) = \lim_{n \to \infty} \frac{\mathbb{E}_{G \sim \mathcal{G}(v(r_i) \cdot n, T_i)}[\sigma_{r,G}(\boldsymbol{s}_i^*(k))]}{n}$$
    be the expected number of infected vertices in the subgraph defined by $T_i$, normalized by the total number of vertices in the whole graph.
Let $S[i, k]$ store a seeding strategy that allocates $k$ seeds in the first $i$ subtrees $T_1, \dots, T_i$, and let $H[i, k]$ be the expected total number of infected vertices corresponding to $S[i, k]$, divided by $n$.
**for** $k = 0, 1, \dots, K$ **do**
    set $S[1, k] = \boldsymbol{s}_1^*(k)$ and $H[1, k] = h(T_1, k)$.
**end for**
**for** each $i = 2, \dots, m$ **do**
    **for** $k = 0, 1, \dots, K$ **do**
        $k_i = \operatorname*{argmax}_{k_i \in \{0,1,\dots,k\}} H[i-1, k-k_i] + h(T_i, k_i)$;
        set $S[i, k]$ be the strategy that allocates $k - k_i$ seeds among $T_1, \dots, T_{i-1}$ according to $S[i-1, k-k_i]$ and puts the remaining $k_i$ seeds in the leaf of $T_i$ with the highest density;
        set $H[i, k] = H[i-1, k-k_i] + h(T_i, k_i)$;
    **end for**
**end for**
**Output:** the seeding strategy $S[m, K]$.
```

Algorithm 8.1: The InfMax algorithm

Finally, it is straightforward to see the time complexity of Algorithm 8.1, in terms of the number of evaluations of $\Sigma_{r,\mathcal{G}(n,T)}(\cdot)$.

Theorem 8.24. *Algorithm 8.1 requires $O_I(|V_T|K^2)$ computations of $\Sigma_{r,\mathcal{G}(n,T)}(\cdot)$.*

8.7 Conclusion and Future Work

In this chapter, we presented an influence maximization algorithm which finds optimal seeds for the stochastic hierarchical blockmodel, assuming the weights of tree nodes do not fall into a narrow regime between $\Omega(1/n^2)$ and $O(1/n^{1+1/r})$. As a crucial observation behind the algorithm, when the root of the tree has weight $\omega(1/n^{1+1/r})$, our results show that the optimal seeding strategy is to put all the seeds together. Our results provide a formal verification for the intuition that one should put the seeds close to each other to maximize the synergy effect in a nonsubmodular cascade model.

Removing Limitations One obvious future direction is to extend our algorithm such that it works for weights of tree nodes between $\Omega(1/n^2)$ and $O(1/n^{1+1/r})$ as well. Related to this, Schoenebeck and Tao [66] shows that InfMax for the complex contagion on the stochastic hierarchical blockmodel is NP-hard to approximate to within factor $n^{1-\varepsilon}$ if vertices have non-homogeneous thresholds, i.e., each vertex v has a individual threshold $r_v \in \mathbb{Z}^+$ such that v is infected when it has at least r_v infected neighbors. It is unknown whether this inapproximability result carries over to the homogeneous case where all agents have the same threshold.

It is also interesting to see if our main result Theorem 8.11 still holds without the proper separation assumption. We only use this assumption in the proof of Proposition 8.15. To remove the proper separation assumption, more insight is needed on the behavior of the cascade in the critical leaves. As a next step for this, one might consider the case when leaves t_1 and t_2 have weights $c_1 n^{-1/r}$ and $c_2 n^{-1/r}$ respectively, and their parent t has weight $dn^{-1/r}$ with $d < c_1$ and $d < c_2$; it is an interesting open problem to see that if it is still optimal to either put all the seeds in t_1 or to put all the seeds in t_2. We conjecture this is true.

Extension One way to extend our results is to relax the assumption that the network is known. For example, can the network be learned from observing previous cascades, or by experimenting with them? Or, can they be elicited from agents with limited, local knowledge? Another direction would be to leverage these results to create heuristics that work well on real-world networks. A final direction would be more careful empirical studies (particularly experiments) about the nature of various cascades (e.g. submodular versus nonsubmodular).

BIBLIOGRAPHY

[1] J. A. Adell and P. Jodrá. Exact kolmogorov and total variation distances between some familiar discrete distributions. *Journal of Inequalities and Applications*, 2006(1):64307, 2006.

[2] R. Angell and G. Schoenebeck. Don't be greedy: Leveraging community structure to find high quality seed sets for influence maximization. *WINE*, 2016.

[3] A. Arora, S. Galhotra, and S. Ranu. Debunking the myths of influence maximization. In *Proceedings of the 2017 ACM International Conference on Management of Data-SIGMOD'17*, 2017.

[4] L. Backstrom, D. P. Huttenlocher, J. M. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *ACM SIGKDD*, 2006.

[5] E. Balkanski, N. Immorlica, and Y. Singer. The importance of communities for learning to influence. In *Advances in Neural Information Processing Systems*, pages 5862–5871, 2017.

[6] D. P. Bertsekas and J. N. Tsitsiklis. *Introduction to probability*, volume 1. Athena Scientific Belmont, MA, 2002.

[7] S. Bharathi, D. Kempe, and M. Salek. Competitive influence maximization in social networks. In *WINE*, 2007.

[8] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier. Maximizing social influence in nearly optimal time. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 946–957. SIAM, 2014.

[9] D. Centola. The spread of behavior in an online social network experiment. *Science*, 329(5996):1194–1197, 2010.

[10] D. Centola and M. Macy. Complex contagions and the weakness of long ties. *American journal of Sociology*, 113(3):702–734, 2007.

[11] J. Chalupa, P. L. Leath, and G. R. Reich. Bootstrap percolation on a bethe lattice. *Journal of Physics C: Solid State Physics*, 12(1):L31, 1979.

[12] N. Chen. On the approximability of influence in social networks. *SIAM Journal on Discrete Mathematics*, 23(3):1400–1415, 2009.

[13] W. Chen. An issue in the martingale analysis of the influence maximization algorithm imm. In *International Conference on Computational Social Networks*, pages 286–297. Springer, 2018.

[14] W. Chen and B. Peng. On adaptivity gaps of influence maximization under the independent cascade model with full adoption feedback. In *ISAAC 2019: The 30th International Symposium on Algorithms and Computation*, 2019.

[15] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *ACM SIGKDD*, pages 199–208. ACM, 2009.

[16] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1029–1038. ACM, 2010.

[17] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *2010 IEEE International Conference on Data Mining*, pages 88–97. IEEE, 2010.

[18] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 88–97. IEEE, 2010.

[19] W. Chen, L. V. Lakshmanan, and C. Castillo. Information and influence propagation in social networks. *Synthesis Lectures on Data Management*, 5(4):1–177, 2013.

[20] W. Chen, T. Lin, Z. Tan, M. Zhao, and X. Zhou. Robust influence maximization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 795–804. ACM, 2016.

[21] W. Chen, B. Peng, G. Schoenebeck, and B. Tao. Adaptive greedy versus non-adaptive greedy for influence maximization. In *AAAI*, 2020.

[22] S. Cheng, H. Shen, J. Huang, G. Zhang, and X. Cheng. Staticgreedy: solving the scalability-accuracy dilemma in influence maximization. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 509–518. ACM, 2013.

[23] A. Clauset, C. Moore, and M. E. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

[24] P. DiMaggio. Structural analysis of organizational fields: A blockmodel approach. *Research in organizational behavior*, 1986.

[25] I. Dinur. The pcp theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.

[26] P. Domingos and M. Richardson. Mining the network value of customers. In *ACM SIGKDD*, 2001.

[27] R. Durrett. *Lecture notes on particle systems and percolation.* Brooks/Cole Pub Co, 1988.

[28] J. W. Essam. Percolation theory. *Reports on Progress in Physics*, 43(7):833, 1980.

[29] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[30] W. Feller. *An introduction to probability theory and its applications*, volume 2. John Wiley & Sons, 2008.

[31] S. Galhotra, A. Arora, and S. Roy. Holistic influence maximization: Combining scalability and efficiency with opinion-aware models. In *Conference on Management of Data*, pages 743–758. ACM, 2016.

[32] J. Gao, G. Ghasemi, J. J. Jones, and G. Schoenebeck. Complex contagions in charitable donations. 2019.

[33] S. Goldberg and Z. Liu. The diffusion of networking technologies. In *SODA*, 2013.

[34] J. Goldenberg, B. Libai, and E. Muller. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review*, 9(3):1–18, 2001.

[35] J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001.

[36] D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of AI Research*, 42:427–486, 2011.

[37] A. Goyal, W. Lu, and L. V. Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th international conference WWW*, pages 47–48. ACM, 2011.

[38] A. Goyal, W. Lu, and L. V. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 211–220. IEEE, 2011.

[39] M. Granovetter. Threshold models of collective behavior. *American Journal of Sociology*, 83(6):1420–1443, 1978. URL http://www.journals.uchicago.edu/doi/abs/10.1086/226707.

[40] K. Han, K. Huang, X. Xiao, J. Tang, A. Sun, and X. Tang. Efficient algorithms for adaptive influence maximization. *Proceedings of the VLDB Endowment*, 11 (9):1029–1040, 2018.

[41] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[42] S. Janson, T. Łuczak, T. Turova, and T. Vallier. Bootstrap percolation on the random graph $g_{N,P}$. *The Annals of Applied Probability*, 22(5):1989–2047, 2012.

[43] K. Jung, W. Heo, and W. Chen. IRIE: Scalable and robust influence maximization in social networks. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 918–923. IEEE, 2012.

[44] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *ACM SIGKDD*, pages 137–146, 2003.

[45] D. Kempe, J. M. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *ICALP*, pages 1127–1138, 2005.

[46] J. H. Kemperman. On the optimum rate of transmitting information. In *Probability and information theory*, pages 126–169. Springer, 1969.

[47] S. Khanna and B. Lucier. Influence maximization in undirected networks. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1482–1496. Society for Industrial and Applied Mathematics, 2014.

[48] L. Le Cam et al. An approximation theorem for the poisson binomial distribution. *Pacific Journal of Mathematics*, 10(4):1181–1197, 1960.

[49] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[50] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. In *EC*, pages 228–237, 2006.

[51] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.

[52] D. A. Levin and Y. Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[53] Q. Li, W. Chen, X. Sun, and J. Zhang. Influence maximization with ε-almost submodular threshold functions. In *NIPS*, pages 3804–3814, 2017.

[54] Y. Li, J. Fan, Y. Wang, and K.-L. Tan. Influence maximization on social graphs: A survey. *IEEE Trans. on Knowledge and Data Engineering*, 30(10):1852–1872, 2018.

[55] T. M. Liggett. *Interacting particle systems*, volume 276. Springer Science & Business Media, 2012.

[56] Y. Lim, A. Ozdaglar, and A. Teytelboym. A simple model of cascades in networks, 2015.

[57] V. Lyzinski, M. Tang, A. Athreya, Y. Park, and C. E. Priebe. Community detection and classification in hierarchical stochastic blockmodels. *arXiv*, 2015.

[58] E. Mossel and S. Roch. Submodularity of influence in social networks: From local to global. *SIAM J. Comput.*, 39(6):2176–2188, 2010.

[59] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1): 265–294, 1978.

[60] N. Ohsaka, T. Akiba, Y. Yoshida, and K.-i. Kawarabayashi. Fast and accurate influence maximization on large networks with pruned monte-carlo simulations. In *AAAI*, pages 138–144, 2014.

[61] B. Peng and W. Chen. Adaptive influence maximization with myopic feedback. In *Advances in Neural Information Processing Systems*, pages 5575–5584, 2019.

[62] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *ACM SIGKDD*, pages 61–70, 2002.

[63] D. M. Romero, B. Meeder, and J. Kleinberg. Differences in the mechanics of information diffusion across topics : Idioms , political hashtags , and complex contagion on twitter. In *WWW*, pages 695–704. ACM, 2011. URL http://dl.acm.org/citation.cfm?id=1963503.

[64] G. Sadeh, E. Cohen, and H. Kaplan. Sample complexity bounds for influence maximization. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[65] M. Salek, S. Shayandeh, and D. Kempe. You share, I share: Network effects and economic incentives in P2P file-sharing systems. In *International Workshop on Internet and Network Economics*, pages 354–365. Springer, 2010.

[66] G. Schoenebeck and B. Tao. Beyond worst-case (in)approximability of nonsubmodular influence maximization. In *International Conference on Web and Internet Economics*, pages 368–382. Springer, 2017.

[67] G. Schoenebeck and B. Tao. Influence maximization on undirected graphs: Towards closing the $(1-1/e)$ gap. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, pages 423–453. ACM, 2019.

[68] G. Schoenebeck and B. Tao. Beyond worst-case (in)approximability of nonsubmodular influence maximization. *ACM Trans. Comput. Theory*, 11(3): 12:1–12:56, Apr. 2019. ISSN 1942-3454. doi: 10.1145/3313904. URL http://doi.acm.org/10.1145/3313904.

[69] G. Schoenebeck, B. Tao, and F.-Y. Yu. Think globally, act locally: On the optimal seeding for nonsubmodular influence maximization. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[70] G. Schoenebeck, B. Tao, and F.-Y. Yu. Limitations of greed: Influence maximization in undirected networks re-visited. In *International Conference on Autonomous Agents and Multi-Agent Systems*, 2020.

[71] Y. Tang, X. Xiao, and Y. Shi. Influence maximization: Near-optimal time complexity meets practical efficiency. In *SIGMOD international conference on Management of data*, pages 75–86. ACM, 2014.

[72] Y. Tang, Y. Shi, and X. Xiao. Influence maximization in near-linear time: A martingale approach. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1539–1554. ACM, 2015.

[73] Y. S. Thibaut Horel. Maximization of approximately submodular functions. In *NIPS*, 2016.

[74] H. C. White, S. A. Boorman, and R. L. Breiger. Social structure from multiple networks. i. blockmodels of roles and positions. *American Journal of Sociology*, 81(4):730–780, 1976. URL http://www.jstor.org/stable/2777596.

[75] Y. Yang, J. Jia, B. Wu, and J. Tang. Social role-aware emotion contagion in image social networks. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

APPENDIX A

Omitted Proofs in Chapter 4

A.1 Proof of Lemma 4.7

Given a seed s in the complete graph G, we calculate the probability that an arbitrary vertex $v \in V \setminus \{s\}$ is infected according to Remark 4.4. Consider the reverse random walk without repetition starting from v as described in Remark 4.4. It reaches s in one move with probability $\frac{1}{n-1}$, and it reaches s in t moves with probability $\frac{1}{n-1}\prod_{i=1}^{t-1}\frac{n-1-i}{n-1}$ since $\prod_{i=1}^{t-1}\frac{n-1-i}{n-1}$ is the probability that the random walk never reaches s and never comes back to any vertices that have been visited within the first $t-1$ moves and $\frac{1}{n-1}$ is the probability that the random walk moves to s in the t-th move. Putting this together, v will be infected by s with probability

$$\frac{1}{n-1}+\sum_{t=2}^{n-1}\frac{1}{n-1}\prod_{i=1}^{t-1}\frac{n-1-i}{n-1}=\frac{1}{n-1}\sum_{t=1}^{n-1}\prod_{i=0}^{t-1}\frac{n-1-i}{n-1}.$$

Simple calculations reveal an upper bound for this probability.

