Maximizing Expected Value of Information in Decision Problems by Querying on a Wish-to-Know Basis

by

Robert W. Cohn

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
2016

Doctoral Committee:

Professor Satinder Singh Baveja, Co-Chair
Professor Edmund H. Durfee, Co-Chair
Professor Richard L. Lewis
Professor Michael P. Wellman

2016

To my late grandfather, Seymour B. Cohn, whose seminal contributions to engineering and casual mastery of seemingly all matters of life will always inspire me.

A C K N O W L E D G M E N T S

First and foremost I would like to thank my co-advisors, Satinder Singh and Ed Durfee, for their combined efforts in training me as a researcher. I am deeply grateful for Ed's tireless guidance and patience throughout the entirety of my experience at Michigan, always making himself available whenever I needed help in overcoming obstacles, whether they be research-related or life-related, and his willingness to explore the theoretical paths I am drawn to while training me to properly ground my work in rigorous empirical studies. I am also deeply grateful for Satinder's efforts in conveying to me his vast array of sharp insights into research, regarding technical and practical matters alike, and his welcoming aid in my theoretical endeavors despite my humble mathematical background.

I also thank my other doctoral committee members, Michael Wellman and Rick Lewis, for their consistently insightful feedback throughout the proposal, dissertation defense, and thesis writing stages of my time at Michigan. In particular, I could always count on Michael Wellman to ask sharp questions, which often led to fruitful discussions that helped to shape some of the deepest facets of my research, and I thank Rick for his encouragement early on that motivated me to continue pursuing the theoretical avenues that ended up becoming the foundation of my dissertation.

I would like to thank the AI faculty of the Michigan CSE department as a whole for their willingness to openly discuss and debate research topics in nearly any context, and I am deeply grateful to John Laird for his personal involvement in arranging the initial funding that brought me to Michigan to begin as a masters student. I also thank the staff of the CSE department as a whole, and I am particularly grateful to Dawn Freysinger for her cheerful help on numerous occasions in guiding me through the formal processes required at various points in the program.

I also thank my colleagues, especially my group, past and present. In particular, I am grateful to Jonathan Sorg, Erik Talvitie, and Alex Kulesza for their mentorship, and to Nan Jiang, Jeshua Bratman, Ananda Narayan, and XiaoXiao Xu for the conversations and comradery we shared.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Maximizing Expected Value of Information in Decision Problems by Querying on a Wish-to-Know Basis

by

Robert W. Cohn

Co-Chairs: Satinder Singh Baveja and Edmund H. Durfee

An agent acting under uncertainty regarding how it should complete the task assigned to it by its human user can learn more about how it should behave by posing queries to its human user, provided its user is capable of responding to them. Asking too many queries, however, may risk requiring undue attentional demand of the user, and so the agent should prioritize asking the most valuable queries. For decision-making agents, Expected Value of Information (EVOI) measures the value of a query, and so given a set of queries the agent can ask, the agent should ask the query that is expected to maximally improve its performance by selecting the query with highest EVOI in its set.

Unfortunately, to compute the EVOI of a query, the agent must consider how each possible response would influence its future behavior, which makes query selection particularly challenging in settings where planning the agent's behavior would be expensive even without the added complication of considering queries to ask, especially when there are many potential queries the agent should consider. The focus of this dissertation is on developing

query selection algorithms that can be feasibly applied to such settings.

The main novel approach studied, Wishful Query Projection (WQP), is based on the intuition that the agent should consider which query to ask on the basis of obtaining knowledge that would help it resolve a particular dilemma that it *wishes* could be resolved, as opposed to blindly searching its entire query set in hopes of finding one that would give it valuable knowledge. In implementing WQP, this dissertation contributes algorithms that are founded upon the following novel result: for myopic settings, when the agent can ask any query as long as the query has no more than some set number of possible responses, the best query takes the form of asking the user to choose from a specified subset of ways for the agent to behave. Through a combination of empirical and theoretical analysis, the work presented shows that WQP selects queries with near-optimal EVOI when the agent's query set is (1) balanced in the range of queries it contains; and (2) rich in terms of the highest contained query EVOI.

CHAPTER 1

Introduction

> *"If information value theory and associated decision theoretic structures do not in the future occupy a large part of the education of engineers, then the engineering profession will find that its traditional role of managing scientific and economic resources for the benefit of man has been forfeited to another profession."*
> – Ronald A. Howard, 1966.

Designing agents to effectively represent and reason over their uncertainty about their world is central in artificial intelligence, since for many settings the agent can benefit by adapting its behavior to take advantage of knowledge it gains as it acts in the world that cannot be feasibly specified beforehand. For instance, it may be infeasible to provide the agent with a complete model of its environment, or it may be that the agent's task depends on unclear and/or ambiguous instructions provided by a human user. As an example, consider a web assistant agent tasked by its human user to recommend housing options in Boston. The agent is unlikely to have complete information regarding the user's preferences among housing options in trading off factors such as cost, proximity to transportation hubs, population density, neighborhood safety, and structural properties.

Empowering such a housing recommendation agent with the ability to *query* its user for more information can allow it to dynamically adapt its planning process according to the individual preferences of its user. For example, suppose the agent can ask its user any yes/no question. The agent may be able to significantly narrow the space of options it considers by asking the user whether she would be willing to pay up to x additional dollars to live on a cul-de-sac, whether living near a subway stop would be worth living in an area with unreliable parking access, or whether she would prefer a particular housing option over another. With a set of such queries at its disposal, the agent can acquire valuable knowledge about its user's preferences that it can use to improve its task performance.

Beyond the challenges of designing the agent's representation of its environment, its uncertainty representation, its space of decisions, and the set of queries it can ask, the computational problem of determining what the agent should ask presents significant challenges. In principle, the agent should ask the query that it expects, upon learning its response, will best improve its ability to make good decisions – that is, it should ask the query with highest Expected Value of Information (EVOI). However, computing the EVOI associated with a particular query requires the agent to reason about how asking the query would update its knowledge and allow it to improve its own decision-making process, implying that the computational complexity of EVOI-based query selection is inherently tied to that of optimal planning.

Selecting from a large query set, then, can present significant computational challenges in settings where optimal planning under uncertainty would be expensive even if the agent did not consider any queries. In such settings, researchers typically either consider small query sets that can be exhaustively searched (which limits the agent's ability to be selective about what knowledge it gains), or employ query selection approaches that do not offer formal approximation guarantees. An example of the latter approach is to select queries on the basis of some measure of global uncertainty reduction instead of EVOI, which can often be done more efficiently at the cost of sacrificing the quality of the query selected. The cost can be substantial, however, in situations where the agent is uncertain about many factors of its world, but only a key subset play an important role in the agent's decision problem. For example, suppose the housing recommendation agent is highly uncertain about the user's preferences regarding high altitudes. Since altitude is not a pertinent factor when only considering housing options in Boston, the agent should focus its queries on learning about more important factors even though the agent might be less uncertain about them.

In this dissertation I study the computational problem of selecting the query with maximum expected value of information for decision-making agents, focusing on the relationship between uncertainty reduction and expected value of information. Drawing from insights I obtain through analytical and empirical comparisons of techniques for query selection, I contribute a new approach for query selection called *Wishful Query Projection (WQP)* founded upon the following principle. Selecting the query that maximizes expected value of information can be efficiently approximated by factoring the problem into two steps: first, find a good ("wishful") query to ask without considering what the agent can ask, and second, select a query that the agent *can* ask that reduces the agent's uncertainty in a similar way. Towards better understanding the computational properties and structure of expected value of information, along with the potential effectiveness of utilizing WQP

for query selection, I present research that focuses on examining the structure and computational properties of the problems posed by each of the aforementioned two steps of WQP. In addition, I present an empirical study that evaluates the closeness of approximation and computational savings afforded by WQP algorithms compared to existing uncertainty-based techniques and brute-force EVOI maximization.

1.1 Problem Statement

This dissertation is concerned with the computational problem of how an agent can select the query with maximum Expected Value of Information (EVOI) from some specified askable set, in settings where optimal planning is computationally expensive. Here, the EVOI of a query corresponds to the expected improvement in value associated with the agent improving its decision as a function of the query's response compared to the best decision before asking the query, and hence EVOI is the natural criterion for query selection when queries are used to improve the agent's performance in its decision problem.

However, the fact that EVOI directly measures expected value improvement means that the complexity of computing EVOI is related to the complexity of optimal planning: in order to compute the EVOI of a query, the agent must, for each possible response to the query, compute the posterior Bayes-optimal decision value under the associated posterior distribution, which requires optimal planning. EVOI-based query selection for large query sets, then, presents a challenge in settings where optimal planning would be expensive even if the agent did not consider queries. Lest query selection be limited to small query sets that limit the extent to which the agent can acquire valuable knowledge in a variety of states of uncertainty, the number of EVOI computations that need to be performed should be sublinear in the size of the query set, or ideally, independent of the size of the query set. For some cases, structure in the query set, decision problem, and/or form of uncertainty can allow only a subset of the queries to be considered at no penalty (Viappiani and Boutilier, 2010) or can allow EVOI computations to be simplified at no penalty (Bonilla et al., 2010a). More generally, a good query selection algorithm must either replace EVOI computations with a simpler comptuation, restrict attention to only a subset of queries, or both, while offering formal approximation guarantees regarding the query they select relative to the best query in the set.

This dissertation evaluates EVOI-based query selection algorithms on the basis of:

(A) The complexity of the algorithm relative to the complexity of optimal planning and the size of the query set,

(B) The extent to which the algorithm offers formal guarantees pertaining to the EVOI of the query selected relative to the EVOI of the best query in the agent's query set; and

(C) The empirical performance of the algorithm as compared to brute-force EVOI maximization, baselines, and applicable state-of-the-art query selection algorithms, measured by

  (C1) The EVOI of the query selected, as a function of relevant properties of the agent's query set and/or decision problem; and

  (C2) The computation time (in seconds) the algorithm spent to select a query, as a function of the size of the query set.

I have presented a high-level description of the query selection problem this dissertation will examine towards providing researchers/practitioners new algorithmic tools for designing decision-making agents that intelligently query their human users; in Section 2.2 I will formally specify the first two points above by defining what it means for a query selection algorithm to be efficient and approximate in the context of this dissertation.

1.2 Approach

As I discussed in Section 1.1, the research presented in this dissertation is directed towards developing query selection algorithms that can be feasibly applied to select from large query sets on the basis of maximizing expected value of information (EVOI) in settings where optimal planning computations serve as the main bottleneck in performing EVOI computations. The algorithms developed in this dissertation limit the number of optimal planning computations that must be performed by restricting EVOI computations to a space that can be efficiently searched, and I describe two novel approaches for doing so next.

1.2.1 Uncertainty-based Query Filtering

The first approach is to restrict attention to a promising subset of the queries based on how much they are expected to reduce the agent's uncertainty, without considering whether or not they are actually useful for improving the agent's behavior, and then computing EVOI only for this manageable subset. The first hypothesis of this dissertation is that selecting a query in this hybrid fashion can find queries with higher EVOI compared to selecting queries on the basis of EVOI (computed directly) or on the basis of uncertainty reduction, given the same computational constraints. I evaluate a heuristic version of this approach to query selection in a sequential decision making setting in Chapter 3.

1.2.2 Decision Query Projection

The second approach is the focus of most of this dissertation, and applies the following intuition. Consider the thought process a human might take leading up to asking a good question. Certainly, one would not proceed by enumerating all questions that can be generated by natural language. Instead, one would perform this search with some kind of goal in mind. For example, reconsidering the housing option example but where a human realtor assumes the role of the agent, the realtor may have two neighborhoods containing properties that stand out as the best candidates according to what she knows about her customer's preferences, but can only show options in one of the neighborhoods due to time constraints. Then, even though she cannot realistically ask the customer which of all of the options in the two neighborhoods would be best, there might be a factor that options in each neighborhood have in common, such as whether or not they are close to good schools. Then, the realtor might ask the customer whether they have children in order to narrow down which neighborhood would be most promising to consider. Here, the realtor would never consider questions about less important factors like the preferred color of the house, or about negligibly related factors such as their political affiliation.

The key takeaway here is that the agent should search its query set on the basis of finding a query that can give it information which it has already determined to be useful, instead of blindly searching its query set in the hope that it can find a query that will give it useful information. More specifically, if the agent can efficiently find an ideal query (henceforth, a *wishful* query) that it would select if only it could be asked, it can search the set of queries that it *can* ask on the basis of finding a query similar to that wishful query in terms of *how* they reduce the agent's uncertainty. With this approach the agent restricts EVOI computations to an easily searchable *wishful query set*, and uses the result to drastically simplify the task of searching its query set so that it becomes a *query projection* problem which no longer requires performing optimal planning computations. I will refer to this abstract query selection approach as *Wishful Query Projection* (WQP).

A concrete WQP algorithm requires specifying the following two components:

1. *Wishful query selection:* Specify and efficiently search an appropriate space of queries to obtain a wishful query; and

2. *Query projection:* Efficiently find a query that the agent can ask that best approximates the wishful query.

The second and main hypothesis of this dissertation is that WQP can be implemented to yield efficient and approximate query selection algorithms (I specify technical definitions

for efficient and approximate in Section 2.2). I consider wishful query selection in Chapter 4 and query projection in Chapter 5, and then study the empirical performance of several WQP algorithms in Chapter 6 in terms of how they depend on key properties of the agent's query set.

1.3 Contributions

This dissertation provides four main contributions to the decision-theoretic knowledge acquisition community. Two of these contributions are query selection algorithms that are designed to approximately maximize EVOI while avoiding optimal planning computations to the extent possible. The other two contributions are theoretical results that are used in this dissertation to derive one of these algorithms (an implementation of the abstract Wishful Query Projection (WQP) approach discussed in the previous section), and also serve as contributions in their own rights. I provide brief descriptions of each contribution, ordered by the chapters in which they appear, below.

Hybrid Algorithm. The first algorithm, introduced in Chapter 3, is designed for action query selection in sequential decision-making settings. To select a query while avoiding optimal planning computations, the Hybrid algorithm begins by employing an algorithm contributed by related work to cheaply select a promising subset of queries to consider, and then Hybrid selects a query by performing EVOI computations (which require optimal planning) only for queries in this subset. Also in Chapter 3, Hybrid is empirically evaluated in a car-driving problem.

Wishful Query Selection. Chapter 4 investigates an abstract setting where the agent can ask any query, so long as its query has exactly k possible responses. The main result is that the agent can ignore all queries except those contained in a particular subset. Specifically, the agent can restrict its attention to decision queries, which are queries that ask the user to select their preferred decision out of a specified set. This result provides insight into the question of what types of queries the agent should consider.

Decision Query Projection. Chapter 5 investigates the problem of efficiently finding a query that the agent can ask that has similar EVOI to that of a particular decision query. The main result is that the loss in EVOI incurred by replacing a decision query with an arbitrary query can be upper bounded as a function of how uncertain the agent would be on average, after asking the replacement query, about what the response would have been if it

had asked the decision query instead.

DEER Algorithm. The second algorithm, DEER, is introduced in Chapter 5, and implements the abstract WQP approach discussed in the previous section. DEER draws from the second and third contributions described above to implement the two steps of WQP, and by combining their accompanying theoretical results, upper bounds on the loss in EVOI in asking DEER's selected query compared to the best query in the set are derived. DEER is empirically evaluated in a housing option recommendation problem in Chapter 6.

I note that the first and second contributions are published (Cohn et al., 2011, 2014) while the third and fourth have yet to be published at the time of writing.

CHAPTER 2

Background

In this dissertation I consider an agent that is faced with acting according to the uncertain preferences of its user, and focus on how the agent can efficiently solve the *query selection problem*: that is, how the agent can efficiently determine which of some allowed set of queries would be best to ask its user. In this chapter I begin by introducing the technical concepts that are core in this dissertation, and then I use them to formalize and expound upon the problem statement I outlined in Section 1.1. Then I present a brief survey of the rich body of related work to this problem, highlighting the associated limitations that the contributions of this dissertation address.

2.1 Preliminaries

Here I present a general framework to formalize (1) the uncertain decision problem faced by the agent, (2) the semantics of queries; and (3) the Expected Value of Information criterion used in this dissertation to measure the value of queries. I formalize the settings I consider in each subsequent chapter as special cases of this framework, except for Chapter 5 where I adopt this framework in its full generality.

2.1.1 Decision-Making under Uncertainty

I assume that the agent's uncertainty takes the form of some arbitrary distribution ψ over a (possibly continuous) parameter space Ω, where each $\omega \in \Omega$ specifies a possible model. Throughout this disseration I refer to ψ as the agent's "uncertainty." Examples include uncertainty over parameter values in parameterized reward functions or transition functions, as well as other noise parameters in decision problems. I assume that the agent is faced with the one-shot decision problem of selecting from a finite set U of possible decisions (such as which of a set of housing options to recommend to a human user), where each

decision could for instance be an action, an open-loop plan, or a policy even though the agent's choice among them is single-shot.

I emphasize that when I refer to the agent's "decision", I am referring to the agent's future behavior (however that is expressed); this is not to be confused with an individual action taken in a sequential decision making setting. For each $\omega \in \Omega$ and decision $u \in U$, there is an associated value denoted V_ω^u that captures the expected utility of decision u in model ω. To illustrate, in the housing recommendation example discussed in Chapter 1, the decision set could be a set of possible housing options to recommend to the human user, and the space of models could correspond to a space of weight vectors, specifying the relative values of neighborhood safety versus saving monetary costs versus proximity to transportation hubs, with the value associated with each possible housing option being a linear function of the uncertain weight vector.

The expected value of decision u under distribution ψ can be written as

$$V_\psi^u = \mathbb{E}_{\omega \sim \psi}[V_\omega^u] = \int_\Omega \psi(\omega) V_\omega^u d\omega,$$

where $\omega \sim \psi$ denotes model-parameters (ω) sampled from distribution ψ over Ω, and the Bayes-optimal decision that maximizes expected value under ψ is

$$u_\psi^* = \arg\max_{u \in U} \{V_\psi^u\},$$

and the associated Bayes-optimal value is

$$V_\psi^* = \max_{u \in U} \{V_\psi^u\} = V_\psi^{u_\psi^*}.$$

Throughout this thesis, I often informally refer to the problem of computing Bayes-optimal decisions/values as "optimal planning".

Note that while I have assumed that the agent has some way to compute the value V_ω^u of decision u in some model ω (and Bayes-optimal decisions/values for uncertainty ψ), I have purposely not committed to how it does so because my focus here is on developing and studying query selection algorithms that do not depend on any special structure in the decision model or form of uncertainty, leaving room for these algorithms to be adapted to take advantage of any available known structure in practice. However, as I discussed in Section 1.1 this disseration is focused on settings where optimal planning is expensive. For instance, in the housing recommendation example discussed in the previous chapter,

optimal planning could be expensive if the agent's decision problem were to take the form of a valued constraint satisfaction problem (which are known to be NP-hard (Cohen et al., 2006)), or optimal planning could be expensive simply due to the need to consider a large number of candidate housing options.

2.1.2 Querying

I assume that the agent can ask a single query and observe its response before executing a decision. Next I define queries and show how their responses can be used to improve the agent's decision. First, note that ψ, the distribution over model parameters, is a sufficient statistic of the agent's knowledge about which decision is best, and thus any information relevant to the value of decisions that the agent receives as a response to a query can be incorporated via a Bayes update to ψ. Specifically, assume the agent poses query q to the user, and the user responds with the j^{th} out of a finite number of possible responses to q. Then the posterior distribution of ψ given response j to query q is written as

$$\psi(\omega|q=j) = \frac{Pr(q=j|\omega)\psi(\omega)}{\int_\Omega Pr(q=j|\omega')\psi(\omega')d\omega'}. \tag{2.1}$$

Note that the likelihood function associated with a query q, $\Pr(q=j|\omega)$ for each response j, completely defines how asking q and observing its response can impact the agent's knowledge. Intuitively, this is the agent's model for how the user would respond to the query conditioned on parameter ω, and so the agent's prediction for what the response to the query would be if the agent were to ask the query depends on the agent's uncertainty ψ over ω.

I will use $\psi|Y=y$ to denote the posterior distribution induced when ψ is updated to incorporate the knowledge that $Y=y$ for a random variable Y. After receiving response j for query q, the agent's best decision based on the updated knowledge is $u^*_{\psi|q=j}$ with a new expected value of $V^*_{\psi|q=j}$. Note that by abuse of notation, the query q is treated here as a random variable over responses j with its distribution implicitly determined by ψ. This allows the query's mathematical effect on the agent's uncertainty to be considered directly, abstracting away its semantic meaning. The expected value of asking query q, denoted $V^*_{\psi|q}$, is thus $\mathbb{E}_{j\sim q;\psi}[V^*_{\psi|q=j}]$, where the expectation is over the probability of response j to query q given prior knowledge ψ. Note that ψ here represents the parameters that determine the agent's uncertainty, as opposed to a realization of ω, which I signify with a semi-colon. For clarity, throughout this dissertation I use semi-colons instead of conditional symbols "|" in order to distinguish functional dependencies of distributions on non-random parameters,

from distributions resulting from conditioning on realizations of random variables.

In expectation, if a query and its response induces a new Bayes-optimal decision, that decision can only improve the agent's expected value:

$$\begin{aligned} V^*_\psi &= \max_u \mathbb{E}_{\omega\sim\psi}[V^u_\omega] = \max_u \left[\mathbb{E}_{j\sim q;\psi}\mathbb{E}_{\omega\sim\psi|q=j}[V^u_\omega]\right] \\ &\leq \mathbb{E}_{j\sim q;\psi}\left[\max_u \mathbb{E}_{\omega\sim\psi|q=j}[V^u_\omega]\right] \text{ (by Jensen's inequality)} \\ &= \mathbb{E}_{j\sim q;\psi}[V^*_{\psi|q=j}] = V^*_{\psi|q}. \end{aligned}$$

This disseration is concerned with the question of how an agent decides what to ask its human user. The Expected Value of Information ($EVOI$) associated with query q measures the expected increase in value associated with asking q, updating to a new Bayes-optimal decision, and never asking another query:

$$\begin{aligned} EVOI(q;\psi) &= \mathbb{E}_{j\sim q;\psi}\left[V^*_{\psi|q=j} - V^{u^*_\psi}_{\psi|q=j}\right] && (2.2) \\ &= \mathbb{E}_{j\sim q;\psi}\left[V^*_{\psi|q=j}\right] - \mathbb{E}_{j\sim q;\psi}\left[V^{u^*_\psi}_{\psi|q=j}\right] \\ &= V^*_{\psi|q} - V^*_\psi. && (2.3) \end{aligned}$$

Equation 2.3 holds since the law of iterated expectations directly implies that $\mathbb{E}_{j\sim q;\psi}\left[V^{u^*_\psi}_{\psi|q=j}\right] = V^{u^*_\psi}_\psi$. Intuitively, this can be understood as follows: prior to asking the query q, the set of posterior distributions, and their probabilities of occurring, are directly determined by the agent's prior distribution ψ; hence, the expected value of the agent's prior Bayes-optimal decision should not change when evaluated as an expectation across each possible posterior distribution, even though it may change for each posterior individually.

Lastly, note that for any q and ψ, $EVOI(q;\psi) \geq 0$ (to see this, consider Equation 2.2 and note that $V^*_{\psi|q=j} \geq V^{u_\psi}_{\psi|q=j}$ for any q, j, and u).

Note that in this setting, the agent can only ever ask a single query, as opposed to settings where the agent can select a conditional sequence of queries (query trees). When I consider the latter setting, I explicitly refer to it as the *nonmyopic setting* to distinguish it from the setting specified here, which I refer to as the *myopic* setting since researchers often assume that the agent can only ask a single query as an approximation in settings where it can ask multiple queries (Dittmer and Jensen, 1997).

I note that using myopic or nonmyopic to characterize properties of settings is an abuse of terminology, since these terms are usually used to characterize agent reasoning processes as opposed to settings in which they act. In particular, *single-shot* and *sequential* would be

the more conventional terms for how myopic and nonmyopic are used here, but I use this notation in order to make clear the distinction between how the query selection problem is characterized (myopic versus nonmyopic) and how the posterior decision-making problem is characterized (single-shot versus sequential).

2.2 Query Selection Problem

I now formalize the query selection problem considered in this dissertation, and specify the non-empirical properties of query selection algorithms that this dissertation aims to achieve, as outlined in Section 1.1.

A *Q-EVOI-optimal query* (sometimes referred to as a myopically optimal query selected from Q) for some *askable query set* Q maximizes $EVOI$ given prior knowledge ψ:

$$q^* = \arg\max_{q \in Q} [V^*_{\psi|q} - V^*_{\psi}] = \arg\max_{q \in Q} V^*_{\psi|q}.$$

When the context is clear, I will often refer to a Q-EVOI-optimal query as simply EVOI-optimal, and I will assume that queries contained in Q each have exactly k possible responses for some fixed, finite $k \geq 2$ unless otherwise noted.

The *query selection problem* is to compute q^*. The main objective of this dissertation is to contribute algorithms for solving the query selection problem *approximately* and *efficiently*. More specifically, in the context of this dissertation,

- An algorithm solves the query selection problem *approximately* if it offers nontrivial formal guarantees pertaining to the EVOI of the query selected relative to the EVOI of the Q-EVOI-optimal query q^*; and
- An algorithm solves the query selection problem *efficiently* if its computational complexity is such that the number of optimal planning computations performed by the algorithm is independent of the size of the query set.

For example, consider the *Exhaustive* algorithm that evaluates the EVOI of every query in Q. Exhaustive is approximate (in this case, exact) since doing so is guaranteed to compute q^*, which is the best possible formal guarantee. However, Exhaustive is not efficient since doing so requires performing $|Q|$ EVOI computations, each of which have complexity $\mathcal{O}(kB\Pi^*)$, where B represents the complexity of a Bayesian update and Π^* represents the complexity of optimal planning under a posterior distribution (i.e., computing the value of a Bayes-optimal decision).

At the other extreme, consider the *Random* algorithm that selects a query at random. Random is not approximate since in the worst-case, where all queries but one have zero EVOI, in expectation Random selects a query with EVOI approaching zero as $|Q|$ grows since it is equally likely to choose any query in the set, independent of their relative EVOIs, and so it cannot offer any nontrivial approximation guarantee for the general case. However, Random is efficient since it selects a query with complexity that is independent of optimal planning complexity.

Now that I have formally specified the query selection problem considered in this dissertation, I survey related work and describe the associated limitations towards designing query selection algorithms that I address through the contributions of this dissertation.

2.3 Related Work

The query selection problem considered in this dissertation is related to an extremely rich body of work spanning such research communities as active learning, active sensing, preference elicitation, and bayesian experimental design, and to recurring themes in such research communities as optimal planning, reinforcement learning, and human/agent interaction. I begin the following presentation of related work by considering two possible ways to express the query selection problem studied in this dissertation as a special case of frameworks studied in related work, and then I consider the most closely related work which lies at the intersection. I would be remiss not to emphasize, however, that the vastness of related work necessarily prohibits the subsequent instances I describe from comprising an exhaustive survey.

To begin, first consider an abstract treatment of the query selection problem where querying corresponds to choosing an action in a sequential decision-making setting (that is, ignoring any structure in the effect of query responses on the agent's world). From this perspective, the query selection problem becomes a type of sequential optimal planning problem, and I discuss how the query selection problem fits into the vast literature of sequential decision-making problems in Section 2.3.1 and the limitations of treating it directly as such. In addition, since the value of a query (to the agent) relates to how it can make use of its response to improve its behavior, the extent to which the agent can efficiently perform optimal planning under uncertainty (independent of asking queries) plays a key role in how efficiently it can perform query selection; in Section 2.3.1 I also provide technical background on several types of optimal planning problems that I use to exemplify situations where optimal planning becomes the computational bottleneck.

Second, many researchers have considered problems where the objective is to determine

the best "piece of knowledge" to acquire under a variety of criteria (that is, settings where the acquired knowledge is not necessarily used to improve an agent's behavior; I give examples below), and I discuss how the query selection problem fits into the rich body of knowledge acquisition literature in Section 2.3.2. Overall, the query selection problem considered in this dissertation lies within the intersection of optimal planning problems and knowledge acquisition problems, and I make liberal use of techniques contributed in each. I discuss other work lying within this intersection in Section 2.3.3.

2.3.1 Sequential Decision-Making

At its core, the query selection problem considered in this dissertation is a type of sequential decision-making problem. Namely, the agent faces a 2-stage sequential planning problem in which the first stage corresponds to selecting a query to ask, and the second stage corresponds to acting in its world under its updated uncertainty upon observing the response to its query. Note that the second stage may, in turn, correspond to a nested sequential decision-making problem if the agent will act over multiple time steps after observing the response to its query. In this section I discuss several types of sequential decision-making problems in detail, and explain why the sequential decision-making literature does not address the challenges presented in the types of query selection problems examined in this dissertation.

2.3.1.1 Markov Decision Processes

In a *Markov Decision Process (MDP)* (Bellman, 1957), the agent's environment is modeled as a tuple $M = \langle S, A, T, R \rangle$ where the components represent the state space, action space, transition function, and reward function, respectively. Throughout this chapter I will assume that the agent's action space A is finite. At each time step, the agent observes its current state s, takes action a, probabilistically transitions to state s' according to $T(s'|s, a)$, and receives reward $R(s') \in [r_{\min}, r_{\max}]$. A policy π maps $S \to A$, and the expected value of acting according to π in MDP ω beginning from state s is denoted $V^{\pi}_{\omega}(s)$ and defined as $\sum_{t=0}^{\infty} \gamma^t r_t$, where γ is a discount factor $\in (0, 1]$ and r_t is the reward received at time t. Let Π, the agent's policy space, denote the set of all possible policies. The optimal policy for MDP ω, denoted π^*_{ω}, can be computed by algorithms from dynamic programming, e.g., value iteration or policy iteration (Sutton and Barto, 1998), or algorithms from linear programming (Bertsekas and Tsitsiklis, 1996).

2.3.1.2 Partially Observable MDPs and Belief State MDPs

Partially Observable MDPs (POMDPs) (Pineau et al., 2006) are an extension of MDPs where the agent is assumed to act in an underlying MDP $M = \langle S, A, T, R \rangle$, except the agent cannot directly observe its current state at each time step, instead maintaining a belief state based on observations it sees at each time step. At each time step the agent's belief state (probability distribution) over current states s is ψ, the agent takes action a, receives reward r according to $R(s)$, probabilistically transitions to state s' according to $T(s'|s, a)$, and then observes observation x which is stochastically emitted as a function of the action a it took in the state s it occupied and the state s' it transitioned to. More formally, the observation $x \in X$ is sampled from the observation function Ω where for all states $s, s' \in S$ and actions $a \in A$, $\sum_{x \in X} \Omega(x|s, a, s') = 1$. The agent then updates its belief state from ψ to ψ' using Bayes rule, conditioning on the action a it took and the observation x it observed (here, it is assumed that the agent can do so because it knows all parameters of the POMDP).

A *belief state MDP* representation can be constructed from a POMDP representation of an uncertain sequential decision-making problem so that the POMDP is represented directly as an MDP. In a belief state MDP, the agent's action space is the same, but its state space directly corresponds to the space of possible belief states Ψ, and Bayes updates performed by the agent to account for observations are abstracted away, instead represented directly in the transition function T. Here, $T(\psi'|\psi, a)$ corresponds to the probability that the agent's posterior belief state will be ψ' when it executes action a in belief state ψ, where the probability is determined by accounting for the distribution over observations emmitted from the distribution over next states given it executes a in ψ. The reward function $R(\psi)$, then, is the expectation of $R(s)$ over the distribution ψ over states s.

2.3.1.3 Key Challenges in Query Selection

The agent's query selection problem could conceivably be represented with either a belief state MDP or POMDP, where queries correspond to actions and query responses correspond to observations, and the agent's state in the underlying MDP corresponds to a realization ω of its uncertainty, which is not affected by the agent's actions. Such a representation might be especially natural if the agent faces a nested sequential decision-making problem upon observing the response to its query. However, solving query selection this way becomes computationally infeasible as the size of the query set increases beyond a small number of queries. For the case of using a belief state MDP, the size of the representation would grow linearly in the number of queries and their possible responses, as would the corresponding

optimal planning complexity even ignoring the complexity of constructing the representation. Worse, methods for solving POMDPs often have exponential runtime or convergence rates in the size of the observation set, and researchers typically focus on cases where the observation sets are small and the size of the state space or length of planning horizon is the limiting factor (Pineau et al., 2006).

Thus, neither belief state MDP representations nor POMDP representations afford tractable out-of-the-box solutions to the query selection problem considered in this dissertation. Instead, the approaches considered in this dissertation deal with these issues by exploiting structure in query selection problems that is not present in general sequential decision-making problems. Namely, treating the query selection problem as a generic sequential decision-making problem ignores the fact that the agent's state in the underlying MDP never changes, and so queries affect the agent's belief state only in constrained ways, compared to general state-action representations where the agent's belief state could be affected in arbitrary ways. In fact, I will show in Chapter 4 that there is structure in the space of potential uncertainty updates in the form of a partial ordering over all possible queries in terms of their EVOI, and this is the main point of leverage used by the WQP algorithms I develop in Chapter 5.

2.3.2 Knowledge Acquisition

A rich literature exists on the subject of knowledge acquisition in settings where queries have special structure and/or are evaluated on the basis of criteria other than EVOI. In fact, many settings have been considered in related work where the problem of selecting a single query to ask, however the query set and objective are defined, has exploitable structure and can be solved efficiently, either exactly or approximately. In such cases, other aspects of the problem become the computational bottleneck, such as the cost of Bayesian updates, and often researchers focus on planning query trees instead of single queries, which presents different challenges. I discuss two notable examples below.

2.3.2.1 Active Learning

Active learning (Settles, 2009) is a subfield of machine learning in which the agent actively chooses some or all of the data that it will use, typically in a classification setting. There are two crucial differences between active learning problems and the query selection problem of this dissertation. The first is that in active learning, the agent can ask only queries relating to the label of a datapoint or set of datapoints, whereas in query selection the agent's query set can contain queries that map parameters (in active learning, *hypotheses*) to arbitrary

categorical distributions. The second difference is that in active learning, the agent's objective is usually to ask queries that will best improve a measure of its classification accuracy over the space of possible datapoints when using a particular classification algorithm, as opposed to the decision-theoretic objective considered in this dissertation. Combining the two, there is often an analytical relationship between a query and its value in active learning settings which can be exploited to select queries efficiently (Cohn et al., 1996). Hence, active learning researchers typically focus on the challenges that arise in selecting conditional sequences of queries, i.e., query trees, (Nowak, 2011) for their settings, whereas in the setting considered in this dissertation, the problem of selecting a single query to ask presents significant computational challenges on its own.

2.3.2.2 Active Sensing

Researchers in active sensing (Krause and Guestrin, 2005) focus on knowledge acquistion problems that occur in a variety of engineering applications, where the main challenges involve accounting for noise in interfacing with the physical world, and effectively exploiting structure for problems of interest (Krause et al., 2008). However, in addition to directly addressing challenges that occur in practical settings, the active sensing community has studied a variety of abstract knowledge acquisition problems. In particular, researchers have studied problems where selecting multiple queries, whether they are selected in an unordered or sequential fashion, is infeasible, and a common approximation to this problem is to repeatedly select which single query to ask, without taking into account future queries (Dittmer and Jensen, 1997; Bayer-Zubek, 2004), which is termed *myopic* query selection as opposed to *nonmyopic* query selection.

Towards providing theoretical justification for myopic approximations, Krause and Guestrin (2007) show that for knowledge acquisition problems where the objective function is monotonic and submodular, selecting queries *myopically* offers powerful approximation guarantees (Nemhauser et al., 1978), and similar guarantees apply to settings where multiple queries are selected adaptively (Golovin and Krause, 2011). Although these results do not apply directly to the query selection problem considered in this dissertation, they do apply to the wishful query selection problem considered in Chapter 4 and hence are germane in the implementation and theoretical analysis of the WQP approach developed in this dissertation.