$$
\begin{aligned}
&\frac{1}{n-1}\sum_{t=1}^{n-1}\prod_{i=0}^{t-1}\frac{n-1-i}{n-1}\\
=&\frac{1}{n-1}\left(\sum_{t=1}^{\lceil\sqrt{n}\rceil}\prod_{i=0}^{t-1}\frac{n-1-i}{n-1}+\sum_{t=\lceil\sqrt{n}\rceil+1}^{n-1}\prod_{i=0}^{t-1}\frac{n-1-i}{n-1}\right)\\
=&\frac{1}{n-1}\left(\sum_{t=1}^{\lceil\sqrt{n}\rceil}\prod_{i=0}^{t-1}\frac{n-1-i}{n-1}+\left(\prod_{i=0}^{\lceil\sqrt{n}\rceil-1}\frac{n-1-i}{n-1}\right)\sum_{t=\lceil\sqrt{n}\rceil+1}^{n-1}\prod_{i=\lceil\sqrt{n}\rceil}^{t-1}\frac{n-1-i}{n-1}\right)
\end{aligned}
$$

(distributive law)

$$
<\frac{1}{n-1}\left(\sum_{t=1}^{\lceil\sqrt{n}\rceil}1+1\cdot\sum_{t=1}^{n-1-\lceil\sqrt{n}\rceil}\left(\frac{n-1-\lceil\sqrt{n}\rceil}{n-1}\right)^{t}\right)
$$

(the first two products are replaced by 1, and $\prod_{i=\lceil\sqrt{n}\rceil}^{t-1}\frac{n-1-i}{n-1}\leq\left(\frac{n-1-\lceil\sqrt{n}\rceil}{n-1}\right)^{t-\lceil\sqrt{n}\rceil}$)

$$
<\frac{1}{n-1}\left(\sum_{t=1}^{\lceil\sqrt{n}\rceil}1+\sum_{t=0}^{\infty}\left(\frac{n-1-\lceil\sqrt{n}\rceil}{n-1}\right)^{t}\right)
$$

(the summation is extended to the infinite series)

$$
=\frac{1}{n-1}\left(\lceil\sqrt{n}\rceil+\frac{n-1}{\lceil\sqrt{n}\rceil}\right).
$$

Finally, by linearity of expectation, the expected total number of infected vertices is

$$
1+(n-1)\cdot\frac{1}{n-1}\left(\lceil\sqrt{n}\rceil+\frac{n-1}{\lceil\sqrt{n}\rceil}\right)<3\sqrt{n},
$$

which concludes the lemma.

A.2 Proof of Theorem 4.8

Before we move on, we examine some of the properties of Example 4.9 which will be used later.

Proposition A.1. *The followings are true.*

1. *$\ell\leq k$;*
2. *$\sigma(\{u_1\})=\cdots=\sigma(\{u_k\})$;*
3. *$\sigma(\{v_1\})\geq\cdots\geq\sigma(\{v_{k+\ell}\})$;*
4. *The greedy algorithm will never pick any vertices in $V\setminus\{u_1,\ldots,u_k,v_1,\ldots,v_k\}$;*
5. *For any $i=1,\ldots,k$ and $j=1,\ldots,k+\ell$, we have $\Pr(u_i\to v_j)<\frac{1}{k}+\frac{3}{k^{1.2}}$;*

6. For any $i, j \in \{1, \dots, k\}$ with $i \neq j$, we have $\Pr(u_i \to u_j) < \frac{2k}{k^{1.2}+2k-1}(\frac{1}{k} + \frac{3}{k^{1.2}})$.

Proof. To show 1, suppose $\ell > k$, we will have

$$\begin{aligned}\sum_{i=1}^{k+\ell}|D_i| &\geq \sum_{i=1}^{k}\left\lceil k^{0.8}\left(1-\frac{1}{k}\right)^{i-1}\right\rceil + \sum_{i=k+1}^{k+\ell-1}\left\lceil k^{0.8}\left(1-\frac{1}{k}\right)^{k}\right\rceil \\ &\geq k^{1.8}\cdot\left(1-\left(1-\frac{1}{k}\right)^{k}\right) + k\cdot k^{0.8}\left(1-\frac{1}{k}\right)^{k} \\ &= k^{1.8} > \left\lfloor\left(1-\frac{100}{k^{0.2}}\right)k^{1.8}\right\rfloor,\end{aligned}$$

which violates our construction.

2 follows immediately by symmetry, and 3 is trivial. As for 4, choosing a seed in $D_i \setminus \{v_i\}$ is clearly sub-optimal, as choose v_i as a seed will make all the remaining vertices in D_i infected with probability 1. Choosing a seed in $C_i \setminus \{u_i\}$ is also sub-optimal. If u_i is not seeded, seeding u_i is clearly better. Otherwise, seeding any vertices from $u_1, \dots, u_{i-1}, u_{i+1}, \dots, u_k$ is better (notice that we have a total of k seeds, so there are unseeded vertices among these). This is due to the submodularity: if a clique C_i already contains a seed, putting another seed in the same clique is no better than putting a seed in a new clique that does not contain a seed yet. Finally, given that the greedy algorithm will choose seeds in $u_1, \dots, u_k, v_1, \dots, v_{k+\ell}$, choosing seeds in $\{v_{k+1}, \dots, v_{k+\ell}\}$ is clearly sub-optimal: we should first seed all of $v_1, \dots, v_k$ before seeding any of $v_{k+1}, \dots, v_{k+\ell}$, but we have a total of only k seeds.

To see 5, consider the reverse random walk starting from v_j. Since $\deg(v_j) = k + |D_j| - 1 > k$, it will reach u_i in one step with probability less than $1/k$. It is easy to see that the walk will never reach u_i if it ever reaches any vertex in $V \setminus \{u_1, \dots, u_k, v_1, \dots, v_{k+\ell}\}$. Therefore, the only possibility of the walk reaching u_i is to alternate between $\{u_1, \dots, u_k\}$ and $\{v_1, \dots, v_{k+\ell}\}$. When it reaches a vertex on the u-side, it will move to a vertex on the v-side with probability $(k+\ell)/(k^{1.2}-1+k+\ell)$. When it reaches a vertex on the v-side, it will move to exactly u_i with probability less than $1/k$, as all vertices in $\{v_1, \dots, v_{k+\ell}\}$ have degrees more than k. If we disregard the scenario where the random walk visits a vertex that has already been visited (which can only increase the probability that the random walk reaches u_i), the random walk reaches u_i at Step 3 with probability less than $\frac{1}{k}\cdot\frac{k+\ell}{k^{1.2}-1+k+\ell}$, it reaches u_i at Step 5 with probability less than $\frac{1}{k}\cdot(\frac{k+\ell}{k^{1.2}-1+k+\ell})^2$, and so on. Putting these analyses together,

$$\begin{aligned}\Pr(u_i \to v_j) < \sum_{t=0}^{\infty}\frac{1}{k}\cdot\left(\frac{k+\ell}{k^{1.2}-1+k+\ell}\right)^t &= \frac{1}{k}\cdot\frac{k^{1.2}-1+k+\ell}{k^{1.2}-1} \\ &\leq \frac{1}{k} + \frac{2k}{k(k^{1.2}-1)} < \frac{1}{k} + \frac{3}{k^{1.2}},\end{aligned}$$

where the penultimate inequality uses property 1.

To see 6, the reverse random walk starting from u_j will reach the v-side with probability $(k+\ell)/(k^{1.2}-1+k+\ell) \leq 2k/(k^{1.2}+2k-1)$ (since $\ell \leq k$ by 1). Noticing this, property 5 and Lemma 4.5 conclude 6 immediately. □

Proposition A.2. *Given G constructed in Example 4.9, the greedy algorithm will iteratively pick $v_1, \ldots, v_k$.*

Proof. By 4 in Proposition A.1, we will only consider seeds in $\{u_1, \ldots, u_k, v_1, \ldots, v_k\}$. We will prove this proposition by induction.

For the base step, since choosing v_1 is more beneficial than choosing any of $v_2, \ldots, v_k$, we only need to compare $\sigma(\{v_1\})$ to each of $\sigma(\{u_1\}), \ldots, \sigma(\{u_k\})$. Since $\sigma(\{u_1\}) = \cdots = \sigma(\{u_k\})$, we consider $\sigma(\{u_1\})$ without loss of generality. We aim to find an upper bound for $\sigma(\{u_1\})$ by upper-bounding the probability that each vertex in the graph is infected given a single seed u_1.

Firstly, the expected number of infected vertices in C_1 is at most $3k^{0.6}$ by Lemma 4.7. Next, by 6 in Proposition A.1, each of $u_2, \ldots, u_k$ will be infected with probability less than $\Pr(u_i \to u_j) < \frac{2k}{k^{1.2}+2k-1}(\frac{1}{k}+\frac{3}{k^{1.2}})$. Moreover, if u_i is not infected, all the remaining vertices in C_i will not be infected. If u_i is infected, the total number of infected vertices in C_i is at most $3k^{0.6}$ by Lemma 4.7. Finally, each vertex $v_1, \ldots, v_k$ will be infected with probability less than $\frac{1}{k}+\frac{3}{k^{1.2}}$ by 5 of Proposition A.1. In addition, if certain v_i is infected, then all vertices in D_i will be infected. Putting together, we have

$$\begin{aligned}
&\sigma\left(\{u_1\}\right) \\
\leq & 3k^{0.6} + (k-1) \cdot \frac{2k}{k^{1.2}+2k-1}\left(\frac{1}{k}+\frac{3}{k^{1.2}}\right) \cdot 3k^{0.6} + \left(\frac{1}{k}+\frac{3}{k^{1.2}}\right)\left\lfloor \left(1-\frac{100}{k^{0.2}}\right)k^{1.8} \right\rfloor \qquad (\dagger) \\
\leq & 3k^{0.6} + k \cdot \frac{2k}{k^{1.2}}\frac{2}{k} \cdot 3k^{0.6} + \frac{1}{k}\left(1-\frac{100}{k^{0.2}}\right)k^{1.8} + \frac{3}{k^{1.2}}\left(1-\frac{100}{k^{0.2}}\right)k^{1.8} \\
< & 3k^{0.6} + 12k^{0.4} + k^{0.8} - 100k^{0.6} + 3k^{0.6} \qquad (\ddagger) \\
< & k^{0.8}.
\end{aligned}$$

On the other hand, we have $\sigma(\{v_1\}) \geq |D_1| = \lceil k^{0.8} \rceil > \sigma(\{u_1\})$. Therefore, the first seed that the greedy algorithm will pick is v_1, which concludes the base step of the induction.

For the inductive step, suppose $v_1, \ldots, v_t$ have been chosen by the greedy algorithm in the first t iterations. We aim to show that the greedy algorithm will pick v_{t+1} next. By symmetry, with $v_1, \ldots, v_t$ being seeded, the marginal increment of $\sigma(\cdot)$ by seeding each of $u_1, \ldots, u_k$ is the same. Thus, we only need to show that $\sigma(\{v_1, \ldots, v_{t+1}\}) - \sigma(\{v_1, \ldots, v_t\}) > \sigma(\{v_1, \ldots, v_t, u_1\}) - \sigma(\{v_1, \ldots, v_t\})$.

To calculate a lower bound for $\sigma(\{v_1, \ldots, v_{t+1}\}) - \sigma(\{v_1, \ldots, v_t\})$, we first evaluate the probability $\Pr(\{v_1, \ldots, v_t\} \to v_{t+1})$. In order for the reverse random walk starting from v_{t+1} to reach one of $v_1, \ldots, v_t$, it must reach one of $u_1, \ldots, u_k$ in the first step, and then "escape" from the clique in the second step. The probability that the walk escapes from the clique, $(k+\ell)/(k^{1.2}-1+k+\ell)$, is clearly an upper bound of $\Pr(\{v_1, \ldots, v_t\} \to v_{t+1})$. Therefore, with seeds $v_1, \ldots, v_t$, the expected number of infected vertices in D_{t+1} is at most $(k+\ell)/(k^{1.2}-1+k+\ell)\times|D_{t+1}|$. On the other hand, when v_{t+1} is further seeded, all vertices in D_{t+1} will be infected. By only considering the marginal gain on the expected number of infected vertices in D_{t+1}, we have

$$\sigma(\{v_1,\ldots,v_{t+1}\})-\sigma(\{v_1,\ldots,v_t\})>\left(1-\frac{k+\ell}{k^{1.2}-1+k+\ell}\right)\cdot\left\lceil k^{0.8}\left(1-\frac{1}{k}\right)^t\right\rceil$$

$$>\left(1-\frac{2k}{k^{1.2}}\right)k^{0.8}\left(1-\frac{1}{k}\right)^t>k^{0.8}\left(1-\frac{1}{k}\right)^t-2k^{0.6}.$$

To find an upper bound for $\sigma(\{v_1,\ldots,v_t,u_1\})-\sigma(\{v_1,\ldots,v_t\})$. We note that all vertices in $D_1,\ldots,D_t$ are infected with probability 1 with seeds $v_1,\ldots,v_t$, and we have

$$\begin{aligned}&\sigma(\{v_1,\ldots,v_t,u_1\})-\sigma(\{v_1,\ldots,v_t\})\\=&\sum_{w\in V\setminus(D_1\cup\cdots\cup D_t)}\left(\Pr(\{v_1,\ldots,v_t,u_1\}\to w)-\Pr(\{v_1,\ldots,v_t\}\to w)\right)\\<&\sum_{w\in V\setminus(D_1\cup\cdots\cup D_t)}\Pr(\{u_1\}\to w), \qquad\qquad \text{(By submodularity)}\end{aligned}$$

so we only need to consider the expected number of infected vertices with the graph containing only one seed u_1 and with vertices in $D_1\cup\cdots\cup D_t$ disregarded. Therefore, if we split $\sigma(\{v_1,\ldots,v_t,u_1\})-\sigma(\{v_1,\ldots,v_t\})$ into three terms as it is in (†), the first two terms regarding the expected number of infections on the k cliques are the same as they appeared in (†), which are less than $3k^{0.6}+12k^{0.4}$ as computed at step (‡).

By excluding $D_1, \ldots, D_t$ for the third term, we have

$$\begin{aligned}
&\sigma(\{v_1, \ldots, v_t, u_1\}) - \sigma(\{v_1, \ldots, v_t\})\\
<&3k^{0.6} + 12k^{0.4} + \left(\frac{1}{k} + \frac{3}{k^{1.2}}\right)\left(\left\lfloor \left(1 - \frac{100}{k^{0.2}}\right) k^{1.8} \right\rfloor - \sum_{i=1}^{t} \left\lceil k^{0.8}\left(1 - \frac{1}{k}\right)^i \right\rceil\right)\\
\leq&3k^{0.6} + 12k^{0.4} + \left(\frac{1}{k} + \frac{3}{k^{1.2}}\right)\left(\left(1 - \frac{100}{k^{0.2}}\right) k^{1.8} - \sum_{i=1}^{t} k^{0.8}\left(1 - \frac{1}{k}\right)^i\right)\\
\leq&3k^{0.6} + 12k^{0.4} + k^{0.8}\left(1 - \frac{100}{k^{0.2}}\right) - k^{0.8}\left(1 - \left(1 - \frac{1}{k}\right)^t\right)\\
&\quad + \frac{3}{k^{1.2}} \cdot \left(\left(1 - \frac{100}{k^{0.2}}\right) k^{1.8} - \sum_{i=1}^{t} k^{0.8}\left(1 - \frac{1}{k}\right)^i\right)\\
&\hfill \left(\text{since } \textstyle\sum_{i=1}^{t} k^{0.8}\left(1 - \frac{1}{k}\right)^i = k^{1.8}\left(1 - \frac{1}{k}\right)^t\right)\\
\leq&3k^{0.6} + 12k^{0.4} + k^{0.8}\left(1 - \frac{1}{k}\right)^t - 100k^{0.6} + \frac{3}{k^{1.2}} \cdot k^{1.8}\\
&\hfill \left(\text{since } \left(\left(1 - \tfrac{100}{k^{0.2}}\right) k^{1.8} - \textstyle\sum_{i=1}^{t} k^{0.8}\left(1 - \frac{1}{k}\right)^i\right) \leq k^{1.8}\right)\\
<&k^{0.8}\left(1 - \frac{1}{k}\right)^t - 50k^{0.6}\\
<&\sigma(\{v_1, \ldots, v_{t+1}\}) - \sigma(\{v_1, \ldots, v_t\}),
\end{aligned}$$

which concludes the inductive step. □

We are now ready to prove Theorem 4.8. Let $S = \{v_1, \ldots, v_k\}$ be the set of seeds selected by the greedy algorithm, and let $S^* = \{u_1, \ldots, u_k\}$ be the optimal seeds (we do not even need to show that this is optimal: if there were seeds with quality better than S^*, the approximation guarantee is even smaller than $\frac{\sigma(S)}{\sigma(S^*)}$).

By only considering infected vertices in $D_1, \ldots, D_{k+\ell}$, we have

$$\sigma(S^*) > \frac{k}{k^{0.8} + k}\left\lfloor \left(1 - \frac{100}{k^{0.2}}\right) k^{1.8} \right\rfloor > \frac{k}{k^{0.8} + k}\left(k^{1.8} - 50k^{1.6}\right),$$

since each of $v_1, \ldots, v_{k+\ell}$ will be infected with probability at least $\frac{k}{k^{0.8}+k}$ (notice that even v_1, with the highest degree among $v_1, \ldots, v_{k+\ell}$, has degree only $k^{0.8} + k - 1$).

Now consider $\sigma(S)$. Given seed set S, each of $u_1, \ldots, u_k$ will be infected with probability $\frac{k}{k^{1.2}+k-1}$, and each of $v_{k+1}, \ldots, v_{k+\ell}$ will be infected with probability at most $\frac{k}{k^{1.2}+k-1}$, as the reverse random walk starting from any of $v_{k+1}, \ldots, v_{k+\ell}$ needs to reach one of $u_1, \dots, u_k$ before reaching a seed in S. Therefore,

$$\begin{aligned}
&\sigma(S) \\
\leq &\sum_{i=1}^{k} \left\lceil k^{0.8}\left(1-\frac{1}{k}\right)^i \right\rceil + k \cdot \frac{k}{k^{1.2}+k-1} \cdot 3k^{0.6} + \ell \cdot \frac{k}{k^{1.2}+k-1} \left\lceil k^{0.8}\left(1-\frac{1}{k}\right)^k \right\rceil \\
\leq &\sum_{i=1}^{k} k^{0.8}\left(1-\frac{1}{k}\right)^i + k + \frac{3k^{2.6}}{k^{1.2}} + k \cdot \frac{k}{k^{1.2}} k^{0.8} \qquad \text{(since } \lceil x \rceil \leq x+1 \text{ and } \ell \leq k\text{)} \\
= &k^{1.8}\left(1-\left(1-\frac{1}{k}\right)^k\right) + k + 3k^{1.4} + k^{1.6}.
\end{aligned}$$

Finally, the approximation guarantee of the greedy algorithm on the instance described in Example 4.9 is at most

$$\begin{aligned}
\frac{\sigma(S)}{\sigma(S^*)} &\leq \frac{k^{1.8}\left(1-\left(1-\frac{1}{k}\right)^k\right) + k + 3k^{1.4} + k^{1.6}}{\frac{k}{k^{0.8}+k}\left(k^{1.8} - 50k^{1.6}\right)} \\
&\leq \left(1-\left(1-\frac{1}{k}\right)^k\right) \frac{k^{1.8}(k^{0.8}+k)}{k(k^{1.8}-50k^{1.6})} + \frac{k+3k^{1.4}+k^{1.6}}{\frac{1}{2} \times \frac{1}{2}k^{1.8}} \\
&\qquad\qquad\qquad\qquad \text{(since } \tfrac{k}{k^{0.8}+k} > \tfrac{1}{2} \text{ and } 50k^{1.6} \ll \tfrac{1}{2}k^{1.8}\text{)} \\
&= \left(1-\left(1-\frac{1}{k}\right)^k\right)\left(1+\frac{51k^{0.8}}{k-50k^{0.8}}\right) + \frac{4+12k^{0.4}+4k^{0.6}}{k^{0.8}} \\
&\leq \left(1-\left(1-\frac{1}{k}\right)^k\right) + O\left(\frac{1}{k^{0.2}}\right),
\end{aligned}$$

which concludes Theorem 4.8.

A.3 Proofs of Properties of Max-k-Coverage

We prove all the lemmas in Sect. 4.4.1 here. Notice that the lemmas are restated for the ease of reading. Again, for all the lemmas in this section, we are considering a Max-k-Coverage instance $(U, \mathcal{M}, k)$ where $\mathcal{S} = \{S_1, \dots, S_k\}$ denotes k subsets output by the greedy algorithm and $\mathcal{S}^* = \{S_1^*, \dots, S_k^*\}$ denotes the optimal solution.

We first define a useful notion called a Max-k-Coverage instance *with restriction*.

Definition A.3. Given a Max-k-Coverage instance $(U, \mathcal{M}, k)$ and a subset $U' \subseteq U$, the Max-k-Coverage instance $(U, \mathcal{M}, k)$ *with restriction on* U' is another Max-k-Coverage instance $(U', \mathcal{M}', k')$ where $\mathcal{M}' = \{S \cap U' : S \in \mathcal{M}\}$.

We begin by proving the following lemma, which compares the k subsets output by the greedy algorithm with arbitrary ℓ subsets. This is a more general statement than saying that the greedy algorithm always achieves a $(1-(1-1/k)^k)$-approximation.

Lemma A.4. *Given a* Max-k-Coverage *instance* $(U, \mathcal{M}, k)$*, let* $\mathcal{S}'$ *be an arbitrary collection of* ℓ *subsets, we have* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - 1/\ell)^k) \operatorname{val}(\mathcal{S}')$.

Proof. Fix an arbitrary ℓ, we prove this lemma by induction on k. To prove the base step for $k = 1$, the subset in $\mathcal{S}'$ with the largest size covers at least $\frac{1}{\ell} \operatorname{val}(\mathcal{S}')$ elements, so the first subset picked by the greedy algorithm should cover at least $\frac{1}{\ell} \operatorname{val}(\mathcal{S}')$ elements. Thus, for $k = 1$, $\operatorname{val}(\mathcal{S}) \geq (1/\ell) \operatorname{val}(\mathcal{S}') = (1 - (1 - 1/\ell)^k) \operatorname{val}(\mathcal{S}')$.

For the inductive step, suppose this lemma holds for $k = k_0$, we aim to show that it holds for $k = k_0 + 1$. Let $\mathcal{S} = \{S_1, \ldots, S_{k_0+1}\}$ be the output of the greedy algorithm. By the same analysis above, $|S_1| \geq \frac{1}{\ell} \operatorname{val}(\mathcal{S}')$. Consider the Max-k-Coverage instance $(U' = U \setminus S_1, \mathcal{M}', k_0)$ which is the instance $(U, \mathcal{M}, k_0 + 1)$ with restriction on $U \setminus S_1$. Since the greedy algorithm selects subsets based on *marginal* increments to $\operatorname{val}(\cdot)$, $(S_2 \setminus S_1), \ldots, (S_{k_0+1} \setminus S_1)$ will also be the k_0 subsets picked by the greedy algorithm on the restricted instance. By the induction hypothesis, we have

$$\left| \left(\bigcup_{i=2}^{k_0+1} S_i \right) \setminus S_1 \right| = \left| \bigcup_{i=2}^{k_0+1} (S_i \setminus S_1) \right| \geq \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0} \right) \left| \bigcup_{S \in \mathcal{S}'} (S \setminus S_1) \right|$$

$$= \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0} \right) \left| \left(\bigcup_{S \in \mathcal{S}'} S \right) \setminus S_1 \right|.$$

We then discuss two different cases.

If $|S_1 \cap (\bigcup_{S \in \mathcal{S}'} S)| \leq \frac{1}{\ell} |\bigcup_{S \in \mathcal{S}'} S| = \frac{1}{\ell} \operatorname{val}(\mathcal{S}')$, then $|(\bigcup_{S \in \mathcal{S}'} S) \setminus S_1| \geq \frac{\ell-1}{\ell} \operatorname{val}(\mathcal{S}')$ and

$$\operatorname{val}(\mathcal{S}) = |S_1| + \left| \left(\bigcup_{i=2}^{k_0+1} S_i \right) \setminus S_1 \right| \geq \frac{1}{\ell} \operatorname{val}(\mathcal{S}') + \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0} \right) \cdot \frac{\ell - 1}{\ell} \operatorname{val}(\mathcal{S}')$$

$$= \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0+1} \right) \operatorname{val}(\mathcal{S}'),$$

which concludes the inductive step.

If $|S_1 \cap (\bigcup_{S \in \mathcal{S}'} S)| > \frac{1}{\ell} |\bigcup_{S \in \mathcal{S}'} S| = \frac{1}{\ell} \operatorname{val}(\mathcal{S}')$, let $|S_1 \cap (\bigcup_{S \in \mathcal{S}'} S)| = (\frac{1}{\ell} + c) \operatorname{val}(\mathcal{S}')$

for some $c \in (0, 1 - \frac{1}{\ell}]$, and we have

$$\begin{aligned}
\operatorname{val}(\mathcal{S}) =& |S_1| + \left| \left(\bigcup_{i=2}^{k_0+1} S_i \right) \setminus S_1 \right| \\
\geq& \left(\frac{1}{\ell} + c \right) \operatorname{val}(\mathcal{S}') + \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0} \right) \cdot \left(1 - \frac{1}{\ell} - c \right) \operatorname{val}(\mathcal{S}') \\
=& \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0+1} + c \left(1 - \frac{1}{\ell} \right)^{k_0} \right) \operatorname{val}(\mathcal{S}') \\
>& \left(1 - \left(1 - \frac{1}{\ell} \right)^{k_0+1} \right) \operatorname{val}(\mathcal{S}'),
\end{aligned}$$

which concludes the inductive step as well. □

The lemma below shows that, if the first subset picked by the greedy algorithm is one of the subsets in the optimal solution, then the barrier $1 - (1 - 1/k)^k$ can be overcome.

Lemma 4.11. *If* $S_1 \in \mathcal{S}^*$, *then* $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - \frac{1}{k})^k + \frac{1}{4k^2}) \operatorname{val}(\mathcal{S}^*)$.

Proof. Assume $S_1 = S_1^*$ without loss of generality. In order to be picked by the greedy algorithm, S_1^* should also be the subset in $\mathcal{S}^*$ with the largest size. Therefore, $|S_1| = |S_1 \cap (\bigcup_{i=1}^k S_i^*)| = (\frac{1}{k} + c) \operatorname{val}(\mathcal{S}^*)$ for some $c \geq 0$, and $|(\bigcup_{i=2}^k S_i^*) \setminus S_1| = |(\bigcup_{i=1}^k S_i^*) \setminus S_1| = (1 - \frac{1}{k} - c) \operatorname{val}(\mathcal{S}^*)$. By applying Lemma A.4 on the instance with restriction $U \setminus S_1$, we have $|(\bigcup_{i=2}^k S_i) \setminus S_1| \geq (1 - (1 - \frac{1}{k-1})^{k-1}) |(\bigcup_{i=2}^k S_i^*) \setminus S_1| = (1 - (1 - \frac{1}{k-1})^{k-1})(1 - \frac{1}{k} - c) \operatorname{val}(\mathcal{S}^*)$. Putting together,

$$\begin{aligned}
\operatorname{val}(\mathcal{S}) &= |S_1| + \left| \left(\bigcup_{i=2}^{k} S_i \right) \setminus S_1 \right| \\
&\geq \left(\frac{1}{k} + c \right) \operatorname{val}(\mathcal{S}^*) + \left(1 - \left(1 - \frac{1}{k-1} \right)^{k-1} \right) \left(1 - \frac{1}{k} - c \right) \operatorname{val}(\mathcal{S}^*) \\
&= \left(1 - \left(1 - \frac{1}{k} \right) \left(1 - \frac{1}{k-1} \right)^{k-1} + c \left(1 - \frac{1}{k-1} \right)^{k-1} \right) \operatorname{val}(\mathcal{S}^*) \\
&\geq \left(1 - \left(1 - \frac{1}{k} \right) \left(1 - \frac{1}{k} \right)^{k-1} \left(1 - \frac{1}{(k-1)^2} \right)^{k-1} \right) \operatorname{val}(\mathcal{S}^*) \\
&\qquad \text{(since } \left(1 - \tfrac{1}{k}\right) \left(1 - \tfrac{1}{(k-1)^2}\right) = 1 - \tfrac{1}{k-1} \text{ and } c \left(1 - \tfrac{1}{k-1}\right)^{k-1} \geq 0\text{)} \\
&\geq \left(1 - \left(1 - \frac{1}{k} \right)^{k} \left(1 - \frac{1}{(k-1)^2} \right) \right) \operatorname{val}(\mathcal{S}^*) \\
&\geq \left(1 - \left(1 - \frac{1}{k} \right)^{k} + \frac{1}{(k-1)^2} \left(1 - \frac{1}{k} \right)^{k} \right) \operatorname{val}(\mathcal{S}^*).
\end{aligned}$$

The lemma follows from noticing $(1-\frac{1}{k})^k \geq \frac{1}{4}$ and $\frac{1}{k-1} > \frac{1}{k}$. □

Next, we show that, in order to have the tight approximation guarantee $1-(1-1/k)^k$, the first subset picked by the greedy algorithm must intersect almost exactly $1/k$ fraction of the elements covered by the k optimal subsets.

Lemma 4.12. *If* $\frac{|S_1\cap(\bigcup_{i=1}^k S_i^*)|}{\mathrm{val}(\mathcal{S}^*)} \notin [\frac{1}{k}-\varepsilon, \frac{1}{k}+\varepsilon]$ *for some* $\varepsilon > 0$ *which may depend on* k*, then* $\mathrm{val}(\mathcal{S}) \geq (1-(1-1/k)^k+\varepsilon/4)\,\mathrm{val}(\mathcal{S}^*)$.

Proof. By the same argument in the first paragraph of the proof of Lemma A.4, we have $|S_1| \geq \frac{1}{k}\,\mathrm{val}(\mathcal{S}^*)$. On the other hand, considering the instance with restriction on $U \setminus S_1$, the greedy algorithm, picking subsets based on marginal increments, will pick $(S_2\setminus S_1), \ldots, (S_k\setminus S_1)$ as the first $k-1$ seeds in the restricted instance. Applying Lemma A.4, we have $|(\bigcup_{i=2}^k S_i)\setminus S_1| \geq (1-(1-1/k)^{k-1})|(\bigcup_{i=1}^k S_i^*)\setminus S_1|$.

If $\frac{|S_1\cap(\bigcup_{i=1}^k S_i^*)|}{\mathrm{val}(\mathcal{S}^*)} > \frac{1}{k}+\varepsilon$, let $\frac{|S_1\cap(\bigcup_{i=1}^k S_i^*)|}{\mathrm{val}(\mathcal{S}^*)} = \frac{1}{k}+c$ where $c > \varepsilon$. The last paragraph of the proof of Lemma A.4 can be applied here, and we have

$$\mathrm{val}(\mathcal{S}) \geq \left(\frac{1}{k}+c\right)\mathrm{val}(\mathcal{S}^*) + \left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)\left(1-\frac{1}{k}-c\right)\mathrm{val}(\mathcal{S}^*)$$

$$= \left(1-\left(1-\frac{1}{k}\right)^k + c\left(1-\frac{1}{k}\right)^{k-1}\right)\mathrm{val}(\mathcal{S}^*) > \left(1-\left(1-\frac{1}{k}\right)^k+\frac{\varepsilon}{4}\right)\mathrm{val}(\mathcal{S}^*),$$

since $(1-\frac{1}{k})^{k-1} > \frac{1}{4}$ and $c > \varepsilon$.

If $\frac{|S_1\cap(\bigcup_{i=1}^k S_i^*)|}{\mathrm{val}(\mathcal{S}^*)} < \frac{1}{k}-\varepsilon$, let $\frac{|S_1\cap(\bigcup_{i=1}^k S_i^*)|}{\mathrm{val}(\mathcal{S}^*)} = \frac{1}{k}-c$ where $c \in (\varepsilon, \frac{1}{k})$. We have

$$\left|\left(\bigcup_{i=2}^k S_i\right)\setminus S_1\right| \geq \left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)\left|\left(\bigcup_{i=1}^k S_i^*\right)\setminus S_1\right|$$

$$= \left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)\left(1-\frac{1}{k}+c\right)\mathrm{val}(\mathcal{S}^*).$$

Adding S_1, we have

$$\begin{aligned}\mathrm{val}(\mathcal{S}) =&|S_1| + \left|\left(\bigcup_{i=2}^k S_i\right)\setminus S_1\right|\\ \geq&\frac{1}{k}\,\mathrm{val}(\mathcal{S}^*) + \left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)\left(1-\frac{1}{k}+c\right)\mathrm{val}(\mathcal{S}^*)\\ =&\left(1-\left(1-\frac{1}{k}\right)^k + c\left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)\right)\mathrm{val}(\mathcal{S}^*)\\ >&\left(1-\left(1-\frac{1}{k}\right)^k+\frac{\varepsilon}{4}\right)\mathrm{val}(\mathcal{S}^*),\end{aligned}$$

since $1-(1-\frac{1}{k})^{k-1} > \frac{1}{4}$ (this holds $k \geq 2$; if $k=1$, the premise of the lemma will not hold as we will then have $S^* = \{S_1\}$) and $c > \varepsilon$. □

The next lemma shows that, in order to have the tight approximation guarantee $1-(1-1/k)^k$, the first subset output by the greedy algorithm must not cover a number of elements that is significantly more than $1/k$ fraction of the number of elements in the optimal solution.