2.3.3 Querying in Decision Problems

I now discuss the most closely related work to this dissertation, where researchers consider query selection problems with objectives related to improvement in decision-making problems. A key factor distinguishing related work in query selection for decision-theoretic settings is the criterion used for selecting queries. Broadly, criteria used fall into one of four categories: Expected Value of Information (EVOI), direct uncertainty reduction, model elimination, and maximum regret minimization. Which criterion is appropriate depends on the nature of the agent's model of its decision problem. In the setting studied in this dissertation, the agent maintains an explicit distribution over models that define the value of each possible decision, and hence EVOI is the appropriate criterion since EVOI measures the expected value improvement induced by a query. In contrast, in settings where the agent similarly considers a space of models that determine the value of each possible decision but does not maintain an explicit distribution over the space of models (i.e., non-Bayesian settings), querying so as to maximize the number of models eliminated, or querying so as to minimize worst-case loss (maximum regret minimization) are examples of appropriate criteria (Braziunas, 2007). Lastly, in settings where the space of models do not prescribe value to each possible decision, but instead prescribe probabilities over which decision is optimal (Wilson et al., 2012), uncertainty-based criteria such as directly reducing the agent's uncertainty in the model space or directly reducing the agent's uncertainty in which decision is optimal may be appropriate (Abbas, 2004).

Various authors have contributed EVOI-based query selection algorithms that can be used to select queries from particular query sets defined with respect to special classes of decision problems and/or forms of uncertainty (Boutilier et al., 2003; Bonilla et al., 2010b; Dittmer and Jensen, 1997; Chajewska et al., 2000; Braziunas and Boutilier, 2005). In particular, most work in EVOI-based query selection takes place in single-shot decision-making settings where computing an optimal decision conditioned on knowledge of the true model parameter is not the computational bottleneck, and these authors tend to focus on reducing the cost of computing Bayes updates or other forms of Bayesian inference required for EVOI computation in their settings. Nevertheless, for the settings these authors consider the exhaustive algorithm for EVOI-based query selection is still impractical, albeit for different reasons, and these authors have considered ways to approximate EVOI-based query selection by exploiting structure in the agent's uncertainty representation, decision value representation, and/or query set.

For example, Viappiani and Boutilier (2010) study preference elicitation in a recommender system setting where the agent's goal is to recommend to the user an item maximizing the user's preferences, but can ask *choice queries* first, which ask the user to state

their preferred item out of a set of k options. In fact, their *items* are just a different way of referring to decisions in this dissertation, and the *noiseless choice queries* they consider correspond exactly to the *k-response decision queries* I define in Chapter 4. In these terms, they address EVOI-based query selection of k-response decision queries, and show that a greedy algorithm that incrementally builds the set of decisions comprising the decision query enjoys the guarantee that the $EVOI$ of the k-response decision query constructed is within a factor of $1 - (\frac{k-1}{k})^k$ (at worst 63%) of optimal. While their results apply directly only to variations of decision query selection problems, the fact that decision query selection can be approximated efficiently plays a crucial role in implementing the WQP approach for query selection considered in this dissertation. I will discuss their theoretical results and algorithms in great detail in Section 4.5 of Chapter 4, and I will study an extended decision query selection setting in Chapter 6.

2.4 Summary

In this chapter I formally defined the agent's uncertain decision-making problem, formally defined queries, showed how the agent can use the knowledge provided by responses to queries to potentially improve its decision, and then introduced Expected Value of Information (EVOI) as the principal metric used in this dissertation to measure the value of asking a query. Then I specified the agent's query selection problem and expounded upon the problem statement stated in Section 1.1 to define what it means for a query selection algorithm to be *efficient* and *approximate*. Finally, I presented background material on sequential decision-making and surveyed related work, placing the query selection problem considered in this dissertation at the intersection of optimal planning and knowledge acquisition. In the next chapter I study several query selection problems in sequential decision-making settings, where the challenge of dealing with expensive optimal planning computations is paramount.

CHAPTER 3

Query Selection in Sequential Decision Making

A semi-autonomous agent acting in a sequential decision-making environment should act autonomously whenever it can do so confidently, and seek help from its human user when it cannot. In this chapter I study how an agent can select a good query to ask on the basis of Expected Value of Information (EVOI) when the agent is acting under uncertainty in a sequential decision-making problem.

Recall that in Chapter 2 I formalized the agent's query selection problem as one that takes place in the context of an abstract single-shot decision-making framework, where the agent has uncertainty over a space of models that prescribe value to each of the agent's possible decisions. In this formulation, it is assumed that the agent asks a single query, observes the query's response, and then makes the posterior Bayes-optimal decision. While the step of making the posterior Bayes-optimal decision is single-shot in that no future decisions are to be made, this formulation makes no assumptions regarding structure in the agent's decision set (other than that it is of finite cardinality) or structure in how the model space maps decisions to values.

In this chapter I consider several types of query selection problems where decisions correspond to policies defined in sequential decision-making problems. In such cases, queries update the agent's knowledge about parameters of the problem (such as rewards) which in turn update the agent's knowledge about how policies map to values. Here, the main computational challenge in query selection is that the single-shot problem of selecting from the set of policies in light of the response to its query requires performing an (often) expensive optimal planning computation. This limits the number of queries that can be feasibly evaluated on the basis of EVOI, since evaluating the EVOI of a single query requires as many optimal planning computations as there are possible responses to the query.

This chapter is organized as follows. First I consider reward and transition query selection, which directly query parameters of the agent's model distribution. I show how parameterizing the agent's prior with a set of independent Dirichlet distributions allows Bayes

updates for these types of queries to be performed efficiently, and discuss approximate methods for overcoming the challenges of continuous response spaces and Bayes-optimal planning under transition/reward uncertainty. In the context of reward and transition query selection, I empirically investigate the extent to which myopic query selection limits the agent's ability to select good queries compared to nonmyopic query selection. Then I consider action queries, which are often natural query forms (Chernova and Veloso, 2009) but provide only indirect information about the agent's MDP distribution and thus present additional computational challenges. I study their application in settings where the agent has uncertainty only in its reward function, and show how this structural assumption leads to straightforward simplifications of EVOI. Even with these simplifications, however, the issue of expensive optimal planning computations limits the number of queries that can be evaluated, and I empirically compare several types of strategies for action query selection that trade off between computational requirements and selected query EVOI. I find that a hybrid method, which narrows the space of queries to consider on the basis of cheap uncertainty-reduction-based evaluations before performing expensive EVOI evaluations, can both quickly find good queries and eventually settle on near-EVOI-optimal queries given enough computation time, recommending its application in computation-restricted scenarios.

3.1 Setting

Next I introduce the Bayesian MDP representation used to represent the agent's uncertain decision-making problem in this chapter. Note that Bayesian MDPs extend MDPs, which I discussed in Section 2.3.1.1 of Chapter 2.

3.1.1 Bayesian MDPs

An agent may have uncertainty about any of the parameters of the MDP, such as the transition function T and/or the reward function R. This uncertainty translates into a distribution over possible MDPs, which in this chapter I will assume all have the same state and action spaces. However the distribution over MDPs is represented, let the agent's uncertainty be denoted ψ. The expected value for policy π under uncertainty ψ is $\mathbb{E}_{\omega\sim\psi}[V_\omega^\pi]$, where $\omega \sim \psi$ denotes parameters ω sampled from the distribution ψ over a space of parameters Ω, where the mapping from parameters $\omega \in \Omega$ to a corresponding MDP is left implicit. The *Bayes-optimal policy* under uncertainty ψ is $\pi_\psi^* = \arg\max_\pi \left\{\mathbb{E}_{\omega\sim\psi}[V_\omega^\pi]\right\}$, and the associated expected optimal value function for ψ is therefore $\mathbb{E}_{\omega\sim\psi}\left[V_\omega^{\pi_\psi^*}\right]$. As shorthand, I will

use V^*_ψ to denote the expected optimal value function for ψ, and V^π_ψ to denote the expected value function associated with π for ψ.

3.1.2 EVOI in Sequential Decision-Making

Like the abstract setting specified in Chapter 2, in this chapter I will assume that the agent can ask only a single query before it makes its decision. For the sequential decision-making settings considered in this chapter, the agent's decision set corresponds to its policy space Π, and I will assume that the agent occupies state s_c when it asks its query, and then immediately observes the response to its query in the same state.

In principle, each $\pi \in \Pi$ should prescribe an action for each state s and uncertainty ψ pair, since the agent may gain additional knowledge about the underlying model as it continues to act in the world after observing the response to its query. However, in this chapter I will assume that once the agent observes the response to its query and updates ψ, it never updates its uncertainty again, which allows Π to be simplified so that each $\pi \in \Pi$ prescribes actions as a function of state only. Thus, in this chapter query selection ignores any potential information the agent might gather from sources other than queries. For example, observing future state transitions might induce updates to its transition function uncertainty, and taking that into account could affect the relative usefulness of some queries. This assumption allows optimal planning computations in the settings I describe subsequently to be tractably approximated, and I emphasize that this assumption is *in addition* to the assumption in the abstract formulation specified in Chapter 2 that the agent can ask only a single query. Since both assumptions are myopic in that potentially impactful future events are ignored, in this chapter I refer to them as the *myopic assumptions* or *myopic conditions*, formally specified as follows:

1. The agent asks exactly one query in its current state s_c before taking any actions; and

2. After the agent observes the response to its query and updates its uncertainty, the agent never acquires additional knowledge that would induce further updates to its uncertainty.

As discussed at the end of Section 2.1.2, throughout this dissertation I use the terms "myopic" and "nonmyopic" to characterize the structure of knowledge acquisition processes, and I use the terms "single-shot" and "sequential" to characterize the structure of decision-making processes, in order to avoid confounding these concepts with the fact that query selection followed by decision selection is always sequential regardless of the structure of each. As introduced in Chapter 2, Expected Value of Information (EVOI) assesses

the goodness of a query in terms of how much value the agent is expected to gain from it. Since in this setting the agent's decision set corresponds to its policy space Π, and the value of a policy is a function of the agent's current state s_c, the EVOI associated with asking a query q (and receiving its response) under uncertainty ψ while occupying state s_c, under the myopic assumptions above, is defined as follows:

$$\begin{aligned} EVOI(q, \psi, s_c) &= \mathbb{E}_{j \sim q;\psi}\left[V^{\pi^*_{\psi|q=j}}_{\psi|q=j}(s_c)\right] - V^{\pi^*_\psi}_{\psi}(s_c) \\ &= \mathbb{E}_{j \sim q;\psi}\left[V^*_{\psi|q=j}(s_c)\right] - V^*_\psi(s_c), \end{aligned} \tag{3.1}$$

and the EVOI-based query selection problem when the agent can ask queries in some set Q is to select the query satisfying

$$\arg\max_{q \in Q} EVOI(q, \psi, s_c),$$

which will, in expectation, most increase the agent's value under the myopic assumptions above.

3.2 Selecting Transition and Reward Queries

In this section I will focus on two types of queries for which computing posterior distributions is straightforward, and will make the (somewhat unrealistic) assumption that the user is capable of perfectly responding to them. I now describe those query types and how the agent can parameterize its MDP uncertainty to efficiently incorporate their responses.

3.2.1 Dirichlet Priors and Transition/Reward Query Semantics

Transition queries ask for the outgoing transition probabilities associated with taking some specified action a in state s, while *reward queries* ask for the reward that would be received upon entering some specified state s. In this way, the agent can ask a reward or transition query to improve its knowledge of the uncertain parameters of the underlying MDP in which it must act, potentially allowing it to improve its policy in light of the response to its query. Although in principle an agent might have simultaneous reward and transition uncertainty and be able to ask both reward and transition queries, I will consider separate settings for each.

3.2.1.1 Dirichlet Transition Uncertainty Parameterization and Transition Queries

Suppose the agent knows the reward function of the user's model of the true MDP exactly, but has only a prior distribution over a space of transition functions. For a particular state-action pair, this translates to a distribution over *categorical* distributions (arbitrary discrete distributions) governing the transition probabilities to each possible next state, assuming the state space is finite. Absent prior knowledge that correlates the transition functions between different state-action pairs, the natural choice (oft-used in the literature, e.g. see Dearden et al. (1999)) to parameterize the agent's prior distribution over transition functions is as a set of independent Dirichlet distributions, one for each state-action pair.

Dirichlet distributions of order $|S|$ are supported by the space of $|S|$-dimensional categorical distributions (i.e., a sample is $|S|$ real numbers $\omega_1, \omega_2, \ldots, \omega_{|S|}$ that sum to 1 with each $\omega_i \geq 0$), and are parameterized by an $|S|$-dimensional vector of real numbers θ (often informally referred to as "counts") with the following pdf:

$$\text{Dir}(\omega; \theta) = \frac{1}{B(\theta)} \prod_{i=1}^{|S|} \omega_i^{\theta_i - 1},$$

where $B(\theta)$ is a normalizing constant. With this representation, the agent's model space Ω is the space of $|S| \cdot |A|$-dimensional vectors of categorical distributions, and letting $\omega_{s,a}$ denote the distribution corresponding to the outgoing state transition probabilities when action a is executed in state s and letting $\theta_{s,a}$ denote the parameters of the corresponding Dirichlet distribution, the agent's prior ψ over transition functions $\omega \in \Omega$ is parameterized as

$$\psi(\omega) = \prod_{(s,a) \in S \times A} \text{Dir}(\omega_{s,a}; \theta_{s,a}).$$

This representation for transition function uncertainty is often used in the literature because it allows the agent to easily incorporate observations of state-action-state transitions to update its uncertainty as it acts in the world (Dearden et al., 1999). Namely, the Bayes update for ψ upon observing a transition from state s to state s_j upon executing action a simply corresponds to incrementing the j^{th} component of $\theta_{s,a}$ by 1. Even though such non-query observations are not accounted for by EVOI as defined in Equation 3.1 (due to the second myopic assumption), in practice the agent could benefit by performing Bayes updates and potentially updating its policy as it observes transitions after asking its query.

With the agent's transition uncertainty parameterization made concrete, I now discuss how the agent can incorporate responses to transition queries. A *transition query* q asks about some state-action pair (s, a), and the response specifies $\omega_{s,a}$: the categorical distri-

bution over possible next states when action a is taken in state s. Let Q_T denote the set of all such queries (note that Q_T is implicitly a function of the state and action space, which are known to the agent). Since the Dirichlet distributions for each state-action pair are independent of each other and the user's response is assumed to match the parameters of the underlying MDP, the Bayes update for ψ to incorporate the response j to a transition query q about (s, a) simply corresponds to replacing the corresponding Dirichlet distribution $\psi_{s,a}$ with the (point distribution on) the categorical distribution specified by j. This ensures that all MDPs sampled from the post-query distribution $\psi|q = j$ would have the transition probabilities for (s, a) set to agree with j.

The *transition query selection problem* for an agent currently occupying state s_c is to select the transition query q^* with highest EVOI as defined in Equation 3.1:

$$q^* = \arg\max_{q \in Q_T} EVOI(q, \psi, s_c).$$

I have provided a simple way for the agent to represent its transition uncertainty, allowing the agent to efficiently perform Bayes updates to incorporate responses to transition queries. Next I discuss reward queries and show how a similar uncertainty parameterization can be used for reward uncertainty instead of transition uncertainty.

3.2.1.2 Dirichlet Reward Uncertainty Parameterization and Reward Queries

Suppose the agent knows the transition probabilities of the MDP exactly but begins with a prior over a space of reward functions, where recall in this chapter I have assumed that reward is a function of state only. Further, suppose rewards can only take on values defined within some (known) finite set Γ and may be a stochastic function of state. Then, absent prior knowledge correlating the rewards between different states, the agent can use exactly the same parameterization for its reward uncertainty as the parameterization for transition uncertainty described above, except ψ is parameterized as $|S|$ independent Dirichlet distributions instead of $|S| \cdot |A|$ of them, and each Dirichlet distribution is of order $|\Gamma|$ instead of order $|S|$.

A *reward query* q asks about some state s, and the response specifies the categorical distribution over the possible reward values Γ for s. Let Q_R denote the set of reward queries (note that Q_R is implicitly a function of the state space, which is known to the agent). Similarly to the case for transition queries, the agent can trivially update its uncertainty ψ to $\psi|q = j$ upon receiving response j to reward query q by replacing the corresponding Dirichlet component with the (point distribution on) the categorical distribution specified by j.

The *reward query selection problem* for an agent currently occupying state s_c is to select the reward query q^* with highest EVOI as defined in Equation 3.1:

$$q^* = \arg\max_{q \in Q_R} EVOI(q, \psi, s_c).$$

I have provided simple representations for transition and reward uncertainty that afford efficient Bayes updates for transition and reward queries, respectively. Next I discuss how EVOI can be approximately computed for reward and transition queries.

3.2.2 Computing Expected Value of Information for Reward and Transition Queries

Recall that Equation 3.1 defines EVOI in this chapter as follows:

$$EVOI(q, \psi, s_c) \quad = \quad \mathbb{E}_{j \sim q;\psi}\left[V^*_{\psi|q=j}(s_c)\right] - V^*_{\psi}(s_c).$$

Now, even though Bayes updates can be efficiently performed for these simple query types when using the above described uncertainty parameterizations, it is not clear that Equation 3.1 can be generally computed exactly due to three computational challenges: (1) the expectation over responses in Equation 3.1 is generally not analytically computable, (2) it is generally infeasible to compute Bayes-optimal policies; and (3) it is generally infeasible to exactly compute the expected value of Bayes-optimal policies.

Challenge (1) is mainly due to the continuous nature of reward and transition query responses, and one simple way to overcome this is to approximate the expectation using Monte-Carlo methods, sampling possible responses from the distribution $\Pr(q = j; \psi)$ by sampling MDPs ω from ψ and then sampling a response from the distribution $\Pr(q = j|\omega)$ for each sampled ω, and finally averaging the posterior Bayes-optimal value for each sampled posterior distribution. More formally, suppose n models $\{\omega_j\}_{j=1}^n$ are sampled from ψ, and let $J_q(\omega_j)$ denote the response to query q conditioned on model ω_j. Then,

$$\mathbb{E}_{j \sim q;\psi}\left[V^*_{\psi|q=j}(s_c)\right] \approx \sum_{j=1}^{n} \frac{1}{n} V^*_{\psi|q=J_q(\omega_j)}(s_c). \tag{3.2}$$

Computing this, however, in turn requires computing the Bayes-optimal value associated with each sampled induced posterior, and doing so accurately is key as this corresponds to computing the extent to which the query response's change to the agent's uncertainty is actually useful to the agent in terms of allowing it to improve its policy, and in turn its value. The expensive nature of posterior Bayes-optimal planning computations in sequen-

tial decision-making settings makes challenges (2) and (3) the main obstacles for tractably selecting queries in sequential decision-making settings. For the reward and transition queries considered in this section, I handle (2) with the mean-MDP method below, and handle (3) with Monte-Carlo methods, sampling MDPs from the appropriate distribution and averaging their Bayes-optimal values. Of course, for Monte-Carlo approximations, increased sampling improves approximation accuracy; for the experiments that follow, I empirically determined that 200 (500) samples approximated transition (reward) queries well.

The mean-MDP method is one way to approximately compute Bayes-optimal policies. For the case of unknown reward functions, the optimal policy with respect to the mean MDP is also Bayes-optimal (Ramachandran and Amir, 2007), and I will discuss how this structure can be further exploited when I discuss action query selection in Section 3.3. For the case of unknown transition probabilities in acyclic domains (i.e., domains where the agent can never return to any state it has previously visited), the same result holds. However, in the general case the mean-MDP method is only an approximation. Nevertheless, for the remainder of this section I refer to the optimal mean-MDP policy as Bayes-optimal even if approximate. Note that an advantage of using independent Dirichlet distributions for both cases is that the mean-MDP is trivial to compute: the counts θ map directly to empirical probabilities for the mean-MDP parameters.

Utilizing the mean-MDP method, each $V^*_{\psi|q=J_q(\omega_j)}(s_c)$ in Equation 3.2 is approximated as follows. Letting $\overline{\psi}|q = J_q(\omega_j)$ denote the mean-MDP associated with $\psi|q = J_q(\omega_j)$ (the posterior distribution for ψ induced upon incorporating response $J_q(\omega_j)$ to query q), and supposing m samples $\{\omega_j^i\}_{i=1}^m$ are drawn from $\psi|q = J_q(\omega_j)$,

$$V^*_{\psi|q=J_q(\omega_j)}(s_c) \approx \frac{1}{m}\sum_{i=1}^{m} V^{\pi^*_{\overline{\psi}|q=J_q(\omega_j)}}_{\omega_j^i}(s_c). \tag{3.3}$$

Note that using Equations 3.2 and 3.3 to approximate EVOI for reward and transition queries reduces the computation to a set of optimal planning computations, one for each sampled mean-MDP. Even though optimal planning in MDPs is well-studied and myriad algorithms exist for doing so, optimal planning for many sequential decision-making problems of interest is expensive; in such cases, the fact that many optimal planning computations must be performed to approximate the EVOI of even a single query means that the agent will be limited to considering only a few queries at best. For the experiments in this section, I will examine small domains where this is not an issue, but in the next section, where I discuss action query selection, I will return to this point and discuss strategies for approximating EVOI to avoid optimal planning computations to the extent possible. More-

over, in Chapter 5 I will discuss general strategies for avoiding optimal planning computations that apply to arbitrary types of queries and uncertain decision problems.

I will refer to the algorithm that selects a query by evaluating the EVOI of each one via the approximations specified above as *Expected Myopic Gain (EMG)*.

3.2.3 Empirical Results

Recall from Section 3.2.2 that the EVOI-optimal query (which EMG approximates) with respect to the query set being selected from is the optimal query for the agent to ask in the sequential decision-making setting considered in this chapter under the following two *myopic conditions*, or *myopic assumptions*:

1. The agent asks exactly one query in its current state s_c before taking any actions; and
2. After the agent observes the response to its query and updates its uncertainty, the agent never acquires additional knowledge that would induce further updates to its uncertainty.

In this section I examine EMG's performance on a pair of problem domains that are small enough to exhaustively compare EMG's choice against other possibilities, but that are interesting enough to allow for preliminary testing of the degree to which the above conditions are critical to EMG's (approximate) optimality. The first, called the "tree" domain, takes the form of a (acyclic) tree shown in Figure 3.1 (left), where the agent starts in state 0 and takes 2 actions, ending in one of the 4 "terminal" states $3-6$. In each of states 0, 1, and 2, the agent has two action choices, depicted with "solid" and "dotted" lines. For example, taking action s (solid) in state 0 has probability p_0^s and $1-p_0^s$ of transitioning into states 1 and 2, respectively. The agent knows the transition topology (e.g., from state 0 the next state is either 1 or 2). For transition queries the agent also knows the expected rewards ($R4 = 0.4$ and $R5 = 1.0$ and zero for the other states). For reward queries, the agent also knows the transition probabilities (p_0^s = p_1^s = p_2^s = .9 and p_0^d = p_1^d = p_2^d = .8), that states $0-2$ have zero reward, and that rewards for states $3-6$ can be $\{0, 1, 2\}$.

The second domain, called "grid" (Figure 3.1 right), has nine states, where the agent starts in state 0, and state 8 is the terminal goal state (the episode ends once the agent reaches it). The agent has a choice of four different actions in each non-goal state, where each action can stochastically transition to neighboring states as indicated by the solid arrows. The agent knows this topology. For transition queries, the agent also knows the rewards of all states ($R5 = -10$, $R6 = -2$, $R8 = 1$, and the rest are all 0). For reward

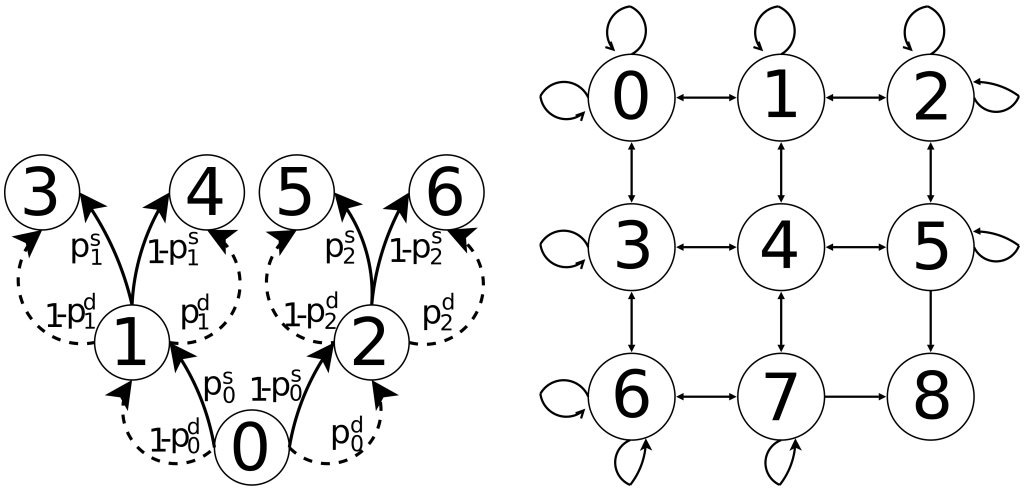


Figure 3.1: The "tree" and "grid" domains.

queries, the agent also knows the actions' transition probabilities: each action (conceptually, North, South, East, and West) moves in its corresponding direction with probability $.9$ and in the opposite direction with probability $.1$, and knows that rewards can be $\{-10, -2, 0, 1\}$ for all states.

As described in the previous section, for both query types in both domains, the agent's prior is a set of independent Dirichlet distributions. All counts are initialized to 1 so that the prior is uniform over multinomial transition (reward) probabilities for transition (reward) queries.

3.2.3.1 Experimental Procedure

For a particular type of query and a particular domain, the agent computes the Bayes-optimal policy given its initial knowledge, and it also uses EMG to compute an approximately EVOI-optimal query. To evaluate the actual *value* of asking a query, I adopt the following procedure. I randomly draw some number (10,000 unless otherwise noted) of sample (fully specified) MDPs from the prior distribution. For a given sampled MDP and a query, I use the sampled MDP's answer to the query to update the agent's distribution over MDPs, from which it computes a (possibly new) Bayes-optimal policy. The query's value for the MDP is the difference in long-term expected reward from executing the new versus the old policy in the sampled MDP. Note that this value does *not* take into account reward or transition knowledge that the agent could acquire and exploit as it takes actions after asking its query. While this does not impact values computed in the tree domain due to its acyclicity, it does impact values computed in the grid domain. However, when these values are treated as correct, both myopic assumptions are met and EMG should be expected to

choose the best query on average for this definition of query value.

Given such a measure for the value of each query, I compare EMG to other query selection methods by studying the average value of the query selected by each method across the sample MDP set. The first method I analyze is *random*, which simply chooses a query randomly. I execute this strategy by choosing a query randomly for *each* sample MDP (as opposed to using the same query throughout the sample MDP set). The next query selection method, *Best-Fixed*, asks the query with maximum average value across the *sample MDP set* as opposed to the query with maximum expected value across the *MDP distribution*. The last method, *omni*scent, chooses the query with highest value separately for each sample MDP (so it may choose a different query for each sample).

Note that neither Best-Fixed nor Omni are realizable query selection methods, as they both identify queries to ask using after-the-fact information that the agent could not possibly have. I include them in order to get a sense for how well EMG performs relative to upper bounds (Best-Fixed is an upper bound since both it and EMG must select one query for the entire sample set, and Best-Fixed chooses the query with best average value; omni is an upper bound since it selects the best query for each individual sample, resulting in a set of optimal queries across the sample set).

As an example, consider reward queries in the tree domain and consider drawing 2 MDPs as the sample set. Random will choose a random state to ask about for each sample; suppose it chooses states 0 and 1, respectively. EMG will choose to ask about state 3 (as explained in the next section). Best-fixed will look at the two sample MDPs in the set and choose to ask about whatever state yields the highest average value across both sample MDPs; suppose that is state 4. Omni will choose the best state to ask about for each sample; suppose it chooses states 1 and 2, respectively. The evaluation of each query selection method then consists of averaging the value of each method's queries throughout the sample set: in this example, random's choice of (0,1), EMG's choice of (3,3), Best-Fixed's choice of (4,4), and omni's choice of (1,2).

Due to symmetries in the environments, an arbitrary choice sometimes is made over equally good (in expectation) queries, and what happens experimentally is that due to the luck of the draw sometimes the EMG choice is the same as the Best-Fixed query for the specific trials, and sometimes it is not. To account for this variability, I present results that average the value of each query selection method over 10 independently-generated experiments of 10000 sampled MDPs each.

Query	Domain	Rand	EMG	BF	Omni
Trans	Tree	0.0137	0.0202	0.0203	0.0473
Trans	Grid	0.382	0.715	0.740	3.12
Reward	Tree	0.0530	0.0576	0.0586	0.127
Reward	Grid	0.851	1.23	1.34	4.04

Table 3.1: Results for the optimality under myopic assumptions experiment.

3.2.3.2 EMG Optimality under Myopic Assumptions

Recall that in the experimental setup described above, both myopic assumptions are met since query value is empirically measured assuming that the agent does not exploit any post-query knowledge it could acquire. Thus, EMG should select the most valuable query on average. To verify this claim, I begin by comparing in Table 3.1 the average values of EMG, Best-Fixed, omni, and random queries in the tree and grid domains.

For transition queries in the tree domain, the EMG query is arbitrarily either p_2^s or p_2^d. Though derivable analytically, it can be easily understood why asking about an action in state 2 is rational. State 2 dominates state 1 because (since they are equally likely to be reached) a better decision in state 2 in expectation will be more beneficial given the high reward of state 5. Comparing states 0 and 2, information about an action in either would be expected to equally increase the probability of reaching state 5. However, such a better decision in state 0 also *decreases* the probability of reaching state 4 (the only other positive reward), while a better decision in state 2 leaves the probability of reaching state 4 unchanged. Hence, asking about a state 2 action is the optimal choice (in expectation). The top row in Table 3.1 shows that EMG performed considerably better than random, and approaches Best-Fixed as one would expect which verifies EMG's optimality in this scenario. Omni provides an upper bound for how well the agent could do if it could miraculously guess the right query for every sample MDP.

In the grid domain, the EMG transition query asks about an action in state 4 (intuitively, the point on the best path to goal state 8 where a misstep is most costly). With 4 actions in state 4 that are initially indistinguishable, its chances of guessing the best for a particular set of trials is smaller than in the tree domain, so EMG is more noticeably lower than Best-Fixed. The optimal reward query in the tree domain is to ask about state 3, since the agent has the most control in getting to state 3 if it finds out it has high reward, and also the most control in avoiding state 3 if it finds out it has low reward. For reward queries in the grid domain, the symmetry of transition probabilities means that asking about states 1 and 3 are equivalently the best choice, since the agent must pass through at least one of them on its path to the goal and it may choose which to pass through. Table 3.1 shows that

in the grid domain, EMG's performance beats that of random and approaches Best-Fixed. However, even though the gap between EMG and Best-Fixed is to be expected in light of the explanation just given, the clear evidence for EMG's optimality in the tree domain does not translate to the grid domain.

Accounting for cost: In principle, even the optimal query should only be made when its expected gain is no less than its *cost*. The value of each entry in Table 3.1 provides an empirically-derived maximum "cost" (in terms of value) the agent should be willing to incur for posing that query; otherwise in expectation it would do better by autonomously following its initial Bayes-optimal policy than asking for help. Here I confirmed that EMG predicts the value of its query well by comparing the expected gain computed by EMG for the optimal query with the corresponding empirical gain in Table 3.1. For example, 100 independent executions of EMG for transition queries on the tree domain yielded an average expected gain of 0.0212 (st.dev. 0.0018), which closely estimated the corresponding empirical gain in Table 3.1, 0.0202. But I next examine how EMG's predictions about query value and its willingness to query can change as its optimality conditions are violated.

3.2.3.3 Post-Query Updates

Throughout the remainder of the experiments, I investigate how EMG's performance is affected when its myopic assumptions are violated. Recall that EMG's myopic optimality conditions require that once the agent receives the user's response to its query and builds its new Bayes-optimal policy, it will not receive any additional knowledge and so it cannot update its policy any further. For example, an agent executing in a state space that includes cycles (like the grid domain) could potentially learn from its experiences to update its model so as to build a better policy given it might return to that state. EMG does not consider how a state-action pair that can be learned at runtime might be less valuable to ask about beforehand.

Because of the myriad ways in which an agent could gather and incorporate knowledge at runtime, I examine the same phenomenon but in a simpler way. In these experiments, I assume that *after* it asks its one query, the agent is also unexpectedly (to it) given information for either 1 or 2 randomly-selected queries, where a random query could duplicate what the agent asked, and in experiments with 2 random queries they are distinct. The entries in the table represent the average value for knowing both the answer to its query *and the post-query information*. Here, unlike EMG, Best-Fixed takes into account which random query response(s) the agent will observe, and selects the query that is best in combination with each sampled random query response(s) across the sampled MDP set.

Query	Domain	Rand	EMG	BF	Omni
Trans	Tree-1	0.0242	0.0298	0.0299	0.0530
Trans	Grid-1	0.739	1.03	1.05	3.18
Trans	Tree-2	0.0336	0.0382	0.0385	0.0566
Trans	Grid-2	1.05	1.35	1.35	3.35
Reward	Tree-1	0.0796	0.0819	0.0825	0.129
Reward	Grid-1	1.46	1.82	1.84	4.04
Reward	Tree-2	0.102	0.103	0.104	0.131
Reward	Grid-2	1.97	2.27	2.28	4.08

Table 3.2: Results with 1 and 2 post-query updates.

As shown in Table 3.2, EMG tracks Best-Fixed closely, suggesting that EMG's initial query choice (which is the same as for the previous set of experiments) tends to be near-optimal despite not anticipating the possible additional information. However, it would not be difficult to construct other problems where EMG performs much more poorly, such as problems where it is known up-front that the same information that EMG asks for will definitely also be in the post-query information; being myopic, EMG cannot use this knowledge, whereas a nonmyopic approach could.

Notice also in Table 3.2 the narrowing gap between EMG/Best-Fixed and random, and with omni. This is not surprising, since given more randomly-chosen additional information, one would expect the agent's initial query choice to make less difference. This has implications on the value of incurring the overhead of EMG (random might become preferable) as well as on whether to query at all.

Accounting for cost: As the impact of the EMG query is diluted by other gathered information, the value-added by the query will fall. For example, I experimentally determined that the value-added of the EMG query in row 1 Table 3.2 (by comparing to the case where no query is asked but random information is received) is 0.0164, and for row 3 is 0.0121. EMG computes a gain of 0.0212 for its query, and thus EMG could myopically overestimate the acceptable "cost" for querying the user.

3.2.3.4 Sequences of Queries

EMG also assumes that the agent can ask only a single query. If the agent can ask multiple queries, then EMG can make incorrect decisions by failing to factor into its gain calculations the (exploration) value a query has for asking better queries downstream. In general, computing optimal sequences of queries to reliably derive such a strategy involves solving Bayes-Adaptive Markov decision processes (Duff, 2003), which is widely considered

Query	Domain	Rand	EMG	BF	Omni
Trans	Tree	0.0254	0.0386	0.0379	0.0592
Trans	Grid	0.640	1.17	1.25	3.98
Reward	Tree	0.0907	0.0960	0.0968	0.134
Reward	Grid	1.60	2.31	2.13	4.42

Table 3.3: Results for the sequence of queries experiment.

intractable (Braziunas and Boutilier, 2008).

The hypothesis I will empirically test is as follows: If an agent can ask two queries, then using EMG to determine these queries yields an expected added value competitive with the best possible query sequence. I tested this hypothesis in the two domains by slightly modifying the experimental procedure: for each trial (sampled true MDP), after computing and asking the EMG query and incorporating the response, the agent ran EMG a second time to identify the most valuable (next) query to ask given its new distribution over MDPs. After the response to this, the Bayes-optimal policy given the combined information of the two responses was evaluated against the true MDP. For each of the 10,000 trials, I also computed the value for every possible fixed *pair* of queries.