Lemma A.5. *If* $|S_1| \geq (\frac{1}{k}+\varepsilon)\operatorname{val}(\mathcal{S}^*)$ *for some* $\varepsilon > 0$ *which may depend on* k*, then* $\operatorname{val}(\mathcal{S}) \geq (1-(1-1/k)^k + \varepsilon/8)\operatorname{val}(\mathcal{S}^*)$.

Proof. If $|S_1 \cap (\bigcup_{i=1}^k S_i^*)|/\operatorname{val}(\mathcal{S}^*) \notin [\frac{1}{k}-\frac{\varepsilon}{2}, \frac{1}{k}+\frac{\varepsilon}{2}]$, Lemma 4.12 directly implies this lemma. Suppose $|S_1 \cap (\bigcup_{i=1}^k S_i^*)|/\operatorname{val}(\mathcal{S}^*) \in [\frac{1}{k}-\frac{\varepsilon}{2}, \frac{1}{k}+\frac{\varepsilon}{2}]$. Since $|S_1| \geq (\frac{1}{k}+\varepsilon)\operatorname{val}(\mathcal{S}^*)$, we have $|S_1 \setminus (\bigcup_{i=1}^k S_i^*)| > \frac{\varepsilon}{2}\operatorname{val}(\mathcal{S}^*)$. Let $|S_1 \cap (\bigcup_{i=1}^k S_i^*)| = (\frac{1}{k}+c)\operatorname{val}(\mathcal{S}^*)$ where $c \in [-\frac{\varepsilon}{2}, \frac{\varepsilon}{2}]$. By the same analysis in the last paragraph of the proof of Lemma 4.12 (which uses Lemma A.4 as well),

$$\begin{aligned}
\operatorname{val}(\mathcal{S}) =& |S_1| + \left|\left(\bigcup_{i=2}^k S_i\right) \setminus S_1\right| \\
\geq& \left(\frac{1}{k}+\varepsilon\right)\operatorname{val}(\mathcal{S}^*) + \left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)\left(1-\frac{1}{k}+c\right)\operatorname{val}(\mathcal{S}^*) \\
=& \left(1-\left(1-\frac{1}{k}\right)^k + c\left(1-\left(1-\frac{1}{k}\right)^{k-1}\right)+\varepsilon\right)\operatorname{val}(\mathcal{S}^*) \\
>& \left(1-\left(1-\frac{1}{k}\right)^k + \frac{\varepsilon}{8}\right)\operatorname{val}(\mathcal{S}^*).
\end{aligned}$$

For the last inequality, it holds trivially if $c \geq 0$, and it holds for $c < 0$ as $c(1-(1-\frac{1}{k})^{k-1}) + \varepsilon > c + \varepsilon \geq \frac{\varepsilon}{2}$. □

The next two lemmas show that, in order to have the tight approximation guarantee $1-(1-1/k)^k$, those optimal subsets must be almost disjoint and the first subset output by the greedy algorithm must not cover too many elements that are not covered by the optimal subsets.

Lemma 4.13. *If* $\sum_{i=1}^k |S_i^*| > (1+\varepsilon)\operatorname{val}(\mathcal{S}^*)$ *for some* $\varepsilon > 0$ *which may depend on* k*, then* $\operatorname{val}(\mathcal{S}) \geq (1-(1-\frac{1}{k})^k + \frac{\varepsilon}{8k})\operatorname{val}(\mathcal{S}^*)$.

Proof. If $\sum_{i=1}^k |S_i^*| > (1+\varepsilon)\operatorname{val}(\mathcal{S}^*)$, the subset in $\mathcal{S}^*$ with the largest size should contain more than $\frac{1+\varepsilon}{k}\operatorname{val}(\mathcal{S}^*)$ elements, implying that $|S_1| > \frac{1+\varepsilon}{k}\operatorname{val}(\mathcal{S}^*)$. Lemma A.5 then implies that $\operatorname{val}(\mathcal{S}) \geq (1-(1-\frac{1}{k})^k + \frac{\varepsilon}{8k})\operatorname{val}(\mathcal{S}^*)$. □

Lemma 4.14. *If* $|S_1 \setminus (\bigcup_{i=1}^k S_i^*)| > \varepsilon \operatorname{val}(\mathcal{S}^*)$ *for some* $\varepsilon > 0$ *which may depend on* k*, then* $\operatorname{val}(\mathcal{S}) \geq (1-(1-1/k)^k + \varepsilon/16)\operatorname{val}(\mathcal{S}^*)$.

Proof. If $|S_1 \cap (\bigcup_{i=1}^k S_i^*)| < (\frac{1}{k} - \frac{\varepsilon}{2}) \operatorname{val}(\mathcal{S}^*)$, Lemma 4.12 implies this lemma. Otherwise, we have

$$\begin{aligned}|S_1| = \left|S_1 \setminus \left(\bigcup_{i=1}^k S_i^*\right)\right| + \left|S_1 \cap \left(\bigcup_{i=1}^k S_i^*\right)\right| &\geq \varepsilon \operatorname{val}(\mathcal{S}^*) + \left(\frac{1}{k} - \frac{\varepsilon}{2}\right) \operatorname{val}(\mathcal{S}^*) \\ &= \left(\frac{1}{k} + \frac{\varepsilon}{2}\right) \operatorname{val}(\mathcal{S}^*),\end{aligned}$$

and Lemma A.5 implies this lemma. □

Finally, the lemma below shows that, in order to have the tight approximation guarantee $1 - (1 - 1/k)^k$, those subsets in the optimal solution must have about the same size.

Lemma 4.15. *If there exists $S_i^* \in \mathcal{S}^*$ such that $|S_i^*| < (\frac{1}{k} - \varepsilon) \operatorname{val}(\mathcal{S}^*)$ for some $\varepsilon > 0$ which may depend on k, then $\operatorname{val}(\mathcal{S}) \geq (1 - (1 - \frac{1}{k})^k + \frac{\varepsilon}{8k}) \operatorname{val}(\mathcal{S}^*)$.*

Proof. Assume $|S_1^*| < (\frac{1}{k} - \varepsilon) \operatorname{val}(\mathcal{S}^*) \leq (\frac{1}{k} - \varepsilon) \sum_{i=1}^k |S_i^*|$ without loss of generality. We have

$$\sum_{i=2}^k |S_i^*| > \left(\frac{k-1}{k} + \varepsilon\right) \sum_{i=1}^k |S_i^*| \geq \left(\frac{k-1}{k} + \varepsilon\right) \operatorname{val}(\mathcal{S}^*).$$

Therefore,

$$\max_{2 \leq i \leq k} |S_i^*| > \left(\frac{1}{k} + \frac{\varepsilon}{k-1}\right) \operatorname{val}(\mathcal{S}^*) > \left(\frac{1}{k} + \frac{\varepsilon}{k}\right) \operatorname{val}(\mathcal{S}^*).$$

By the nature of the greedy algorithm,

$$|S_1| \geq \max_{2 \leq i \leq k} |S_i^*| > \left(\frac{1}{k} + \frac{\varepsilon}{k}\right) \operatorname{val}(\mathcal{S}^*),$$

and Lemma A.5 implies this lemma. □

We remark that we only include those properties that are useful in our analysis, while there are some other important properties for Max-k-Coverage that are not listed here.

A.4 Alternative Models for Linear Threshold Model on Undirected Graphs

As mentioned in the last subsection of Sect. 4.5, we will discuss alternative or more general ways to define LTM on undirected graphs, and discuss whether our results in Sect. 4.3 and Sect. 4.4 extend to those new settings.

Weighted undirected graphs with symmetric weights A seemingly natural way to define LTM on undirected edge-weighted graphs is to define edge-weighted undirected graphs such that the weights satisfy the constraints that, 1) for each vertex v, $\sum_{u\in\Gamma(v)} w(u,v) \leq 1$ (as it is in LTM for general directed graphs), and 2) $w(u,v) = w(v,u)$ for any pair $\{u,v\}$ (so that the graph is undirected). However, this model is unnatural in reality, because it disallows the case that a popular vertex exercises significant influence over many somewhat lonely vertices. Consider an extreme example where the graph is a star, with a center u and n leaves $v_1,\ldots,v_n$. The constraint $\sum_{i=1}^{n} w(v_i,u) \leq 1$ implies that there exists at least one v_i such that $w(v_i,u) \leq \frac{1}{n}$, and furthermore, $w(u,v_i) = w(v_i,u) \leq \frac{1}{n}$. In this case, even if u is the only neighbor of v_i, u still has very limited influence to v_i just because u has a lot of other neighbors. In reality, it is unnatural to assume that a node's being popular reduces its influence to its neighbors.

The LTM constraint $\sum_{u\in\Gamma(v)} w(u,v) \leq 1$ makes the above model with symmetrically weighted graphs unnatural. Moreover, this constrains is particular to LTM. For ICM which does not have this constraint, it is much more natural to consider graphs with symmetric edge weights $\forall\{u,v\}: w(u,v) = w(v,u)$, and this is indeed the model studied most often in the past literature, including Khanna and Lucier's work [47].

Weighted undirected graphs with normalization A more natural way to define LTM on graphs that are both edge-weighted and undirected is to start with an edge-weighted undirected graph $G = (V,E,w')$ without any constraint and then normalize the weight of each edge (u,v) such that $w(u,v) = \frac{w'(u,v)}{\sum_{u'\in\Gamma(v)} w'(u',v)}$ and $w(v,u) = \frac{w'(u,v)}{\sum_{v'\in\Gamma(u)} w'(u,v')}$, as mentioned in the last subsection of Sect. 4.5. After normalization, we have, for each $v\in V$, $\sum_{u\in\Gamma(v)} w(u,v) = 1$, so this is a valid linear threshold model. Notice that, after the normalization, the weights of the two anti-parallel directed edges (u,v) and (v,u) may be different. Even though they had the same weight before the normalization (to maintain the undirected feature). In the corresponding live-edge interpretation, each v chooses one of its incoming edges to be "live" with probability proportional to the edge-weights (instead of choosing one uniformly at random as in Theorem 4.2).

Theorem 4.8 holds naturally under this more generalized model. However, Theorem 4.10 no longer holds, and the barrier $1-(1-1/k)^k$ is tight even up to lower order terms: for any positive function $f(k)$ which may be infinitesimal, there is always an example where the greedy algorithm achieves less than a $(1-(1-1/k)^k+f(k))$-approximation. Example 4.9 can be easily adapted to show this. Let $m \gg k$ be a sufficiently large number that is divisible by k^k such that both $m^{0.1}$ and $\sqrt{m}$ are integers. Increase the size of $C_1,\ldots,C_k$ to $m^{0.1}$. Increase the sizes of the stars such that $|D_i| = m(1-\frac{1}{k})^{i-1}$ for $i=1,\ldots,k$ and $|D_{k+1}| = \cdots = |D_{k+\ell-1}| = m(1-\frac{1}{k})^k$, where ℓ and $|D_{k+\ell}|$ are set such that $\sum_{i=1}^{k+\ell} |D_i| = km - k\sqrt{m}$. Set the weights of the edges in each D_i to be extremely small, say $1/m^{100}$, and set the weights of the remaining edges to be 1. After normalizing the weights, the weight of each edge connecting v_i to each of the remaining vertices in D_i is still 1, the weight of each edge (u_i,v_j) (for

$i = 1, \ldots, k$ and $j = 1, \ldots, k+\ell$) becomes $\frac{1}{k+(|D_j|-1)/m^{100}} \approx \frac{1}{k}$, the weight of each edge (v_j, u_i) (again for $i = 1, \ldots, k$ and $j = 1, \ldots, k+\ell$) becomes $\frac{1}{k+\ell+|C_i|-1} = \Theta(\frac{1}{m})$ which can be made much smaller than $f(k)$. By a similar argument, the greedy algorithm will choose $\{v_1, \ldots, v_k\}$, while the optimal seed set is $\{u_1, \ldots, u_k\}$. We have $\frac{\sigma(S)}{\sigma(S^*)} \leq \frac{mk(1-(1-1/k)^k)+o(m^{0.1})}{km-k\sqrt{m}}$, which can be less than $(1-(1-1/k)^k + f(k))$ for sufficiently large m.

Lemma 4.16 and Lemma 4.17 rely crucially on the fact that each vertex v should choose its incoming live edge *uniformly at random*, and Lemma 4.18 also relies on this. This explains why the proof of Theorem 4.10 fails to work for this edge-weighted setting.

Unweighted undirected graphs with slackness In the previous setting, as well as the unweighted setting used in this paper, we have $\sum_{u\in\Gamma(v)} w(u,v)$ equals exactly 1. Equivalently, each v chooses exactly one incoming live edge. The most general LTM allows that $\sum_{u\in\Gamma(v)} w(u,v)$ may be strictly less than 1, or that each v can choose no incoming live edge with certain probability.

To define a model that incorporates this feature, we consider a more general model where each vertex v has a parameter $\vartheta_v \in [0,1]$ (given as an input to the algorithm) such that each vertex v chooses no incoming live edge with probability $1-\vartheta_v$, and, with probability ϑ_v, it chooses an incoming edge being live uniformly at random. Equivalently, given an undirected unweighted graph $G = (V, E)$, we assign weights to the edges such that $w(u,v) = \frac{\vartheta_v}{\deg(v)}$ and consider the standard LTM on directed graphs. Notice that we could further generalize this to allow weighted graphs, and then normalize the weights of the edges such that the sum of the weights of all incoming edges of each vertex v is exactly ϑ_v. However, this is a model that is even more general than the one in the last subsection (the model in the last subsection is obtained by setting $\vartheta_v = 1$ for all v from this model), and we know that the ratio $1-(1-1/k)^k$ is tight even up to infinitesimal additive $f(k)$. Thus, in this subsection, we consider the unweighted setting with the $(1-\vartheta_v)$ slackness for each vertex v. We will show that both Theorem 4.8 and Theorem 4.10 hold under this setting. It is clear that Theorem 4.8 holds, as we are considering a more general model.

To see that Theorem 4.10 holds, we first observe that Lemma 4.5, Lemma 4.16 and Lemma 4.17 hold with exactly the same proofs. To see that the remaining part of the proof of Theorem 4.10 can be adapted to this setting, we need to show that Lemma 4.18 holds, and we need to establish that InfMax under this setting is still a special case of Max-k-Coverage so that Proposition 4.21, 4.19 and 4.20 hold.

Note that Lemma 4.18 is also true for this new setting with slackness, and it can be proved by a simple coupling argument if knowing Lemma 4.18 for the original setting without slackness is true. Alternative, it can be proved directly by a similar arguments used in Sect. 3.5.2, and we include such a proof in Appendix A.5 for completeness.

We will use a more general version of Max-k-Coverage with weighted elements, where each element e_i has a positive weight $w(e_i)$, and the objective function we are maximizing becomes $\mathrm{val}(\mathcal{S}) = \sum_{e \in \bigcup_{S \in \mathcal{S}} S} w(e)$. All the lemmas in Sect. 4.4.1 hold for the weighted MAX-K-COVERAGE with exactly the same proofs. The interpretation of an INFMAX instance to a MAX-K-COVERAGE instance is almost the same as it is given in Sect. 4.2.1. The elements are tuples in $V \times H$ where H is the set of all possible realizations. Notice that here $|H| = \prod_{v \in V}(\deg(v)+1)$, as an extra outcome that an vertex chooses no incoming live edge is possible now. The weight of the element (v,g) equals to the probability that g is sampled. Therefore, $\sigma(S) = \sum_{v \in V} \Pr(S \to v) = \sum_{v \in V} \sum_{g\text{: } v \text{ is reachable from } S \text{ under } g} \Pr(g \text{ is sampled}) = \sum_{(v,g)\text{: } v \text{ is reachable from } S \text{ under } g} w((v,g))$. Let $\Sigma(S)$ be the same as before (which is the set of all elements (v,g) that are "covered" by S, or equivalently, the set of all (v,g)'s such that v is reachable from S under g). We have $\sigma(S) = \sum_{(u,g) \in \Sigma(S)} w((u,g))$.

Finally, Proposition 4.21, 4.19 and 4.20 hold with the following changes to the proof.

- every $|\Sigma(S)|$ is changed to its weighted version $\sum_{(u,g) \in \Sigma(S)} w((u,g))$;
- $\prod_{v \in V} \deg(v)$ is changed to 1 (Notice that we had $|H| = \prod_{v \in V} \deg(v)$ before, but we have $\sum_{g \in H} \Pr(g \text{ is sampled}) = 1$ now).

A.5 Proof of Lemma 4.18 Including Slackness

Recall that, in LTM on undirected graphs with slackness, each vertex has a parameter $\vartheta_v \in [0,1]$ that is given as an input to the algorithm. With probability $1 - \vartheta_v$, vertex v chooses no incoming live edge, and with probability ϑ_v, vertex v chooses one of its incoming edges as the live edge uniformly at random. (See the last subsection of Append. A.4.)

We prove the following lemma in this section.

Lemma A.6. *Consider LTM on undirected graphs with slackness. For any $v \in V$, we have $\sigma(\{v\}) = \deg(v) + 1$.*

This lemma is a generalization to Lemma 4.18, as the linear threshold model in Definition 4.1 used in Chapter 4 is a special case with the slackness of each vertex being 0.

This lemma also fills in the last piece of the proof that Theorem 4.10 holds for the setting with unweighted undirected graphs with slackness.

As mentioned, the arguments is largely identical to the one in Sect. 3.5.2.

We first show that Lemma 4.18 holds for trees.

Lemma A.7. *Suppose G is a tree, we have $\sigma(\{v\}) \leq \deg(v) + 1$.*

Proof. We assume without loss of generality that G is rooted at v. Consider an arbitrary vertex $u \neq v$ at the second last level with children $u_1, \ldots, u_t$ being leaves of T. We have $\deg(u) = t + 1$. Suppose u's parent s is infected by v with probability x ($x = 1$ if $s = v$). Then u will be infected with probability $\frac{x\vartheta_u}{t+1}$, and each u_i

of $u_1, \ldots, u_t$, having degree 1, will be infected with probability ϑ_{u_i} if u is infected. Therefore, the expected number of infected vertices in the subtree rooted at u is

$$\frac{x\vartheta_u}{t+1}\left(1+\sum_{i=1}^{t}\vartheta_{u_i}\right)+\left(1-\frac{x\vartheta_u}{t+1}\right)\cdot 0x \leq \frac{x\vartheta_u}{t+1}(t+1)=x\vartheta_u.$$

This suggests that, if we contract the subtree rooted at u to a single vertex u, the expected total number of infected vertices can only increase for this change of the graph G, since the degree of u becomes 1 after this contraction, making the infection probability of u become $x\vartheta_u$. We can keep doing this contraction until G becomes a star with center v, and the expected number of infected vertices can only increase during this process. The lemma follows. □

We define the *lift* of an undirected graph G with respect to a vertex $a \subseteq V$, which is a new undirected graph $\widehat{G}_a$ that shares the same vertex a with G plus a lot of new vertices. We will then define a coupling between sampling live-edges in G and sampling live-edges in $\widehat{G}_a$. Given the seed v, this coupling reveals an upper bound of $\sigma(\{v\})$. In particular, we will show $\sigma_G(\{v\}) \leq \sigma_{\widehat{G}_v}(\{v\})$, where $\sigma_G(\cdot)$ and $\sigma_{\widehat{G}_v}(\cdot)$ denote the function $\sigma(\cdot)$ with respect to the graphs G and $\widehat{G}_v$ respectively.

Let

$$\mathcal{P}_a=\{P=((v_1,v_2),(v_2,v_3),\ldots,(v_{t-1},v_t)) : v_1=a; v_2,\ldots,v_t\neq a; \forall i\neq j : v_i\neq v_j\}$$

be the set of all simple paths P that start from vertex a but never come back to a.

Definition A.8. Given an undirected graph $G=(V,E)$ and $a \in V$, the *lift* of G with respect to a, denoted by $\widehat{G}_a=(\widehat{V},\widehat{E})$, is an undirected graph defined as follows.

- The vertex set is $\widehat{V}=\{a\}\cup V_P$, where $V_P=\{v_P : P\in\mathcal{P}_a\}$ is the set of vertices corresponding to the simple paths in $\mathcal{P}_a$.
- For each $v_P \in V_P$, include $(a, v_P)\in\widehat{E}$ if P is a path of length 1 that starts from a; for each $v_{P_1}, v_{P_2}\in V_P$, include (v_{P_1}, v_{P_2}) if $|P_2|=|P_1|+1$ and P_2, P_1 share the first $|P_1|$ common edges (or $|P_1|=|P_2|+1$ and P_1, P_2 share the first $|P_2|$ common edges, since $\widehat{G}_a$ is undirected).
- If $P\in\mathcal{P}_a$ is a path ending at a vertex in G that is adjacent to a, add a dummy vertex in $\widehat{G}$ and connect this vertex to v_P.

It is easy to see that $\widehat{G}_a$ is a tree (that can be viewed as) rooted at a. The vertices in the tree $\widehat{G}_a$ correspond to all the paths in $\mathcal{P}_a$ starting at a. For any path $P\in\mathcal{P}_a$ with v being its ending vertex, $\deg(v_P)$ in $\widehat{G}_a$ equals to $\deg(v)$ in G.

Let $\Psi : E\to 2^{\widehat{E}}$ be the function mapping an undirected edge in G to its counterparts in $\widehat{G}_a$:

$$\Psi(e)=\begin{cases}\{(a,v_P)\mid P=((a,v))\} & \text{if } e=(a,v)\\ \{(v_{P_1},v_{P_2})\mid P_2=(P_1,e)\} & \text{Otherwise.}\end{cases}$$

Notice that in the above definition, $\Psi(e)$ contains only a single edge (a, v_P) with $P = ((a, v))$ being the length-one path connecting a and v if $e = (a, v)$, while $\Psi(e)$ contains the set of all (v_{P_1}, v_{P_2}) such that P_2 is obtained by appending e to P_1. Let $\Phi : V \to 2^{\widehat{V}}$ represent the vertex correspondence:

$$\Phi(v) = \begin{cases} \{v\} & \text{if } v = a \\ \{v_P \mid P \text{ ends at } v\} & \text{Otherwise.} \end{cases}$$

From our definition, it is easy to see that $\Psi(e_1) \cap \Psi(e_2) = \emptyset$ if $e_1 \neq e_2$, and $\Phi(u) \cap \Phi(v) = \emptyset$ if $u \neq v$. Moreover, since $\mathcal{P}_a$ contains only paths, for any vertex v and edge e in G, each path in $\widehat{G}_a$ connecting a to a leaf (recall that $\widehat{G}_a$ is a tree) can intersect each of $\Psi(e)$ and $\Phi(v)$ at most once.[1]

Finally, to let the inequality $\sigma_G(\{v\}) \leq \sigma_{\widehat{G}_v}(\{v\})$ make sense, we need to specify the parameter ϑ for each vertex in $\widehat{G}_v$. This is done in a natural way: for each vertex $w \in \Phi(v)$ in $\widehat{G}_v$, set ϑ_w for vertex w in $\widehat{G}_v$ be the same as ϑ_v for vertex v in G.

Lemma A.9.

$$\sigma_G(\{v\}) \leq \sigma_{\widehat{G}_v}(\{v\}).$$

Proof. We will define a coupling between the process of revealing live-edges in G and the process of revealing live-edges in $\widehat{G}_v$. Let χ_G be the edge-revelation process in G, and $\chi_{\widehat{G}_v}$ be the edge-revelation process in $\chi_{\widehat{G}_v}$, where in both processes, each edge is viewed as two anti-parallel directed edges, and we always reveal all the incoming edges for a vertex u simultaneously by choosing exactly one incoming edge uniformly at random with probability ϑ_u. We will couple χ_G with another edge-revelation process $\chi'_{\widehat{G}_v}$ of $\widehat{G}_v$.

We consider the following coupling. In each iteration where all the incoming edges of u, denoted by $(u_1, u), (u_2, u), \ldots, (u_{\deg(u)}, u)$, are revealed such that at most one of them is live, we reveal all the incoming edges for each $v_P \in \Phi(u)$ as follows.

- If none of $(u_1, u), (u_2, u), \ldots, (u_{\deg(u)}, u)$ is live in G, then v_P chooses no live incoming edge.
- For each P' such that $v_{P'}$ is a neighbor of v_P, there must exists $u_i \in \{u_1, \ldots, u_{\deg(u)}\}$ such that either that P' is obtained by appending (u, u_i) to P or that P is obtained by appending (u_i, u) to P'. Reveal the directed edge $(v_{P'}, v_P)$ such that it is live if and only if (u_i, u) is live in G.

[1] To see this for each $\Psi(e)$, suppose for the sake of contradiction that the path from v_P to the root contains two edges $(v_{P_1}, v_{P_2}), (v_{P_3}, v_{P_4})$ such that $(v_{P_1}, v_{P_2}), (v_{P_3}, v_{P_4}) \in \Psi(e)$ for some edge e. Assume without loss of generality that the order of the four vertices on the path according to the distances to the root is $(v_{P_1}, v_{P_2}, v_{P_3}, v_{P_4})$. It is easy to see from our construction that $P_1 \subsetneq P_2 \subsetneq P_3 \subsetneq P_4$. As a result, $(v_{P_1}, v_{P_2}), (v_{P_3}, v_{P_4}) \in \Psi(e)$ implies that P_2 is the path obtained by appending e to P_1, and P_4, containing P_2, P_3, is obtained by appending e to P_3, which further implies that P_4 is a path that uses the edge e twice, contradicting to our definition that $\mathcal{P}_a$ contains only simple paths.

The corresponding claim for each $\Phi(v)$ can be shown similarly.

- If there is a live edge $(v_{P'}, v_P)$ revealed in the above step, make all the remaining directed edges connecting to v_P not be live. If no live edge is revealed in the above step and one of $(u_1, u), (u_2, u), \ldots, (u_{\deg(u)}, u)$ in G is live, it must be that (a, u) is an edge in G and u has chosen (a, u) being the live edge. In this case, let the edge between v_P and the dummy vertex being live (See the third bullet point of Definition A.8).

This defines a coupling between χ_G and $\chi'_{\widehat{G}_v}$. It is easy to check that each $v_P \in \widehat{V}$ chooses exactly one of its incoming edges uniformly at random with probability ϑ_{v_P} and chooses no incoming edge with probability $1 - \vartheta_{v_P}$ in this coupling, which is the same as it is in the process $\chi_{\widehat{G}_v}$. The difference is that, there are dependencies between the revelations of incoming edges for different vertices in $\widehat{G}_v$: if both $v_P, v_{P'} \in \widehat{V}$ belongs to the same $\Phi(u)$ for some $u \in V$, the incoming edges for v_P and $v_{P'}$ are revealed in the same way.

Although the two processes $\chi'_{\widehat{G}_v}$ and $\chi_{\widehat{G}_v}$ are not the same, we will show that the expected number of vertices that are reachable from v by live edges is the same in both $\chi'_{\widehat{G}_v}$ and $\chi_{\widehat{G}_v}$. It suffices to show that, for each $v_P \in \widehat{V}$, all the *vertices* in the path connecting v_P to the seed v are considered independently (meaning that the incoming edges for v_{P_1} on the path are revealed independently to the revelations of the incoming edges of v_{P_2}), since this would imply that the probability v_P is connected to a seed is the same in both $\chi'_{\widehat{G}_v}$ and $\chi_{\widehat{G}_v}$, and the total number of vertices reachable from v by live edges is the same by the linearity of expectation. We only need to show that there do not exist two vertices on this path that are in the same set $\Phi(u)$ for some $u \in V$, since the incoming edges of each $v_{P_1} \in \Phi(u_1)$ are revealed independently to the revelations of the incoming edges of each $v_{P_2} \in \Phi(u_2)$ whenever $u_1 \neq u_2$. This is true due to that all the paths in $\mathcal{P}_v$ are simple paths, as remarked in the paragraph below where we define function $\Phi(\cdot)$.

Following the same analysis before, we can show that the number of the vertices reachable from v in χ_G is always at most the number of vertices reachable from v in $\chi'_{\widehat{G}_v}$. The lemma concludes here. □

Since v has the same degree in G and $\widehat{G}_v$, Lemma A.9 and Lemma A.7 implies Lemma 4.18 in the slackness setting.

APPENDIX B

Generalizing Results in Chapter 5 for General Threshold Model

In this section, we show that all our theoretical results in Chapter 5 hold for submodular general threshold model. In Sect. B.1, we define the two feedback models, the full-adoption and the myopic, based on the general threshold model. In Sect. B.2, we justify that all our results in Chapter 5 hold for submodular general threshold model. Notice that, however, our empirical results in Sect. 5.7 depend on the reverse reachable set technique, which is only compatible with the triggering model.

B.1 General Threshold Model and Feedback

$I_{G,F}$ in Definition 2.1 can be viewed as a random function $I_{G,F} : \{0,1\}^{|V|} \to \{0,1\}^{|V|}$. In addition, if the thresholds of all the vertices are fixed, this function becomes deterministic. Correspondingly, we define a *realization* of a graph $G = (V, E)$ as a function $\phi : V \to (0, 1]$ which encodes the thresholds of all vertices. Let $I^{\phi}_{G,F} : \{0,1\}^{|V|} \to \{0,1\}^{|V|}$ be the deterministic function corresponding to the general threshold model $I_{G,F}$ with vertices' thresholds following realization ϕ. We will interchangeably consider ϕ as a function defined above or a $|V|$ dimensional vector in $(0,1]^{|V|}$, and we write $\phi \sim (0,1]^{|V|}$ to mean a random realization is sampled such that each θ_v is sampled uniformly at random and independently as it is in Definition 2.1.

In the remaining part of this section, we define the *full-adoption feedback model* and the *myopic feedback model* corresponding to the general threshold model.

When the seed-picker sees that a vertex v is not infected (v may be a vertex adjacent to $I^{\phi}_{G,F}(S)$ in the full-adoption feedback model, or a vertex adjacent to S in the myopic feedback model), the seed-picker has certain partial information about v's threshold. Specifically, let IN_v be v's infected in-neighbors that are observed by the seed-picker. By seeing that v is not infected, the seed-picker knows that the threshold of v is in the range $(f_v(IN_v), 1]$, and the posterior distribution of θ_v is the uniform distribution on this range.

Let the *level* of a vertex v, denoted by o_v, be a value which either equals a character $\checkmark$ indicating that it is infected, or a real value $\vartheta_v \in [0, 1]$ indicating that $\theta_v \in (\vartheta_v, 1]$. Let $O = \{\checkmark\} \cup [0, 1]$ be the space of all possible levels. A *partial realization* φ is a function specifying a level for each vertex: $\varphi : V \to O$. We say that a partial realization φ *is consistent with* the full realization ϕ, denoted by $\phi \simeq \varphi$, if $\phi(v) > \varphi(v)$ for any $v \in V$ such that $\varphi(v) \neq \checkmark$.

Definition B.1. Given a general threshold model $I_{G=(V,E),F}$ with a realization ϕ, the *full-adoption feedback* is a function $\Phi^{\mathfrak{f}}_{G,F,\phi}$ mapping a seed set $S \subseteq V$ to a partial realization φ such that

- $\varphi(v) = \checkmark$ for each $v \in I^{\phi}_{G,F}(S)$, and
- $\varphi(v) = f_v(I^{\phi}_{G,F}(S) \cap \Gamma(v))$ for each $v \notin I^{\phi}_{G,F}(S)$.

Definition B.2. Given a general threshold model $I_{G=(V,E),F}$ with a realization ϕ, the *myopic feedback* is a function $\Phi^{\mathfrak{m}}_{G,F,\phi}$ mapping a seed set $S \subseteq V$ to a partial realization φ such that

- $\varphi(v) = \checkmark$ for each $v \in S$, and
- for each $v \notin S$, $\varphi(v) = \checkmark$ if $f_v(S \cap \Gamma(v)) \geq \phi(v)$, and $\varphi(v) = f_v(S \cap \Gamma(v))$ if $f_v(S \cap \Gamma(v)) < \phi(v)$.

Notice that, in both definitions above, a vertex v that does not have any infected neighbor (i.e., $v \notin S$ such that $I^{\phi}_{G,F}(S) \cap \Gamma(v) = \emptyset$ for the full-adoption feedback model or $S \cap \Gamma(v) = \emptyset$ for the myopic feedback model) always satisfies $\varphi(v) = 0$, as $f_v(\emptyset) = 0$ by Definition 2.1.