As seen in Table 3.3, the sequence of two EMG queries together yielded an average value similar to that of the average value of the Best-Fixed-pair (omni and random query-pair values are also given for completeness). In fact, in some cases, EMG did even better. EMG has an advantage over any fixed-pair, because EMG is *not* fixed: it can choose a different second query based on the response to the first. On the other hand, EMG is at a disadvantage because it does not look ahead to how the first query can "set up" the second.[1] As the data shows, sometimes EMG's advantage of following a policy (conditionally branching to a different second query) outweighs being myopic, and other times it does not. Note that experimentally evaluating the space of all possible 2-step policies to see how close EMG gets to the best such policy would require overcoming additional nontrivial computational challenges, since the space of 2-step policies is continuous (for 2-step policies, the second query to ask is allowed to be a function of the continuous response to the first) and cannot be explicitly enumerated unlike the small finite set of query *pairs* the agent can ask.

Accounting for cost: The agent can of course ask even longer sequences of queries. Figure 3.2a shows, for the transition-query grid domain, a comparison between the value of the posterior policies induced by sequential EMG or random queries, as well as the optimal (in

[1] If each query EMG considered was instead about a pair of pieces of information, EMG could find the pair with the highest expected gain, but then would be limited to considering a fixed pair.

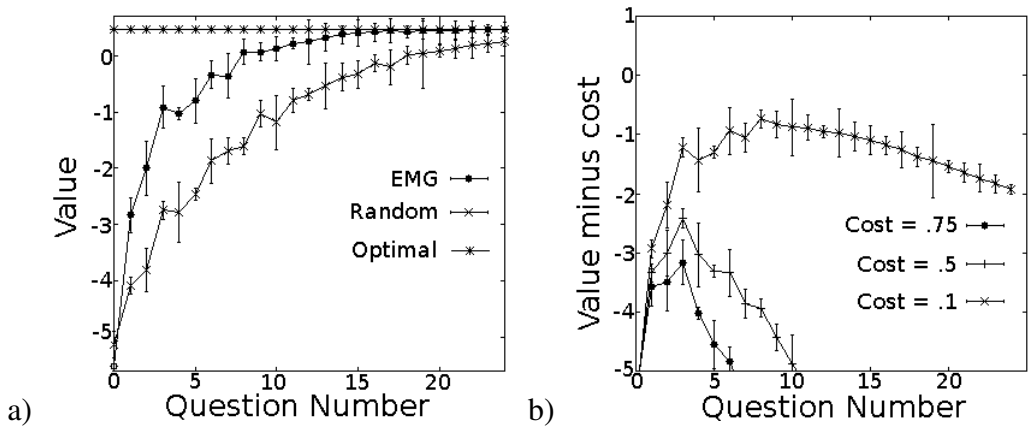


Figure 3.2: a) Value of the policy obtained after a sequence of EMG queries, compared with a random query sequence and the optimal policy with full information. b) Value of the policy obtained after a sequence of EMG queries, minus the total cost to obtain it for various costs of querying.

expectation) policy the agent is attempting to obtain. As expected, repeatedly asking EMG queries improves agent value faster than a sequence of random queries. The important thing to note, however, is in general the improvement in value for each additional query decreases as the sequence gets longer. This suggests that, if EMG is used for problems where multiple (even unlimited) queries can be asked, there will likely be a tipping point where the cost of querying exceeds the expected gain of doing so. This is shown in Figure 3.2b, where a few different costs for querying are shown. The higher the cost, the fewer queries can be asked before the expected net value to the agent starts falling.

3.2.3.5 Online Query Asking

Finally, I try relaxing the myopic assumption that the agent ask its query before taking any action. Now, an agent could benefit by waiting to ask its query until after taking some actions. In the tree domain for transition queries, for example, as discussed above if the agent must ask its query before it begins acting, it should ask about an action in state 2. However, the response will not change the agent's action choice in state 0. If it takes its action in state 0 first, taking it to state 1 or 2 with equal probability in expectation, then it can ask a query that improves its action choice in whichever of those states it ended up in. Its expected value increases because it improves the chances of reaching a non-zero reward state however its action in state 0 turns out, compared to asking first (where information it gets about an action in state 2 will only be useful 50% of the time in expectation).

This suggests a heuristic strategy for deciding when to query: if the Bayes-optimal policies for every possible answer to the (proposed) EMG query all prescribe the same

Query	EMG	EMG+AWINA	EMG+Random
Trans	0.715	0.869	0.750
Reward	1.23	1.23	1.18

Table 3.4: Results for the online experiments (grid domain).

action for the current state, there is no advantage in asking the query at this time. The agent should instead take an action. It can then repeat this procedure, postponing asking a query until it has reached a state where that query will impact its next action choice.

This *Ask When Impacts Next Action (AWINA)* heuristic clearly would work in the simple example considered so far. However, because the tree domain involves taking exactly two actions, it does not provide much latitude for more comprehensive evaluation, so I use the grid world (with all other EMG optimality conditions met) to compare standard (prior to any actions) EMG to the AWINA heuristic. I also consider the case where the decision about asking the EMG query is made randomly (with probability .5) at each time step (until a query has been asked), and the results are given in Table 4 (note that the agent updates its uncertainty upon observing transitions or rewards as it acts before asking its query). These results suggest that, in the grid domain, the agent will do well to use EMG with the AWINA heuristic when dealing with transition queries because it is better off waiting until it knows which negative state it has wandered near. Note that even waiting randomly to ask does better than always asking at the start. For reward queries, in contrast, finding out anything about the reward landscape before setting off is useful, so AWINA does not improve EMG, and EMG+random does worse than the others. In summary, AWINA can sometimes help, but a more complete exploration for additional interesting heuristics is an area for future research (I touch on this topic in Chapter 7).

3.3 Selecting Action Queries under Reward Uncertainty

While in the previous section I considered reward and transition queries which are tied directly to the agent's uncertainty regarding the parameters of the underlying MDP, in this section I consider action queries which the agent can use to indirectly improve its knowledge. Of the many types of queries one could consider asking the user, action queries are arguably quite natural for a human to respond to, as they ask what action the user would take if teleoperating the agent in a particular state.

When teleoperating, the user chooses actions according to her model of the world. I will assume that the agent fully knows the user's model of world dynamics (i.e., the agent has full knowledge of the transition function), but has an incomplete model of the user's

rewards, and thus risks acting counter to the user's true rewards. Further, I will assume that the only way the agent can learn about the underlying rewards is to ask a query, i.e., there is no "reward signal" present so the reward function is solely used as a means to parameterize the agent's policy.

The agent's **objective** is to identify the query that will maximize its gain in expected long-term value with respect to the user's true rewards and the agent's current state, which corresponds to identifying the query with highest EVOI (Equation 3.1). Recall that this objective is myopic because it ignores future queries that could be made, such as if the agent could ask a sequence of queries, or wait to query later.

This section is organized as follows. First, I show how EVOI can be simplified to take advantage of the structure associated with the agent being uncertain only about the reward function, to create a new EVOI-based algorithm for action query selection: EMG-AQS. Second, I compare the EMG-AQS algorithm with an uncertainty-based algorithm called Active Sampling (AS) (Lopes et al., 2009) that applies directly to the problem setting, but selects action queries based on maximally reducing uncertainty in policy representations. Third, I develop a new hybrid algorithm combining EMG-AQS and AS, which I show in a computation-time-limited scenario can combine the strengths of EMG-AQS and AS.

3.3.1 Computing Expected Value of Information for Action Queries

In this section I assume that the agent has uncertainty only over the reward function, i.e., uncertainty over the user's preferences for how it should act. To learn more about the user's preferences over how the agent should act, the agent can ask action queries. An *action query* asks the user for what action she would take in some state s, which corresponds to asking for the optimal action for state s given knowledge of the underlying reward function ω. Note that independent Dirichlet priors cannot account for the correlations introduced among state-rewards when incorporating the response to an action query (no such correlations are introduced for the reward and transition queries studied in the previous section), and hence are not a viable choice to parameterize the agent's uncertainty when it considers action queries. Instead, I assume that the set of possible reward functions is finite, so that the agent's uncertainty ψ is expressed as an arbitrary discrete probability distribution (i.e., a categorial distribution) over a finite set Ω of reward functions.

Let Q_A denote the set of all action queries (which is implicitly a function of the state space of the underlying MDP, which is known to the agent). Recall that EVOI is specified as follows:

$$EVOI(q, \psi, s_c) = \mathbb{E}_{j \sim q;\psi}\left[V^*_{\psi|q=j}(s_c)\right] - V^*_{\psi}(s_c), \tag{3.4}$$

For the case considered in this section where the agent only has reward uncertainty, drastic simplification applies. First, as described in Section 3.2.2, the mean-MDP method can be used to exactly compute Bayes-optimal values for the case of reward uncertainty only. Specifically, the expected value of a policy over a reward distribution is its value for the single *mean*-reward function, denoted $\bar{\psi}$, which implies that the Bayes-optimal policy for ψ is the optimal policy for $\bar{\psi}$. (See proof of Thm. 3 by Ramachandran and Amir 2007.) Thus, Equation 3.4 can be simplified as

$$EVOI(q, \psi, s_c) = \mathbb{E}_{j \sim q;\psi}\left[V^*_{\overline{\psi}|q=j}(s_c)\right] - V^*_{\overline{\psi}}(s_c),$$

where $\overline{\psi}|q = j$ denotes the mean-reward function for the posterior distribution of ψ induced upon incorporating response j to q, or more formally, $\overline{\psi}|q = j$ is defined as $\mathbb{E}_{\omega \sim \psi|q=j}\left[\omega\right]$.

Updating ψ to incorporate the response j to an action query q to obtain $\psi|q = j$ is not trivial as is the case for the reward and transition queries considered in the previous section, but fortunately this problem is closely connected to related work. Namely, this is exactly the problem solved in Bayesian Inverse Reinforcement Learning (BIRL) (Ramachandran and Amir, 2007), and like Lopes et al. (2009) I use BIRL to perform Bayes updates over the reward space in this section.

In BIRL, the starting assumption is a noisy-model of the user's action selection. For the MDP given by reward function ω, there is an associated action-value function Q^* such that $Q^*(s, a, \omega)$ is the expected value obtained when the start state is s, the first action is a, and the optimal policy is followed thereafter. The user is assumed to respond with action j to an action query q for state s with probability $\Pr(q = j|\omega) = \frac{1}{Z_\omega} e^{\alpha Q^*(s,a,\omega)}$, where Z_ω is the normalization term and α is a noise (or confidence) parameter such that the larger the α, the more confident the agent is that the response received is indeed optimal for the user. Setting α lower can help in situations in which the user's responses are noisy, or inconsistent with respect to all rewards in the reward space. Given a response j to query q about state s, and a current distribution ψ over rewards, the posterior distribution $\psi|q = j$ over rewards is defined by the Bayes update

$$\psi(\omega|q = j) = \frac{1}{Z}\Pr(q = j|\omega)\psi(\omega), \tag{3.5}$$

where again Z is the appropriate normalization term. The finiteness of the reward set allows the agent to tractably update ψ exactly according to Equation 3.5 upon receiving the response to an action query, since the agent can precompute (or cache) value functions

for each reward parameter, resulting in substantial computational savings when updating the reward distribution (provided the set of reward functions is small). Although it can be more natural from the agent designer's perspective to use continuous reward spaces to account for many possible policies, performing BIRL in continuous reward spaces comes at a significant computational cost. (See Lopes et al. (2009) and Ramachandran and Amir (2007) for Monte Carlo methods for approximating BIRL in continuous reward spaces.)

Combining the above model for action query responses with the finiteness of the agent's action space, Equation 3.3.1 can be explicitly computed as a weighted sum over the $|A|$ possible responses as follows:

$$\begin{aligned} EVOI(q, \psi, s_c) &= \sum_{j=1}^{|A|} \Pr(q = j; \psi) V^*_{\overline{\psi}|q=j}(s_c) - V^*_{\overline{\psi}}(s_c) \\ &= \sum_{j=1}^{|A|} V^*_{\overline{\psi}|q=j}(s_c) \sum_{\omega \in \Omega} \psi(\omega) \Pr(q = j|\omega) - V^*_{\overline{\psi}}(s_c). \end{aligned} \tag{3.6}$$

Assuming that the value caching scheme above is used to allow each $\Pr(q = j|\omega)$ and $\overline{\psi}|q = j$ to be explicitly computed in an efficient manner, the main computational bottleneck associated with using Equation 3.6 to approximate EVOI is the computation of each Bayes-optimal posterior value $V^*_{\overline{\psi}|q=j}(s_c)$, which are not always cached by the above scheme since the mean-rewards $\overline{\psi}|q = j$ are a convex combination of members of Ω and thus not necessarily a member of Ω.

This completes the description of how the agent updates its reward function distribution after each query, and utilizes the structure present in reward function distributions to simplify EVOI computations. I will refer to the algorithm that exhaustively computes the EVOI of every action query in this manner and then selects the best one as *EMG-based action query Selection (EMG-AQS)*.

Reward function parameterizations. Parameterizing the reward function allows for a compact representation of the reward function distribution, even when defined over an infinite state space. Consider a distribution over a discrete parameter space Ω and a function ϕ that maps $\omega \in \Omega$ to a reward function R. The mean reward function can be safely used for computing policies and values optimal with respect to a reward distribution, but the reward function associated with the mean reward parameters can be safely used instead only when $\mathbb{E}_{\omega \sim \psi}[\phi(\omega)] = \phi(\mathbb{E}_{\omega \sim \psi}[\omega])$ (an extension of Theorem 3 in Ramachandran and Amir 2007). Due to linearity of expectation, this equality holds when $\phi(\omega)$ is a linear

function of ω, but not necessarily otherwise. Therefore I use the mean reward parameters when this equality holds, but otherwise use the mean reward function, calculated as $\mathbb{E}_{\omega\sim\psi}[\phi(\omega)] = \sum_{\omega\in\Omega}\phi(\omega)\psi(\omega)$. I explain in more detail how I make use of parameterized reward functions in the descriptions of the experiments that follow.

3.3.2 Active Sampling

Another approach for action query selection is Active Sampling (AS), due to Lopes et al. (2009). Intuitively, AS reduces the agent's uncertainty in the user's policy by querying the state that has maximum mean entropy (uncertainty) in its action choices. Next I describe AS in more detail.

Each MDP has a stochastic optimal policy (stochastic only in that ties between actions are broken with uniform probability), and the action-choice probability for a state for each action is binned into X uniform intervals between 0 and 1. Given a current distribution over rewards ψ, the mean-entropy for state s, $\bar{H}(s)$, is given by

$$\bar{H}(s) = -\frac{1}{|A|}\sum_{a\in A}\sum_{x=0}^{X-1}\mu_{sa}(x)\log\,\mu_{sa}(x), \text{ where}$$

$$\mu_{sa}(x) = \sum_{\omega\in\Omega}\Big\{\pi^*(a|\omega;s)\in I_x\Big\}\psi(\omega).$$

Here $I_x = (\frac{x}{X}, \frac{x+1}{X}]$ for $x \neq 0$ and $I_0 = [0, \frac{1}{X}]$, $\psi(\omega)$ is the probability of reward function r given distribution ψ over rewards, and $\{\pi^*(a|r;s)\in I_x\}$ is 1 if the probability of action a in state s under the optimal policy given r falls in the interval I_x and 0 otherwise. The AS algorithm queries the state with maximum mean-entropy.

Note that using AS for action query selection affords computational advantages over using EMG-AQS. In particular, AS never needs to compute posterior Bayes-optimal policies/values, and in fact never needs to compute Bayes-optimal policies/values at all when the value caching scheme described in the previous section is used (once the cost of precomputing the optimal value function for each candidate reward function is paid). However, this advantage over EMG-AQS comes at the tradeoff of less accurate query selection. For example, the dynamics of the world may dictate that some states are less likely to be reached than others, especially when taking into account the agent's current state. Also, taking the wrong action in some states may be catastrophic, whereas in others benign. Minimizing policy uncertainty does not consider these factors, and thus is only a proxy for achieving the agent's objective of maximizing its expected value.

Connection to Value-Loss Minimization. Even though the query selection strategy adopted by AS does not improve the agent's expected value directly as the strategy adopted by EMG-AQS does, Theorem 3.1, a new result that I prove below, implies that querying the state with maximum mean-entropy reduces an *upper bound* on the agent's *expected value loss* compared to if it were somehow able to act optimally for the underlying MDP. (I will refer back to Theorem 3.1 when I analyze the properties of another uncertainty-based algorithm in Chapter 5, but for now the reader can safely skip to the empirical comparisons of Section 3.3.4 if desired.)

Theorem 3.1. *Let* $\mu(a|s,\psi) = \mathbb{E}_{\omega\sim\psi}\big[\Pr(a = \pi^*_\omega(s)|\psi)\big]$*, and let* $H_\mu(\psi, s') = H(\mu(\cdot|s',\psi))$*, where* H *denotes the Shannon entropy function, i.e., for some discrete probability distribution* Y*,* $H(Y) = -\sum_i Y(i)\log(Y(i))$*. Then for any distribution* ψ *over a set of MDPs* Ω*,* $\mathbb{E}_{\omega\sim\psi}[V^*_\omega(s) - V^{\pi^*_\psi}_\omega(s)] \leq (1 - e^{-\max_{s'} H_\mu(\psi,s')})|\Omega|(V_{max} - V_{min})$.

Proof. Let policy μ_ψ be defined as follows: $\mu_\psi(s) = \arg\max_a \mu(a|s,\psi)$ (with ties broken arbitrarily), and let $p = \min_s \max_a \mu(a|s,\psi)$. Then,

$$
\begin{aligned}
& E_{\omega\sim\psi}[V^*_\omega(s) - V^{\pi^*_\psi}_\omega(s)] \leq E_{\omega\sim\psi}[V^*_\omega(s) - V^{\mu_\psi}_\omega(s)] \leq E_{\omega\sim\psi}[V^*_\omega(s) - V^{\mu_\psi}_\omega(s)] \\
= \;& \sum_\omega \psi(M)(V^*_\omega(s) - V^{\mu_\psi}_\omega(s)).
\end{aligned}
$$

Now if $\psi(\omega) > (1-p), \forall s'\, \mu_\psi(s') = \pi^*_\omega(s')$ and consequently $\forall s'\, V^*_\omega(s') = V^{\mu_\psi}_\omega(s')$. To see this, note that $\forall\omega$ if $\exists s' : \mu_\psi(s') \neq \pi^*_\omega(s')$, then $\psi(\omega) + p \leq 1$, and the preceeding claim follows. Thus,

$$
\begin{aligned}
& \sum_\omega \psi(\omega)(V^*_\omega(s) - V^{\mu_\psi}_\omega(s)) = \sum_{\omega:\psi(\omega)\leq 1-p} \psi(\omega)(V^*_\omega(s) - V^{\mu_\psi}_\omega(s)) \\
\leq \;& \sum_{\omega:\psi(M)\leq 1-p} (1-p)(V^*_\omega(s) - V^{\mu_\psi}_\omega(s)) \\
\leq \;& \sum_{\omega:\psi(M)\leq 1-p} (1-p)(V_{\max} - V_{\min}) \\
\leq \;& (1-p)|\Omega|(V_{\max} - V_{\min}) \\
\leq \;& (1 - e^{\max_{s'} H_\mu(\psi,s')})|\Omega|(V_{\max} - V_{\min}). \quad \textbf{(Lemma 5.1)}
\end{aligned}
$$

□

The last inequality is due to Lemma 5.1 which I will prove in Chapter 5. Note that if $H_\mu(\psi) = 0, \mathbb{E}_{\omega\sim\psi}[V^*_\omega(s) - V^{\pi^*_\psi}_\omega(s)] = 0$, which implies that the upper bound evaluates

to zero when $\max_{s'} H_\mu(\psi)$ is zero, at which point the agent would be guaranteed to act optimally. This suggests that querying the state in which the agent's policy entropy is highest, as done by AS, is a sensible surrogate for EVOI.

3.3.3 Computation-Limited Scenarios and Hybrid Approach

So far I have described two approaches for evaluating and selecting action queries: EMG-AQS, which directly maximizes EVOI in a costly manner, and AS, which indirectly maximizes EVOI using a cheaper uncertainty-based criterion. In practice, however, computation time may be limited if neither method can feasibly evaluate all possible queries.

To perform query selection in such a scenario, a simple heuristic strategy would be to evaluate, using EMG-AQS or AS, as many randomly sampled queries from the agent's query set as possible before the available computation time is exceeded. In implementing such a strategy, it is unclear whether EMG-AQS or AS would be the better choice, since although AS can evaluate many more queries than EMG-AQS given the same time limit, it could conceivably select a less valuable query since its uncertainty reduction criterion is only a proxy for EVOI.

One way to potentially improve this strategy would be to select a promising subset of queries to evaluate instead of evaluating as many random queries as time allows. Namely, the *Hybrid* approach, the main contribution of this chapter, uses AS's cheap evaluations to identify a promising subset of queries for EMG-AQS to consider. More specifically, Hybrid uses AS's evaluation metric to select the top m queries out of a set of n randomly sampled queries, and then uses EMG-AQS's evaluation metric to evaluate as many of those m queries as time allows. I empirically compare the effectiveness of using Hybrid as opposed to EMG-AQS or AS alone for computation-limited scenarios in Experiments 2 and 3 of the next section.

3.3.4 Comparisons

I now empirically compare the relative suitability of EMG-AQS versus AS for choosing the most valuable queries to pose to the user. *Prima facie*, one might expect EMG-AQS to perform better, as it is Bayes-optimal for selecting a single query assuming that only a single query can be asked. However, neither method is Bayes-optimal when selecting a sequence of queries, or when limited computation time is available so that only some of the possible queries can be evaluated. I test the former condition in the first experiment, the latter condition in the second experiment, and both combined in the third experiment. In

the second and third experiments, I include the Hybrid method method in the comparisons as well.

The principal metric of comparison that I use is *policy loss*, which is the difference in value between the optimal policy and the policy based on uncertain knowledge. In the experiments that follow, I report average policy loss over trials (error bars shown are confidence intervals for p-value 0.05), where each trial uses a reward function whose parameters are uniformly randomly drawn from the reward parameter space. Both strategies begin with uniform priors.

3.3.4.1 Puddle World

I conducted the first set of experiments in the Puddle World, as used in the evaluations of Lopes et al. (2009). The puddle world is a 21x21 gridworld, and so has 441 states (locations). The agent can move (deterministically) in any of the 4 cardinal directions to a neighboring state (unless it bumps into a grid edge in which case it does not change state). The agent operates over an infinite time horizon, with discount factor $\gamma \in [0, 1)$. The reward function is

$$f(w, s) = r_1 \exp(\frac{-||s_1 - s||}{2\sigma^2}) + r_2 \exp(\frac{-||s_2 - s||}{2\sigma^2}),$$

where s_1,s_2 represent the centers of reward emitting regions, and r_1,r_2 constitute the sign and magnitude of the reward emitted from each. The pre-set scalar σ determines the rate of decay of the magnitude of reward emitted from each center. The parameterization permits cases with one goal and one puddle (where one of the r parameters is positive and the other is negative), and similarly two goals or two puddles.

Like in the experiments of Lopes et al. (2009), the agent's start state is always $(0, 0)$. While their formulation uses a continuous state space, their deterministic direction and distance of movement actions resulted in a *de facto* discretization of space. Similarly, in this experiment the agent explicitly represents the state space discretely, which also allows for much more accurate and rapid Q-value approximations than would be computed through function approximation. I discretized the parameter space to allow the reward decay scalars to independently take on values -1 or 1 and the centers of the reward emitting regions to be located on a 5x5 discretization of the state space. This results in a set of reward parameters of size 2500.

The user in this experiment is modeled as the optimal policy given the actual reward parameters for the particular trial: a response to a query is the action in this policy corresponding to the state being asked about.

Experiment 1.

Here I perform the comparison across a range of discount factors. Since the agent's value is always measured with respect to the start state, a smaller discount factor reduces the benefit of collecting rewards in the future – good queries will inform the agent how to behave near the start state. As the discount factor increases, however, rewards the agent receives in the future are increasingly important. I test over a range of discounts to gain insights into each method's sensitivity to corresponding changes in query quality.

Figure 3.3 shows the performance of each method for a sequence of queries, across discount factors 0.2, 0.9, 0.99, and 0.999. Note the different scaling of the *y-axes*: a smaller discount factor leads to lower policy values, which in turn leads to lower policy loss magnitudes.

On all graphs, EMG-AQS shows the best performance for its first query. This is expected because EMG-AQS's first query choice is Bayes-optimal, so no method can do better in expectation for the first query. For the low discount factor of 0.2, EMG-AQS significantly outperforms AS across the entire sequence. In fact, AS's performance is comparable to Random, a strategy that simply selects random queries. Because EMG-AQS directly aims to increase the agent's expected discounted reward, it outperforms AS which instead improves decisions that are most uncertain regardless of when those decisions might be needed. Similarly, for discount 0.9, which is the exact (discretized) setting Lopes et al. (2009) use to test their method, EMG-AQS outperforms AS for queries 2-4.

One might expect that, by raising the discount factor closer to 1, the relative benefits of EMG-AQS would diminish. Indeed, with a discount of 0.99, the difference between EMG-AQS and AS shrinks. In fact, AS outperforms EMG-AQS for queries 4 and 5. This is not entirely surprising: both approaches make myopic decisions based on different criteria, so even though EMG-AQS will greedily make Bayes-optimal decisions at each point in the query sequence, the combination of queries asked by AS can be better.

Surprisingly, though, this trend does not persist as the discount factor is raised to 0.999: EMG-AQS once again outperforms AS. By investigating the queries being asked, I discovered that the explanation for this is essentially the other side of the coin for the larger discount factors. That is, when the discount factor is high enough, the penalties incurred by, for example, wandering too near the puddle on the way to the goal location become negligible compared to the accumulation of reward the agent gets as it hovers around the goal. Because of this, EMG-AQS focuses on queries that effectively map the goal region rather than the puddle region, while AS is not biased either way.

Hence, in the Puddle World, EMG-AQS effectively biases its queries towards the start

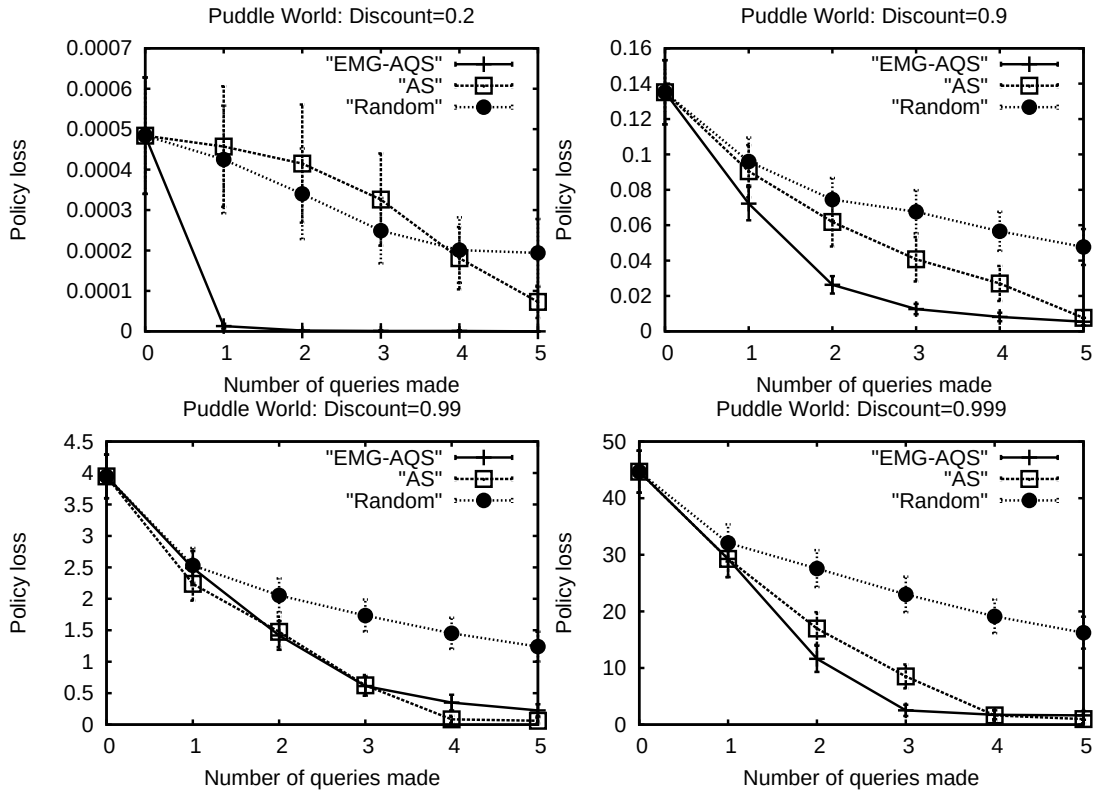


Figure 3.3: Average policy loss for EMG-AQS, AS, and Random on the Puddle World across a sequence of queries shown, from left to right, for discount 0.2, 0.9, 0.99, and 0.999. All error bars are confidence intervals with p-value = $.05$.

state (for low discounts) or towards finding the best region to occupy in steady state (for high discounts), while AS is more evenhanded. These results suggest that EMG-AQS makes better choices by more accurately assessing whether querying to improve reward collection in the short-term versus reward collection in the long-term would be more valuable, though in a balanced situation where this choice is less impactful, AS can be better. Next I further test the methods over sequences of queries, but also take into account the computational efficiency of the methods.

3.3.4.2 Driving Domain

To test the robustness of the trends observed in the Puddle World, and to take computational efficiency into account, I ran additional comparisons between EMG-AQS and AS in an implementation of the Driving Domain (Figure 3.4a), which is a traffic navigation problem often used in studies of apprenticeship learning (Abbeel and Ng, 2004).

Dynamics: At each discrete time step the agent controls a car on a highway by taking one

of three actions: move left, no action, or move right. Attempting to move off the edge results in a no action. Other cars are present, which move at random continuously-valued constant speeds (this makes the state space infinite) and never change lanes. The agent lacks an explicit transition function, but can forward simulate to sample trajectories influenced by its choices of actions.

State Features: The agent represents state as a vector of features, consisting of three parts. The first has 5 binary features to specify the agent's current lane. The second contains 3 binary features to specify whether the agent is colliding with, tailgating, or trailing another car. The third part consists of 3 sets of features, one for the agent's current lane, and one for each adjacent lane, each containing 20 binary features specifying whether the agent's car (disregarding lane) will collide with another car in $2x$ timesteps, where x ranges from 0 to 19. This part encodes an agent's left, forward, and right view of traffic and takes into account car velocities.

Rewards: The reward function is parameterized by a weight vector w as $f(w, s) = \phi(s) \cdot w$, where $\phi(s)$ is a vector of reward features corresponding to state s. In particular, $\phi(s)$ contains four binary features, which specify whether the agent is colliding with, tailgating or trailing another car, and whether the agent is currently driving on the shoulder. In the experiments, the agent has *a priori* knowledge that the reward weight corresponding to driving on the shoulder is -0.2, but must learn which of 1000 possible assignments of the other parameters best encodes user preferences.

Practicality of Action-Queries: Since in these evaluations the user is a policy, queries can be answered by invoking the user policy. However, in a practical scenario with a human user, the agent should translate its representation of state into one that a human could more easily understand. Since a hypothetical state includes information about car positions and velocities as well as the agent car's position, the agent could present a state to the human user as a simulated video clip, or an annotated picture.

Value Calculations: Due to the infinite state space, the optimal value function for a given reward parameter can no longer be calculated exactly, and so I employ Sarsa(λ) (Sutton and Barto, 1998) with linear function approximation to approximate value functions. This often makes the user suboptimal with respect to the reward function chosen for a particular trial. I accordingly set α to 40.0, which represents relatively high confidence in user responses but allows robustness to possible inconsistencies in responses. Unlike the experiments in

the Puddle World, the agent makes action-queries to learn the user's driving preferences *before* it enters the world. As a result, there is no notion of current or start state, and so policy values, including those used in our policy loss performance metric, are computed as an expectation over possible start states.

Query Space: The infinite state space also makes it impossible to exhaustively consider and rank every possible query with either EMG-AQS or AS, resulting in a *de-facto* computation-time limitation. Given only a subset of possible queries can be evaluated, the agent needs a means to approximately find each query evaluation algorithm's maximum. In the following experiments I compare the strategies discussed in Section 3.3.3 for doing so. Namely, I consider using EMG-AQS or AS to evaluate as many randomly (uniformly) sampled queries as possible, and also consider using the Hybrid method discussed in detail in Section 3.3.3. The more computation time is allowed, the larger the set of queries that can be evaluated, and in turn the greater the chances of finding a good query among them. In the experiments, I assess the impact of various computation time limits.

Experiment 2.

In this experiment, I measure the policy loss for each method asking one query, allotting each method the same amount of time to select from a large set of randomly-selected candidate queries. The implementation of AS used here (using precomputed values) evaluates a single query about 200 times faster than EMG-AQS (also using precomputed values), since EMG-AQS performs the time-consuming operation of formulating hypothetical policies for each of the answers to the query. I.e., EMG-AQS must perform optimal planning computations for every query evaluated, unlike AS. I also test the Hybrid method described in Section 3.3.3, using AS to rank 16 randomly-selected queries, and then using the remaining allotted time for EMG-AQS to evaluate them in order until time expires and then return the best found.

Figure 3.4b shows that AS's superior speed allows it to find a better query more quickly than EMG-AQS, but does not benefit from additional time unlike EMG-AQS, allowing EMG-AQS to outperform AS once at least 3 seconds are available for computation. The Hybrid algorithm, on the other hand, gets the best of both worlds and meets or surpasses the performance of both methods at all points. Intuitively, this is due to AS's ability to quickly find queries whose answers are very uncertain, allowing the slower but more effective EMG-AQS to determine which of those is most helpful in terms of value gain.

Experiment 3.

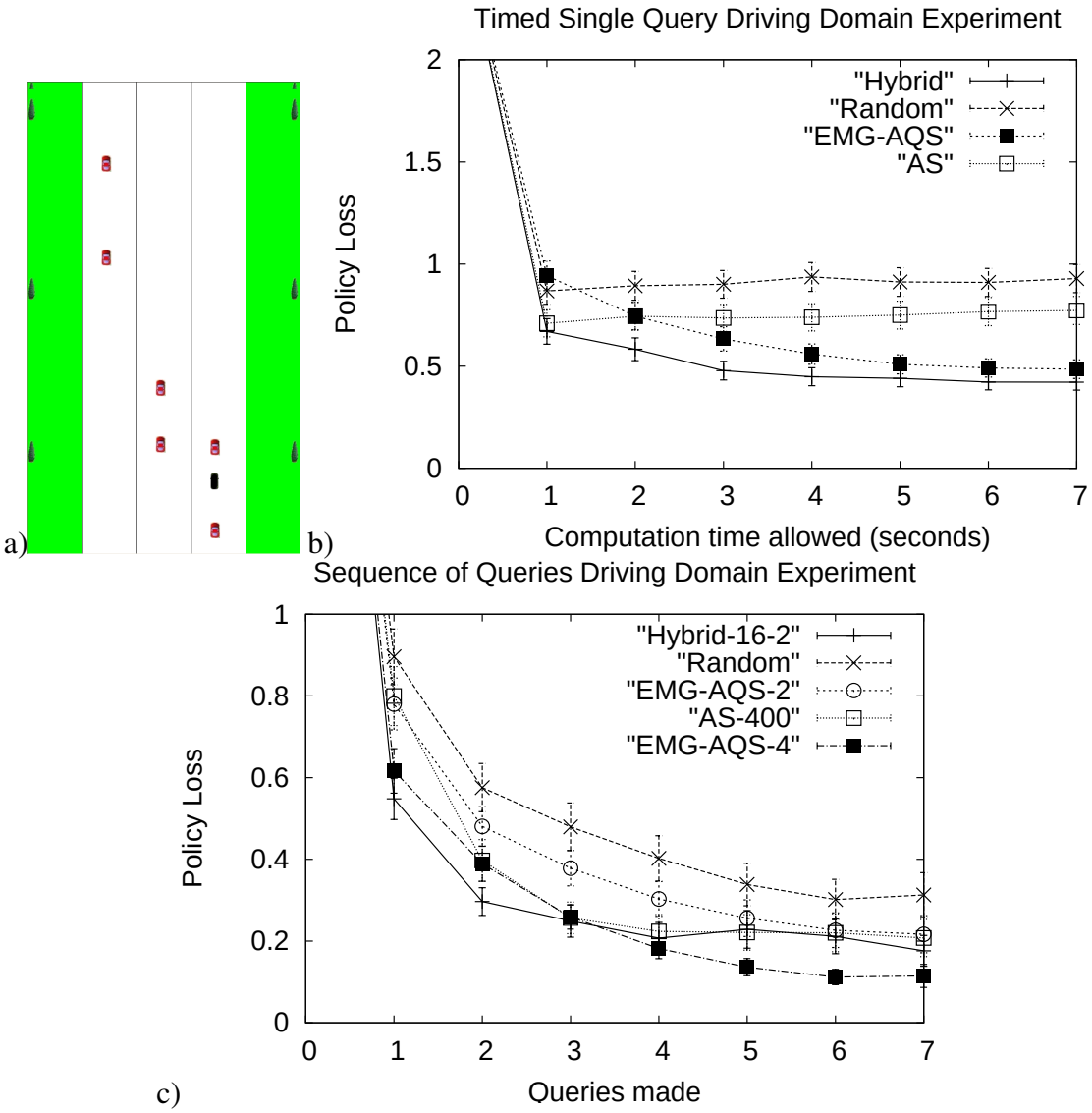


Figure 3.4: a) The Driving Domain. Average policy loss for various query strategies according to b) computation time allotted and c) number of queries made. Error bars are confidence intervals for p-value 0.05.

Here I test how the strategies perform over a sequence of queries in the Driving Domain, when again computation time is limited. I employ three strategies that each take roughly 2 seconds to select a query per step: AS applied to a set of 400 random queries, EMG-AQS applied to a set of 2 random queries, and Hybrid where AS selects 2 queries from a set of 16 random queries and EMG-AQS selects the best of those (these correspond to T=2 for each strategy on the previous graph). I also employ EMG-AQS applied to a set of 4 random queries, which consumes 4 seconds of computation per step, as well as Random.

Figure 3.4c shows that AS-400 outperforms EMG-AQS-2, but that the Hybrid method,

which adds a small computational cost (about 4%) to EMG-AQS-2, outperforms both. EMG-AQS-4, consuming twice the computation time, outperforms the rest (except when beaten by Hybrid for the first two queries). I also ran the strategies over larger query sets, and found that EMG-AQS and Hybrid continued to improve as they were allowed to sort through more random queries, while AS did not. These results build on the trends observed for a single query: EMG-AQS only outperforms AS when a certain level of computation is available, but Hybrid's usage of AS as a query filter can boost EMG-AQS's performance when computation is limited to get the best of both worlds.

3.4 Conclusions

In this chapter I considered three types of queries that an agent acting in an uncertain sequential decision-making problem can ask. For reward and transition queries, which query parameters of the agent's uncertainty over MDP models directly, I showed that Bayes updates for query responses can be performed efficiently when the agent's uncertainty is representable by independent Dirichlet distributions. Then I discussed several straightforward approximations that can be applied to reduce EVOI computations to repeated sequential optimal planning computations, which in turn can be performed by standard optimal planning methods, producing the Expected Myopic Gain (EMG) algorithm for selecting transition and reward queries. I then presented empirical results that demonstrated EMG's near-optimality when its assumptions were met, and that it continued to be effective even when the assumptions were violated.

Then I introduced action queries, and developed a new action query selection method, called Expected Myopic Gain-based Action Query Selection (EMG-AQS), which implements EVOI-based action query selection by performing Bayes updates using Bayesian Inverse Reinforcement Learning, and takes advantage of the structure present when the agent has only reward uncertainty to simplify optimal planning computations. Even so, the optimal planning computations required by EMG-AQS are expensive and I presented an empirical comparison between EMG-AQS and an existing action query selection method, Active Sampling (AS), which has the computational advantage that it does not need to perform expensive optimal planning computations to select a query. Although EMG-AQS is Bayes-optimal for a single query, I tested the effects of two conditions under which EMG-AQS is not Bayes-optimal: performance over a sequence of queries, and performance given a fixed amount of time allotted to select queries. Under the former condition, I found that in most cases EMG-AQS outperformed AS over a sequence of queries due to its superior discretion between querying to improve short-term versus long-term reward collection. Under

the latter condition, I found that when little computation time was available, AS outperformed EMG-AQS, but once enough time was available, EMG-AQS outperformed AS. In addition, I devised the Hybrid algorithm, which performed better than either method alone in settings with limited computation time.

CHAPTER 4

Wishful Query Selection: Selecting from the k-Response Query Set

For typical settings in which the agent can query its user, the agent is restricted to asking a query from some specified set, such as the set of action-queries in the context of sequential decision-making considered in Chapter 3. Intuitively, the impact of asking queries in terms of improving the agent's performance partially depends on the usefulness of the set of queries it has access to. A natural question to ask, then, is how to determine when the agent's query set is expressive enough, i.e., how can the agent's designer know whether or not it would be useful to consider adding more queries to the agent's query set? One way to formally pose this question is as a query selection problem, where the agent can ask *any* k-response query (a query with k possible responses, with k fixed to some integer ≥ 2): can the agent consider only those queries contained in some subset of the set of all k-response queries without missing the most valuable k-response query? If so, that subset should be considered "expressive enough."

To this end, in this chapter I study the question of what the agent should ask its user in order to maximize EVOI when the *only* restriction on what the agent can ask is that it can only have k possible responses (where $k \geq 2$), in an abstract setting in which a decision-making agent must query *before* making its "decision" when (1) the agent may ask only a single query (myopic query selection); and (2) the agent may ask n queries (nonmyopic query selection).

I show that, in myopic settings, where the goal is to select a query so as to maximize EVOI without considering any future queries that could be asked, the set of k-response *decision queries*, each of which ask for the best decision out of some subset, is sufficiently general in that there is no benefit in considering any k-response queries beyond k-response decision queries. This result dovetails with recent work by Viappiani and Boutilier (2010), who contribute efficient approximate algorithms for k-response decision query selection that exploit the submodularity of a lower-bound for decision query EVOI.

In addition, I consider a nonmyopic setting where the goal is to select a depth-n k-response query tree instead of a single query. Here I show that the set of depth-n trees constructed from k-response decision queries is not sufficiently general, in that there are cases where more valuable depth-n k-response query trees exist outside the set of depth-n k-response decision query trees. However, I also show that the set of depth-n trees constructed from k-response decision-*set* queries is sufficiently general. Finally, I show the computational result that depth-n k-response query tree selection can be reduced to k^n-response decision query selection, where the algorithms contributed by Viappiani and Boutilier (2010) directly apply.

The results developed in this chapter provide guidance for how queries can be designed to be as informative as possible. In particular, they provide theoretical justification for using only decision/decision-set queries (in the myopic/nonmyopic settings) when these ideal types of queries can be asked. Moreoever, as I will show in Chapter 5 the results of this chapter for the myopic setting provide a solution to the wishful query selection problem proposed in Chapter 1, and hence play a crucial role in implementing the Wishful Query Projection (WQP) approach for query selection developed in this dissertation.

4.1 Problem Formulation

In this chapter I will use the models and notation specified in Section 2.1 of Chapter 2, focusing on the study of the k-response query set, which I define next.

There has been much past research on the subject of selecting Q-EVOI-optimal queries from specific query sets Q. For example, **Action Queries**, which ask about the optimality of actions in particular states of a Markov decision process (MDP), are a popular form of query recently studied in the areas of learning by demonstration (Chernova and Veloso, 2009) and active inverse reinforcement learning (Melo and Lopes, 2013; Judah et al., 2012). (See detailed discussion of action query selection in Chapter 3.) Specifically, action queries take the form "What is the best action to take in state s?" When the number of actions is finite, an action query can have only a finite number of responses. However, the number of possible action queries depends on the number of possible states, which can be large or infinite in many problems of interest.

As another example, **Bound Queries** ask whether the true value of some dimension of the unknown parameter exceeds a particular threshold, and thus cleanly map to the agent's uncertainty representation (Chajewska et al., 2000; Braziunas and Boutilier, 2005). For example, given uncertainty about the reward value of a goal state, a bound query might ask "Is the reward of this state above 0.5?" Bound-queries are binary-response queries

that have two possible responses (intuitively, "yes" and "no"); however, there are infinitely many such queries when the parameter being queried about is continuous.

One attribute that many query sets used by the research community have in common is that they contain only queries that have k possible responses for some fixed integer $k \geq 2$. For example, action query sets in sequential decision making settings are typically comprised of queries that each have $|A|$ possible responses (where $|A|$ is the size of the action space), and pairwise comparison query sets correspond to binary-response decision query sets (Bonilla et al., 2010b). The following three types of k-response query sets are at the core of this chapter:

k-Response Queries. Let Q_k denote the set of all k-response queries. Other than being limited to a fixed number of responses, this class is unconstrained. For example, Q_2 includes the Bound Queries mentioned previously.

k-Response Decision Queries. Let D_k denote the set of all k-response *decision queries*, where a query $q \in D_k$ asks which out of k decisions is best given knowledge of the uncertain parameter. As noted above, the semantics of a query are defined by its likelihood function $\Pr(q = j|\omega)$; thus the decision query q over decisions $\{u_i\}_{i=1}^k$ is defined so that $\Pr(q = j|\omega) = \delta(\arg\max_i\{V_\omega^{u_i}\} = j)$, where δ is the indicator-function that takes on the value of one if the equality in its argument is satisfied and zero otherwise. (I assume that in the event of a tie, the response with smallest index is chosen, unless stated otherwise.)

k-Response Decision-Set Queries. Let H_k denote the set of all k-response *decision-set queries*, where a query $q \in H_k$ asks which out of k finite *sets* of decisions contains the best decision. More formally, if q queries sets $\{U_i\}_{i=1}^k$, where each $U_i \subseteq U$, then $\Pr(q = j|\omega) = \delta(\arg\max_i\{\max_{u \in U_i} V_\omega^u\} = j)$ (with ties broken in the same manner as above).

Note that the size of Q_k is infinite (in general) while the sizes of H_k and D_k are finite, since the number of decisions is finite. Furthermore, $Q_k \supset H_k \supset D_k$.

4.2 Summary of Theoretical Results

The main results of this chapter stem from studying the following comparisons.

- Myopic query selection: Compare k-response queries with k-response decision queries

in terms of the most valuable queries they contain.

- Nonmyopic query selection: Compare depth-n k-response query trees, depth-n k-response decision query trees, and depth-n k-response decision-set query trees with each other in terms of the most valuable query trees they contain.

As contributions, these comparisons show that, for all finite decision problems and for all uncertainty ψ over them:

1. (*In a myopic setting, the agent can safely consider only decision queries.*) I show in Section 4.3 that the set of k-response decision queries is *EVOI-sufficient*: the EVOI-optimal k-response decision query has EVOI at least as high as any other k-response query.

2. (*In a nonmyopic setting, the agent can safely consider only decision-set queries.*) I show in Section 4.4 that the set of depth-n k-response decision-set query trees is *EVOI-sufficient*: the EVOI-optimal depth-n k-response decision-set query tree has EVOI at least as high as any depth-n k-response query tree.

3. (*In a nonmyopic setting, the agent cannot be limited to only decision queries.*) I show in Section 4.4 that the set of depth-n k-response decision query trees is not *EVOI-sufficient*: the EVOI-optimal depth-n k-response decision query tree may have lower EVOI than the EVOI-optimal depth-n k-response query tree.

4.3 Myopic k-Response Query Selection

The first query selection problem I consider in this chapter is the myopic k-response query selection problem:

$$q_k^* = \arg\max_{q \in Q_k} EVOI(q; \psi), \tag{4.1}$$

for some fixed $k \geq 2$. That is, what should the agent ask when the only restriction is that its query must have exactly k responses? Before tackling that problem, consider what the agent should ask when there is no restriction at all in what it can ask, i.e., when even k is unrestricted.

Prior to asking its query, the agent's Bayes-optimal decision u_ψ^* may be suboptimal with respect to the true realization of the uncertain parameter, inducing expected loss

$$\mathbb{E}_{\omega \sim \psi}[V_\omega^* - V_\omega^{u_\psi^*}].$$

Note that this upper-bounds the maximum EVOI a query could have, and that any query allowing the agent to behave optimally thereafter in response (i.e., allowing the agent to adopt $\max_u V^u_\omega$ under any $\omega \in \Omega$ as a function of the query response) achieves this EVOI. Thus, the query that asks "Which decision in U has highest value under the true realization of the uncertain parameter?" must have the highest EVOI out of any possible query. This result implies that in the extreme case where $k = |U|$ (recall that U corresponds to the agent's decision set), one solution to Equation 4.1 is the query that asks "What is the optimal decision?" which, in fact, is a member of the set of $|U|$-response decision queries.

Returning to the k-response query selection problem (Equation 4.1), the agent will, in general, no longer have the ability to completely resolve its uncertainty regarding which decision it should execute via a single query (typically, $k \ll |U|$): but does the agent need to consider all k-response queries? A desirable property of a k-response query set Q would be that for all finite decision problems and for all uncertainty ψ over them, Q always contains a k-response query with EVOI as high as any other k-response query, so that there would be no benefit in considering any k-response queries beyond those contained by Q. More formally, I will say that a query set Q is *k-response EVOI-sufficient* (hereafter, simply *EVOI-sufficient* with the constraint k on the number of responses left implicit) if Q satisfies

$$\sup_{q \in Q} EVOI(q, \psi) = \sup_{q' \in Q_k} EVOI(q', \psi).$$

Next I derive the first and main result of this chapter: Theorem 4.2, which states that the set of k-response *decision* queries is EVOI-sufficient. Intuitively, this means that the above result that the $|U|$-response decision query has the highest possible EVOI for any unrestricted-response query generalizes to the D_k-EVOI-optimal query having the highest possible EVOI for any *k-response query* (where k is fixed). This reduces the (myopic) k-response query selection problem to the (myopic) k-response decision query selection problem, and I discuss algorithms for the latter in Section 4.5.

To prove that the set of k-response decision queries is EVOI-sufficient, I will begin by proving the following lemma, and then I will show how it directly leads to Theorem 4.2. (Theorem 4.2 is actually a special case of Lemma 4.1, and in Chapter 6 I will use Lemma 4.1 again to prove another result.)

Lemma 4.1. *Consider an arbitrary k-response query q, and suppose the agent will execute decision u_j upon receiving response j to q for $j = 1, 2, \ldots, k$. Let d_q be the k-response decision query over $u_1, u_2, \ldots, u_k$. Then,*

$$\sum_{j=1}^{k} \Pr(q=j) V_{\psi|q=j}^{u_j} \leq \sum_{j=1}^{k} \Pr(d_q=j) V_{\psi|d_q=j}^{u_j}.$$

Proof.

$$\begin{aligned}
\sum_{j=1}^{k} \Pr(q=j) V_{\psi|q=j}^{u_j} &= \int_{\Omega} \psi(\omega) \sum_{j=1}^{k} \Pr(q=j|\omega) V_{\omega}^{u_j} d\omega \\
&\leq \int_{\Omega} \psi(\omega) \sum_{j=1}^{k} \Pr(q=j|\omega) \max_{i \in 1,\ldots,k} V_{\omega}^{u_i} d\omega \\
&= \int_{\Omega} \psi(\omega) \max_{i \in 1,\ldots,k} V_{\omega}^{u_i} \sum_{j=1}^{k} \Pr(q=j|\omega) d\omega \\
&= \int_{\Omega} \psi(\omega) \max_{i \in 1,\ldots,k} V_{\omega}^{u_i} d\omega \\
&= \sum_{j=1}^{k} \Pr(d_q=j) V_{\psi|d_q=j}^{u_j}.
\end{aligned}$$

□

The last step follows since the following two quantities are equivalent: (1) the expected highest value under the uncertain parameter out of the k posterior-executed decisions; and (2) the expected value associated with asking which of the the k posterior-executed decisions has highest value under the uncertain parameter and then executing that decision.

Intuitively, Lemma 4.1 can be interpreted as follows. Consider the expected value associated with asking a k-response query q and then executing some arbitrary decision $u_j \in U$ upon observing response j to q (note that these decisions may not be Bayes-optimal decisions with respect to the corresponding posteriors). Lemma 4.1 implies that this value can only be improved by replacing q with the k-response decision query asking about those decisions $u_1, u_2, \ldots, u_k$.

However, typically the agent will ask a query in order to improve the expected value of its decision. That is, upon asking a k-response query q and observing response j, it executes the posterior Bayes-optimal decision as opposed to some arbitrary decision. Since Lemma 4.1 is true for any arbitrary set of decisions executed for the posterior distributions, it applies to this case where the decisions are posterior Bayes-optimal as well. Intuitvely, this implies that if the agent will use the response to a k-response query q to choose the best of decisions $\{u_j\}_{j=1}^{k}$ (which may contain duplicates), the agent can only do better by getting straight to the point and asking which of those decisions is best directly instead of

asking q. This gives rise to the query improvement procedure I discuss next.

Query improvement procedure. Consider the procedure that replaces an arbitrary k-response query q with the k-response decision query d_q, which asks which of q's posterior Bayes-optimal decisions is best given full knowledge of the uncertain parameter. Theorem 4.2 shows that the set of k-response decision queries is EVOI-sufficient by showing that this procedure can only improve the query, in that d_q must have EVOI at least as high as the original query q:

Theorem 4.2. *(The set of k-response decision queries is EVOI-sufficient.)*

For all finite decision problems and for all uncertainty ψ over them, the EVOI-optimal k-response decision query has EVOI equal to that of the EVOI-optimal k-response query:

$$\sup_{q \in Q_k} \{\textit{EVOI}(q, \psi)\} = \max_{q' \in D_k} \{\textit{EVOI}(q', \psi)\}.$$

Proof. Consider an arbitrary k-response query q and recall that the decision $u^*_{\psi|q=j}$ is the Bayes-optimal decision for the posterior distribution $\psi|q = j$ (the posterior induced by the j^{th} response to query q). Let d_q denote the k-response decision query that asks for the optimal decision in the set $\{u^*_{\psi|q=j}\}_{j=1}^k$. Then,

$$\begin{aligned}
\sum_{j=1}^{k} \Pr(q = j) V^*_{\psi|q=j} &= \sum_{j=1}^{k} \Pr(q = j) V^{u^*_{\psi|q=j}}_{\psi|q=j} \\
&\leq \sum_{j=1}^{k} \Pr(d_q = j) V^{u^*_{\psi|q=j}}_{\psi|d_q=j} \quad \textbf{(By Lemma 4.1)} \\
&\leq \sum_{j=1}^{k} \Pr(d_q = j) V^*_{\psi|d_q=j},
\end{aligned}$$

implying that $\text{EVOI}(q, \psi) \leq \text{EVOI}(d_q, \psi)$. Since this is true for any k-response query q, it is also true for the Q_k-EVOI-optimal query, so the D_k-EVOI-optimal query must be Q_k-EVOI-optimal. □

Thus, I have shown that the set of k-response decision queries is EVOI-sufficient. This means that when restricted to asking k-response queries, there is no loss in considering only k-response decision queries.

Recursive query improvement. As discussed above, a query q can be improved (in terms of EVOI) to a decision query d_q that asks which of q's posterior Bayes-optimal decisions has highest value under the uncertain parameter. However, note that the j^{th} decision asked about by d_q might not be posterior Bayes-optimal when j is the response to d_q. For example, the j^{th} decision u_j asked about by d_q might have lower value than another decision u'_j for all $\omega \in \Omega$ that prescribe response j to d_q. As a result, applying this query improvement procedure to d_q may yield another query d'_q that has higher EVOI than d_q. In general, repeating this query improvement procedure recursively would converge to a k-response decision query q^*_k with the property that when j is the response, the j^{th} decision in the set queried by q^*_k is Bayes-optimal[1]. I will refer to k-response decision queries that have this property as *locally optimal k-response decision queries*, and will revisit this point in Section 4.5 where I discuss algorithms for k-response decision query selection.

4.4 Nonmyopic k-Response Query Selection

Now I turn to the nonmyopic setting and ask: what query *trees* of depth-n should the agent consider when each query is constrained to having k responses?[2] Here, a *depth-n query tree* prescribes an i^{th} query to ask as a function of the responses to the $i-1$ queries already asked. I begin the analysis by defining EVOI for query trees. Then I show that the set of depth-n k-response *decision-set* query trees contains a query tree with EVOI at least as high as any other depth-n k-response query tree. Lastly, I show that the same is *not* true for depth-n k-response decision query trees. That is, Theorem 4.2 does not generalize to nonmyopic query selection.

4.4.1 Expected Value of Information for Query Trees

Let $M_n(Q)$ represent the set of all depth-n query trees that select queries only from query set Q. When a depth-n query tree μ is used to select n queries, the result is a trajectory of queries and responses. Let $X(\mu)$ denote the random variable where $\Pr(X(\mu) = j)$ represents the probability that the j^{th} such trajectory is realized when μ is used to select n queries.

Similarly to asking a single query, using μ to select n queries results in an updated

[1] Applying the query improvement procedure recursively until strict EVOI improvement is no longer possible must converge in a finite number of recursions, since there are a finite number of decisions and hence a finite number of k-response decision queries.

[2] Note that this is a departure in the context of this dissertation, as I do not study nonmyopic query selection anywhere else.

posterior at the leaves; let $\psi|X(\mu) = j$ refer to the posterior distribution induced when the j^{th} query and response trajectory is realized, which leads to a possibly new Bayes-optimal decision $V^*_{\psi|X(\mu)=j}$. Thus, the EVOI associated with μ under ψ can be written as

$$\text{EVOI}(\mu, \psi) = V^*_{\psi|X(\mu)} - V^*_{\psi},$$

exposing the intuitive fact that the EVOI of any depth-n k-response query tree can be thought of as the EVOI of an equivalent k^n-response query, as stated by the following lemma:

Lemma 4.3. *(Depth-n k-response query trees can be represented as k^n-response queries.)*

For all finite decision problems and for all uncertainty ψ over them, for all depth-n k-response query trees μ, there exists a k^n-response query q^μ such that the EVOI of μ is equal to the EVOI of q^μ, implying that

$$\sup_{\mu \in M_n(Q_k)} \{\textit{EVOI}(\mu, \psi)\} \leq \sup_{q \in Q_{k^n}} \{\textit{EVOI}(q, \psi)\}.$$

Proof. Consider an arbitrary depth-n k-response query tree μ. Since there are k^n possible trajectories of queries and responses resulting from using μ to select a trajectory of n queries, it is possible to construct a k^n-response query q^μ such that $\Pr(q^\mu = j|\omega) = \Pr(X(\mu) = j|\omega)$. Thus, $\text{EVOI}(\mu, \psi) = \text{EVOI}(q^\mu, \psi)$ since μ and q^μ are interchangable in terms of their effect on ψ. □

I will say that a depth-n k-response query tree set M is *Depth-n k-Response EVOI-sufficient* if M always contains a query tree with EVOI at least as high as any other depth-n k-response query tree. (Hereafter, I omit the dependence on k and n and refer to such query tree sets as EVOI-sufficient.)

Next I show that the set of depth-n k-response decision-*set* query trees is EVOI-sufficient, while the set of depth-n k-response decision query trees is not EVOI-sufficient.

4.4.2 Decision Queries and Decision-Set Queries in Nonmyopic Query Selection

First I show that the set of depth-n k-response decision-set query trees is EVOI-sufficient. The following lemma provides a key step in proving this fact, reversing the relationship shown by Lemma 4.3 for the special case of depth-n k-response decision-set query trees and k^n-response decision queries:

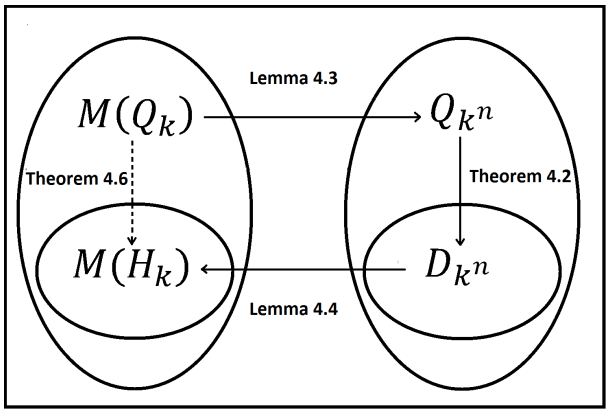


Figure 4.1: Diagram illustrating the main steps used to prove Theorem 4.6. Each arrow represents a statement that, for any query/query tree contained in the set at the tail of the arrow, a query/query tree with equal or higher EVOI must exist in the set at the head of the arrow. The three solid arrows, together with the fact that $M(H_k) \subset M(Q_k)$, imply Theorem 4.6 (represented by the dotted arrow).

Lemma 4.4. *(k^n-Response decision queries can be represented as depth-n k-response decision-set query trees.)*

For all finite decision problems and for all uncertainty ψ over them, for all k^n-response decision queries q, there exists a depth-n k-response decision-set query tree μ^q such that the EVOI of q is equal to the EVOI of μ^q, implying that

$$\max_{q \in D_{k^n}} \{\textit{EVOI}(q, \psi)\} \leq \max_{\mu \in M_n(H_k)} \{\textit{EVOI}(\mu, \psi)\}.$$

Proof. Let q denote any k^n-response decision query. I will prove the result by showing that it is always possible to construct a depth-n k-response decision-set query tree μ^q so that $\text{EVOI}(\mu^q, \psi) = \text{EVOI}(q, \psi)$.

μ^q can be constructed as follows. Let $Z_k(S)$ be any function that partitions a set S into k disjoint sets (where $|S|$ divisible by k), and let U^q denote the set of decisions queried by q. Then, construct μ^q such that $\mu^q(\psi)$ is the query that asks the decision-set query about sets $Z_k(U^q)$, and let $\mu^q(\psi|\mu^q(\psi) = j)$ be the query that asks the decision-set query about sets $Z_k\Big(Z_k(U^q)_j\Big)$, and so on. When taken together, responses to queries selected when using μ^q to select a trajectory of n k-response decision-set queries exactly determine which decision out of the original set U^q has maximum value for every possible ω.

Thus, if, under a particular ω, invoking μ^q to select a trajectory of n k-response decision-set queries determines that decision u_j^q is best out of the set, the response to q under ω would be j, and vice-versa; thus, for all $\omega, \Pr(X(\mu^q) = j|\omega) = \Pr(q = j|\omega)$, which implies that $\text{EVOI}(\mu^q, \psi) = \text{EVOI}(q, \psi)$. □

As a side note, the above construction implies that the agent can restrict attention to k-response decision-set queries containing decision sets of size k^{n-1} or less, compared to the set of all k-response decision-set queries, which includes those containing decision sets of size up to $|U|$.

Corollary 4.5. *For all finite decision problems and for all uncertainty ψ over them, the EVOI-optimal depth-n k-response decision-set query tree can be constructed using decision-set queries whose decision sets contain no more than k^{n-1} decisions.*

I now put together Lemma 4.3, Lemma 4.4, and Theorem 4.2 to prove that the EVOI-optimal depth-n k-response query tree has EVOI no higher than the EVOI-optimal depth-n k-response decision-set query tree (see Figure 4.1 for a visualization of the proof below):

Theorem 4.6. *(The set of depth-n k-response decision-set query trees is EVOI-sufficient.)*

For all finite decision problems and for all uncertainty ψ over them, the EVOI-optimal depth-n k-response query tree has EVOI equal to the EVOI of the EVOI-optimal depth-n k-response decision-set query tree:

$$\sup_{\mu \in M_n(Q_k)} \{\textit{EVOI}(\mu, \psi)\} = \max_{\mu' \in M_n(H_k)} \{\textit{EVOI}(\mu', \psi)\}.$$

Proof. Invoking Lemma 4.3, Theorem 4.2, and Lemma 4.4 sequentially,

$$\begin{aligned}
\sup_{\mu \in M_n(Q_k)} \{\text{EVOI}(\mu, \psi)\} &\leq \sup_{q \in Q_{k^n}} \{\text{EVOI}(q, \psi)\} \\
&= \max_{q' \in D_{k^n}} \{\text{EVOI}(q', \psi)\} \\
&\leq \max_{\mu' \in M_n(H_k)} \{\text{EVOI}(\mu', \psi)\},
\end{aligned}$$

implying that

$$\sup_{\mu \in M_n(Q_k)} \{\text{EVOI}(\mu, \psi)\} = \max_{\mu' \in M_n(H_k)} \{\text{EVOI}(\mu', \psi)\},$$

since $M_n(H_k) \subseteq M_n(Q_k)$.

□

I now show that the set of depth-n k-response decision query trees is not EVOI-sufficient, by constructing an example where no depth-2 binary-response decision query tree can have EVOI as high as the EVOI-optimal depth-2 binary-response decision-set query tree:

Theorem 4.7. *(The set of depth-n k-response decision query trees is not EVOI-sufficient.)*

There exist finite decision problems under uncertainty ψ where the optimal depth-n k-response decision query tree has lower EVOI than the optimal depth-n k-response query tree, i.e., where

$$\sup_{\mu \in M_n(Q_k)} \{\textit{EVOI}(\mu, \psi)\} > \max_{\mu' \in M_n(D_k)} \{\textit{EVOI}(\mu', \psi)\}.$$

Proof. Consider a decision problem with four possible decisions u_1, u_2, u_3 and u_4, where all $4!$ orderings over the values of each decision are supported by ψ and no two decisions have the same value under any parameter. Lemma 4.3 and Lemma 4.4 together imply that the EVOI-optimal depth-n k-response query tree has EVOI equal to that of the EVOI-optimal k^n-response decision query. Thus, here the EVOI-optimal depth-2 binary-response query tree has EVOI equal to that of the EVOI-optimal 4-response decision query q^*. In this problem, asking q^* would allow the agent to act optimally as a function of the response to q^* since there are only four possible decisions, which is achievable by the EVOI-optimal depth-2 binary-response query tree by Lemma 4.4. However, no depth-2 binary-response decision query tree exists that can meet this requirement.

To see this, suppose that the agent's first query asks which is the better of u_1 and u_2, and that the response is u_1. Since all possible orderings of the decision values are supported by ψ, this still leaves any of u_1, u_3, and u_4 as candidates for being the best out of all four decisions, and similarly, any subsequent query asked will leave the agent uncertain about which is the better of two decisions (three if it asks the same query again), hence in this case there is no depth-2 binary-response decision query tree in which the agent asks about u_1 and u_2 first that ensures the agent will act optimally after asking two queries. Due to symmetry, the same is true no matter which query is asked first, implying that no depth-2 binary-response decision query tree guarantees that the agent will act optimally after asking 2 queries. □

4.5 Algorithms for k-Response Query Selection

Recall that this chapter is focused on studying the k-response query selection problem. In Section 4.3 I showed that the k-response query selection problem can be reduced to k-response decision query selection since the set of k-response decision queries is EVOI-sufficient (Theorem 4.2). Next I discuss algorithms for myopic k-response decision query selection.

4.5.1 Myopic k-Response Query Selection

The problem of selecting the EVOI-optimal k-response decision query has been studied by Viappiani and Boutilier (2010) (the decision queries of this chapter correspond to their "noiseless choice queries"). In particular, they show that greedily constructing a k-response decision query (where, at each iteration, the next decision to add to the set asked about is determined by maximizing EVOR, a lower bound for EVOI defined below which can be computed efficiently) closely approximates EVOI-optimal k-response decision query selection. Next I summarize their results, which connect with Theorem 4.2.

To begin, define the *Expected Value of Recommendation* (EVOR) of a k-response decision query d as follows, letting u_j^d denote the decision associated with the j^{th} response to d:

$$EVOR(d, \psi) = \mathbb{E}_{\omega \sim \psi}\left[\max_j V_\omega^{u_j^d}\right] - V_\psi^*.$$

EVOR is the same as EVOI applied to decision queries, except when the response is j, EVOR considers the expected value associated with executing decision u_j^d, as opposed to EVOI which considers the expected value associated with executing the posterior Bayes-optimal decision $u^*_{\psi|d=j}$. Since the Bayes-optimal posterior decision when u_j is the response to a decision query may not be u_j, EVOR is a lower-bound for EVOI, i.e., for any k-response decision query $d \in D_k$,

$$EVOR(d, \psi) \leq EVOI(d, \psi). \tag{4.2}$$

Recall from Section 4.3 that starting with an arbitrary k-response query and recursively applying the query improvement procedure (presented below the proof of Theorem 4.2) eventually converges to a *locally optimal k-response decision query*, which has the property that when decision u_j is the response, u_j is the new Bayes-optimal decision. Letting D_k^* denote the set of locally optimal k-response decision queries, this implies that

$$\max_{d \in D_k^*} EVOI(d, \psi) = \max_{d' \in D_k} EVOI(d', \psi), \tag{4.3}$$

and that for any locally optimal k-response decision query d (i.e., $d \in D_k^*$),

$$EVOR(d, \psi) = EVOI(d, \psi). \tag{4.4}$$

Combining these three facts, we have that

$$\begin{aligned}\arg\max_{d\in D_k} EVOI(d,\psi) &= \arg\max_{d\in D_k^*} EVOI(d,\psi)\\ &= \arg\max_{d\in D_k^*} EVOR(d)\\ &= \arg\max_{d\in D_k} EVOR(d),\end{aligned}$$

i.e., the EVOI-optimal k-response decision query can be computed by maximizing EVOR instead of EVOI. Furthermore, EVOR is monotonic and submodular in k, implying EVOR maximization can be closely approximated by greedily constructing the query on the basis of EVOR (Nemhauser et al., 1978), and Equation 4.4 implies that the same procedure closely approximates EVOI maximization as well. Next I summarize the properties of the aforementioned greedy construction procedure compared to an exhaustive aproach.

Let the computational complexity of executing a single Bayes update be $\mathcal{O}(B)$, where B is a measure of the size of the problem (which I leave undefined here because I will simply count how many such updates are performed by the different algorithms).

Exhaustive k-Response Decision Query Selection Algorithm. Exhaustively evaluate each possible k-response decision query and select the best one. This has computational complexity $\mathcal{O}(|U|^k kB)$.

Greedy k-Response Decision Query Selection Algorithm. Approximate the EVOI-optimal k-response decision query by greedily constructing the set of k decisions it asks about as follows. Begin with the empty set, and on the first iteration, add the prior Bayes-optimal decision u_ψ^* to the set. Then on each subsequent i^{th} iteration, expand the set of $i-1$ decisions added so far to include the decision that maximizes the EVOR of the i-response decision query that asks about those decisions. This algorithm enjoys the guarantee that the EVOI of the k-response decision query constructed is within a factor of $1-(\frac{k-1}{k})^k$ (at worst 63%) of EVOI-optimal (due to monotonicity and submodularity of EVOR, as explained above), and has computational complexity $\mathcal{O}(k^2|U|B)$.

4.5.2 Nonmyopic k-Response Query Selection

Although Viappiani and Boutilier (2010) do not discuss algorithms for nonmyopic query selection, the same algorithms as described above can be applied in the nonmyopic setting by exploiting the theoretical results provided by this chapter.

Namely, combining Theorem 4.2 with Theorem 4.6 implies that the EVOI-optimal

depth-n k-response query tree can be computed through two steps: (1) compute the EVOI-optimal k^n-response decision query q^*; (2) contruct a depth-n k-response decision-set query tree μ^* yielding the same EVOI as q^*.

Working backwards, step (2) can be implemented by the procedure described in the proof of Lemma 4.4, which involves computing any size-k partition of the k^n decisions queried by q^* for all k^n nodes of the tree. Since each of these k^n computations is $\mathcal{O}(k^n)$, step (2) has complexity $\mathcal{O}(k^{2|U|})$. Implementing step (1) by either the exhaustive or the greedy algorithm above, then, yields the two algorithms below for depth-n k-response decision-set query tree selection.

Exhaustive Depth-n k-Response Decision-set Query Tree Selection Algorithm. This algorithm implements step (1) using the exhaustive k-response decision query selection algorithm above, and so its computational complexity is $\mathcal{O}(|U|^{k^n} k^n B)$.