After properly defining the two feedback models, the definition of the adaptive policy π, as well as the definitions of the functions $\mathcal{S}^{\mathfrak{f}}(\cdot,\cdot,\cdot), \mathcal{S}^{\mathfrak{m}}(\cdot,\cdot,\cdot), \sigma^{\mathfrak{f}}(\cdot,\cdot), \sigma^{\mathfrak{m}}(\cdot,\cdot)$, are exactly the same as they are in Sect. 5.2.1. The definitions of the adaptivity gap and the greedy adaptivity gap are also the same as they are in Sect. 5.2.2.

B.2 Extending of Our Results to General Threshold Model

We will show in this section that all our results can be extended to the submodular general threshold model. Recall that a general threshold model is submodular means that all the local influence functions f_v's are submodular. In this section, whenever we write $I_{G,F}$, we refer to the general threshold model in Definition 2.1, not the triggering model in Definition 2.1.

B.2.1 Infimum of Greedy Adaptivity Gap

Theorem 5.6 is extended as follows.

Theorem B.3. *For the full-adoption feedback model,*

$$\inf_{G,F,k:\ I_{G,F} \text{ is } \mathtt{ICM}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))} = \inf_{G,F,k:\ I_{G,F} \text{ is } \mathtt{LTM}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))}$$

$$= \inf_{G,F,k:\ I_{G,F}\ \textit{is submodular}} \frac{\sigma^{\mathfrak{f}}(\pi^g,k)}{\sigma(S^g(k))} = 1-\frac{1}{e}.$$

The same result holds for the myopic feedback model.

Recall that Theorem 5.6 can be easily implied by Lemma 5.8, Lemma 5.9 and Theorem 5.10. Since Lemma 5.8 and Lemma 5.9 are for specific models `ICM` and `LTM` which are compatible with both the triggering model and the general threshold model, their validity here is clear. Following the same arguments, Theorem B.3 can be implied by Lemma 5.8, Lemma 5.9 and the following theorem which is the counterpart to Theorem 5.10.

Theorem B.4. *If $I_{G,F}$ is a submodular general threshold model, then we have both*

$$\sigma^{\mathfrak{f}}(\pi^g,k) \geq \left(1-\frac{1}{e}\right)\max_{S\subseteq V,|S|\leq k}\sigma(S) \qquad \textit{and} \qquad \sigma^{\mathfrak{m}}(\pi^g,k) \geq \left(1-\frac{1}{e}\right)\max_{S\subseteq V,|S|\leq k}\sigma(S).$$

Similar to the proof of Theorem 5.10, Theorem B.4 can be proved by showing the three propositions: Proposition 5.11, Proposition 5.12 and Proposition 5.13. It is straightforward to check that Proposition 5.12 and Proposition 5.13 hold for the general threshold model with exactly the same proofs. Now, it remains to extend Proposition 5.11 to the general threshold model, which is restated and proved below.

Proposition B.5. *Given a submodular general threshold model $I_{G,F}$, any $S \subseteq V$, any feedback model (either full-adoption or myopic) and any partial realization φ that is a valid feedback of S (i.e., $\exists\phi : \varphi = \Phi^{\mathfrak{f}}_{G,F,\phi}(S)$ or $\exists\phi : \varphi = \Phi^{\mathfrak{m}}_{G,F,\phi}(S)$, depending on the feedback model considered), the function $\mathcal{T} : \{0,1\}^{|V|} \to \mathbb{R}_{\geq 0}$ defined as $\mathcal{T}(X) = \mathbb{E}_{\phi\simeq\varphi}[|I^{\phi}_{G,F}(S\cup X)|]$ is submodular.*

Proof. Fix a feedback model, $S \subseteq V$ and φ that is a valid feedback of S. Let $T = \{v \mid \varphi(v) = \checkmark\}$ be the set of infected vertices indicated by the feedback of S. We consider a new general threshold model $I_{G',F'}$ defined as follows:

- G' is obtained by removing vertices in T from G (and the edges connecting from/to vertices in T are also removed);
- For any $v \in V' = V\setminus T$, $\Gamma(v)\cap T$ is the set of in-neighbors of v that are removed. Define $f'_v(Y) = \frac{f_v((\Gamma(v)\cap T)\cup Y)-\varphi(v)}{1-\varphi(v)}$ for each subset Y of v's in-neighbors in the new graph G': $Y \subseteq \Gamma(v)\cap V'$.

Notice that f'_v is a valid local influence function. f'_v is clearly monotone. For each $v \in V'$, we have $\varphi(v) = f_v(\Gamma(v)\cap T)$, as this is exactly the feedback received from the fact that v has not yet infected. It is then easy to see that f'_v is always non-negative and $f'_v(\emptyset) = 0$.

A simple coupling argument can show that

$$\mathop{\mathbb{E}}_{\phi\simeq\varphi}\left[\left|I^{\phi}_{G,F}(S\cup X)\right|\right] = \sigma_{G',F'}(X\setminus T) + |T|. \tag{B.1}$$

To define the coupling, for each $v \in V'$, the threshold of v in G, θ_v, is coupled with the threshold of v in G' as $\theta'_v = \frac{\theta_v - \varphi(v)}{1-\varphi(v)}$. This is a valid coupling: by $\phi \simeq \varphi$, we know that θ_v is sampled uniformly at random from $(\varphi(v), 1]$, which indicates that the marginal distribution of θ'_v is the uniform distribution on $(0, 1]$, which makes $I_{G',F'}$ a valid general threshold model.

Under this coupling, on the vertices V', the cascade in G with seeds $S \cup X$ and partial realization φ is identical to the cascade in G' with seeds $X \setminus T$. To see this, consider an arbitrary vertex $v \in V'$ and let IN_v and IN'_v be v's infected neighbors in G and G' respectively. For induction hypothesis, suppose the two cascade processes before v's infection are identical. We have $IN_v = IN'_v \cup (\Gamma(v) \cap T)$ and $IN'_v \cap (\Gamma(v) \cap T) = \emptyset$. It is easy to see from our construction that v is infected in G if and only if v is infected in G':

$$f_v(IN_v) \geq \theta_v \Leftrightarrow f'_v(IN'_v) = \frac{f_v(IN_v) - \varphi(v)}{1 - \varphi(v)} \geq \theta'_v.$$

This proves Eqn. (B.1).

Finally, since each $f_v(\cdot)$ is assumed to be submodular, it is easy to see that each $f'_v(\cdot)$ is submodular by our definition. Thus, $I_{G',F'}$ is a submodular model. This, combined with Eqn. (B.1), proves the proposition. □

B.2.2 Supremum of Greedy Adaptivity Gap

All the results in Sect. 5.4 about the supremum of the greedy adaptivity gap can be extended easily to the submodular general threshold model. In particular, Lemma 5.16 and Lemma 5.18 are under `LTM`, which is compatible with the submodular general threshold model. Theorem 5.14 and Theorem 5.15 are proved by providing an example with a diffusion model that is a combination of `ICM` and `LTM`, and the diffusion model constructed in Definition 5.19 can be easily described in the formulation of the general threshold model, since both `ICM` and `LTM` can be described in the general threshold model.

APPENDIX C

Omitted Proofs in Chapter 8

C.1 Proof of Lemma 8.13

The proof will follow the structure of the proof sketch in the main body of this paper.

Let E be the event that at least one leaf (or tree node) is activated at the end of the cascade. By our definition, $P_{\boldsymbol{k}} = \lim_{n\to\infty} \Pr(E)$. Given a seeding strategy $\boldsymbol{k}$, let $\sigma(\boldsymbol{k}) := \mathbb{E}_{G\sim\mathcal{G}(n,T)}[\sigma_{r,G}(\boldsymbol{k})]$ be the expected number of infected vertices, $\sigma(\boldsymbol{k} \mid E) := \mathbb{E}_{G\sim\mathcal{G}(n,T)}[\sigma_{r,G}(\boldsymbol{k}) \mid E]$ be the expected number of infected vertices conditioning on event E, and $\sigma(\boldsymbol{k} \mid \neg E) := \mathbb{E}_{G\sim\mathcal{G}(n,T)}[\sigma_{r,G}(\boldsymbol{k}) \mid \neg E]$ be the expected number of infected vertices conditioning on that E does not happen. We have

$$\sigma(\boldsymbol{k}) = \Pr(E) \cdot \sigma(\boldsymbol{k} \mid E) + (1 - \Pr(E)) \cdot \sigma(\boldsymbol{k} \mid \neg E),$$

and

$$\Sigma_{r,T}(\boldsymbol{k}) = \lim_{n\to\infty} \frac{\sigma(\boldsymbol{k})}{n} = P_{\boldsymbol{k}} \cdot \lim_{n\to\infty} \frac{\sigma(\boldsymbol{k} \mid E)}{n} + (1 - P_{\boldsymbol{k}}) \cdot \lim_{n\to\infty} \frac{\sigma(\boldsymbol{k} \mid \neg E)}{n}. \tag{C.1}$$

To prove Lemma 8.13, it is sufficent to show the following two claims:

1. First, we show that $1 - P_{\boldsymbol{k}} > 0$ implies $\sigma(\boldsymbol{k} \mid \neg E) = o(n)$, so the second term in (C.1) is always 0 (Sect. C.1.1).

2. Second, to conclude the proof, it suffices to show that $\sigma(\boldsymbol{k} \mid E) = cn + o(n)$ for some constant c which does not depend on $\boldsymbol{k}$, which implies that the first term in (C.1) is monotone in $P_{\boldsymbol{k}}$ (Sect. C.1.2).

These two claims correspond to the second and the third paragraphs in the sketch of the proof.

The following proposition is useful for proving both claims.

Proposition C.1. *Suppose the root of T has weight $\omega(1/n^{1+1/r})$ and consider a leaf t. If there are $\Theta(n)$ infected vertices in $V \setminus V(t)$, then these infected vertices outside $V(t)$ will infect $\omega(1)$ vertices in $V(t)$ with probability $1 - o(1)$.*

Proof. Let $X = \Theta(n)$ be the number of infected vertices in $V\setminus V(t)$. For each $u \in V(t)$ and $v \in V \setminus V(t)$, we assume that the probability p_{uv} that the edge (u, v) appears satisfies $p_{uv} = \omega(1/n^{1+1/r})$ and $p_{uv} = o(1/n)$, where $p_{uv} = \omega(1/n^{1+1/r})$ holds since the root of T has weight $\omega(1/n^{1+1/r})$, and assuming $p_{uv} = o(1/n)$ may only decrease the number of infected vertices in $V(t)$ if the least common ancestor of the two leaves containing u and v has weight $\Omega(1/n)$. Let p be the minimum probability among those p_{uv}'s, and we further assume that each edge (u, v) appears with probability p, which again may only reduce the number of infected vertices in $V(t)$.

For each vertex $u \in V(t)$, by only accounting for the probability that it has exactly r neighbors among those X outside infected vertices, the probability that u is infected is at least

$$\rho := \binom{X}{r} p^r (1-p)^{X-r} = \omega\left(n^r \cdot \left(\frac{1}{n^{1+1/r}}\right)^r \left(1 - \frac{1}{n}\right)^n\right) = \omega\left(\frac{1}{n}\right),$$

and the expected number of infected vertices in $V(t)$ is at least $v(t)n \cdot \rho = \omega(1)$.

Let Y be the number of vertices in $V(t)$ that are infected due to the influence of $V \setminus V(t)$, so we have $\mathbb{E}[Y] = v(t)n\rho$. Applying Chebyshev's inequality,

$$\Pr\left(Y \le \frac{1}{2}v(t)n\rho\right) \le \Pr\left(|Y - \mathbb{E}[Y]| \ge \frac{1}{2}v(t)n\rho\right)$$

$$\le \frac{\mathrm{Var}(Y)}{(\frac{1}{2}v(t)n\rho)^2} = \frac{v(t)n\rho(1-\rho)}{\frac{1}{4}v(t)^2n^2\rho^2} = o(1),$$

where we have used the fact that $n\rho = \omega(1)$ and the variance of the Binomial random variable with parameter n, p is $np(1-p)$. Therefore, with probability $1 - o(1)$, the number of infected vertices in $V(t)$ is at least $\frac{1}{2}v(t)n\rho = \omega(1)$. □

C.1.1 Proof of the First Claim

We consider two cases: 1) T contains no critical or supercritical leaf; 2) T contains at least one critical or supercritical leaf.

If there is no critical or supercritical leaf in T, given that the total number of seeds $K = \Theta(1)$ is a constant, Theorem 8.7 shows that, with high probability, there can be at most $2K = \Theta(1)$ infected vertices even without conditioning on that E has not happened. To be specific, we can take the maximum weight $w^*(t)$ over all the leaves, and assume the entire graph is the Erdős-Rényi graph $\mathcal{G}(n, w^*(t))$. This makes the graph denser, so the expected number of infected vertices increases. We further assume that we have not conditioned on $\neg E$, this further increases the expected number of infected vertices. However, even under these assumptions, Theorem 8.7 implies that the total number of infected vertices is less than $2K$ with high probability. Thus, $\sigma(\boldsymbol{k} \mid \neg E) = o(n)$ even without assuming $1 - P_{\boldsymbol{k}} > 0$.

Suppose there is at least one critical or supercritical leaf, and $\Pr(\neg E) = \Theta(1)$ (equivalently, $1 - P_{\boldsymbol{k}} > 0$, as given in the statement of the first claim). To show that $\sigma(\boldsymbol{k} \mid \neg E) = o(n)$, it suffices to show that, conditioning on there being $\Theta(n)$ infected vertices, E happens with probability $1 - o(1)$. This is because, if $\Pr(\neg E) = \Theta(1)$ and

$\Pr(\neg E \mid \sigma(\boldsymbol{k}) = \Theta(n)) = o(1)$, then

$$\Pr\left(\sigma(\boldsymbol{k}) = \Theta(n) \mid \neg E\right) = \frac{\Pr(\sigma(\boldsymbol{k}) = \Theta(n)) \cdot \Pr(\neg E \mid \sigma(\boldsymbol{k}) = \Theta(n))}{\Pr(\neg E)} = o(1),$$

which implies $\sigma(\boldsymbol{k} \mid \neg E) = o(n)$.

Now, suppose there are $\Theta(n)$ infected vertices; to conclude the claim, we will show that E happens with probability $1 - o(1)$. Since the number of leaves is a constant, there exists $t' \in L_T$ such that the number of infected vertices in $V(t')$ is $\Theta(n)$. Let t be a critical or supercritical leaf (we have supposed there is at least one critical or supercritical leaf). Theorem 8.8 and Corollary 8.10 indicate that, with probability $1-o(1)$, the number of infected vertices in $V(t)$ is either a constant or $v(t)n$. Therefore, if $t' = t$, with probability $1 - o(1)$, those $\Theta(n)$ infected vertices in $V(t)$ will activate t, so E happens with probability $1 - o(1)$. If $t' \neq t$, let $X = \Theta(n)$ be such that with probability $1 - o(1)$ the number of infected vertices in $V(t')$ is more than X, then the total number of vertices in $V(t)$ that are infected by those X vertices in $V(t')$ is $\omega(1)$ (with high probability) according to Proposition C.1. Theorem 8.8 and Corollary 8.10 show that, with high probability, those $\omega(1)$ infected vertices in $V(t)$ will further spread and activate t, which again says that E happens with probability $1 - o(1)$.

C.1.2 Proof of the Second Claim

As an intuitive argument, Proposition C.1, Theorem 8.8, and Corollary 8.10 show that, when E happens, with high probability, a single activated leaf will activate all the critical and supercritical leaves, and the number of vertices corresponding to all the critical and supercritical leaves is fixed and independent of $\boldsymbol{k}$; based on the tree structure and the number of infected outside vertices, the number of infected vertices in a subcritical leaf may vary; however, we will see that the seeding strategy $\boldsymbol{k}$, adding only a constant number of infections, is too weak to significantly affect the number of infected vertices in a subcritical leaf.

To break it down, we first show that all critical and supercritical leaves will be activated with high probability if E happens. This is straightforward: Proposition C.1 shows that an activated leaf can cause $\omega(1)$ infected vertices in every other leaf with high probability, and Theorem 8.8 and Corollary 8.10 indicate that those critical and supercritical leaves will be activated by those $\omega(1)$ infected vertices with high probability.

Lastly, assuming all critical and supercritical leaves are activated, we show that the number of infected vertices in any subcritical leaf does not significantly depend on $\boldsymbol{k}$. We do not need to worry about those seeds that are put in the critical or supercritical leaves, as all vertices in those leaves will be infected later. As a result, we only need to show that a constant number of seeds in subcritical leaves has negligible effect to the cascade.