Greedy Depth-n k-Response Decision-set Query Tree Selection Algorithm. This algorithm approximates step (1) using the greedy k-response decision query selection algorithm described above, and so it has computational complexity $\mathcal{O}(k^{2n}|U|B)$ while offering the guarantee that the EVOI of the query tree computed is within a factor of $1 - (\frac{k^n-1}{k^n})^{k^n}$ (again, at worst 63%) of EVOI-optimal.

Thus, the computational problem of selecting an EVOI-optimal depth-n k-response query tree can be reduced to selecting an EVOI-optimal k^n-response decision query.

4.6 Discussion

In this chapter, I considered the problem of selecting a query when the only restriction on what queries can be asked is that they must have exactly k possible responses (for some $k \geq 2$). In the myopic setting, I proved that the set of k-response decision queries is EVOI-sufficient, which intuitively means that there is no benefit in considering additional k-response queries beyond decision queries. In the nonmyopic setting, where queries are used to construct depth-n query trees, I showed that the set of depth-n k-response decision query trees is *not* EVOI-sufficient, but that the more general set of depth-n k-response decision-*set* query trees is in fact EVOI-sufficient.

I then discussed algorithms developed in related work that can be directly applied to provably approximate k-response decision query selection, and moreover, I showed that the same algorithms apply to selecting depth-n k-response decision-set query trees, since depth-n k-response decision-set query tree selection can be reduced to k^n-response deci-

sion query selection.

I note that while I showed in this chapter that decision queries and decision-set queries are EVOI-sufficient in the myopic and nonmyopic settings respectively, I did not discuss how humans may understand and answer queries from these sets. In some application domains, this may indeed be a practical challenge. In particular, it is clear that decision-set queries would be difficult for humans to answer unless the component decision-sets were to correspond to, say, well-understood (by humans) categories of decisions. In the next chapter I take a step towards answering these types of questions by providing a principled way to choose a query from some askable set to replace a *decision* query, but studying how decision and decision-set queries can best be approximately conveyed in practical applications and how to take into account the cognitive burden they impose when evaluating them are important directions for future work, and I discuss this subject further in Chapter 7.

As another direction, in the next chapter I show how, in the myopic setting, the theoretical properties of decision queries and the computational tractability of approximately selecting from them can be exploited to implement the first step of the WQP approach discussed in Section 1.2 of Chapter 1.

CHAPTER 5

Selecting from Arbitrary Subsets of k-Response Queries via Wishful Query Projection

In Chapter 3, I compared query selection algorithms for sequential decision-making settings when the agent was restricted to asking queries from particular structured query sets, and empirically found that, for the action query set, combining an uncertainty-based approach with an EVOI-based approach resulted in a more effective tradeoff between minimizing computational requirements and maximizing EVOI compared to using either approach alone. Then, in Chapter 4, I considered query selection in a general setting with minimal structure in the underlying decision problem and query set; namely, I considered the problem of selecting the query with highest EVOI assuming that *any* k-response query can be asked when the agent is faced with an abstract single-shot decision-making problem. There, I showed that the agent can ignore all k-response queries except k-response decision-queries without sacrificing EVOI. As a result, selecting from the space of all k-response queries reduces to k-response decision query selection.

In settings like those studied in Chapter 3, however, the agent can ask its user only queries that lie within some specified subset of all k-response queries, e.g., queries whose meanings the agent and user mutually understand, and whose required effort on the part of the user to answer is acceptable. Query selection in such cases can become computationally difficult in the absence of exploitable structure in the query set, and finding the best query may be compared to finding a needle in a haystack. In this chapter I draw from the intuition and theoretical results developed in Chapter 4 to develop general principles for designing query selection algorithms that apply to arbitrary query sets, specifically addressing the problem of selecting the EVOI-optimal query from some arbitrary *askable* set of k-response queries with all other details of the problem abstracted away.

As a general approach to solving this type of query selection problem, I implement the *Wishful Query Projection* (WQP) proposed in Chapter 1, and approach the problem of selecting from an arbitrary set Q of k-response queries as follows: first, compute the

EVOI-optimal k-response decision query d^* even though it cannot necessarily be asked; then, *project* d^* into Q using a query similarity criterion to find a query like d^* that *can* be asked. I show that implementing WQP with an uncertainty-based similarity criterion yields an algorithm offering a formal guarantee regarding the EVOI loss of the query selected as a function of how similar the askable query set is to the k-response decision query set, connecting back to the empirical success I found in Chapter 3 with a different type of hybrid EVOI-based and uncertainty-based approach.

5.1 Askable Query Selection Setting

In this chapter I study the same setting as the myopic setting studied in Chapter 4, where recall the agent is a single-shot decision maker with finite decision set U and arbitrary uncertainty ψ over arbitrary model space Ω with each $\omega \in \Omega$ prescribing value V_ω^u to each decision $u \in U$, where the problem is to select a k-response query to ask with the objective of maximizing EVOI. Unlike Chapter 4, however, here the set Q of queries to select from consists of only a *subset* of all k-response queries, where the particular subset cannot be chosen but is known to the agent. I will refer to $Q \subseteq Q_k$ as the *askable query set*. In this setting, the objective is to solve the *askable query selection problem*:

$$q^* = \arg\max_{q \in Q} EVOI(q; \psi), \tag{5.1}$$

where $Q \subseteq Q_k$. I will also make the simplifying assumption that Q is a finite query set. This assumption makes it possible to specify and study the properties of query selection algorithms that evaluate every askable query under different criteria, since without making additional assumptions about the structure of the askable query set, it is unclear how askable query selection could be accomplished in any principled manner without enumerating the askable query set. Although enumerating the askable query set in practice is likely to be infeasible, my intention is to provide analysis of general query selection techniques that might be combined with domain-specific techniques for scaling to particular settings. Examples of finite k-response query sets are the k-response decision query set (studied in detail in Chapter 4), structured subsets of the k-response decision query set (studied in detail in Chapter 6), the k-response action query set for sequential decision-making settings with finite action sets (studied in Chapter 3), and sets of queries that directly query about parameters of the decision problem (such as finite subsets of the reward and transition queries studied in Chapter 3).

5.2 Decision Entropy and Query Response-Entropy

Throughout this chapter I will use ψ_U to denote the discrete distribution over which decision in U has the highest value induced by ψ, where

$$\psi_U(u) = \eta \int_\Omega \psi(\omega)\delta(u = \arg\max_{u'} V_\omega^{u'}),$$

where η is a normalization constant.

I will use H to denote the Shannon entropy function, where for some discrete probability distribution Y, $H(Y) = -\sum_i Y(i)\log(Y(i))$. I will also apply H to queries so that $H(q)$ denotes the entropy over the discrete distribution over the responses to q under ψ (note that ψ is implicit unless stated otherwise); i.e., the *response-entropy* of a query q is defined as

$$H(q) \triangleq -\sum_j \Pr(q = j)\log(\Pr(q = j)), \tag{5.2}$$

where $\Pr(q = j) = \int_\Omega \Pr(q = j|\omega)d\psi(\omega)$.

Now, consider two queries q and q'. Observing response j to q updates the agent's distribution ψ over model parameters, so it also updates the agent's distribution over what the response to q' would be if asked. By asking q the agent can only improve (in expectation) its prediction of what the response would be if q' were to be asked, in that the expected posterior response-entropy of q' induced by asking q can be no higher than $H(q')$ (due to Jensen's inquality (Kuczma, 2009)). I use $H(q'|q = j)$ to denote the posterior response-entropy over q' upon observing response j to q, and I will use $H(q'|q)$ as shorthand to denote the expected posterior response-entropy of q' induced by observing the response to q, or more formally,

$$H(q'|q) \triangleq \mathbb{E}_{j\sim q;\psi}\big[H(q'|q = j)\big].$$

5.3 Approaches for Askable Query Selection

The naïve approach to solving the askable query selection problem would be to evaluate the EVOI of every query in Q, and to then select the one with highest EVOI: in this chapter I will refer to the algorithm following this approach as *Exhaustive*. Since every EVOI com-

putation requires, for each of the k possible responses, a Bayes update (whose complexity I will denote as B) and a posterior optimal planning computation (whose complexity I will denote as Π^*), the computational complexity of Exhaustive is $\mathcal{O}\big(|Q|kB + |Q|k\Pi^*\big)$. For the settings considered in this dissertation, the complexity of optimal planning dominates that of performing a single Bayes update, and hence the second term, $|Q|k\Pi^*$, is the main concern.

Exhaustive can be used to select only from small query sets, then, since the larger the askable query set, the more optimal planning computations must be performed. However, larger query sets can allow better queries to be asked across the variety of states of uncertainty in which the agent may find itself, so the extent to which an agent can take advantage of asking queries is tied to how many queries it considers. An efficient query selection algorithm, then, should have complexity scaling separately as a function of $|Q|$ and Π^*, so that the additional computation required when the query set is expanded is not a function of optimal planning complexity. One family of algorithms that typically have this property is uncertainty-based algorithms.

Uncertainty-based query selection algorithms can be used to select a query purely on the basis of *how much* the agent would learn by asking the query, as a proxy for expensive EVOI computations which take into accout *how useful* doing so would be. As discussed in Chapter 2, uncertainty-based algorithms are commonly used in non-decision-theoretic settings, such as Active Learning, where the natural objective is often to minimize some measure of the agent's uncertainty. Related work (Guo and Sanner, 2010; Boutilier et al., 2003), and the work described in Chapter 3, has studied the use of various uncertainty-based criteria as heuristics for EVOI-based query selection in decision-theoretic settings, where their main computational advantage is that they need not solve any optimal planning problems in order to select a query, provided computing Bayes updates for the askable query set does not require doing so.

Next I introduce MEDER, an uncertainty-based algorithm for askable query selection which I use extensively both in this chapter and in Chapter 6 to benchmark the theoretical properties and empirical performance of the algorithms I develop later in this chapter.

5.4 MEDER Query Selection Algorithm

Here I discuss a query selection algorithm that reduces a measure of the agent's *decision* uncertainty directly as a proxy for maximizing EVOI. Formally, the *Maximum Expected Decision Entropy Reduction* (MEDER) algorithm selects the query that maximally reduces the agent's expected posterior decision entropy. (The idea of selecting a query to reduce

the agent's expected posterior decision entropy is not new; e.g., see Wilson et al. (2012) who study a similar approach in a sequential decision-making setting.) MEDER selects a query from an askable set Q according to the following objective:

$$q_{\text{MEDER}} = \arg\max_{q \in Q} EER(q, \psi), \text{ where } EER(q, \psi) = H(\psi_U) - H(\psi_U|q).$$

For concreteness, I will assume that MEDER, like Exhaustive, applies its criterion to all askable queries in order to select the best one. MEDER, then, effectively replaces the EVOI computations of Exhaustive with EER computations. Without further assumptions, EER computations would have complexity scaling with $|U|$, which is the same as the complexity of optimal planning absent exploitable structure. In practice, however, criteria similar to decision-entropy can be computed much more efficiently than EVOI for some settings (e.g., see the Active Sampling algorithm of Lopes et al. (2009) discussed in Chapter 3). For this reason, I will distinguish decision-entropy computations from other decision-related computations by using E to represent the complexity of decision-entropy computations. The complexity of MEDER, then, is $\mathcal{O}\big(|Q|kB + |Q|kE\big)$.

Note that MEDER's EER criterion is just one choice out of many possible uncertainty-based criteria. Namely, a variety of other criteria have been studied in active learning settings that could conceivably be considered here (such as Fisher Information (Zhang and Oles, 2000), KL-Divergence (McCallum and Nigam, 1998), or version space shrinking (Dasgupta, 2004)). Additionally, the agent's uncertainty over model parameters (ψ) could be minimized instead of its uncertainty over which decision is best. While a comprehensive study of how MEDER might compare to other uncertainty-based algorithms is out of the scope of this dissertation, MEDER's approach of maximizing EER has two appealing properties in the context of askable query selection. First, unlike MEDER many other uncertainty-based approaches make additional assumptions regarding the form of the agent's uncertainty representation or query set, which do not apply to the general setting considered in this chapter. Second, although MEDER is uncertainty-based in that its objective of reducing the agent's overall decision uncertainty does not take into account how queries improve the agent's decision-making in terms of *value*, it does not completely ignore the agent's decision problem as many other uncertainty-based algorithms would.

In fact, using MEDER to approximate EVOI-based query selection is principled in that reducing EER reduces an upper bound on the loss associated with the agent's expected value in acting under uncertainty, as compared to if it were to act optimally. This bound, stated as Corollary 5.3 below, plays an important role in the subsequent derivations of

theoretical guarantees for the WQP algorithms introduced in Section 5.5. Next I prove a simple lemma which I apply in a key step in the derivation of Lemma 5.2, which then leads directly to Corollary 5.3. Intuitively, Lemma 5.1 provides a lower bound for how peaked a discrete probability distribution must be as a function of its entropy:

Lemma 5.1. *Let y be an arbitrary discrete probability distribution with sample space $\{1, 2, \ldots, k\}$, and let $j_{max} \triangleq \arg\max_j y(j)$. Then $y(j_{\max}) \geq e^{-H(y)}$.*

Proof.

$$H(y) \triangleq -\sum_{j=1}^{k} y(j) \log(y(j)) \geq -\sum_{j=1}^{k} y(j) \log(y(j_{\max})) = -\log(y(j_{\max})).$$

Thus, $-\log(y(j_{\max})) \leq H(y)$, implying that

$y(j_{\max}) \geq e^{-H(y)}$.

□

Note that when $H(y) = 0$, Lemma 5.1 implies that $y(j_{\max}) \geq 1$ (which is tight for the Dirac delta distribution: the *minimum* entropy distribution), and when $H(y) = -\log(\frac{1}{k})$, Lemma 5.1 implies that $y(j_{\max}) \geq \frac{1}{k}$ (which is tight for the uniform distribution: the *maximum* entropy distribution). Hence, the bound is tight for the two extreme values of $H(y)$; however, the bound is loose for values in between.

Intuitively, when applied to the agent's decision uncertainty ψ_U, Lemma 5.1 implies that as the agent's decision entropy $H(\psi_U)$ decreases, the extent to which the most likely decision to be optimal under ψ_U must be more likely to be optimal than the rest of the decisions increases. The following lemma uses this fact to upper bound the expected value loss of the Bayes-optimal decision as a function of the agent's decision entropy:

Lemma 5.2. *For any decision problem and uncertainty ψ over them, $\mathbb{E}_{\omega \sim \psi}[V_\omega^* - V_\omega^{u_\psi^*}] \leq (1 - e^{-H(\psi_U)})(V_{\max} - V_{\min})$, where ψ_U denotes the discrete distribution over which decision in U has highest value under the uncertain model parameter.*

Proof. Let $x = \arg\max_x \psi_U(x)$ (note that $x \in U$, i.e., x is a decision as opposed to an

index). Then,

$$
\begin{aligned}
& \mathbb{E}_{\omega\sim\psi}[V_\omega - V_\omega^{u^*_\psi}] \leq \mathbb{E}_{\omega\sim\psi}[V_\omega - V_\omega^x] \\
= \;& \psi_U(x)\mathbb{E}_{\omega\sim\psi|x=\arg\max_u V_\omega^u}[V_\omega - V_\omega^x] + (1-\psi_U(x))\mathbb{E}_{\omega\sim\psi|x\neq\arg\max_u V_\omega^u}[V_\omega - V_\omega^x] \\
= \;& (1-\psi_U(x))\mathbb{E}_{\omega\sim\psi|x\neq\arg\max_u V_\omega^u}[V_\omega - V_\omega^x] \\
\leq \;& (1-\psi_U(x))(V_{\max} - V_{\min}) \\
\leq \;& (1-e^{-H(\psi_U)})(V_{\max} - V_{\min}). \quad \textbf{(Lemma 5.1)}
\end{aligned}
$$

□

A simple application of Jensen's inequality (Kuczma, 2009) yields a similar bound that is a function of the expected posterior entropy of ψ_U when q is asked:

Corollary 5.3. *For any query q,*

$$\mathbb{E}_{j\sim\psi;q}\left[\mathbb{E}_{\omega\sim\psi|q=j}[V_\omega^* - V_\omega^{u^*_{\psi|q=j}}]\right] \leq (1-e^{-H(\psi_U|q)})(V_{\max} - V_{\min}).$$

Note that MEDER's objective of maximizing EER corresponds to minimizing the expected posterior entropy of ψ_U, since $H(\psi)$ does not depend on q. Corollary 5.3, then, implies that even though MEDER is uncertainty-based in nature, its focus on *reducing* the agent's decision-entropy in particular is formally tied, albeit indirectly, to *improving* the agent's decisions. Even so, at the extreme, a query with high EER can have zero EVOI, as illustrated by the following example:

Example 5.1. *Suppose U contains four decisions u_1, u_2, u_3, and u_4, the askable query set Q contains $q_{1,4}$ and $q_{2,3}$ (which ask about u_1 and u_4, u_2 and u_3 respectively), and that Ω consists of three possible models ω_1, ω_2, and ω_3, where ψ assigns them equal probability $\frac{1}{3}$. Further, suppose values range from -300 to 3.*

Now suppose u_1, u_2, and u_3 are all risky decisions in that, for each one, u_j has value j for ω_j, but the lowest possible value (-300) for the other two ω. On the other hand, u_4 is a safe decision in that u_4 has value 0 for all ω. Note that u_4 is the prior Bayes-optimal decision since it has expected value 0, while all other decisions have expected value that is at most -199. For the purpose of this example, assume that the user breaks ties uniformly when answering a query (i.e., for ω_1, when $q_{2,3}$ is asked, both decisions have value -300 for ω_1 so the probability is 0.5 for each response).

Consider which of the two askable queries would be better to ask. First, $EVOI(q_{2,3};\psi) = 0$ *because knowing its response will not change the Bayes-optimal decision. On the other hand,* $EVOI(q_{1,4}) = \frac{1}{3}$ *because* $\Pr(q_{1,4} = 1) = \frac{1}{3}$*, and* $\psi|q_{1,4} = 1$ *assigns probability* 1 *to* ω_1*, allowing the agent to change its decision to* u_1 *which has value* 1 *for* ω_1 *when the response is* u_1*, but effecting no change when the response is* u_4*. In summary,* $q_{1,4}$ *has EVOI of* $\frac{1}{3}$ *while* $q_{2,3}$ *has EVOI of* 0*.*

Using EER as a proxy for EVOI fails in this example, since u_4 *is not optimal for any* ω*, resulting in* $q_{1,4}$ *having EER of* 0 *as opposed to* $q_{2,3}$ *which has positive EER.*

While Example 5.1 shows that reducing EER does not necessarily lead to improvements in the agent's decision-making, Corollary 5.3 implies that reducing EER does reduce an upper bound on the extent to which the agent's decision value differs from that of the optimal decision. Further, in the limit where $H(\psi_U) = 0$, the loss evaluates to 0, since in such a case the agent has no uncertainty regarding the optimal decision and thus is guaranteed to act optimally by selecting the Bayes-optimal decision. Thus, MEDER will select the EVOI-optimal query when the askable query set contains a query allowing the agent to completely eliminate its decision uncertainty – i.e., when the query set contains a query equivalent to the $|U|$-response decision query d^*_U that asks the user to select from the entire set U of decisions.

However, the askable query set typically will not contain d^*_U, since it would be burdensome to ask the agent's user in that it essentially asks the user to program the agent's policy according to the user's preferences/knowledge. (Avoiding this burden is arguably one of the main motivations for incorporating query-asking into agents in the first place!) In cases where the askable query set does not contain d^*, then, MEDER can be viewed as the algorithm that selects the query that maximally reduces the entropy *over the responses to* d^*_U (to see this, note that $\psi_U(u_j) = \Pr\left(d^*_U = u_j;\psi\right)$). In this sense, using MEDER for query selection treats learning the response of d^*_U as the goal, but MEDER can run into issues like the one illustrated in Example 5.1 when achieving that goal with a single query from the askable query set is unrealistic.

5.5 Wishful Query Projection

I have shown that MEDER's objective of maximizing EER is related to minimizing the agent's expected value loss (Corollary 5.3), and that maximizing EER can be viewed

as minimizing the agent's response-entropy over the best possible query d^*_U. However, MEDER can be misled into selecting minimally valuable queries when learning the response to d^*_U is an unrealistic goal given the queries contained by the askable query set.

Suppose instead that the agent were to treat learning the response to a more realistic, but still valuable, decision query d^* as its goal when selecting an askable query. If the agent can find an askable query similar to d^*, this should result in the agent selecting a query with high EVOI to the extent that d^* has high EVOI. This novel goal-based query selection approach is at the core of the Wishful Query Projection (WQP) approach for query selection developed in this dissertation.

Consider the following way to factor EVOI-based query selection for arbitrary k-response query sets into two steps to select a query q: 1) solve the D_k-EVOI-optimal query selection problem to obtain query d^*, which may not be askable; and 2) project d^* into the askable query set by finding an askable query q similar to d^*. Note that step 1) here specifies an implementation for the wishful query selection step of WQP as presented in Section 1.2 of Chapter 1, and henceforth when I refer to WQP I will assume its first step is implemented this way. The remainder of this chapter is focused on implementing step 2): the query projection step.

In WQP, all computations related to optimal planning are frontloaded into the first step, which can be approximated efficiently (as discussed in Section 4.5). The second step's task of searching the askable query set for a query similar to d^*, then, can focus on finding an askable query similar to d^* without taking into account its effect on the agent's decision-making since queries similar to d^* are known to have high EVOI, which should find an askable query with high EVOI to the extent that the askable query set contains a query similar to d^*. I introduce an uncertainty-based implementation of this approach for query selection next, along with an approximation for it, and analyze in detail the corresonding computational and EVOI-maximizing properties of the two algorithms.

5.5.1 Directed Expected Entropy Reduction (DEER)

The first WQP algorithm I develop is *Directed Expected Entropy Reduction (DEER)*, which selects a k-response query from the askable query set Q according to the following steps:

- Compute the D_k-EVOI-optimal k-response query d^*:
  $d^* = \arg\max_{d \in D_k} EVOI(d; \psi)$
- Select the query q from Q that minimizes the agent's uncertainty over what the response to d^* would be:
  $q = \arg\min_{q \in Q} H(d^*|q)$

The first step of DEER computes d^*, the D_k-EVOI-optimal k-response decision query. Recall that d^* has the highest possible EVOI out of all k-response queries, making it the ideal query if only the agent could ask it. In the second step, DEER recasts the query selection problem to finding a query similar to d^*, which will produce a query with high EVOI to the extent that the askable query set Q contains such a similar query. Specifically, the second step selects the query which, in expectation (over the responses to q), induces the lowest posterior entropy over the responses to d^*. For example, the ideal match would be d^* itself (supposing $d^* \in Q$), since there can be no uncertainty regarding the response to d^* conditioned on its own response (recall that decision queries are answered deterministically). Connecting back to the discussion of MEDER in Section 5.4, the second step is equivalent to applying MEDER to a reduced version of the original problem where U is replaced by the subset of its decisions that are queried by d^* – so in this way, DEER is a hybrid EVOI-based and uncertainty-based query selection algorithm.

5.5.1.1 DEER Computational complexity

First, consider the computational complexity of the first step. As discussed in Chapter 4, Viappiani and Boutilier (2010) showed that the D_k-EVOI-optimal query can be computed exactly with complexity $\mathcal{O}(|U|^k kB)$ via evaluating the EVOR of every k-response decision query, or approximately with complexity $\mathcal{O}(|U|k^2 B)$ via greedy construction where again EVOR is used in place of EVOI.

Naïvely, DEER's second step can be computed with complexity $\mathcal{O}(|Q|kE + |Q|kB)$ by evaluating the expected posterior entropy over d^* induced by each askable query q. However, note that

$$H(d^*|q) = H(q|d^*) - H(q) + H(d^*) \quad \textbf{(Bayes' Rule for Conditional Entropy)} \,,$$

and so the agent can equivalently perform the second step by solving

$$q = \arg\min_{q \in Q} H(q|d^*) - H(q)$$

instead, which requires computing only k Bayes updates since the posteriors of d^* can be computed before iterating through the askable query set. This implements the second step with complexity $\mathcal{O}(|Q|kE + kB)$.

Combining the two steps, DEER's complexity is $\mathcal{O}(k(|U|^k B + |Q|E))$ when the first step is computed exactly, or $\mathcal{O}(k(|U|Bk + k|Q|E))$ when the first step is approximated via

greedy construction. In contrast, the computational complexity of exhaustive EVOI evaluation is $\mathcal{O}(|Q|kB + |Q|k\Pi^*)$, where Π^* represents the complexity of solving an optimal planning problem. In the absence of structure in the decision problem, the complexity of optimal planning is $\mathcal{O}(|U|)$, and so in such a setting DEER's complexity is factored into the sum of two terms, one of which depends on optimal planning complexity while the other depends on the size of the query set, compared to exhaustive query selection which must perform $|Q|k$ optimal planning computations. Thus, DEER achieves the disseration-wide computational goal of allowing the query set to be expanded *without* requiring additional computation that scales with optimal planning complexity.

Revisiting Example 5.1.

Once again consider Example 5.1, where MEDER selected $q_{2,3}$, which has EVOI = 0.0, instead of $q_{1,4}$, which has EVOI = $\frac{1}{3}$. Now consider which query DEER will select. For the first step, d^* is $q_{3,4}$ since knowing that u_3 yields the highest value yields the highest value improvement compared to u_1 or u_2. Then, DEER will select $q_{1,4}$. To see this, note that $\frac{1}{3}$ of the time the response to $q_{1,4}$ will be u_4, and conditioned on that the response to $q_{3,4}$ will always be u_4, inducing zero posterior response entropy for d^*, and $\frac{2}{3}$ of the time the posterior response entropy for d^* becomes $-\log(\frac{1}{2})$. In contrast, the response to $q_{2,3}$ always induces $-\log(\frac{1}{2})$ posterior response entropy for d^*. Thus, unlike MEDER, DEER selects the right query because the askable query set contains a query providing similar information to that of d^*, unlike the query about all decisions d^*_U which MEDER aims to approximate.

5.5.1.2 EVOI-loss performance bound for DEER

Here I derive a performance guarantee for DEER in the form of an upper bound on the *EVOI-loss* associated with the query it selects. More formally, the EVOI-loss of a query q selected from askable query set Q is defined as $\max_{q' \in Q} EVOI(q'; \psi) - EVOI(q; \psi)$. For example, the algorithm that selects a query by exhaustively evaluating the EVOI of every askable query always produces a query with zero EVOI-loss. Also, note that EVOI-loss is always nonnegative.

Recall from Chapter 4 that a locally optimal k-response decision query is defined as a k-response decision query with the property that (under ψ) the decision associated with the j^{th} response to the query is posterior Bayes-optimal conditioned on the j^{th} response to the query, and also recall that the set of locally optimal k-response decision queries is denoted

as D_k^*, which is a subset of D_k, the set of all k-response decision queries.

Recall that DEER selects a query by replacing the D_k-EVOI-optimal k-response decision query d^* with one contained in the askable query set whose response, in expectation, will induce the lowest posterior entropy over the responses to d^*. In this section I show that DEER is principled by deriving Theorem 5.5: an upper bound on how much worse the query DEER selects can be than the EVOI-optimal askable query q^*, in terms of EVOI. To begin, I establish Lemma 5.4, which shows that DEER's projection step is principled in that the EVOI lost in replacing a locally optimal k-response decision query with an arbitrary k-response query can be upper bounded as a function of the expected posterior response-entropy of d^* upon receiving a response to q:

Lemma 5.4. *For any $d \in D_k^*$ and $q \in Q$,*

$$EVOI(d;\psi) - EVOI(q;\psi) \leq \Big(1 - e^{-\mathbb{E}_{j\sim\psi;q}\left[H(d|q=j)\right]}\Big)\Big(V_{\max} - V_{\min}\Big),$$

where $V_{\max} = \sup_\omega V_\omega^*$ and $V_{\min} = \inf_{\omega,u} V_\omega^u$.

Proof. Let U_d denote the set of decisions queried by d. Then,

$$\begin{aligned}
& EVOI(d;\psi) - EVOI(q;\psi) = \mathbb{E}_{i\sim\psi;d}\Big[V^*_{\psi|d=i}\Big] - \mathbb{E}_{j\sim\psi;q}\Big[V^*_{\psi|q=j}\Big] \\
\leq\ & \mathbb{E}_{i\sim\psi;d}\Big[V^*_{\psi|d=i}\Big] - \mathbb{E}_{j\sim\psi;q}\Big[\max_{u\in U_d} V^u_{\psi|q=j}\Big] \\
\leq\ & \mathbb{E}_{\omega\sim\psi}\Big[\max_{u\in U_d} V^u_\omega\Big] - \mathbb{E}_{j\sim\psi;q}\Big[\max_{u\in U_d} V^u_{\psi|q=j}\Big] \ \textbf{(Since } d \in D_k^*\textbf{)} \\
=\ & \mathbb{E}_{j\sim\psi;q}\Big[\mathbb{E}_{\omega\sim\psi|q=j}\Big[\max_{u\in U_d} V^u_\omega\Big] - \max_{u\in U_d} V^u_{\psi|q=j}\Big] \ \textbf{(Law of iterated expectations)} \\
=\ & \mathbb{E}_{j\sim\psi;q}\Big[\mathbb{E}_{\omega\sim\psi|q=j}\Big[\max_{u\in U_d} V^u_\omega - V_\omega^{\arg\max_{u\in U_d} V^u_{\psi|q=j}}\Big]\Big] \\
\leq\ & \mathbb{E}_{j\sim\psi;q}\Big[\Big(1 - e^{-H(d|q=j)}\Big)\Big(V_{\max} - V_{\min}\Big)\Big] \ \textbf{(Corollary 5.3 with } U_d \textbf{ as } U \textbf{ and } \psi|q=j \textbf{ as } \psi\textbf{)}
\end{aligned}$$

□

Note that when $q = d$, $H(d|q = j) = 0$ for all responses j; in this case the right-hand side of the bound evaluates to 0, predicting correctly that there can be no loss in asking q in place of d. Otherwise, the loss increases smoothly as a function of the expected posterior entropy over d conditioned on the response to q.

Recall from Chapter 4 that the D_k-EVOI-optimal k-response decision query has the highest possible EVOI out of all k-response queries (Theorem 4.2): combining this fact with Lemma 5.4 yields an upper bound that is a function of how similar Q and D_k^* are in terms of the extent to which a query can be chosen from Q to minimize the expected

posterior entropy over the responses to an arbitrary locally optimal k-response decision query. I will use $H(Q, D_k^*)$ to represent this measure of similarity between Q and D_k^*, formally defined as

$$H(Q, D_k^*) = \max_{d \in D_k^*} \min_{q \in Q} \mathbb{E}_{j \sim \psi; q}\big[H(d|q = j)\big].$$

Then,

Theorem 5.5. *(Upper bound on DEER EVOI-loss.) For any askable k-response query set Q, if DEER selects q, then*

$$\max_{q^* \in Q} EVOI(q^*; \psi) - EVOI(q; \psi) \leq \Big(1 - e^{-H(Q, D_k^*)}\Big)\Big(V_{\max} - V_{\min}\Big).$$

Proof.

$$\begin{aligned}
\max_{q^* \in Q} EVOI(q^*; \psi) - EVOI(q; \psi) &\leq \max_{d^* \in D_k^*} EVOI(d^*; \psi) - EVOI(q; \psi) \quad \textbf{(Theorem 4.2)} \\
&\leq \max_{d^* \in D_k^*} \Big\{\Big(1 - e^{-H(q^*|q)}\Big)\Big(V_{\max} - V_{\min}\Big)\Big\} \quad \textbf{(Lemma 5.4)} \\
&\leq \Big(1 - e^{-H(Q, D_k^*)}\Big)\Big(V_{\max} - V_{\min}\Big).
\end{aligned}$$

□

Intuitively, Theorem 5.5 states that the richer the askable query set is in terms of the extent to which, for any locally optimal k-response decision query d, it contains a similar query to d, the better the guarantee associated with the EVOI loss of the query selected by DEER. Thus, DEER can be expected to select nearly Q-EVOI-optimal queries when Q is nearly as rich as the set of k-response decision queries D_k.

As a side note, a similar bound can be obtained for MEDER since MEDER's objective corresponds to minimizing the agent's expected posterior response-entropy for d_U^*, which recall is the ideal query d_U^* asking about all decisions (this bound is weaker than DEER's, however, since learning the response to d_U^* is typically unrealistic as discussed in Section 5.4). The bound is given below, and note that $D_{|U|}^*$ contains only d_U^*.

Corollary 5.6. *(Upper bound on MEDER EVOI-loss.) For any askable k-response query set Q, if MEDER selects q, then*

$$\max_{q^* \in Q} EVOI(q^*; \psi) - EVOI(q; \psi) \leq \Big(1 - e^{-H(D_{|U|}^*, Q)}\Big)\Big(V_{\max} - V_{\min}\Big).$$

EVOI-loss guarantee for greedy approximation of DEER. A similar guarantee applies to DEER when its first step is approximated as follows. First, perform the greedy construction procedure described in Section 4.5 to obtain a k-response decision query d. Then, recursively apply the query improvement procedure discussed in Section 4.3 to d until convergence to a locally optimal k-response decision query d'. The additional step of improving d to d' is needed because in general, the greedy construction procedure is not guaranteed to construct a locally optimal k-response decision query, and the response-entropy based EVOI-loss upper bound provided by Lemma 5.4 on which Theorem 5.5 relies is valid only when the decision query to be replaced is locally optimal. The complexity of each iteration of query improvement has complexity $\mathcal{O}(|U|k + Bk)$ (Viappiani and Boutilier, 2010), and while recursive query improvement is guaranteed to converge in a finite number of recursions as discussed in Section 4.3, little else is known about its computational properties[1]. An EVOI-loss bound for DEER when its first step is approximated in this way is provided by Theorem 5.7 below (I note that the greedily approximated version of DEER tested in Chapter 6 does *not* perform the step of recursive query improvement).

Theorem 5.7. *Suppose the first step of DEER is approximated by the procedure described above. Then for any askable k-response query set Q, if this approximate version of DEER selects q, then*

$$\max_{q^* \in Q} EVOI(q^*; \psi) - EVOI(q; \psi) \leq \left(1 - e^{-H(Q, D_k^*)}\right)\left(V_{\max} - V_{\min}\right) + \frac{(V_{\max} - V_{\min})}{e}.$$

Proof. First, as discussed in Section 4.5, the greedy construction procedure produces a k-response decision query d such that

$$EVOI(d; \psi) \geq \left(1 - (\frac{k-1}{k})^k\right) \max_{d^* \in D_k} EVOI(d^*; \psi).$$

Combining the fact that $\left(1 - (\frac{k-1}{k})^k\right) \geq (1 - \frac{1}{e})$ (Nemhauser et al., 1978) and the fact that recursively applying the query improvement procedure discussed in Section 4.3 produces query d' with EVOI at least as high as the EVOI of d, we have that

$$EVOI(d'; \psi) \geq (1 - \frac{1}{e}) \max_{d^* \in D_k} EVOI(d^*; \psi). \tag{5.3}$$

[1] Viappiani and Boutilier (2010) informally state that in practice it tends to converge quickly for various heuristic ways to choose the initial k-response decision query. For the case considered here, the initial query is greedily constructed and one might expect that recursive query improvement would tend to converge even faster compared to the cases tested by Viappiani and Boutilier (2010).