We say a subcritical leaf t is *vulnerable* if there exists a criticial or supercritical leaf t' such that the least common ancestor of t and t' has weight $\Omega(1/n)$, and we say t

is *not-vulnerable* otherwise. It is easy to see that a vulnerable leaf t will be activated with high probability conditional on E, even if no seed is put into it. Since each $v \in V(t)$ is connected to one of the $v(t')n$ vertices in $V(t')$ with probability $\Omega(1/n)$, the number of infected neighbors of v follows a Binomial distribution with parameter $(v(t')n, p)$ where $p = \Omega(1/n)$. We only consider $p = \Theta(1/n)$, as there can only be more infected vertices if $p = \omega(1/n)$. If $p = \Theta(1/n)$, the Binomial distribution becomes a Poisson distribution with a constant mean λ for $n \to \infty$. In this case, with constant probability $e^{-\lambda}\frac{\lambda^r}{r!}$, v has r infected neighbors. Therefore, v will be infected with constant probability, and $V(t)$ has $\Theta(n)$ vertices that are infected by $V(t')$ outside. The second part of Theorem 8.7 shows that, these $\Theta(n)$ infected vertices will further spread and activate t with high probability. Therefore, the seeds on those vulnerable subcritical leaves have no effect, since vulnerable subcritical leaves will be activated with high probability regardless the seeding strategy.

Let $t_1, \ldots, t_M$ be all the not-vulnerable subcritical leaves. Suppose we are at the stage of the cascade process where all those critical, supercritical and vulnerable subcritical leaves have already been activated (as they will with probability $1 - o(1)$ since we assumed that E has happened) and we are revealing the edges between $V \setminus \bigcup_{m=1}^{M} V(t_m)$ and $\bigcup_{m=1}^{M} V(t_m)$ to consider the cascade process in $\bigcup_{m=1}^{M} V(t_m)$. For each $i = 0, 1, \ldots, r-1$ and each $m = 1, \ldots, M$, let χ_i^m be the number of vertices in $V(t_m)$ that have *exactly* i infected neighbors among $V \setminus \bigcup_{m=1}^{M} V(t_m)$, which can be viewed as a random variable. For each $m = 1, \ldots, M$, let χ_r^m be the number of vertices in $V(t_m)$ that have *at least* r infected neighbors. If there are K_m seeds in $V(t_m)$, we increase the value of χ_r^m by K_m. Let $\boldsymbol{\chi}^m = (\chi_0^m, \chi_1^m, \ldots, \chi_r^m)$. Since $(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$ completely characterizes the expected number of infected vertices in the subcritical leaves (the expectation is taken over the sampling of the edges within every $V(t_i)$ and between every pair $V(t_i), V(t_j)$), we let $\sigma(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$ be the total number of infected vertices in the subcritical leaves, given $(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$. We aim to show that *adding $K_1, \ldots, K_M$ seeds in $V(t_1), \ldots, V(t_M)$ only changes the expected number of infected vertices by $o(n)$.*

Let $(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$ correspond to the case where no seed is added, and $(\bar{\boldsymbol{\chi}}^1, \ldots, \bar{\boldsymbol{\chi}}^M)$ correspond to the case where K_m seeds are added to t_m for each $m = 1, \ldots, M$. The outline of the proof is that, we first show that a) the total variation distance of the two distributions $(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$ and $(\bar{\boldsymbol{\chi}}^1, \ldots, \bar{\boldsymbol{\chi}}^M)$ is $o(1)$; then b) we show that $\sigma(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$ and $\sigma(\bar{\boldsymbol{\chi}}^1, \ldots, \bar{\boldsymbol{\chi}}^M)$ can only differ by $o(n)$ in expectation.

We first note that claim a) can imply claim b) easily. Notice that the range of the function $\sigma(\cdot)$ falls into the interval $[0, n]$. The total variation distance of $(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)$ and $(\bar{\boldsymbol{\chi}}^1, \ldots, \bar{\boldsymbol{\chi}}^M)$ being $o(1)$ implies that

$$\left| \mathop{\mathbb{E}}_{(\boldsymbol{\chi}^1,\ldots,\boldsymbol{\chi}^M)}[\sigma(\boldsymbol{\chi}^1, \ldots, \boldsymbol{\chi}^M)] - \mathop{\mathbb{E}}_{(\bar{\boldsymbol{\chi}}^1,\ldots,\bar{\boldsymbol{\chi}}^M)}[\sigma(\bar{\boldsymbol{\chi}}^1, \ldots, \bar{\boldsymbol{\chi}}^M)] \right| = o(n),$$

by a standard property of total variation distance (see, for example, Proposition 4.5 in [52]).

To show the claim a), noticing that M is a constant and $\boldsymbol{\chi}^{m_1}$ is independent of $\boldsymbol{\chi}^{m_2}$ for any m_1 and m_2 (the appearances of edges between $V(t_{m_1})$ and $V \setminus \bigcup_{m=1}^{M} V(t_m)$

are independent of the appearances of edges between $V(t_{m_2})$ and $V \setminus \bigcup_{m=1}^{M} V(t_m)$), it is sufficient to show that the total variation distance between $\boldsymbol{\chi}^m$ and $\bar{\boldsymbol{\chi}}^m$ is $o(1)$. Each vertex $v \in V(t_m)$ is connected to an arbitrary vertex in a critical or supercritical leaf with probability between $\omega(1/n^{1+1/r})$ (since the root has weight $\omega(1/n^{1+1/r})$) and $o(1/n)$ (otherwise t_m is vulnerable). Since the number of infected vertices in $V \setminus \bigcup_{m=1}^{M} V(t_m)$ is $\Theta(n)$, the number of v's infected neighbors follows a Binomial distribution, $\mathrm{Bin}(n, \theta)$, with mean $n\theta$ between $\omega(1/n^{1/r})$ and $o(1)$, we can use Poisson distribution $\mathrm{Po}(n\theta)$ to approximate it. Formally, the total variation distance is $d_{TV}(\mathrm{Bin}(n,\theta), \mathrm{Po}(n\theta)) \leq n\theta^2 = o(1/n)$. Thus, this approximation only changes the total variation distance of $\boldsymbol{\chi}^m$ by $o(1)$. Observing this, the proposition below shows the total variation distance between $\boldsymbol{\chi}^m$ and $\bar{\boldsymbol{\chi}}^m$ is $o(1)$.

Proposition C.2. *Let λ be such that $\lambda = \omega(1/n^{1/r})$ and $\lambda = o(1)$. Let $Y_1, \ldots, Y_n \in \mathbb{Z}$ be n independently and identically distributed random variables where each Y_i is sampled from a Poisson distribution with mean λ. Let $Z_1, \ldots, Z_n \in \mathbb{Z}$ be n random variables, where the first K of them satisfy $Z_1 = \cdots = Z_K = r$ with probability 1, and the remaining random variables $Z_{K+1}, \ldots, Z_n$ are independently sampled from a Poisson distribution with mean λ. For $i = 0, 1, \ldots, r-1$, let χ_i be the number of random variables in $\{Y_1, \ldots, Y_n\}$ that have value i, and $\bar{\chi}_i$ be the number of random variables in $\{Z_1, \ldots, Z_n\}$ that have value i. Let χ_r be the number of random variables in $\{Y_1, \ldots, Y_n\}$ that have values at least r, and $\bar{\chi}_r$ be the number of random variables in $\{Z_1, \ldots, Z_n\}$ that have values at least r. The total variation distance between $\boldsymbol{\chi} = (\chi_0, \chi_1, \ldots, \chi_r)$ and $\bar{\boldsymbol{\chi}} = (\bar{\chi}_0, \bar{\chi}_1, \ldots, \bar{\chi}_r)$ is $d_{TV}(\boldsymbol{\chi}, \bar{\boldsymbol{\chi}}) = o(1)$ if $K = \Theta(1)$.*

To show that random vectors $\boldsymbol{\chi}$ and $\bar{\boldsymbol{\chi}}$ have a small total variation distance, we first estimate them by Poisson approximations. Note that $\boldsymbol{\chi}$ and $\bar{\boldsymbol{\chi}}$ can be seen as ball and bin processes. There are $r+1$ bins, and n balls. For $\boldsymbol{\chi}$, the probability of ball i in bin ℓ is $\Pr[Y_i = \ell]$ when $0 \leq \ell < r$ and $\Pr[Y_i \geq r]$ for bin r. χ_ℓ is the number of balls in bin ℓ. Therefore, we can simplify the correlation between the coordinates of $\boldsymbol{\chi} = (\chi_0, \chi_1, \ldots, \chi_r)$, and formulate $\boldsymbol{\chi}$ as a $r+1$ coordinate-wise independent Poisson $\boldsymbol{\zeta} = (\zeta_0, \zeta_1, \ldots, \zeta_r)$ with the same expectation $\mathbb{E}[\boldsymbol{\chi}] = \mathbb{E}[\boldsymbol{\zeta}]$ conditioning on $\sum_{0 \leq \ell \leq r} \zeta_\ell = n$. For $\bar{\boldsymbol{\chi}}$, we define $\bar{\boldsymbol{\zeta}}$ similarly.

Then, we upper-bound the total variation distance between those two Poisson vectors $\boldsymbol{\zeta}$ and $\bar{\boldsymbol{\zeta}}$ conditioning on $\sum_{0 \leq \ell \leq r} \zeta_\ell = \sum_{0 \leq \ell \leq r} \bar{\zeta}_\ell = n$. We compute the relative divergence between them and use the Pinsker's inequality [46] to upper bound the total variation distance.

Proof. For $\boldsymbol{\chi}$, there are $r+1$ bins and n balls. Let the probability of ball i in bin ℓ be $p_\ell := \Pr[Y_i = \ell]$ when $0 \leq \ell < r$ and $p_r := \Pr[Y_i \geq r]$ for bin r (note that these probabilities are independent of the index i). For $0 \leq \ell \leq r$, χ_ℓ is the number of balls in bin ℓ. Consider the following Poisson vector $\boldsymbol{\zeta} = (\zeta_0, \zeta_1, \ldots, \zeta_r)$ with parameters $(\lambda_0, \ldots, \lambda_r)$ where $\lambda_\ell = np_\ell$ for $0 \leq \ell \leq r$: each coordinate ζ_ℓ is sampled from a Poisson distribution with parameter λ_ℓ independently. Note that the distribution of

$\boldsymbol{\chi}$ equals to $\boldsymbol{\zeta}$ conditioning on $\sum_{\ell=0}^r \zeta_\ell = n$: for all $\mathbf{k} \in \mathbb{Z}_{\geq 0}^{r+1}$ with $\sum_{\ell=0}^r k_\ell = n$,

$$\Pr(\boldsymbol{\chi} = \mathbf{k}) = \Pr\left(\boldsymbol{\zeta} = \mathbf{k} \mid \sum_{\ell=0}^r k_\ell = n\right) = \frac{n!}{n^n e^{-n}} \prod_{\ell=0}^r \frac{\lambda_\ell^{k_\ell} e^{-\lambda_\ell}}{k_\ell!}. \tag{C.2}$$

The process $\bar{\boldsymbol{\chi}}$ needs more work. In the context of ball and bin process, the first K balls are in bin r with probability 1, and the rest of balls follow the distribution $(p_\ell)_{0\leq \ell\leq r}$ defined above. For $0 \leq \ell \leq r$, $\bar{\chi}_\ell$ is the number of balls in bin ℓ. This non-symmetry makes the connection from $\bar{\boldsymbol{\chi}}$ to a Poisson distribution less obvious. Here, we first use a process $\bar{\boldsymbol{\chi}}'$ to approximate $\bar{\boldsymbol{\chi}}$ where all balls are thrown into the bins independently and identically, and we translate $\bar{\boldsymbol{\chi}}'$ to a Poisson distribution. Before defining $\bar{\boldsymbol{\chi}}'$, note that $\bar{\boldsymbol{\chi}}$ is equivalent to the following process: instead of picking first K indices, we can randomly pick K indices $i_1, i_2, \ldots, i_K$ and let $Z_{i_\iota} = r$ for $0 \leq \iota \leq K$. The other follows the distribution $(p_\ell)_{0\leq\ell\leq r}$. In this formulation, the distribution of the positions of balls are identical, but not independent. Now we define $\bar{\boldsymbol{\chi}}'$ by setting them to be independent: Let the probability of ball i in bin ℓ be $\bar{p}_\ell := (1 - K/n)p_\ell$ when $0 \leq \ell < r$ and $\bar{p}_r := (1-K/n)p_r + K/n$. The positions of balls are now mutually independent in $\bar{\boldsymbol{\chi}}'$. For $0 \leq \ell \leq r$, $\bar{\chi}'_\ell$ is the number of balls in bin ℓ.

Note that the distributions of $\bar{\boldsymbol{\chi}}$ and $\bar{\boldsymbol{\chi}}'$ are different. In particular, the marginal distribution of $\bar{\chi}_r$ is K plus a binomial distribution with parameter $(n-K, p_r)$, and the marginal distribution of $\bar{\chi}'_r$ is a binomial distribution with parameter $(n, \bar{p}_r)$. However, we can show that

$$d_{TV}(\bar{\boldsymbol{\chi}}, \bar{\boldsymbol{\chi}}') = o(1). \tag{C.3}$$

Equivalently, we want to show there exists a coupling between $\bar{\boldsymbol{\chi}}$ and $\bar{\boldsymbol{\chi}}'$ such that the probability of $\bar{\boldsymbol{\chi}} \neq \bar{\boldsymbol{\chi}}'$ is in $o(1)$. First, for all $k_r \geq K$, the distributions of $\bar{\boldsymbol{\chi}}$ conditioning on $\bar{\chi}_r = k_r$ and $\bar{\boldsymbol{\chi}}'$ conditioning on $\bar{\chi}'_r = k_r$ are the same. Therefore, fixing a coupling between $\bar{\chi}_r$ and $\bar{\chi}'_r$, we can extend it to a coupling between $\bar{\boldsymbol{\chi}}$ and $\bar{\boldsymbol{\chi}}'$ such that when an event $\bar{\chi}_r = \bar{\chi}'_r$ happens, $\bar{\boldsymbol{\chi}} = \bar{\boldsymbol{\chi}}'$. Thus, we have $d_{TV}(\bar{\boldsymbol{\chi}}, \bar{\boldsymbol{\chi}}') = d_{TV}(\bar{\chi}_r, \bar{\chi}'_r)$. Now it suffices to show the following claim.

Claim C.3.

$$d_{TV}(\bar{\chi}_r, \bar{\chi}'_r) = o(1).$$

Intuitively, the mean of $\bar{\chi}_r$ and $\bar{\chi}'_r$ are both $n\bar{p}_r$ which is in $\omega(1)$, so the small distinction between them should not matter. We present a proof later for completeness.

Given $\bar{\boldsymbol{\chi}}'$, consider the following Poisson vector $\bar{\boldsymbol{\zeta}} = (\bar{\zeta}_0, \bar{\zeta}_1, \ldots, \bar{\zeta}_r)$ with parameter $(\bar{\lambda}_0, \ldots, \bar{\lambda}_r)$ where $\bar{\lambda}_\ell = n\bar{p}_\ell$ for $0 \leq \ell \leq r$. The distribution of $\bar{\boldsymbol{\chi}}'$ equals to $\bar{\boldsymbol{\zeta}}$ conditioning on $\sum_{\ell=0}^r \bar{\zeta}_\ell = n$: for all $\mathbf{k} \in \mathbb{Z}_{\geq 0}^{r+1}$ with $\sum_{\ell=0}^r k_\ell = n$,

$$\Pr(\bar{\boldsymbol{\chi}}' = \mathbf{k}) = \Pr\left(\bar{\boldsymbol{\zeta}} = \mathbf{k} \mid \sum_{\ell=0}^r k_\ell = n\right) = \frac{n!}{n^n e^{-n}} \prod_{\ell=0}^r \frac{\bar{\lambda}_\ell^{k_\ell} e^{-\bar{\lambda}_\ell}}{k_\ell!}. \tag{C.4}$$

Finally, with (C.2), (C.3), and (C.4), it suffices to upper-bound the total variation distance between $\boldsymbol{\chi}$ and $\bar{\boldsymbol{\chi}}'$. We will prove the following claim later.

Claim C.4.