Then,

$$\max_{q^* \in Q} EVOI(q^*; \psi) \leq \max_{d^* \in D_k^*} EVOI(d^*; \psi) \quad \textbf{(Theorem 4.2)}$$

$$\implies (1 - \frac{1}{e}) EVOI(q^*; \psi) \leq (1 - \frac{1}{e}) \max_{d^* \in D_k^*} EVOI(d^*; \psi)$$

$$\implies (1 - \frac{1}{e}) EVOI(q^*; \psi) \leq EVOI(d'; \psi) \quad \textbf{(Equation 5.3)}$$

$$\implies EVOI(q^*; \psi) \leq EVOI(d'; \psi) + \frac{1}{e} EVOI(q^*; \psi)$$

$$\implies EVOI(q^*; \psi) \leq EVOI(d'; \psi) + \frac{(V_{\max} - V_{\min})}{e}$$

$$\implies EVOI(q^*; \psi) - EVOI(q; \psi) \leq EVOI(d'; \psi) - EVOI(q; \psi) + \frac{(V_{\max} - V_{\min})}{e}$$

$$\implies EVOI(q^*; \psi) - EVOI(q; \psi) \leq \left(1 - e^{-H(Q, D_k^*)}\right)\left(V_{\max} - V_{\min}\right) + \frac{(V_{\max} - V_{\min})}{e}.$$

□

Comparing Theorem 5.7 to Theorem 5.5, the maximum EVOI-loss is the same except an additional penalty of at most $\frac{(V_{\max} - V_{\min})}{e}$ (about 0.37% of $V_{\max} - V_{\min}$) is incurred by the approximate version of DEER compared to the exact version of DEER.

5.5.2 Directed Mistake Volume Minimization (DMVM)

Next I introduce DMVM: an approximation for DEER. Then, I derive a performance guarantee for DMVM and show how it can be specified for an example uncertainty representation. While I do further discuss DMVM in Chapter 6 where I include it in some of the experiments, my focus will be on DEER, so if desired the reader can skip this subsection without risk of missing the core content of the remainder of the dissertation.

DEER's second step, which requires performing an entropy computation for every possible query, can be approximated by minimizing a geometrical criterion that may be cheaper to compute than expected posterior response-entropy for some settings. In particular, *Directed Mistake Volume Minimization (DMVM)* approximates DEER's second step as follows:

$$
\begin{aligned}
&q = \arg\min_{q \in Q} M(d^*|q), \text{ where} \\
&M(d^*|q) = \int_{\Omega} \sum_{j=1}^{k} \Pr(d^* = j|\omega)\delta(J_{d^*}(\omega) \neq f(d^*|q = j))d\omega, \\
&J_{d^*}(\omega) = \arg\max_{j} \Pr(d^* = j|\omega), \text{ and} \\
&f(d^*|q = j) = \arg\max_{i} \Pr(d^*|q = j).
\end{aligned}
$$

Instead of selecting the query that minimizes the expected posterior entropy over responses to d^*, the second step of DMVM selects the query q that minimizes $M(d^*|q)$: the *Mistake Volume* associated with using q to match d^*.

To illustrate what mistake volume measures, consider the straightforward case where queries are binary (k=2) and have deterministic responses when conditioned on ω, and without loss of generality denote the possible responses to either query as "yes" and "no" . Observing the response "yes" to q, then, will eliminate all ω that are not in the subset $\Omega_{q,\text{“}yes\text{”}} \subseteq \Omega$ containing the ω where $\Pr(q = \text{“}yes\text{”}|\omega) = 1.0$, as the rest are inconsistent with the observed response. Then, $\Omega_{q,\text{“}yes\text{”}}$ is split into two subsets: one subset where the response of d^* would be "yes", and one where the response would be "no". The response corresponding to the subset with higher probability mass than the other, then, would be the best response prediction of d^* after observing response "yes" to q, which would correspond above to $f(d^*; q = \text{“}yes\text{”})$ above. Similarly, there is a best response prediction of d^* when the response for q is observed to be "no", which would correspond to $f(d^*; q = \text{“}no\text{”})$. Mistake volume, then, would correspond to the *volume* of the subset of Ω containing those ω where, given ω, the resulting best response prediction of d^* upon observing the response to q *does not match* what the response to d^* would be.

The computational efficiency of DMVM compared to DEER heavily depends on the nature of the askable query set Q, the functional form governing decision values V^u_ω, and the geometric structure of the model space Ω. In particular, DMVM needs to compute weighted volumes over Ω, which could be challenging or even infeasible to compute exactly in high-dimensional spaces (for example, computing the volume of a convex polytope defined by a set of linear inequalities is #P-hard (Dyer and Frieze, 1988)). While in general DMVM may not afford computational advantages over DEER, in Chapter 6 I will study the empirical performance of a particularly efficient algorithm whose query selection criterion is motivated by that of DMVM's, in a setting similar to the illustration described above.

5.5.2.1 DMVM EVOI-loss

Next I derive a bound similar to the one stated in Theorem 5.5 that is a function of DMVM's geometrical similarity criterion for queries instead of DEER's entropy-based similarity criterion. Applying Fano's inequality (Fano and Wintringham, 1961), $H(d^*|q)$ can be replaced with a quantity that is specific to a function or algorithm used to predict responses to queries in D_k^* given a response to a query in Q. Let $f(d^*; q = j)$ be the function that predicts the response to d^* that has highest probability conditioned on the response to q, i.e., $f(d^*|q = j) = \arg\max_i \Pr(d^* = i|q = j; \psi)$. Then,

Lemma 5.8. *For any $d^* \in D_k^*$ and $q \in Q_k$,*

$$H(d^*|q) \leq \mathbb{E}_{j\sim\psi;q}\big[\Pr(d^* \neq f(d^*; q = j)|q = j); \psi\big] \log(k) + 1.$$

Proof. Let $b(d^*; q)$ denote the Bernoulli distribution where a success corresponds to the event that $f(d^*; q = j)$ predicts the correct response to d^*, i.e., $p = \Pr(f(d^*; q = j) = d^*)$. By Fano's inequality,

$$H(d^*|q) \leq H\big(b(d^*; q)\big) + \mathbb{E}_{j\sim\psi;q}\big[\Pr(d^* \neq f(q, j)|q = j); \psi\big] \log(k - 1),$$

which in turn implies that

$$H(d^*|q) \quad \leq \quad \mathbb{E}_{j\sim\psi;q}\big[\Pr(d^* \neq f(d^*; q = j)|q = j); \psi\big] \log(k) + 1.$$

□

Next I further factor $H(d^*|q)$ so that it becomes a function of the mistake volume associated with q and d^*, which leads directly to an EVOI-loss upper bound for DMVM. To do so, I need to introduce some additional definitions:

- Let the *Response Function* $J_{d^*}(\omega)$ for $d^* \in D_k^*$ and $\omega \in \Omega$ be defined as

  $$J_{d^*}(\omega) = \arg\max_j \Pr(d^* = j|\omega).$$

  Note that $J_{d^*}(\omega)$ is deterministic since decision query responses are deterministic conditioned on ω. I will utilize this fact below.

- Let the *Mistake Volume* associated with using the response to q to predict the response to d^* be defined as

$$M(d^*, q) = \int_\Omega \sum_{j=1}^{k} \delta(J_{d^*}(\omega) \neq f(d^*; q = j))d\omega.$$

  Note that mistake volume is the criterion that DMVM minimizes in order to project d^* into the askable query set.

- Let $\mathcal{R}_z$ denote the set of all subsets of Ω such that for all $R \in \mathcal{R}_z$, $\int_R dr \leq z$.

- Finally, let $\overline{p}_\Omega(z; \psi) = \sup_{R \in \mathcal{R}_z} \Pr(\omega \in R; \psi)$.

Applying these definitions,

$$\begin{aligned}
& \mathbb{E}_{j \sim q;\psi}\big[\Pr(d^* \neq f(d^*; q = j) | q = j; \psi)\big] \\
= \; & \mathbb{E}_{\omega \sim \psi}\Big[\mathbb{E}_{j \sim q|\omega}\big[\Pr(d^* \neq f(d^*; q = j)|\omega)\big]\Big] \\
= \; & \int_\Omega \psi(\omega) \sum_{j=1}^{k} \Pr(q = j|\omega) \Pr(J_{d^*}(\omega) \neq f(d^*; q = j)|\omega) \\
= \; & \int_\Omega \psi(\omega) \sum_{j=1}^{k} \delta(J_{d^*}(\omega) \neq f(d^*; q = j))d\omega \\
\leq \; & \overline{p}_\Omega(M(d^*, q); \psi).
\end{aligned}$$

Combining this inequality with Lemma 5.8 yields

$$H(d^*|q) \leq \overline{p}_\Omega(M(d^*, q); \psi) \log(k) + 1, \tag{5.4}$$

which can be understood as an upper bound for $H(d^*|q)$ as a function $\overline{p}_\Omega$, which governs how concentrated the mass of ψ is in a geometric sense (which is independent of q and d^*), and as a function of the mistake volume associated with matching d^* with q, which DMVM minimizes to project d^* into the askable query set.

Analagous to how $H(Q, D_k^*)$ measures similarity between Q and D_k^* in terms of posterior-response entropy, define $M(Q, D_k^*)$ as

$$M(Q, D_k^*) = \min_{q \in Q} \max_{d^* \in D_k^*} M(d^*, q),$$

which measures similarity between Q and D_k^* in terms of mistake volume. Applying this definition yields Theorem 5.9 below:

Theorem 5.9. *(Upper bound on DMVM EVOI-loss.) For any askable k-response query set Q, if DMVM selects q, then*

$$\max_{q* \in Q} EVOI(q^*;\psi) - EVOI(q;\psi) \leq \Big(1 - e^{-\log(k)\overline{p}_\Omega\left(M(Q,D_k^*);\psi\right)-1}\Big)\Big(V_{\max} - V_{\min}\Big).$$

Proof.

$$\begin{aligned}
\max_{q* \in Q} EVOI(q^*;\psi) - EVOI(q;\psi) &\leq \max_{d^* \in D_k^*} EVOI(d^*;\psi) - EVOI(q;\psi) \quad \textbf{(Theorem 4.2)} \\
&\leq \max_{d^* \in D_k^*} \Big\{\Big(1 - e^{-H(d^*|q)}\Big)\Big(V_{\max} - V_{\min}\Big)\Big\} \quad \textbf{(Lemma 5.4)} \\
&\leq \max_{d^* \in D_k^*} \Big\{\Big(1 - e^{-\overline{p}_\Omega(M(d^*,q);\psi)\log(k)-1}\Big)\Big(V_{\max} - V_{\min}\Big)\Big\} \quad \textbf{(Eq. 5.4)} \\
&\leq \Big(1 - e^{-\overline{p}_\Omega\left(M(Q,D_k^*);\psi\right)\log(k)-1}\Big)\Big(V_{\max} - V_{\min}\Big).
\end{aligned}$$

The last inequality follows since DMVM selects q so as to minimize $M(q, d^*)$ and $\overline{p}_\Omega$ is monotonically nondecreasing. □

EVOI-loss guarantee for greedy approximation of DMVM. Recall that when DEER's first step is approximated by greedy construction followed by recursive query improvement until convergence, the EVOI-loss of the query selected can be no more than $\frac{(V_{\max}-V_{\min})}{e}$ more than the EVOI-loss of the query selected by the exact version of DEER (Theorem 5.7). When DMVM's first step is approximated in the same manner, the same increase in maximum EVOI-loss applies, and the proof is analogous to that of Theorem 5.7.

Special Case: Gaussian Uncertainty. Here I show an example for how Theorem 5.9 can be specified for a simple example uncertainty representation. In particular, let ψ take the form of a Gaussian distribution with mean μ and variance σ^2. Then,

$$\begin{aligned}
\overline{p}_\Omega\big(M(Q,D_k^*)\big) &= \Pr\Big(\omega \in [\mu - \frac{M(Q,D_k^*)}{2}, \mu + \frac{M(Q,D_k^*)}{2}]\Big) \\
&= \text{erf}\Big(\frac{M(Q,D_k^*)}{2\sigma\sqrt{2}}\Big), \quad \text{where} \qquad (5.5)
\end{aligned}$$

$$\text{erf}(x) \triangleq \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt.$$

The first equality above holds due to the combination of the facts that 1) the Gaussian pdf is

symmetric over its mean; and 2) the Gaussian pdf is monotonically decreasing in absolute distance from the mean. The second equality holds because for any n,

$$\Pr\Big(\omega \in [\mu - n\sigma, \mu + n\sigma]\Big) = \text{erf}\Big(\frac{n}{\sqrt{2}}\Big);$$

the n in this case is obtained by setting $\frac{M(Q,D_k^*)}{2} = n\sigma$ and solving for n. Plugging this result into Theorem 5.9 yields an EVOI-loss bound for DMVM that applies to the special case of single-dimensional Gaussian uncertainty:

Lemma 5.10. *Let the askable query set Q, and let ψ take the form of a Gaussian distribution with variance σ^2. Then if DMVM selects q,*

$$\max_{q^* \in Q} EVOI(q^*; \psi) - EVOI(q; \psi) \leq \Big(1 - e^{-\log(k) erf\Big(\frac{M(Q,D_k^*)}{2\sigma\sqrt{2}}\Big)+1}\Big)\Big(V_{\max} - V_{\min}\Big).$$

Note that since $\text{erf}(x)$ is monotonically increasing in x, Equation 5.5 implies that $p_\Omega(M(Q, D_k^*))$ is monotonically increasing in $M(Q, D_k^*)$ while monotonically decreasing in σ. This implies that for Gaussian distributions, the error bound stated in Lemma 5.10 prescribes lower worst-case EVOI-loss as 1) D_k^* and Q become more related in terms of maximum mistake volume; and 2) the variance of the agent's uncertainty increases. Intuitively, this can be understood as follows: the more specific the region of Ω the agent needs to learn about, the stronger the required connection between D_k^* and Q (in terms of maximum mistake volume) in order to achieve the same upper bound on EVOI loss.

5.6 Summary of Algorithms and Results

Below I list the algorithms studied in this chapter and the main results I developed for them.

- **Exhaustive (baseline)**
  - Computational complexity: $\mathcal{O}(|Q|kB + |Q|k\Pi^*)$
  - EVOI-loss: 0
- **MEDER (baseline)**
  - Computational complexity: $\mathcal{O}\big(|Q|kB + |Q|kE\big)$
  - EVOI-loss: $\leq \Big(1 - e^{-H(D^*_{|U|}, Q)}\Big)\Big(V_{\max} - V_{\min}\Big)$
- **DEER (novel algorithm)**

- Computational complexity: $\mathcal{O}(k(|U|^k B + |Q|E))$
- EVOI-loss: $\leq \left(1 - e^{-H(D_k^*, Q)}\right)\left(V_{\max} - V_{\min}\right)$

- **DEER with first step approximated by greedy construction (novel algorithm)**[2]
  - Computational complexity: $\mathcal{O}(k(|U|Bk + k|Q|E))$
  - EVOI-loss: $\leq \left(1 - e^{-H(D_k^*, Q)}\right)\left(V_{\max} - V_{\min}\right) + \frac{(V_{\max} - V_{\min})}{e}$

- **DMVM (novel algorithm)**
  - Computational complexity: setting-dependent
  - EVOI-loss: $\leq \left(1 - e^{-\log(k)\overline{p}_\Omega\left(M(Q, D_k^*); \psi\right)^{-1}}\right)\left(V_{\max} - V_{\min}\right)$

5.7 Discussion

In this chapter I focused on developing principles for designing query selection algorithms that apply to any setting where the agent can select a query from some specified set of k-response queries. To this end, I introduced the Wishful Query Projection (WQP) approach for query selection, which operates by first computing the ideal query to ask assuming it can have only k possible responses, and then efficiently finding a similar *askable* query. Intuitively, the main computational advantage of using WQP to select a query is that the query set can be expanded without requiring additional optimal planning computations, since all computations related to optimal planning are frontloaded into the first step. I then presented two specific implementations of WQP, DEER and an approximation of DEER called DMVM, and proved that both offer formal performance guarantees regarding the maximum loss in EVOI of the queries they select compared to the best one in the askable set, as a function of measures of how similar the askable query set is to the k-response decision query set. I also briefly discussed the impact that greedily approximating the first steps of DEER and DMVM has on their computational complexity, and also the impact that it has on their performance guarantees.

I examine the strengths and weaknesses of WQP (mainly focusing on DEER) further in Chapter 6, where I conduct an empirical study of its application (along with DMVM and MEDER) in a particular askable query selection setting. Before moving on to that more specific setting, however, it is useful to better understand how WQP approximates askable query selection in the abstract. Intuitively, WQP makes two approximations to

[2]Results apply only if recursive query improvement until convergence is performed after greedy construction in the first step.

approximate EVOI-optimal query selection. The first approximation is that the askable query set is assumed to contain a query similar to the *best* k-response decision query d^* computed in the first step. When this is not true, there is no guarantee that approximating d^* will lead to a good query. For example, it could be the case that no askable query similar to d^* exists, but that an askable query similar to another k-response decision query d exists, where although d and d^* are dissimilar, d has nearly as high of value as d^*. The second approximation is that a query with higher similarity to d^* is assumed to lead to higher EVOI, which may not be the case. For example, when using DEER, it may be that a specific set of models needs to be eliminated in order for the decision corresponding to one of the responses to d^* to ever be valuable in expectation due to differences in relative value magnitudes among models, where at the same time the posterior response-entropy of d^* is minimized by asking a query that eliminates a different set of models where the response to d^* is equally uncertain even though the differences in value between decisions is unimpactful, resulting in the corresponding query having low EVOI. In the next chapter I will empirically study how these two approximations can affect DEER's performance.

CHAPTER 6

Empirical Study of Wishful Query Projection in Askable k-Response Decision Query Selection

In Chapter 4, I showed that when the agent has the ability to ask any query from the set of all k-response queries, the agent can consider only k-response decision queries at no EVOI penalty (Theorem 4.2), which reduces the general k-response query selection problem to a tractable one where efficient, provably approximate algorithms apply. Then, in Chapter 5, I restricted the agent by assuming that the agent can ask only queries lying within an arbitrary given subset of all k-response queries, where the agent cannot choose the subset. There, I showed that the EVOI-Sufficiency and tractability of the k-response decision query set can be leveraged to select from arbitrary k-response query sets in the form of two Wishful Query Projection (WQP) algorithms – DEER and DMVM. While I showed that DEER and DMVM have appealing computational properties while offering approximation guarantees, I have yet to provide empirical analysis of when using them would be suitable compared to alternative approaches.

In this chapter, I study the efficacy of WQP algorithms when applied to an extension of the standard decision query selection setting where the algorithms of Viappiani and Boutilier (2010) do not directly apply. Namely, I compare WQP algorithms to alternative approaches, such as an algorithm that attempts to make use of structure specific to this setting. To do so, I make liberal use of the algorithms, results, and terminology pertaining to the work of Viappiani and Boutilier (2010) presented in Section 4.5 of Chapter 4.

6.1 Askable Decision Query Setting

Consider an extension of the k-response decision query setting where the decisions that can be used to compose queries can be different than the decisions the agent can execute. Such a case could arise when some of the agent's possible decisions would be too complex to

communicate (such as policies in sequential decision making settings) and/or would require undue cognitive burden on the part of the user to choose from.

For example, consider an agent whose task is to recommend a housing option to its human user who is moving from Chicago to Boston. The relative suitability of options in Boston may depend on factors whose range of possibilities the user is familiar with in Chicago, but unfamiliar with in Boston, such as proximity to public transportation lines or neighborhood crime rates, and so asking the user to choose among options in Boston may be infeasible. However, the agent could indirectly learn about the user's preferences over options in Boston by asking about options in Chicago, whose values for the relevant factors the user better understands.

Formally, consider a single-shot decision-making under uncertainty setting with decision set U and uncertainty ψ over model space of dimension d $\Omega \subseteq \mathbb{R}^d$, where each $\omega \in \Omega$ prescribes value to each $u \in U$ such that

$$V_\omega^u \triangleq \omega^T \phi(u), \tag{6.1}$$

where $\phi : U \mapsto \mathbb{R}^c$ is a mapping from decisions to vectors of c features. I assume that the agent can execute only decisions contained in a subset $D \subseteq U$, and that the agent can ask only k-response decision queries composed of decisions contained in a subset $A \subseteq U$. I refer to D as the *doable decision set* and the set of k-response queries composed of decisions in D as the *doable decision query set* Q_D. Similarly, I refer to A as the *askable decision set* A, and the set of k-response queries composed of decisions in A as the *askable decision query set* Q_A.

This chapter is concerned with the *askable decision query selection* problem, formally defined as

$$q_A^* = \arg\max_{q \in Q_A} EVOI(q; \psi).$$

Note that the EVOI for an arbitrary q can be written as

$$\begin{aligned} EVOI(q;\psi) + V_\psi^* &= \mathbb{E}_{j \sim q;\psi}\left[V_{\psi|q=j}^*\right] && (6.2)\\ &= \mathbb{E}_{j \sim q;\psi}\left[\max_{u \in D} V_{\psi|q=j}^u\right]. && (6.3) \end{aligned}$$

Thus, the EVOI of a decision query, whether it contains decisions from D, A, or both, is a function of how much the query's response will improve the agent's ability to select a decision from D, *not* A. In some cases, the same algorithms devised for the standard decision query setting can be applied here at no penalty. Namely, if $D \subseteq A$, the following

lemma implies that this problem can be reduced to selecting a query from Q_D:

Lemma 6.1. *Let $D \subseteq A$. For any ψ, if $q^* = \arg\max_{q \in Q_D} EVOI(q; \psi)$, then $q^* = \arg\max_{q \in Q_A} EVOI(q; \psi)$.*

Proof. Note that q^* is the Q_D-EVOI-optimal query, and so by Theorem 4.2 no k-response query can have higher EVOI (in particular, no query in Q_A can have higher EVOI than q^*). Since $D \subseteq A$, $q^* \in Q_A$, implying the statement of the lemma. □

In other words, when the set of decisions contained by A subsumes that of D, the additional decisions the agent can ask about beyond D are superfluous, and so the setting can be reduced to the standard decision query selection setting discussed in Section 4.5 of Chapter 4. However, in the general case where it may be that $D \not\subseteq A$, the fact that queries must be constructed from a different decision set than the decision set the agent needs to learn about comes into play, since the query produced by the aforementioned reduction cannot always be asked. Throughout the chapter, I use the following example to discuss the merits of various askable decision query selection algorithms:

Example 6.1. *Suppose $k = 2$, and $U = D = \{u_1, u_2, u_3, u_4, u_5, u_6\}$ is comprised of six possible housing options where the agent's task is to recommend the user her most preferred option. Further, suppose the housing options have the following characteristics:*

- *u_1 and u_2 are both close to a subway stop, are too expensive to be feasible, but u_1 is in a gated community while u_2 is not.*
- *u_3 and u_4 are close and far (respectively) from a subway stop, are in the user's price range, and neither are in a gated community.*
- *u_5 and u_6 are both close to a subway stop, are in the user's price range, but u_5 is in a gated community while u_6 is not.*

Assume the utility associated with a housing option is a function only of distance from a subway stop (close or far), whether or not it is in the user's price range, and whether or not it is located in a gated community. Further, assume the extent of the agent's knowledge of the user's weights on these factors is that the user strongly prefers that the option is in her price range, and that the user has a strong unknown preference in whether or not the option is in a gated community but only a minor preference in distance from a subway stop. That

is, the crucial information missing is whether or not the user would prefer to live in a gated community.

Finally, suppose the agent can ask the user a query only about housing options that are currently available to be shown, which only includes u_1, u_2, u_3, and u_4 (i.e., $A = \{u_1, u_2, u_3, u_4\} \subseteq \{u_1, u_2, u_3, u_4, u_5, u_6\}$).

In this example, which would be the best query to ask? Even though u_1 and u_2 are too expensive to be feasible choices for recommendation, asking the user which she prefers would help reveal her preferences with respect to whether or not she would prefer to live in a gated community. The answer to this query would then allow the agent to choose the better of the two best candidates, u_5 and u_6, and so the query asking about u_1 and u_2 would be Q_A-EVOI-optimal.

Of course, in general the Q_A-EVOI-optimal query can be computed by *exhaustively* evaluating the EVOI of each query in Q_A; I refer to this algorithm as *Askable Enumeration Doable Evaluation (AskEnum-DoEval)*. However, exhaustively evaluating the EVOI of every query in Q_A as AskEnum-DoEval does would become computationally intractable as A and/or D grow in size – namely, AskEnum-DoEval requires $\binom{|A|}{k}$ EVOI computations, each of which have complexity $|D|kB$. (Recall that B represents the complexity of performing a single Bayes-update; for example, B could depend on the number of particles used in a particle filtering scheme.) Thus, AskEnum-DoEval has complexity $\mathcal{O}(kA^k|D|B)$. Next I discuss algorithms for approximating askable decision query selection.

6.1.1 Wishful Query Projection Algorithms

The Wishful Query Projection (WQP) strategy developed in Chapter 5 and adopted by DEER and DMVM can be used to approximate query selection in the askable decision query selection setting. Recall that the general idea is to begin by computing the optimal k-response decision query d^*, and then project d^* into the askable query set Q according to some similarity measure – DEER uses response-entropy-based projection, and DMVM uses mistake-region based projection. Also, recall that the response-entropy of a query q under uncertainty ψ, as defined in Chapter 5 and used frequently throughout this chapter, is defined as

$$H(q;\psi) \triangleq \sum_{j=1}^{k} \Pr(q=j;\psi) \log\big(\Pr(q=j;\psi)\big).$$

Note that DEER and DMVM both solve Example 6.1 in that they ask the EVOI-optimal askable query: the first step selects the query d^* asking about u_5 and u_6, and in their second step both algorithms select the query asking about u_1 and u_2 since it both reduces the response entropy of d^* to zero and has the same mistake region as d^*.

In terms of computation, the first step shared by DEER and DMVM has complexity $\mathcal{O}(k|D|^k B)$ when solved exactly, or complexity $\mathcal{O}(k^2|D|B)$ when approximated greedily[1]. DEER's second step has complexity $\mathcal{O}(k|A|^k B)$, since it exhaustively searches the askable decision query set for the query inducing the least expected posterior entropy over the responses to d^*. Thus, DEER and *DEER-Greedy* have complexity $\mathcal{O}(k|D|^k B + k|A|^k B)$ and $\mathcal{O}(k^2|D| + |A|^k)$ in the askable decision query setting, respectively.

DMVM, on the other hand, can be efficiently implemented in the askable decision query selection setting for the case where queries are binary (k=2) and where decision-values have a linear functional form (assumed in Equation 6.1). Namely, the mistake region volume associated with replacing some binary-response doable decision query with a binary-response askable decision query can be replaced with the angle formed between the two corresponding response boundary hyperplanes without changing the ordering over queries[2]. Hence, under these assumptions the projection step of DMVM can be implemented by selecting a query $\hat{q}$ from the askable decision query set so as to minimize the angle between $\hat{q}$'s response boundary hyperplane and d^*'s response boundary hyperplane:

$$\hat{q} = \arg\max_{q \in Q_A} \left\{ \cos^{-1} \left[\frac{\left(\phi(a_1^{\hat{q}}) - \phi(a_2^{\hat{q}})\right) \cdot \left(\phi(u_1^{d^*}) - \phi(u_2^{d^*})\right)}{||\phi(a_1^{\hat{q}}) - \phi(a_2^{\hat{q}})|| \cdot ||\phi(u_1^{d^*}) - \phi(u_2^{d^*})||} \right] \right\}.$$

Note that each angle in the maximization above can be computed with complexity $\mathcal{O}(c)$ (recall that c is the dimensionality of the decision-feature space), so this implementation of DMVM has complexity $\mathcal{O}(c|A|^k)$. I use this implementation of DMVM in the experiments discussed in Section 6.2; hence, DMVM and *DMVM-Greedy* can be implemented with complexity $\mathcal{O}(|D|^2 B + c|A|^2)$ and $\mathcal{O}(|D|B + c|A|^2)$, respectively, when $k = 2$ and decision-values are linear.

I study the empirical performance DEER and DMVM (and their greedy versions) in the subsequent empirical analysis, where I compare the performance of various askable query selection algorithms in terms of the EVOI of the queries they select and the computation

[1] The greedy versions of DEER and DMVM studied in this chapter do not perform the recursive query improvement step discussed in Section 5.5.1.2, so none of the EVOI-loss bounds derived in Chapter 5 apply to the *greedy* versions of DEER and DMVM in this chapter

[2] Assume that Ω is a d-dimensional hypersphere centered at the origin, so that Ω is trivially bounded and symmetric among its dimensions. In other cases, the described algorithm may only be an approximation of DMVM.

they consume in various scenarios. First, however, I discuss another way to approximate askable decision query selection.

6.1.2 Askable Evaluation

One way to approximate AskEnum-DoEval would be to treat A as the agent's decision set in addition to the set from which queries can be constructed – this would replace each EVOI computation (each of which involves solving optimal planning problems within D) with an approximated EVOI computation (each of which involves solving optimal planning problems only within A). More formally, consider the following approximation to the askable decision query selection problem:

$$\begin{aligned}\arg\max_{q \in Q_A} EVOI(q;\psi) &= \arg\max_{q \in Q_A} \left\{ \mathbb{E}_{j \sim q;\psi} \left[\max_{u \in D} V^u_{\psi|q=j} \right] \right\} \\ &\approx \arg\max_{q \in Q_A} \left\{ \mathbb{E}_{j \sim q;\psi} \left[\max_{a \in A} V^a_{\psi|q=j} \right] \right\}.\end{aligned}$$

This approximation completely ignores the doable decision set, and is equivalent to treating the problem as the standard decision query selection setting with $U = A$.

Solving this approximated version of the problem can be computationally easier than the original in two ways: first, exhaustively solving it has complexity scaling only with the size of A – i.e., $\mathcal{O}(k|A|^k B)$, so it does not depend on the complexity of the agent's decision problem. However, A may still be large enough that enumerating Q_A would be intractable.

The second computational advantage of this approximation is that exhaustive search of Q_A can itself be approximated by greedy construction of the decision set in exactly the same way as in the standard decision query setting, in that doing so offers the same approximation guarantee as in the standard decision query setting (see Section 4.5 of Chapter 4), with the important caveat that the guarantee only applies to the approximated EVOI measure. The complexity of the greedy version, then, is $\mathcal{O}(k^2|A|B)$.

I refer to the algorithm that exhaustively searches Q_A but approximates EVOI this way as *Askable Enumeration Askable Evaluation (AskEnum-AskEval)*, and I refer to its greedy approximation as *Askable Greedy Construction Askable Evaluation (AskGreedy-AskEval)*. In the subsequent analysis and empirical comparisons I consider only the exact version of AskEnum-AskEval.

How much EVOI can be lost by ignoring the doable decision set when selecting a query, as AskEnum-AskEval does? Consider Example 6.1: using AskEnum-AskEval to

select a query would result in choosing to ask about u_3 and u_4, since this query would at least allow the agent to choose the one with the more preferred subway stop proximity, as opposed to asking about u_1 and u_2, which wouldn't help the agent decide between any of the decisions in A. In fact, the query selected by AskEnum-AskEval would have zero EVOI if u_5 and u_6 were better choices than u_3 or u_4 even if the agent were to have known the user's preferences regarding subway stop proximity, such as a scenario where the user's prior gated community preference is biased towards yes.

Here, the reason AskEnum-AskEval is misled is that the most valuable features to learn about for selecting from A are not useful for selecting from D, and vice-versa. In general, the more dissimilar the values of the decisions contained by A and D become, the more potential there is for them to be uninformative about each other. Next I upper bound the extent to which dissimilar values between the askable and doable decision sets can hurt AskEnum-AskEval's EVOI performance.

6.1.3 EVOI-loss upper bound for AskEnum-AskEval

Here I show that the EVOI lost by AskEnum-AskEval can be upper bounded as a function of the extent to which A and D are *dissimilar* in terms of decision values – i.e., the stronger the similarity between A and D, the smaller the maximum EVOI loss in using AskEnum-AskEval. In particular, consider the following measure of relatedness, which considers the maximum value that could be lost (or gained) when some decision $u \in U$ is replaced by the decision $a \in A$ such that a minimizes the worst-case value difference between u and a:

$$\Delta(D, A) \triangleq \max_{u \in D} \min_{a \in A} \max_{\omega \in \Omega : \Pr(\omega) > 0} |V^u_\omega - V^a_\omega|.$$

The following theorem states that the query produced by AskEnum-AskEval can have EVOI no worse than $2\Delta(D, A)$ less than the EVOI of query produced by AskEnum-DoEval (which, recall, is Q_A-EVOI-optimal):

Theorem 6.2. *Let $\hat{q}$ be the query computed by AskEnum-AskEval, let q^* be the query computed by AskEnum-DoEval. Then,*

$$EVOI(q^*; \psi) - EVOI(\hat{q}; \psi) \leq 2\Delta(D, A).$$

Proof.

$$
\begin{aligned}
& EVOI(q^*;\psi) - EVOI(\hat{q};\psi) = \sum_{j=1}^{k} \Pr(q^* = j) V^*_{\psi|q^*=j} - \sum_{j=1}^{k} \Pr(\hat{q} = j) V^*_{\psi|\hat{q}=j} \\
\leq\ & \sum_{j=1}^{k} \Pr(q^* = j) V^*_{\psi|q^*=j} - \sum_{j=1}^{k} \Pr(\hat{q} = j) \max_{a_j \in A} V^{a_j}_{\psi|\hat{q}=j} + \Delta(D, A) \quad \textbf{(See below)} \\
\leq\ & \sum_{j=1}^{k} \Pr(q^* = j) V^*_{\psi|q^*=j} - \sum_{j=1}^{k} \Pr(q^* = j) \max_{a_j \in A} V^{a_j}_{\psi|q^*=j} + \Delta(D, A) \quad \textbf{(See below)} \\
=\ & \sum_{j=1}^{k} \Pr(q^* = j) \big(\max_{u_j \in D} V^{u_j}_{\psi|q^*=j} - \max_{a_j \in A} V^{a_j}_{\psi|q^*=j} \big) + \Delta(D, A) \\
\leq\ & \sum_{j=1}^{k} \Pr(q^* = j) \Delta(D, A) + \Delta(D, A) \\
=\ & 2\Delta(D, A).
\end{aligned}
$$

The first inequality can be understood as follows. Let $u^*_j \triangleq \arg\max_{u_j \in D} V^{u_j}_{\psi|\hat{q}=j}$, and similarly let $a^*_j \triangleq \arg\max_{a_j \in A} V^{a_j}_{\psi|\hat{q}=j}$. Also, note that $V^*_{\psi|\hat{q}=j} = V^{u^*_j}_{\psi|\hat{q}=j}$. The inequality holds because for each j, $V^{a^*_j}_{\psi|\hat{q}=j}$ can be higher than $V^{u^*_j}_{\psi|\hat{q}=j}$ by no more than $\Delta(D, A)$ since for any $a \in A$ there exists some $u \in D$ such that for all $\omega \in \Omega$, $|V^a_\omega - V^u_\omega| \leq \Delta(D, A)$.

The second inequality holds because AskEnum-AskEval chooses $\hat{q}$ so as to maximize

$$\sum_{j=1}^{k} \Pr(\hat{q} = j) \max_{a_j \in A} V^{a_j}_{\psi|\hat{q}=j},$$

which is equivalent to EVOI-optimal decision query selection for the decision problem with decision set A, and so by Theorem 4.2 replacing $\hat{q}$ with any other k-response query (namely, q^*) can only lower the above quantity.

□

I have shown that AskEnum-AskEval is guaranteed to select a query with at most $2\Delta(D, A)$ EVOI loss. Indeed, note that in Example 6.1 above, $\Delta(D, A)$ is at least as large as the extent to which u_5 and u_6 dominate all decisions in A. More generally, unless selecting a decision from A is nearly the same decision problem as selecting a decision from D for all ω (for example, when $\Delta(D, A)$ is small), it is possible that valuable askable decision queries with respect to selecting from A could have very different (even orthogonal) effects on the agent's uncertainty as compared to valuable askable decision queries

with respect to selecting from D.

6.2 Experiments

Next I turn to an empirical evaluation of the askable decision query selection algorithms discussed thus far. Recall that, in the context of this dissertation, the primary purpose of the askable decision query setting is as an experimental testing ground to study the effectiveness of the Wishful Query Projection (WQP) approach developed in Chapter 5. As such, the experiments are specifically designed to expose key factors in query selection problems that practitioners should consider when determining the viability of using WQP for their query selection problem. To ground the analysis, I use DEER as the representative for WQP, since DMVM and DEER-Greedy are derived as approximations for DEER, and I leave a study focused on the unique strengths and weaknesses of DEER-Greedy, DMVM, and DMVM-Greedy for future work (however, I still include these approximate algorithms in the results presented after the main experiment descriptions to provide evidence that they can be effective approximations in some situations). The experiments provide evidence for the following two claims:

- (Inconsistent projection quality can hurt DEER.) The query selected by DEER can become worse relative to that selected by AskEnum-DoEval, MEDER, and baselines (AskEnum-AskEval, Random) as the closeness of the best match in the askable query set to a doable decision query d more strongly depends on the particular d.
- (Weak askable query sets can hurt DEER.) The query selected by DEER can become worse relative to that selected by AskEnum-DoEval, MEDER, and baselines (AskEnum-AskEval, Random) as the extent to which the EVOI of d^* overestimates the EVOI of q^* increases.

The first and second claims above relate to the first and second key approximations made by DEER, respectively, which were discussed in Section 5.7: DEER's first step chooses d^* without considering the extent to which the askable query set contains a good match compared to other decision queries, and DEER's second step chooses a query to match d^* solely on the basis of response-entropy reduction without considering its effect on the agent's policy or value. (Note that these are not assured to comprise an exhaustive characterization of DEER's weaknesses, however.) After I present the main experiment descriptions, I provide some basic results showing that DEER's approximations DMVM, DEER-Greedy, and DMVM-Greedy can be effective in some scenarios while requiring significantly less computation.

6.2.1 Setup

Next I describe the empirical problem space considered in the experiments and the methodology I use to compare the algorithms. In the following experiments, the agent's task is to recommend a housing option in Boston to a user looking for housing. I use the Boston Housing dataset to obtain the list of available houses, which contains a list of 506 vectors of 14 real-numbered statistics concerning a suburb (here, for the sake of illustration I am calling each suburb a housing option). Before the agent recommends a housing option to its user, it can query its user to ask which of two housing options of the agent's choosing the user would prefer – i.e., the agent can ask its user a binary-response decision query. However, the set of housing options the agent can ask its user about may not match the housing options it can recommend to its user – i.e., the agent faces an askable decision query selection problem.

The prior distribution over decision-value feature weights is defined by independent gaussians for each feature with $\sigma = 2.0$ and $\mu = 1.0$, and the prior distribution is approximated throughout the experiments as a uniform distribution over 300 particles that are independently drawn from the prior at each trial (but shared between each algorithm on each trial so as to reduce variance). Note that particle resampling is not necessary here since the agent can ask only one query, i.e., the particle weights do not need to undergo multiple updates, so particle depletion is not a concern. All results are averaged over 300 trials, and all errorbars shown in the figures are confidence intervals with p-value of 0.05.

6.2.2 Empirical study of DEER for Askable Decision Query Selection

I begin with a detailed comparison of the performance of DEER compared to MEDER, AskEnum-AskEval, Random, and AskEnum-DoEval purely in terms of the EVOI of the queries selected; in Section 6.2.3 I show some basic results regarding the performance (in terms of EVOI) of approximating DEER with DMVM, DEER-Greedy, or DMVM-Greedy, and in Section 6.2.4 I compare the computational requirements of all of the aforementioned algorithms.

Throughout the following experiments I consider various ways to construct the askable and doable decision sets (A and D) from the full list contained in the dataset. Using the decision sets as the variable between experiments in this way allows control of several key factors that play a role in the relative performance of the algorithms, including the following two factors which are the subject of the two claims stated above:

F1: (Consistency of projection quality). The extent to which the closeness of the best

match (in terms of expected posterior response-entropy) in the askable query set to a doable decision query d more strongly depends on the particular d; and

F2: (Richness of askable decision query set relative to doable decision query set.) The extent to which the EVOI of q^* is close to the EVOI of d^*.

6.2.2.1 Experiment 6.1: Uniformly Random Decision Sets

In the first experiment, the askable and doable decisions are sampled independently and uniformly from the full decision set on each trial. Hence, the average similarity between the two sets is purely a function of diversity in the decision set and any inherent correlations in decision features. As such, this experiment is exploratory in nature, and compares the algorithms without introducing any additional biases that might benefit one algorithm over another (e.g., without explicitly designing the askable query set to be impoverished or rich compared to the doable query set) in order to formulate initial hypotheses, which I then use as the basis for designing subsequent experiments. This will also serve to show that DEER can perform well in settings that are not specifically designed to have particularly aligned askable and doable decision sets, and that DEER can perform better than the worst-case bound derived in Chapter 5 (Theorem 5.5) would predict.

Figure 6.1a shows the average EVOI of the query selected by each method when the size of D is held fixed at 15, as a function of the size of A. As one would expect, the average EVOI of the query selected by each method tends to increase with the size of A (except for Random, which uses no form of discrimination to select a query, and *DoEnum-DoEval*, which considers doable decision queries instead of askable decision queries). Note that DEER achieves the best performance aside from AskEnum-DoEval, which asks the best askable query q^*.

What makes this a particularly favorable scenario for DEER? One potential explanation would be that the askable query set nearly always contains a nearly identical query to d^*. However, if that were the case both DEER and AskEnum-DoEval would achieve nearly the EVOI that DoEnumDoEval does, which asks d^* directly – but the EVOI of d^* is 0.9 on average (not shown in Figure 6.1a for visual clarity), which is significantly higher. Furthermore, consider Figure 6.1b, which shows the average posterior entropy over the responses to d^* after each algorithm asks their query. While the posterior entropy over the response probabilities for d^* can be lowered significantly, the match is far from perfect on average. This suggests that in this scenario, imperfect projection of d^* into the askable decision query set is forgiving in the sense that higher similarity in information between

an askable decision query q and d^* reliably leads to higher EVOI gains, as opposed to the other extreme where all but near-exact matches could have negligible EVOI. That is, the empirical EVOI-loss in replacing d^* with the query selected by DEER is much lower in this scenario than that prescribed by the upper bound on EVOI-loss stated in Theorem 5.5.

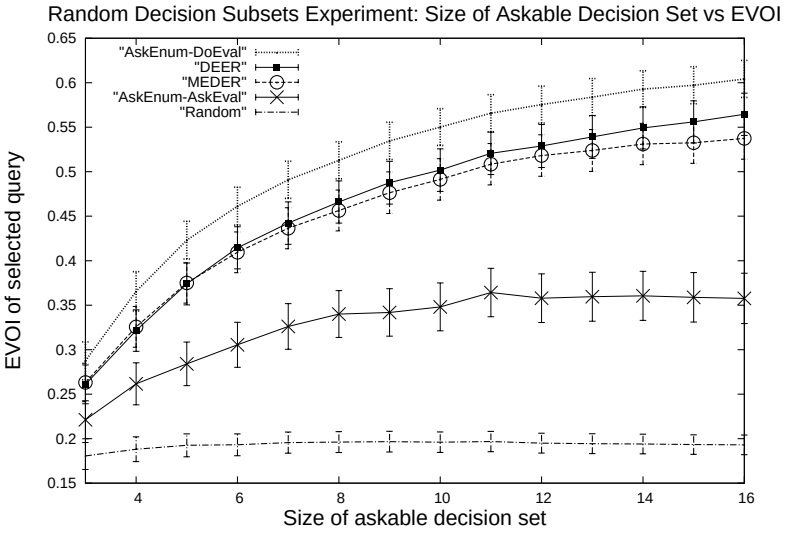
In addition, even though AskEnum-AskEval asked queries with higher EVOI than random on average, it acheived significantly lower EVOI than AskEnum-DoEval. It seems, then, that in this scenario queries must be evaluated on the basis of how they improve the agent's selection of decisions from D as opposed to A, lest significant potential EVOI be lost. Note that according to Theorem 6.2, these results imply that $\Delta(A, D)$ is high for this scenario, which is to be expected since A and D are independently sampled as small subsets from the full decision set.

Finally, notice that MEDER performs nearly as well as DEER at first but then begins to drop off as the askable decision set grows.[3] Why is this? Comparing Figure 6.1a and Figure 6.1b, observe that MEDER's drop off in EVOI as the askable decision set grows larger is accompanied by a growing gap between MEDER's d^* response entropy reduction and that of DEER's. Hence, as the askable query set contains queries more similar to d^*, the EVOI of the query selected by DEER approaches the highest possible EVOI for a binary-response query 4.2, and so directing queries at reducing the response entropy of d^* begins to yield noticeably higher EVOI than the less informed strategy of reducing the entropy over all doable decisions. The next experiment tests this explanation by observing the effect of forcing the askable query set to contain queries that are similar to d^* more often.

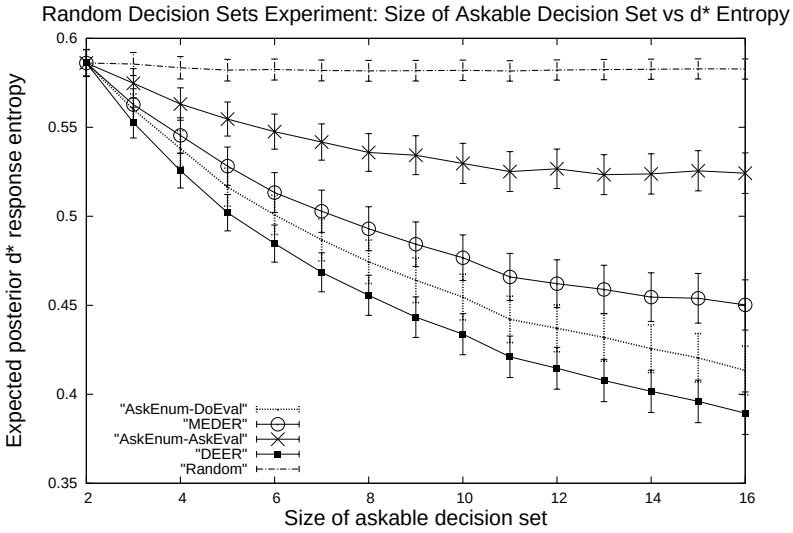
6.2.2.2 Experiment 6.2: Askable Decisions as Doable Decision Subsets

The second experiment considers the same setting as the first experiment, except on each trial the askable decision set is sampled from the doable decision set used for that trial so that $A \subseteq D$. Here the performance of AskEnum-AskEval should improve relative to Experiment 6.1, since askable decision queries will directly improve the agent's ability to select from at least a subset of D. In addition, the askable decision query set will contain a better match to d^* on average, since one or more of the exact decisions composing d^* will be present in the askable set more often than in Experiment 6.1. The explanation for

[3] While Figure 6.1a lacks the statistical significance needed to support this claim, Figure 6.12a (located in the appendix of this chapter) plots EVOI-loss instead of EVOI, and *does* verify the claim with statistical significance. For the experiments in this chapter, EVOI-loss measurements tend to have less variance than EVOI measurements, since the *magnitudes* of the former vary less than the magnitudes of the latter among trials, but I default to reporting EVOI results instead of EVOI-loss results, as EVOI tends to offer better visual clarity for the experiments presented in this chapter.



a)



b)

Figure 6.1: Results for Experiment 6.1– a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of the size of the askable decision set; the size of the doable decision set is fixed at 15. For each trial, both the askable and doable decisions sets are uniformly sampled from the complete Boston housing decision set.

the gap that forms between DEER and MEDER in Figure 6.1a given above would imply, then, that in this scenario the gap should widen since DEER can more effectively project d^*, which brings the projected query's EVOI closer to the highest possible EVOI for q^* (Theorem 5.5), unlike MEDER which asks the query that maximally reduces decision-entropy as a whole with no consideration as to how it might improve the agent's value or decision.

The results are shown in Figure 6.2a and b, which as before show EVOI of query selected and expected posterior d^* response-entropy as a function of the size of the askable decision set. Indeed, the EVOI gap between DEER and MEDER is widened in Figure 6.2a as compared to Figure 6.1a,[4] and Figure 6.2b shows an increase in projection quality compared to that shown in Figure 6.1b, which together corroborate the explanation above for MEDER's degraded performance relative to DEER as a function of the askable decision set size. Note that DEER's advantage in this scenario over MEDER's pure decision entropy reduction approach is evidence that the idea at the heart of Chapter 5 – approximating EVOI maximization by querying to reduce the agent's uncertainty about factors it wishes it could know – can at least sometimes display a meaningful advantage over global measures of uncertainty reduction.

Figure 6.2a also shows that AskEnum-AskEval's performance dramatically improves to approximately match DEER's performance, as compared to the previous experiment. This means that this scenario is a particularly easy askable decision query selection scenario in that treating it like the standard decision query selection setting where $A = D$ does not incur much loss, which benefits AskEnum-AskEval and, indirectly, DEER as well[5].

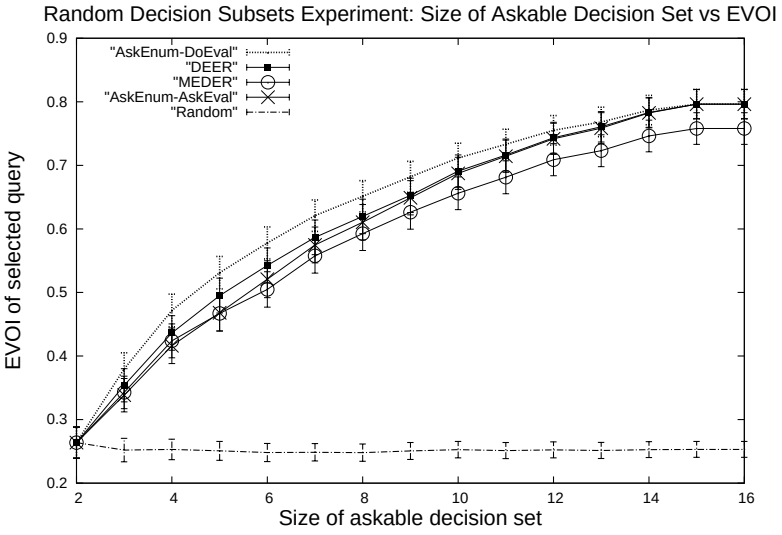
In the next two experiments I study DEER's performance relative to the other algorithms when the scenario is gamed to be hard for DEER by tuning factors F1 and F2.

6.2.2.3 Experiment 6.3: Doable Decisions with High Risk and Varied Askability

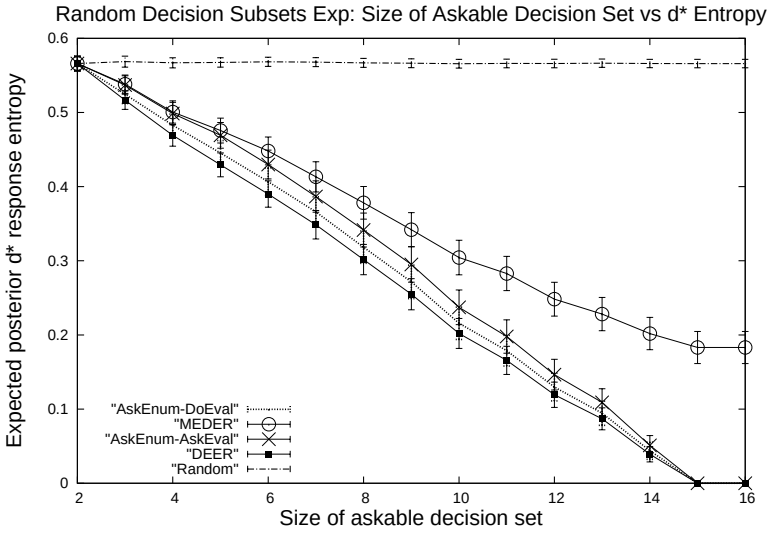
Under what conditions can DEER be misled into selecting a poor query relative to q^*, the query selected by AskEnum-DoEval? Recall that DEER's first main weakness is that it does not consider what queries can actually be asked when it constructs d^* – which leaves open the possibility that DEER may select a poor query when the askable query set does not contain any query giving much information about d^*. However, this alone does not

[4]Measuring EVOI-loss instead of EVOI (shown by Figure 6.12a for Experiment 6.1 and Figure 6.12b for Experiment 6.2) allows this claim to be made with statistical significance; see previous footnote for additional details.

[5]In fact, Theorem 6.4 proven in the appendix of this chapter shows that an algorithm related to DEER (a third WQP algorithm) has EVOI-loss bounded by $\Delta(A, D)$, relating its theoretically favorable conditions to those of AskEnum-AskEval.



a)



b)

Figure 6.2: Results for Experiment 6.2 – a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of the size of the askable decision set; the size of the doable decision set is fixed at 15. For each trial the doable decision set is uniformly sampled from the complete Boston housing decision set, and the askable decision set is then sampled from the doable decision set.

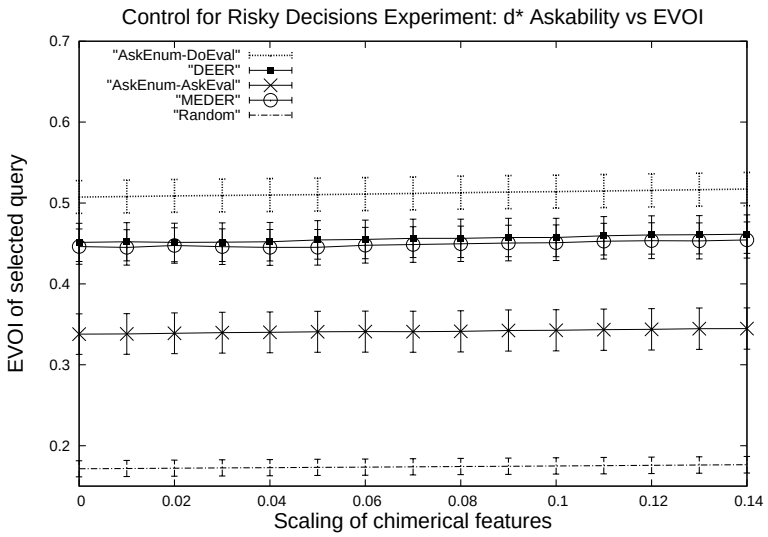
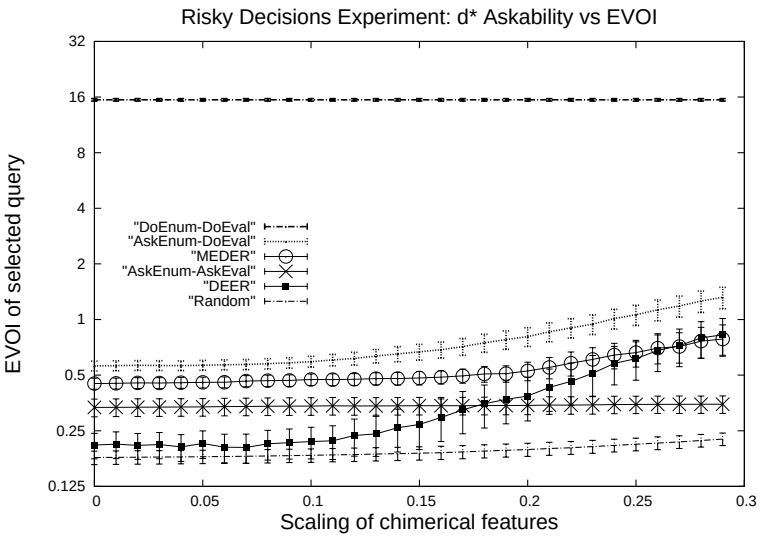


Figure 6.3: Control for Experiment 6.3 – Average EVOI of query selected as a function of ρ, the scaling factor used for the chimerical features (the last two features) of each decision in the askable deicsion set.

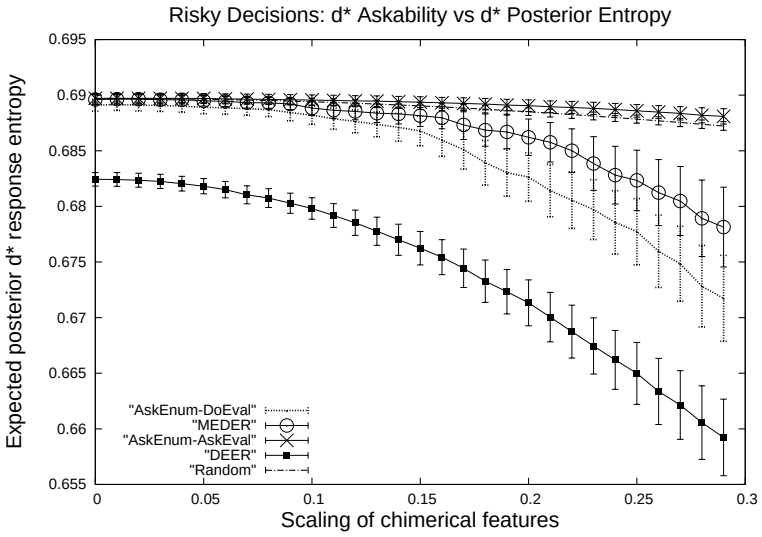
imply that DEER will select a poor query relative to q^* since it could be that no query is particularly informative about selecting from the doable decision set. Hence, for DEER to select a poor query when a good one exists, there must be a good query that is less informative about d^* than the one DEER selects. One way such a situation is possible is when F1– consistency of projection quality– is low, such as when the second-best doable decision query can be closely matched, but d^* cannot.

I construct a scenario with low F1 by augmenting the scenario studied in Experiment 6.1 as follows. First, each time an askable decision set is sampled, the last two features of each decision in the set are multiplied by a scaling factor ρ, where $0 \leq \rho \leq 1$. The smaller ρ is, then, the less can be learned about the weights for these last two features, which I refer to as the *chimerical features* and the *chimerical weights*. Since which in a pair of doable decisions has higher value could conceivably depend heavily on the chimerical features for some doable decision pairs, this change alone does lower F1.

Figure 6.3, which shows the EVOI of the query selected by each algorithm as a function of ρ when the size of the askable and doable decision sets are both fixed to 15, shows that the impact of this change alone is minimal. Intuitively, this is because the relative value of most pairs of doable decisions depends on much more than just the chimerical features, so rarely will this change alone impact the extent to which the askable decision query set



a)



b)

Figure 6.4: Experiment 6.3 – a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of ρ: the scaling factor used for the chimerical features (the last two features) of each decision in the askable deicsion set. The doable decision set is obtained by starting with two specially constructed "risky decisions" and then uniformly sampling additional decisions from the complete Boston housing decision set, and the askable decision set is obtained by uniformly sampling from the complete Boston housing dataset and then scaling the chimerical features (the last two features of each decision) by ρ.

contains a query to match d^*. However, if D were to be constructed so that the better of the two decisions composing d^* depends only on the chimerical weights, DEER should be misled into the futile endeavour of selecting an askable decision query on the basis of learning about the chimerical weights, which should in turn result in poor performance.

To accomplish this, D is also modified so that it always contains two specially constructed decisions that serve as bait for DEER, which I refer to as the *risky* decisions, in addition to ones that are randomly sampled as before. The features for the two risky decisions are all 0.0 except the chimerical features, which are $-20, 10$ and $10, -20$ respectively. These decisions are risky in that they have negative value in expectation, but high value compared to other decisions in D when the weight corresponding to the negative chimerical feature is known to be low (near-zero or negative) and the weight corresponding to the positive chimerical feature is known to be high. As such, the agent would need to significantly improve its knowledge about the chimerical weights in order to make one of these risky decisions Bayes-optimal – and d^*, which always asks about the two risky decisions, does just that, hence setting an effective DEER trap.

Figure 6.4 shows the EVOI of the query selected by each algorithm as a function of ρ when the sizes of the doable and askable decision sets are both fixed to 15 (note that EVOI is given in logscale here). Comparing these results with those shown in Figure 6.3, the presence of the risky decisions causes ρ to have a dramatic effect on the performance of all algorithms except for AskEnum-AskEval, which is to be expected since AskEnum-AskEval ignores D. In particular, reading the graph right-to-left, for $\rho < 0.3$ DEER performs worse than MEDER, and for $\rho < 0.2$ DEER performs worse than AskEnum-AskEval until it degrades to the performance of Random for $\rho = 0$.

As mentioned above, augmenting A by scaling down the chimerical features lowers F1 in the sense that even though no askable decision queries exist that are helpful as a replacement for d^*, other askable decision queries exist that are helpful as a replacement for other, albeit strictly less valuable (Theorem 4.2), doable decision queries. The key conclusion of this experiment is that the lower F1 becomes, the more potential there is for DEER to miss the forest for the trees, and so more generally practitioners should consider DEER most applicable when the askable query set is balanced in its ability to match decision queries, all else being equal.

6.2.2.4 Experiment 6.4: Doable Decisions with Varied Risk and Limited Askability

Note that in addition to F1, F2 is also much lower in the scenario just considered compared to previous scenarios, as evidenced by the much higher magnitude of DoEnum-DoEval's EVOI relative to AskEnum-DoEval's EVOI compared to previous scenarios (recall that

the EVOI results given in Figure 6.4a are in log-scale). Along with F1, F2 also plays an important role in DEER's performance relative to the other algorithms, which I study in the next experiment through further analysis of ρ's effect on DEER's query selection tendencies in the risky decision scenario studied in this experiment, and by further augmenting the scenario to magnify or shrink the riskiness of the risky decisions.

Recall that in the previous experiment I considered a scenario constructed to push on DEER's first weakness: DEER's first step selects d^* as the query to match without considering what queries the askable query set contains. Here I focus on exemplifying and pushing on DEER's second weakness: namely, DEER's second step matches d^* on the basis of expected posterior response-entropy reduction, which does not consider the resulting impact on the agent's value or policy. As discussed in Chapter 5, DEER's second step is in fact equivalent to using MEDER for query selection when the (doable) decision set contains only the decisions d^* asks about, so DEER shares some of MEDER's weaknesses, which become particularly relevant when matching d^* is an unrealistic goal.

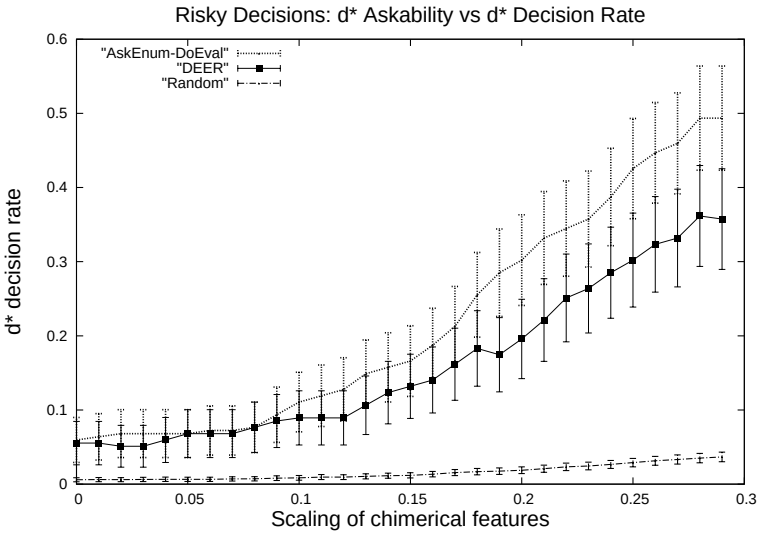
Consider Figure 6.5a, which shows, in the scenario considered in Experiment 6.3, the frequency at which DEER and AskEnum-DoEval would adopt one of the risky decisions asked about by d^* in response to the query they ask, as a function of ρ. As expected, for small values of ρ (say, < 0.1) neither DEER nor AskEnum-DoEval ask a query whose response would cause one of the risky decisions asked about by d^* to be Bayes-optimal for the corresonding posterior very often, since for small values of ρ the agent cannot learn enough about the chimerical features to offset either risky decision's negative prior value. As such, for small values of ρ, AskEnum-DoEval asks queries that allow it to select from the other decisions, resulting in its higher EVOI in Figure 6.4a compared to DEER, which attempts to learn about d^* even though doing so is futile. However, somewhat surprisingly, as ρ grows larger (say, starting at 0.2), AskEnum-DoEval begins asking queries that cause one of the risky decisions to be posterior Bayes-optimal about a third of the time, in expectation over both responses to the query asked, compared to about a fifth of the time for DEER's queries, even though unlike AskEnum-DoEval, DEER is explicitly aiming to reduce the uncertainty over d^*.

To investigate the cause of this phenomenon, I examined the specific queries being asked by AskEnum-DoEval and DEER for larger values of ρ, and found that DEER typically asked queries resulting in about the same posterior d^* response-entropy reduction for each possible response to its queries – which often resulted in neither response giving enough information to cause one of the risky decisions to be Bayes-optimal. In contrast, AskEnum-DoEval typically asked queries that, for one of the responses, significantly decreased the response-entropy of d^* to the extent that one of the risky decisions

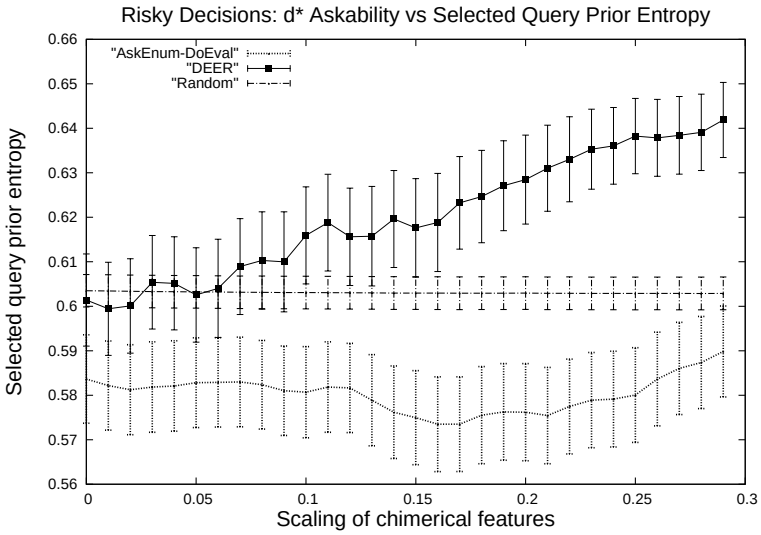
would be Bayes-optimal in the corresponding posterior, whereas for the other response, left the response-entropy of d^* relatively unchanged. In this sense, AskEnum-DoEval asked queries helping it choose between one of the non-risky decisions and one of the risky decisions, which yielded more useful information than DEER's queries, which were aimed at allowing the agent to choose one or the other of the risky decisions, even though most of the time there was not an askable query that yielded enough information about the chimerical features to allow it to do so. Indeed, Figure 6.4b shows that the average prior response-entropy of AskEnum-DoEval's queries tended to be much higher than that of DEER's queries – i.e., AskEnum-DoEval's queries were imbalanced in that one response was much more likely than the other, analogous to asking a question about whether or not an unlikely but influential (if true) statement is true of the world. DEER, on the other hand, selected more even-handed queries that yielded moderate d^* response-entropy reduction for both responses whose weighted sum exceeded that of AskEnum-AskEval's query, even though their responses rarely caused one or both of the risky decisions to be Bayes-optimal (extreme situations like these are responsible for the looseness of the upper bound given in Lemma 5.4).

As the potential for a good match increases as ρ increases, the askable query set becomes rich enough in learning about the chimerical features that DEER's choice of queries that attempt to learn about both risky decisions *do* allow it to select one or the other of the risky decisions, so the extent to which DEER is punished for optimistically matching d^* without taking into account the impact of its query on its decision diminishes. More generally, as the richness of the askable query set increases in terms of the best contained match to d^*, the penalty of using d^* response-entropy reduction to approximate EVOI decreases. This sense of richness in the askable query set is captured by F2, since if the askable query set is rich in that it does contain a query with EVOI close to that of d^*, DEER will select a query with EVOI close to that of d^* – unless d^* was a poor target to begin with, which depends on F1.

In order to study DEER's performance as a function of F2, consider extending the risky decisions scenario considered in Experiment 6.3 so that the negatiave feature associated with each risky decision becomes a parameter. Namely, let the negative feature of each risky decision be multiplied by a scaling factor τ. As τ increases, the *risk* increases in that the prior expected value of each risky decision decreases; however, the potential *reward* also increases because if the agent knew the values of the chimerical feature weights to be negative for one and positive for the other, increasing τ would actually make the more valuable of the two risky decisions even more valuable, leading to an increase in the EVOI of d^*. However, at the same time the knowledge that the agent must gain about the chimeri-

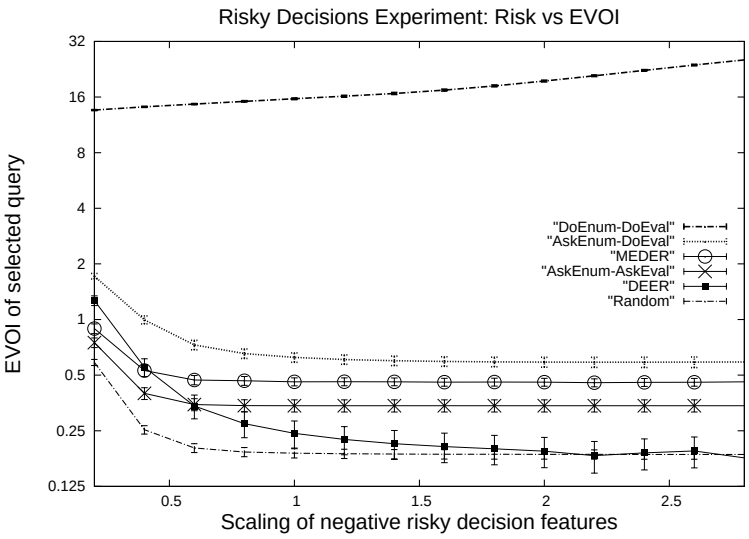


a)

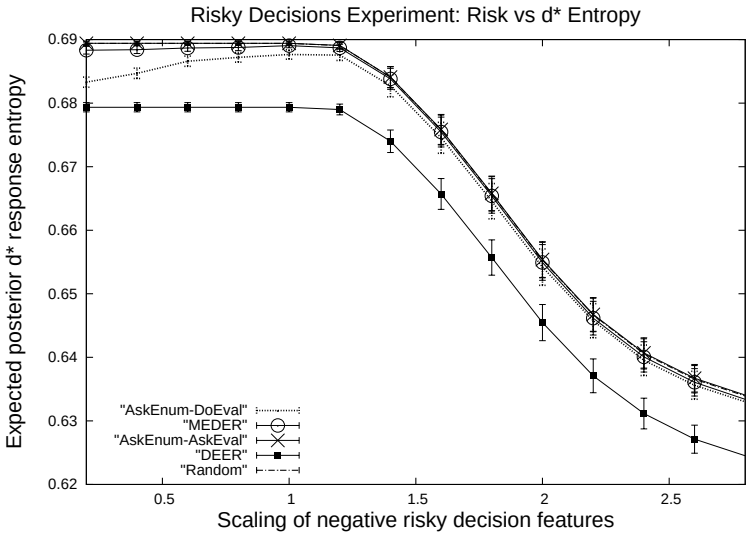


b)

Figure 6.5: a) Average number of decisions contained in the set asked about by d^* that are Bayes-optimal after updating ψ with the selected query's response (which can be 0, 1, or 2 for the binary decision queries considered here, and b) average prior response entropy of query selected, as a function of ρ: the scaling factor used for the chimerical features (the last two features) of each decision in the askable deicsion set. The doable decision set is obtained by starting with two specially constructed "risky decisions" and then uniformly sampling additional decisions from the complete Boston housing decision set, and the askable decision set is obtained by uniformly sampling from the complete Boston housing dataset and then scaling the chimerical features (the last two features of each decision) by ρ.



a)



b)

Figure 6.6: Experiment 6.4 – a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of τ: the scaling factor used for the negative decision features of the two risky decisions present in the doable decision set. The doable decision set is obtained by starting with two specially constructed "risky decisions", scaling their negative features by τ, and then uniformly sampling additional decisions from the complete Boston housing decision set. The askable decision set is obtained by uniformly sampling from the complete Boston housing dataset and then scaling the chimerical features (the last two features of each decision) by $\rho = 0.1$.

cal weights in order to make one of the risky decisions Bayes-optimal increases with τ as well. Hence, in this scenario increasing τ can be viewed as decreasing F2, which makes the problem harder for DEER.