$$d_{TV}(\boldsymbol{\chi}, \bar{\boldsymbol{\chi}}') = o(1).$$

With these claims, we completes the proof:

$$d_{TV}(\boldsymbol{\chi}, \bar{\boldsymbol{\chi}}) \leq d_{TV}(\boldsymbol{\chi}, \bar{\boldsymbol{\chi}}') + d_{TV}(\bar{\boldsymbol{\chi}}, \bar{\boldsymbol{\chi}}') = o(1)$$

by the triangle inequality. □

Proof of Claim C.3. Informally, the mean of $\bar{\chi}_r$ and $\bar{\chi}'_r$ are both $n\bar{p}_r = \omega(1)$, so the small distinction between them should not matter. We formalize these by using Poisson distributions to approximate $\bar{\chi}_r$ (a binomial, $\text{Bin}(n, \bar{p}_r)$) and $\bar{\chi}'_r$ (a transported binomial, $K + \text{Bin}(n-K, p_r)$).

Recall that $\text{Po}(x)$ denotes a Poisson random variable with parameter x. By the triangle inequality, the distance, $d_{TV}(\bar{\chi}_r, \bar{\chi}'_r) = d_{TV}(K + \text{Bin}(n-K, p_r), \text{Bin}(n, \bar{p}_r))$, is less the the sum of the following four terms:

1. $d_{TV}(K + \text{Bin}(n-K, p_r), K + \text{Po}((n-K)p_r))$,
2. $d_{TV}(K + \text{Po}((n-K)p_r), K + \text{Po}(n\bar{p}_r))$,
3. $d_{TV}(K + \text{Po}(n\bar{p}_r), \text{Po}(n\bar{p}_r))$, and
4. $d_{TV}(\text{Po}(n\bar{p}_r), \text{Bin}(n, \bar{p}_r))$.

Now we want to show all four terms are in $o(1)$. By the Poisson approximation [48], for all p, $d_{TV}(\text{Bin}(n,p), \text{Po}(np)) \leq p$, the first and the final term, are less than p_r and $\bar{p}_r$ respectively. Both are in $o(1)$ since $p_r = \Theta(\lambda^r)$.

For the second term, because $d_{TV}(\text{Po}(\lambda_1), \text{Po}(\lambda_2)) \leq \frac{|\lambda_1 - \lambda_2|}{\sqrt{\lambda_1}+\sqrt{\lambda_2}}$ for all λ_1 and λ_2 (see [1]) and $p_r = \Omega(\lambda^r) = \omega(1/n)$,

$$d_{TV}(\text{Po}((n-K)p_r), \text{Po}(n\bar{p}_r)) \leq \frac{n\bar{p}_r - (n-K)p_r}{\sqrt{n\bar{p}_r} + \sqrt{(n-K)p_r}} = \frac{K + Kp_r}{\sqrt{n\bar{p}_r} + \sqrt{(n-K)p_r}} = o(1).$$

Finally, for the third term, let $(x)^+ = \max\{0, x\}$ for all x. Recall that $\bar{\lambda}_r = n\bar{p}_r$. By a definition of total variation distance, we have

$$\begin{aligned}
&d_{TV}(K+\mathrm{Po}(\bar\lambda_r),\mathrm{Po}(\bar\lambda_r))\\
=&\sum_{x\geq 0}\left(\Pr(K+\mathrm{Po}(\bar\lambda_r)=x)-\Pr(\mathrm{Po}(\bar\lambda_r)=x)\right)^+\\
=&\sum_{x\geq K}\left(\Pr(\mathrm{Po}(\bar\lambda_r)=x-K)-\Pr(\mathrm{Po}(\bar\lambda_r)=x)\right)^+ && \text{(the first } K \text{ terms are zero)}\\
=&\sum_{x\geq K}\Pr(\mathrm{Po}(\bar\lambda_r)=x-K)\left(1-\frac{\Pr(\mathrm{Po}(\bar\lambda_r)=x)}{\Pr(\mathrm{Po}(\bar\lambda_r)=x-K)}\right)^+\\
=&\sum_{x\geq 0}\Pr(\mathrm{Po}(\bar\lambda_r)=x)\left(1-\frac{\Pr(\mathrm{Po}(\bar\lambda_r)=x+K)}{\Pr(\mathrm{Po}(\bar\lambda_r)=x)}\right)^+ && \text{(change variable)}\\
=&\sum_{x\geq 0}\Pr(\mathrm{Po}(\bar\lambda_r)=x)\left(1-\frac{(\bar\lambda_r)^K}{(x+1)(x+2)\dots(x+K)}\right)^+
\end{aligned}$$

Because $(x+1)(x+2)\dots(x+K)$ is increasing as x increases, there exists x^* such that $(\bar\lambda_r)^K\leq(x+1)(x+2)\dots(x+K)$ if and only if $x\geq x^*$. Therefore,

$$\begin{aligned}
&d_{TV}(K+\mathrm{Po}(\bar\lambda_r),\mathrm{Po}(\bar\lambda_r))\\
=&\sum_{x\geq 0}\Pr(\mathrm{Po}(\bar\lambda_r)=x)\left(1-\frac{(\bar\lambda_r)^K}{(x+1)(x+2)\dots(x+K)}\right)^+\\
=&\sum_{x\geq x^*}\Pr(\mathrm{Po}(\bar\lambda_r)=x)\left(1-\frac{(\bar\lambda_r)^K}{(x+1)(x+2)\dots(x+K)}\right)\\
=&\Pr(\mathrm{Po}(\bar\lambda_r)\geq x^*)-\Pr(\mathrm{Po}(\bar\lambda_r)\geq x^*+K)\\
=&\sum_{x=x^*}^{x^*+K-1}\Pr(\mathrm{Po}(\bar\lambda_r)=x)\leq K\max_x\Pr(\mathrm{Po}(\bar\lambda_r)=x)
\end{aligned}$$

Now we want to show $\max_x\Pr(\mathrm{Po}(\bar\lambda_r)=x)=o(1)$. Intuitively, since the expectation $\bar\lambda_r=\omega(1)$ is large, the probability mass function $\Pr(\mathrm{Po}(\bar\lambda_r)=x)$ is "flat", and the maximum of the probability mass function is small. Formally, for all x, $\Pr(\mathrm{Po}(\bar\lambda_r)=x+1)/\Pr(\mathrm{Po}(\bar\lambda_r)=x)=\bar\lambda_r/(x+1)$, so the maximum happens at $x_M:=\lfloor\bar\lambda_r\rfloor$.Then we can compute an upper bound of $\Pr(\mathrm{Po}(\bar{\lambda}_r) = x_M)$ by Stirling approximations.

$$\begin{aligned}
\Pr\left(\mathrm{Po}(\bar{\lambda}_r) = x_M\right) = \frac{(\bar{\lambda}_r)^{x_M} e^{-\bar{\lambda}_r}}{x_M!} &\le \frac{(\bar{\lambda}_r)^{x_M} e^{-\bar{\lambda}_r}}{\sqrt{2\pi} x_M^{x_M+1/2} e^{-x_M}} && \text{(Stirling's approximation [30])} \\
=& \frac{1}{\sqrt{2\pi} x_M^{1/2}} \cdot \frac{e^{-\bar{\lambda}_r}}{e^{-x_M}} \cdot \left(\frac{\bar{\lambda}_r}{x_M}\right)^{x_M} \le \frac{1}{\sqrt{2\pi x_M}} \cdot \left(\frac{\bar{\lambda}_r}{x_M}\right)^{x_M} && (\bar{\lambda}_r \ge x_M) \\
\le& \frac{1}{\sqrt{2\pi x_M}} \cdot \left(1 + \frac{\bar{\lambda}_r - x_M}{x_M}\right)^{x_M} \le \frac{1}{\sqrt{2\pi x_M}} \cdot \left(1 + \frac{1}{x_M}\right)^{x_M} \le \frac{e}{\sqrt{2\pi x_M}} = o(1)
\end{aligned}$$

The last one holds because $x_M = \lfloor \bar{\lambda}_r \rfloor = \omega(1)$. □

Proof of Claim C.4. Because the distributions of $\boldsymbol{\chi}$ and $\bar{\boldsymbol{\chi}}'$ are very close to product distributions, the relative entropy between them is easier to compute than the total variation distance. By Pinsker's inequality, if the relative entropy is small, the total variation distance is also small.

$$\begin{aligned}
D_{KL}(\boldsymbol{\chi} \| \bar{\boldsymbol{\chi}}') &= - \sum_{\mathbf{k}: \sum_{\ell=0}^{r} k_\ell = n} \Pr(\boldsymbol{\chi} = \mathbf{k}) \log \frac{\Pr(\bar{\boldsymbol{\chi}}' = \mathbf{k})}{\Pr(\boldsymbol{\chi} = \mathbf{k})} \\
&= - \sum_{\mathbf{k}: \sum_{\ell=0}^{r} k_\ell = n} \Pr(\boldsymbol{\chi} = \mathbf{k}) \log \left(\frac{\prod_{\ell=0}^{r} \frac{\bar{\lambda}_\ell^{k_\ell} e^{-\bar{\lambda}_\ell}}{k_\ell!}}{\prod_{\ell=0}^{r} \frac{\lambda_\ell^{k_\ell} e^{-\lambda_\ell}}{k_\ell!}} \right) && \text{(by Eqn. (C.2) and (C.4))} \\
&= - \sum_{\mathbf{k}: \sum_{\ell=0}^{r} k_\ell = n} \Pr(\boldsymbol{\chi} = \mathbf{k}) \left(\sum_{\ell=0}^{r} k_\ell \log \frac{\bar{\lambda}_\ell}{\lambda_\ell} \right) && \text{(because } \textstyle\sum_{\ell=0}^{r} \lambda_\ell = \sum_{\ell=0}^{r} \bar{\lambda}_\ell) \\
&= - \sum_{\mathbf{k}: \sum_{\ell=0}^{r} k_\ell = n} \Pr(\boldsymbol{\chi} = \mathbf{k}) \left(\sum_{\ell=0}^{r-1} k_\ell \log \left(1 - \frac{K}{n}\right) + k_r \log \left(1 + (1/p_r - 1) \frac{K}{n}\right) \right)
\end{aligned}$$

In the outermost parentheses, everything except k_ℓ and k_r are independent of the summation over $\mathbf{k}$, so we can simplify it as the following:

$$\begin{aligned}
&D_{KL}(\boldsymbol{\chi}\|\bar{\boldsymbol{\chi}}')\\
=&-\left[\log\left(1-\frac{K}{n}\right)\sum_{\mathbf{k}}\Pr(\boldsymbol{\chi}=\mathbf{k})\left(\sum_{\ell=0}^{r-1}k_\ell\right)+\log\left(1+(1/p_r-1)\frac{K}{n}\right)\sum_{\mathbf{k}}\Pr(\boldsymbol{\chi}=\mathbf{k})k_r\right]\\
=&-\left[\log\left(1-\frac{K}{n}\right)\sum_{\ell=0}^{r-1}\mathbb{E}[\chi_\ell]+\log\left(1+(1/p_r-1)\frac{K}{n}\right)\mathbb{E}[\chi_r]\right]\\
=&-\left[(n-\lambda_r)\log\left(1-\frac{K}{n}\right)+\lambda_r\log\left(1+(1/p_r-1)\frac{K}{n}\right)\right] && (\mathbb{E}[\chi_\ell]=\lambda_\ell)\\
=&-n\left[(1-p_r)\log\left(1-\frac{K}{n}\right)+p_r\log\left(1+(1/p_r-1)\frac{K}{n}\right)\right] && (\lambda_r=np_r)
\end{aligned}$$

Now we want to show $(1-p_r)\log\left(1-\frac{K}{n}\right)+p_r\log\left(1+(1/p_r-1)\frac{K}{n}\right)$ is $o(1/n)$. Because $p_r=\Pr(Y_i\geq r)=\Theta(\lambda^r)=\omega(1/n)$ and K is a constant, we can use Taylor expansion to approximate both logs at 1,

$$\begin{aligned}
&(1-p_r)\log\left(1-\frac{K}{n}\right)+p_r\log\left(1+(1/p_r-1)\frac{K}{n}\right)\\
=&-(1-p_r)\frac{K}{n}+p_r(1/p_r-1)\frac{K}{n}+O\left(\frac{1}{n^2}\right)+O\left(\frac{1}{p_rn^2}\right)\\
=&O\left(1/(p_rn^2)\right)=o(1/n) && (\text{because } p_rn=\omega(1))
\end{aligned}$$

Therefore, we have $D_{KL}(\boldsymbol{\chi}\|\bar{\boldsymbol{\chi}}')=o(1)$. By Pinsker's inequality

$$d_{TV}(\boldsymbol{\chi},\bar{\boldsymbol{\chi}}')\leq\sqrt{\frac{1}{2}D_{KL}(\boldsymbol{\chi}\|\bar{\boldsymbol{\chi}}')}=o(1).$$

□

C.2 Proof of Proposition 8.15

By Theorem 8.7 and Corollary 8.10, if no leaf is activated by the local seeds, then there can be at most constantly many infected vertices with high probability. Consider an arbitrary vertex v that is not infected, and let t be the leaf such that $v\in V(t)$. Let K_{in} be the number of infected vertices in $V(t)$ after Phase I and K_{out} be the number of infected vertices outside $V(t)$. By our assumption, $K_{in}=O(1)$ and $K_{out}=O(1)$. We compute an upper bound on the probability that v is infected in the next cascade iteration. Let X_v be the number of v's infected neighbors in $V(t)$ and Y_v be the number of v's infected neighbors outside $V(t)$.

Since the probability that v is connected to each of those K_{out} vertices is $o(n^{-1/r})$, we have

$$\Pr(Y_v\geq r-a)\leq\binom{K_{out}}{r-a}\left(o(n^{-1/r})\right)^{r-a}=o\left(n^{-(r-a)/r}\right)$$

for each $a \in \{0, 1, \ldots, r-1\}$.

Ideally, we would also like to claim that

$$\Pr(X_v \geq a) \leq \binom{K_{in}}{a} w(t)^a = O\left(n^{-a/r}\right), \tag{C.5}$$

so that putting together we have,

$$\Pr(v \text{ is infected}) \leq \sum_{a=0}^{r-1} \Pr(X_v \geq a) \Pr(Y_v \geq r-a) = r \cdot O\left(n^{-a/r}\right) \cdot o\left(n^{-(r-a)/r}\right) = o\left(\frac{1}{n}\right).$$

and conclude that the expected number of infected vertices in the next iteration is $o(1)$, which implies the proposition by the Markov's inequality.

However, conditioning on the cascade in $V(t)$ stopping after K_{in} infections, there is no guarantee that the probability an edge between v and one of the K_{in} infected vertices is still $w(t)$. Moreover, for any two vertices u_1, u_2 that belong to those K_{in} infected vertices, we do not even know if the probability that v connects to u_1 is still independent of the probability that v connects to u_2. Therefore, (C.5) does not hold in a straightforward way. The remaining part of this proof is dedicated to prove (C.5).

Consider a different scenario where we have put K_{in} seeds in $V(t)$ (instead of that the cascade in $V(t)$ ends at K_{in} infections), and let $\bar{X}_v$ be the number of edges between v and those K_{in} seeds (where v is not one of those seeds). Then we know each edge appears with probability $w(t)$ independently, and (C.5) holds for $\bar{X}_v$:

$$\Pr(\bar{X}_v \geq a) \leq \binom{K_{in}}{a} w(t)^a = O\left(n^{-a/r}\right).$$

Finally, (C.5) follows from that $\bar{X}_v$ stochastically dominates X_v (i.e., $\Pr(\bar{X}_v \geq a) \geq \Pr(X_v \geq a)$ for each $a \in \{0, 1, \ldots, r-1\}$), which is easy to see:

$$\Pr\left(X_v \geq a\right) = \Pr\left(\bar{X}_v \geq a \mid \bar{X}_v \leq r-1\right) = \frac{\Pr(a \leq \bar{X}_v \leq r-1)}{\Pr(\bar{X}_v \leq r-1)}$$

$$= \frac{\Pr(\bar{X}_v \geq a) - \Pr(\bar{X}_v \geq r)}{1 - \Pr(\bar{X}_v \geq r)} \leq \Pr\left(\bar{X}_v \geq a\right),$$

where the first equality holds as $\Pr\left(\bar{X}_v \geq a \mid \bar{X}_v \leq r-1\right)$ exactly describes the probability that v has at least a infected neighbors among K_{in} conditioning on v has not yet been infected.