Figure 6.6a shows the EVOI of the query selected by each method (in log-scale) as a function of τ when ρ is fixed to 0.1 – notice that the increase of τ induces increase of the gap between AskEnum-DoEval and DoEnum-DoEval, which reflects decreasing F2. When $\tau < 1$, DEER performs as well or better than the other approximate algorithms because the risky decisions are no longer as risky, and as such DEER's queries that are directed toward learning which is better of the two are useful more often even though its ability to do so is limited (since $\rho = 0.1$)[6]. However, as τ increases, so does the detrimental effect of DEER's entropy-related weakness, quickly resulting in DEER's performance dropping towards that of Random as the closeness in match to d^* the agent needs to change its decision to one of the risky decisions relative to what it can ask becomes increasingly unrealistic, at which point the agent should focus on queries that help it select from the rest of the decisions instead. Figure 6.6b verifies this explanation, since up to $\tau = 1.2$ the extent to which the agent can match d^* changes only negligibly, yet throughout this interval DEER's performance relative to the other algorithms in terms of EVOI significantly drops. Note that the drop in d^* posterior entropy for $\tau > 1.2$ is due to the fact that d^* begins to change increasingly often from asking about the two risky decisions to asking about one risky decision compared to a non-risky decision, which has lower prior entropy but still cannot be usefully matched by an askable query for these values of τ.

The key conclusion of this experiment is that as F2 decreases (i.e., the askable query set becomes more impoverished in terms of the EVOI of its best query compared to the EVOI of the best possible binary-response query d^*), so does the potential for DEER's approach of maximal d^* response-entropy reduction to be misguided in how it acheives posterior entropy reduction. Hence, practitioners should consider DEER most applicable when the askable query set is nearly as rich as the decision query set in terms of the EVOI of queries contained.

6.2.3 Empirical performance of DEER approximations

In the previous section I studied DEER's EVOI-maximizing performance compared to alternative algorithms. However, as I discuss in Section 6.2.4, DEER in its exact form may not

[6]Note that for τ in the interval between 0.0 and approximately 0.1 (Figure 6.6a begins at $\tau = 0.2$), the better of the two risky decisions under the prior is often Bayes-optimal; in this interval the dominant trend is that the EVOI of all algorithms increases until approximately $\tau = 0.2$ (DEER performs nearly as well as AskEnum-DoEval here, unlike the other algorithms), at which point the risky decisions are rarely Bayes-optimal under the prior and the trend described in this experiment becomes dominant.

be computationally viable for some settings. Here I consider DMVM, DEER-Greedy, and DMVM-Greedy as approximations for DEER (see Section 6.1.1 for specifications of these algorithms). Next I summarize the results I obtained when applying these three algorithms to each of the four scenarios considered above, comparing their EVOI performance to that of DEER and Random.

Approximating DEER with DMVM. When applied to the first two scenarios considered above, DMVM and DEER shared very similar EVOI performance (Figures 6.7a and 6.8a, and indeed in both scenarios DMVM lowered the response-entropy of d^* nearly as effectively as DEER (Figures 6.7b and 6.8b). When applied to the third and fourth scenarios considered above, however, DMVM performed significantly worse than DEER, and even worse than Random at many points (Figures 6.9a and 6.10a), which was accompanied by significant differences in the extent to which DMVM was able to lower the response-entropy of d^* compared to DEER (Figures 6.9b and 6.10b).

Greedy approximation of DEER and DMVM. In all four scenarios, DEER-Greedy performed nearly identically to DEER (part a of Figures 6.7 through 6.10). On the other hand, DMVM-Greedy performed nearly identically to DMVM in the first two scenarios (Figures 6.7a and 6.8a) and then significantly better in the third and fourth scenarios (Figures 6.9a and 6.10a). While in general the greedy approximations of both methods have the potential to perform better than their exact counterparts when the greedily constructed decision query happens to be a better target than d^* given the queries available in the askable decision query set, it is unclear why DMVM-Greedy displayed superior performance as compared to DMVM in the third and fourth scenarios while DEER-Greedy did not outperform DEER, but the low sensitivity of the mistake region criterion to the agent's prior compared to that of expected response-entropy reduction means that DMVM-Greedy's query varies less between trials compared to DEER-Greedy's query, which means DMVM-Greedy's choice of query between trials is mostly a function of the askable and doable decision sets sampled. Thus, it is conceivable that, for some points, DMVM-Greedy's less variant queries would be more valuable on average than DEER-Greedy's, but this effect could potentially be flipped were the properties of the decision sets being used to change just enough to change DMVM's typical choice of query.

To summarize, DMVM effectively approximated DEER in the easy scenarios (the first and second), but its performance degraded significantly in the harder scenarios (the third and fourth). On the other hand, greedy approximation of both DEER and DMVM resulted

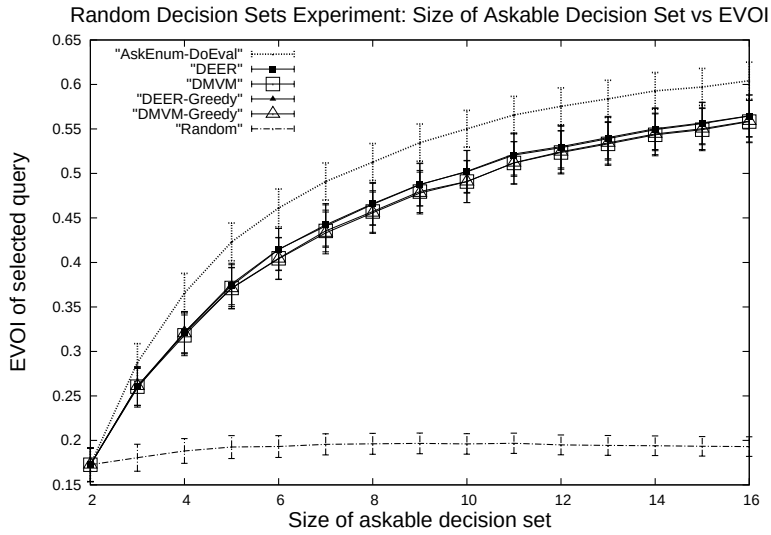
in the same or better performace in all four scenarios. These results suggest that DMVM, DEER-Greedy, and DMVM-Greedy, when applied judiciously, may all be promising ways to implement WQP as approximations for DEER, and in the next section I show that they afford significant computational savings when used in place of DEER.

6.2.4 Computational Performance

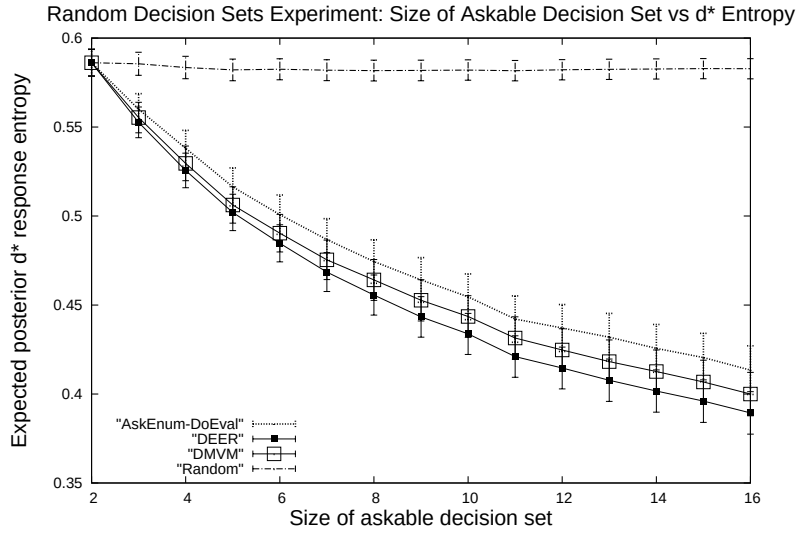
A good query selection algorithm should not only be able to find queries with high EVOI, but it should do so at a reasonable computational cost. In Chapter 5 and throughout this dissertation I have set the computational goal of query selection algorithms to being able to select a query without requiring computation time that scales with the product of optimal planning time and the size of the query set. DEER and DMVM, along with their greedy versions, accomplish this goal from a theoretical perspective; however, their first step of solving for d^*, which depends on optimal planning complexity, may incur significant cost, and they still must enumerate the query set during the projection step (in the absence of exploitable structure). In this section I determine how they compare to alternative algorithms in empirical computational costs for askable decision query selection in the Boston Housing dataset.

I compare the computational costs of the algorithms discussed in this chapter according to how much time, empirically, they take to select a query, as a function of (a) the size of the askable query set; and (b) the size of the underlying optimal planning problem. I quantify these dimensions in the askable query selection problem studied in this chapter as the size of the askable decision set, and the size of the doable decision set, respectively, and the results are shown in Figure 6.11a and b, respectively. Note that in both cases the decision sets are sampled uniformly randomly as in Section 6.2.2.1.

First, consider how DEER and DMVM compare in computation time. DEER and DMVM scale the same asymptotically since they share the same first step of solving for d^*, and both apply their projection criteria by enumerating the askable query set. However, Figure 6.11a shows that the overhead associated with DEER's expected posterior d^* response-entropy criterion is much higher than that of DMVM's mistake region criterion, which is implemented here with a decision-boundary angle criterion (see Section 6.1.1 for details). In fact, in Figure 6.11a, DMVM appears to require negligibly more computation as the askable decision set – and accordingly, the askable decision *query* set – grows, as for DMVM the computation associated with the optimal-planning-related step of solving for d^* dominates that of the projection step. Hence, for DMVM many more askable decision queries could conceivably be added to the set without significantly changing the computa-

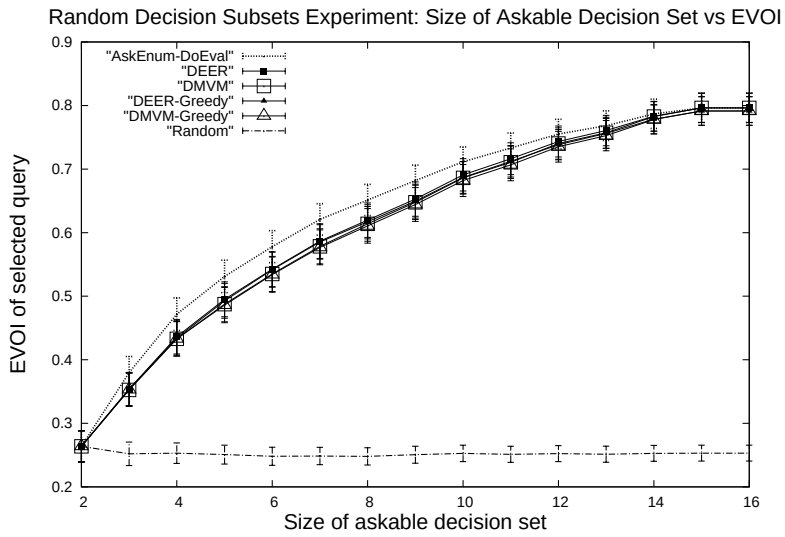


a)

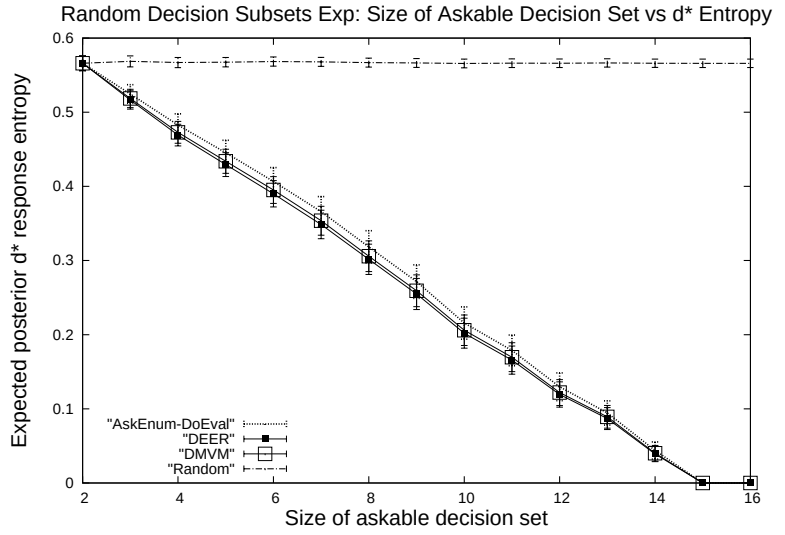


b)

Figure 6.7: Results for Experiment 6.1– a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of the size of the askable decision set; the size of the doable decision set is fixed at 15. For each trial, both the askable and doable decisions sets are uniformly sampled from the complete Boston housing decision set.

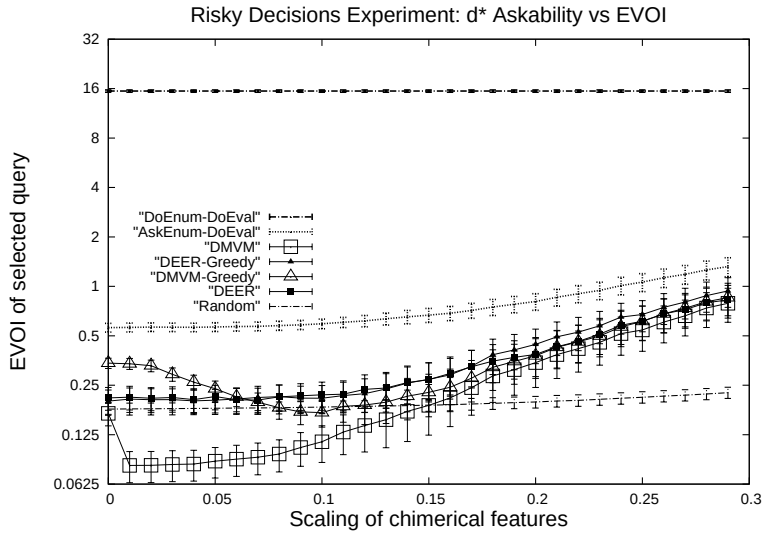


a)

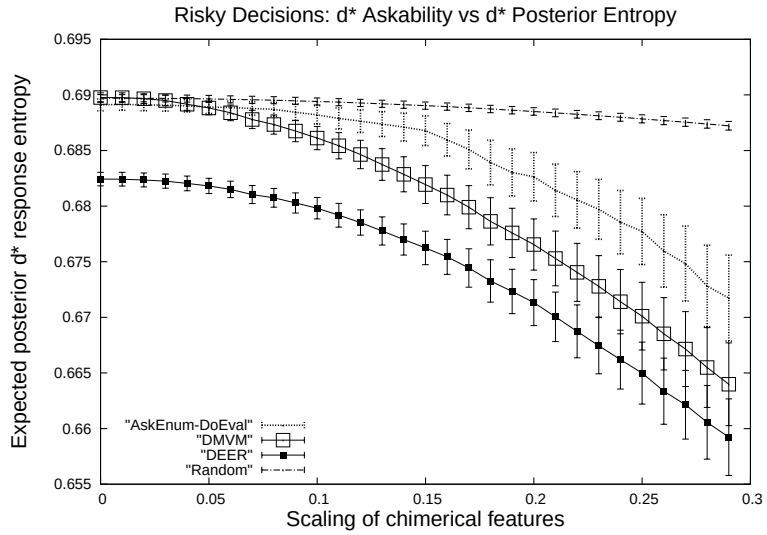


b)

Figure 6.8: Results for Experiment 6.2 – a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of the size of the askable decision set; the size of the doable decision set is fixed at 15. For each trial the doable decision set is uniformly sampled from the complete Boston housing decision set, and the askable decision set is then sampled from the doable decision set.

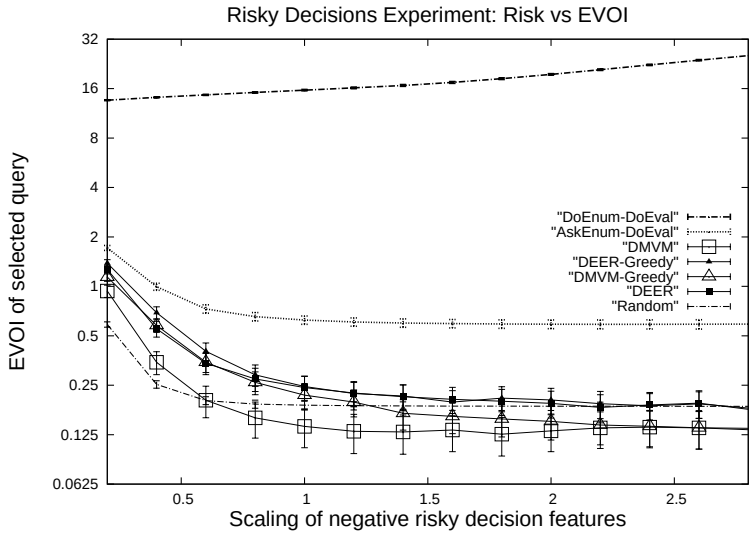


a)

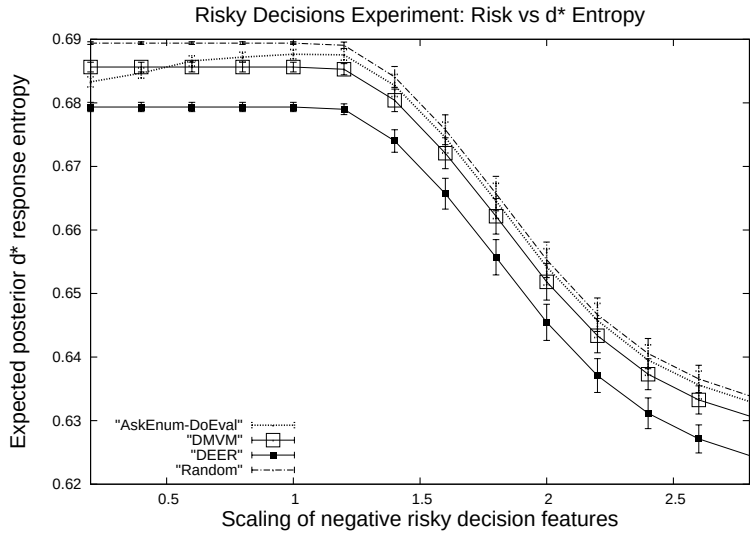


b)

Figure 6.9: Results for Experiment 6.3 – a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of ρ: the scaling factor used for the chimerical features (the last two features) of each decision in the askable deicsion set. The doable decision set is obtained by starting with two specially constructed "risky decisions" and then uniformly sampling additional decisions from the complete Boston housing decision set, and the askable decision set is obtained by uniformly sampling from the complete Boston housing dataset and then scaling the chimerical features (the last two features of each decision) by ρ.



a)



b)

Figure 6.10: Results for Experiment 6.4 – a) Average EVOI of query selected and b) average posterior response entropy of d^* induced by query selected, as a function of τ: the scaling factor used for the negative decision features of the two risky decisions present in the doable decision set. The doable decision set is obtained by starting with two specially constructed "risky decisions", scaling their negative features by τ, and then uniformly sampling additional decisions from the complete Boston housing decision set. The askable decision set is obtained by uniformly sampling from the complete Boston housing dataset and then scaling the chimerical features (the last two features of each decision) by $\rho = 0.1$.

tion time required to select a query.

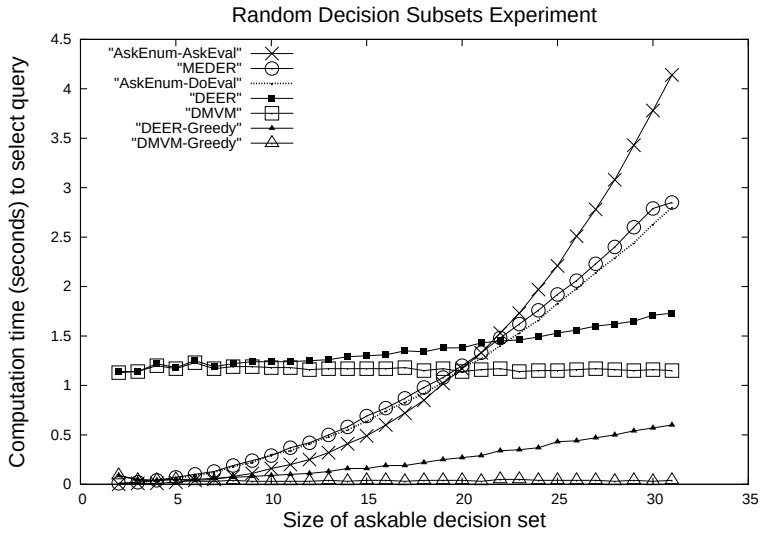
The same relationship holds between DEER-Greedy and DMVM-Greedy, and both benefit from the reduced computational burden of the first step through its greedy construction approximation – the computational savings of this approximation are particularly salient in Figure 6.11b, as the computational growth of the greedy variants as a function of the size of the doable decision set is nearly flat, as opposed to the exact versions whose computation grows quadratically. In fact, DMVM-Greedy requires only between 10 and 50 miliseconds to select a query at all points shown.

When the askable decision set is small, AskEnum-AskEval's approach of ignoring the doable decision set results in significant computational savings as compared to AskEnum-DoEval, but this same trait results in its computation comparing unfavorably to the other algorithms, even the exact AskEnum-DoEval, when the askable decision set grows to be large enough. MEDER, which was implemented exactly, requires similar computation to that of AskEnum-DoEval – this is because computing decision-entropy in the askable decision query setting requires roughly the same computation as optimal planning. In practice, this computation could be approximated heuristically by, for example, considering only a sampled subset of the decisions when computing decision entropy (which would make its computation similar to that of DEER, depending on the size of the sampled set), or other uncertainty-based methods could be used – in this chapter, the purpose of MEDER is to serve as an example of an uncertainty-based method to study its effectiveness in maximizing EVOI compared to DEER, without focusing on its efficient implementation.

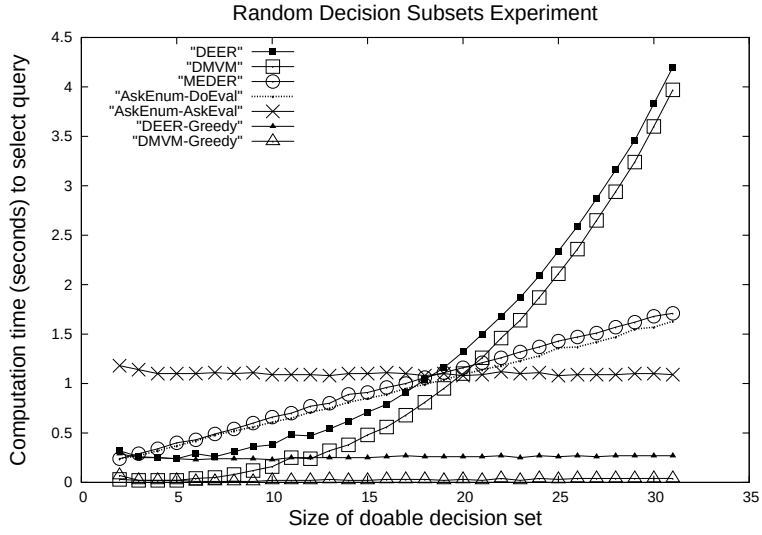
In summary, this section provided the following straightforward computational results: while DEER had high computational costs when implemented exactly, approximating its first step greedily (DEER-Greedy) caused its performance as a function of the size of the underlying decision problem to improve substantially, and approximating its second step through mistake region minimization (DMVM), whose implementation exploited structure in the askable decision query setting studied in this chapter, caused its performance as a function of the size of the askable query set to improve substantially as well; combining the two approximations (DMVM-Greedy) reduced the empirical time required to select a query by several orders of magnitude as compared to DEER and AskEnum-DoEval.

6.3 Discussion

In this chapter I began by defining the *askable decision query selection setting*: an extension of the standard decision query selection setting where the decisions the agent can ask about and the decisions the agent can execute can be different. Here I studied the EVOI-



a)



b)

Figure 6.11: Average computation in seconds to select a query as a function of the size of the askable (a) and doable (b) decision set, where in (a) the size of doable set is fixed at 20, and in (b) the size of the askable set is fixed at 20.

maximizing abilities and computational properties of AskEnum-AskEval (an algorithm that utilizes structure in this setting), MEDER, and Wishful Query Projection (WQP) algorithms (DEER and DMVM). I then presented an empirical study that focused on comparing DEER's performance in terms of query EVOI to that of MEDER, AskEnum-AskEval, and baselines as a means of both providing empirical evidence for the merit of using the WQP approach for query selection, and to better understand what factors should be considered when determining whether WQP would be effective for query selection.

Confirming intuition from Chapter 5, the empirical study showed that, while the relative performance of the algorithms depended on myriad factors, practitioners should consider at least two key factors when assessing the viability of using DEER as a WQP algorithm for their query selection problem. First, the quality of queries selected by DEER can suffer when the askable query set is imbalanced in terms of obtainable information about decisions, so DEER is most applicable when the askable query set contains a wide range of queries. Second, the quality of queries selected by DEER can suffer when the askable query set is impoverished in terms of its highest contained query EVOI compared to the EVOI-optimal decision query, so DEER is most applicable when the askable query set consistently contains queries with close to the highest possible k-response query EVOI. Practitioners should treat these only as guidelines, not strict guarantees, as they directly correlate with DEER's performance only in the sense of upper bounds on the extent to which things can go wrong.

6.4 Appendix

In this appendix I provide additional analysis pertaining to the askable decision query setting, as even though my primary motivation for studying the askable decision query setting in the context of this dissertation was as an empirical testing ground for strengthening intuitions regarding what factors are important to consider when determining the applicability of WQP algorithms compared to alternatives such as MEDER, the setting is interesting in its own right.

Specifically, I provide additional theoretical analysis regarding the EVOI performance of AskEnum-AskEval compared to VBDR, a third WQP algorithm. In addition, I discuss in detail the potential merit of another algorithm for askable decision query selection, which can be viewed as an extension of the greedy decision query construction procedure from the standard decision query selection setting.

6.4.1 Value-based Decision Replacement

AskEnum-AskEval is related to a value-based WQP algorithm. Namely, consider a third projection algorithm which applies only to the askable decision query setting: Value-Based Decision Replacement (VBDR), which selects a query from Q_A as follows. First, d^* is computed. Then, each decision u_j queried by d^* is replaced by a decision a_j so as to minimize $\max_{\omega\in\Omega} |V_\omega^{u_j} - V_\omega^{a_j}|$, producing query $\hat{q}_A$. I show next that VBDR and AskEnum-AskEval are closely related in that a performance guarantee similar to the one stated in Theorem 6.2 applies to VBDR; the result is stated as Theorem 6.4 below. To this end, I first prove Lemma 6.3, which states that any query about decisions in D can be replaced by a query about decisions in A to ensure that a maximum of $\Delta(D, A)$ EVOI loss is incurred:

Lemma 6.3. *For any $q \in Q_D$, there exists some $q' \in Q_A$ such that*

$$|EVOI(q;\psi) - EVOI(q';\psi)| \leq \Delta(D, A).$$

Proof. The proof is by construction. Consider some $q \in Q_D$, and let u_j denote the posterior Bayes-optimal decision under the posterior induced by the j^{th} possible response to q for $\{u_j\}_{j=1}^k$; i.e., $u_j \triangleq \arg\max_{u\in D} V^u_{\psi|q=j}$. Construct query q' to query $\{a_1, a_2, \ldots, a_k\}$, where $a_j = \arg\min_{a_j\in A} \max_{\omega\in\Omega} |V_\omega^{u_j} - V_\omega^{a_j}|$. Then,

$$
\begin{aligned}
EVOI(q;\psi) - V^*_\psi &= \sum_{j=1}^{k} \Pr(q=j) V^*_{\psi|q=j} \\
&= \sum_{j=1}^{k} \Pr(q=j) V^{u_j}_{\psi|q=j} \\
&\leq \sum_{j=1}^{k} \Pr(q=j) \big(V^{a_j}_{\psi|q=j} + \Delta(D,A)\big) \\
&= \sum_{j=1}^{k} \Pr(q=j) V^{a_j}_{\psi|q=j} + \Delta(D,A) \\
&\leq \sum_{j=1}^{k} \Pr(q'=j) V^{a_j}_{\psi|q'=j} + \Delta(D,A) \text{ (\textbf{By Lemma 4.1})}.
\end{aligned}
$$

□

Combining Theorem 4.2 with Lemma 6.3 above yields the EVOI-loss bound for VBDR:

Theorem 6.4. *Let q^* denote the Q_A-EVOI-optimal query, and let $\hat{q}$ denote the query selected by VBDR. Then for any decision problem and uncertainty ψ over them,*

$$EVOI(q^*;\psi) - EVOI(\hat{q};\psi) \leq \Delta(D,A).$$

Proof. Let d^* denote the Q_D-EVOI-optimal query, and recall that $\hat{q}$ is constructed by replacing each decision u_j queried by d^* with decision a_j in order to minimize $\max_{\omega\in\Omega} |V^{u_j}_\omega - V^{a_j}_\omega| \leq \Delta(D,A)$. Then,

$$
\begin{aligned}
EVOI(q^*;\psi) - EVOI(\hat{q};\psi) &\leq EVOI(d^*;\psi) - EVOI(\hat{q};\psi) \text{ \textbf{(By Theorem 4.2)}} \\
&\leq \Delta(D,A) \text{ \textbf{(By Lemma 6.3)}}.
\end{aligned}
$$

□

Theorem 6.4 shows that like AskEnum-AskEval, VBDR is guaranteed to work well when the underlying askable and doable decision problems are similar in terms of the exact value magnitudes prescribed by the parameter space to the available decisions in each set. Intuitively, VBDR and AskEnum-AskEval share a key weakness: they both can fail when the best askable decision queries ask about askable decisions with very low values relative to other askable decisions, since VBDR attempts to match the value magnitudes of the

decisions composing the best doable decision query (which are biased to be high), and AskEnum-AskEval is biased towards using askable decisions with high value magnitudes. In contrast, DEER and DMVM do not share this weakness because their projection criteria consider relative decision values instead of magnitudes.

Next, I conclude my theoretical discussion of askable decision query selection by discussing another specialized approach for the setting: greedily approximating AskEnum-DoEval.

6.4.2 Greedy AskEnum-DoEval

Recall that the Q_A-EVOI-optimal decision query can be computed through the AskEnum-DoEval algorithm described above. However, AskEnum-DoEval's approach of evaluating every combination of k askable decisions requires evaluating $\mathcal{O}(|A|^k)$ queries, which would be infeasible in many settings. It is natural to ask, then, whether adapting the greedy construction procedure of Viappiani and Boutilier (2010) to construct the query from decisions in A while evaluating them with respect to D would yield similar computational benefits and approximation guarantees as in the standard decision query selection setting[7] (refer to Section 4.5 of Chapter 4 for details of the greedy algorithm and notation, both of which are referred to below). I refer to this greedy approximation to AskEnum-DoEval as *Askable Greedy construction Doable Evaluation (AskGreedy-DoEval)*.

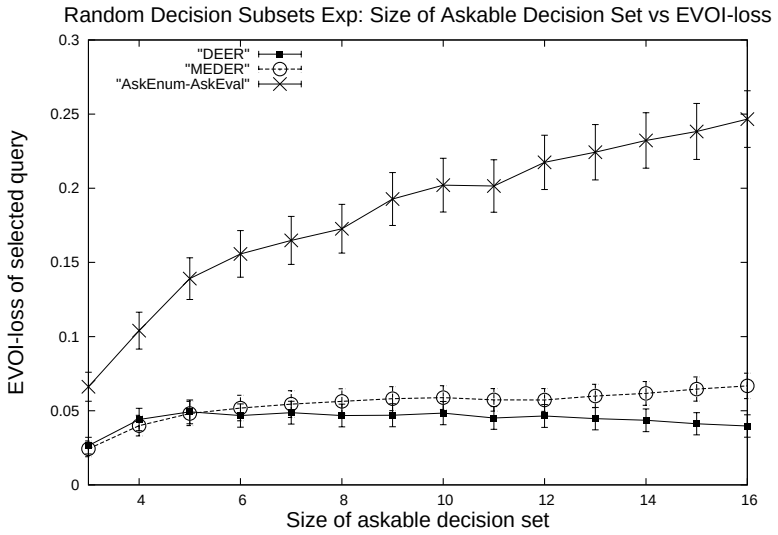
Computationally, the algorithm must perform $\mathcal{O}(k|A|)$ EVOI computations, each of which have complexity $\mathcal{O}(k|D|B)$ (recall that B represents the complexity of bayesian inference, such as the number of particles used in a particle filtering approach). Although AskEnum-DoEval evaluates fewer queries than AskEnum-DoEval, these EVOI computations cannot be replaced by EVOR computations as they can in the standard decision query selection setting (recall that the EVOR of a decision query is defined as the expected value of executing the decision associated with the response chosen by the user when asked the query), because the agent cannot necessarily execute the decisions in the set being constructed. Thus, one of the major computational advantages of the greedy construction procedure over exhaustive search in the standard decision query setting does not apply to the askable decision query setting – recall that EVOR computations are $\mathcal{O}(kB)$, in contrast to EVOI computations which here are $\mathcal{O}(k|D|B)$.

The approximation guarantee that applies to the greedy construction procedure is lost in the askable decision query selection setting as well. To understand why, consider the following argument. As discussed in Section 4.5, the approximation guarantee proven

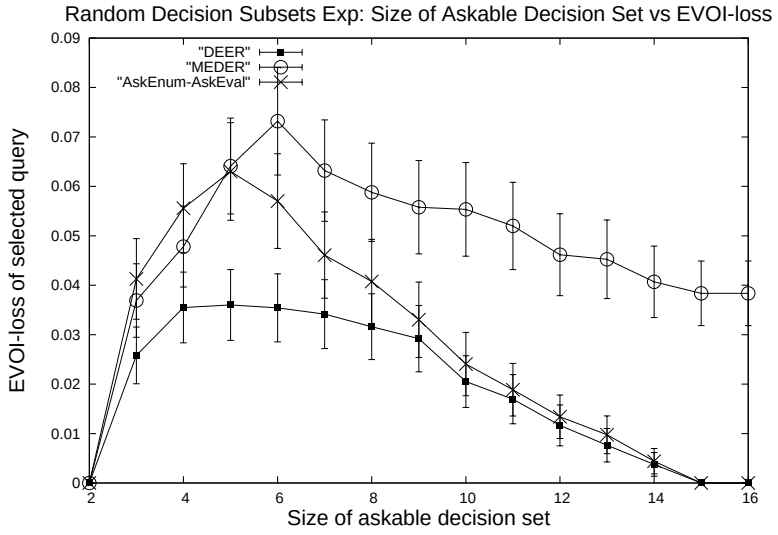
[7] How to choose the first decision to add to the set is unclear. Here, assume that the algorithm begins by adding the askable decision that has highest expected value out of decisions in A.

by Viappiani and Boutilier (2010) is derived by proving and combining the following two results: 1) EVOR is monotonic and submodular (this implies that EVOR can be maximized greedily while offering the guarantee derived from the more general Nemhauser result (Nemhauser et al., 1978); and 2) the decision query with highest EVOI also has highest EVOR. However, in the askable decision query setting, the askable decision query with highest EVOR may not have the highest EVOI, since applied to askable decision queries EVOR would consider how the query would induce improvement in selecting from A instead of D. Thus, one would need to prove that EVOI, instead of EVOR, is monotonic and submodular for queries constructed from decisions in A. However, unlike EVOR, EVOI is nonmonotonic in general for decision queries, and so cannot be nonmonotonic for queries constructed from decisions in A. To see this, consider Example 6.1 above and the Q_A-EVOI-optimal binary-response query asking about $\{u_1, u_2\}$. Recall that even though u_1 and u_2 have low value compared to u_3 and u_4, the information provided by distinguishing between u_1 and u_2 is useful for distinguishing between u_5 and u_6. Now consider adding u_3 to the set so the query becomes $\{u_1, u_2, u_3\}$. The answer will always be u_3, which provides no information to help distinguish between u_5 and u_6, and so the EVOI of the expanded query is zero. Hence, EVOI is nonmonotonic and so the same guarantee cannot directly apply in this setting.

This concludes my discussion of AskGreedy-DoEval; I leave empirical analysis and further theoretical analysis of AskGreedy-DoEval for future work.



a)



b)

Figure 6.12: Supplemental results for Experiments 6.1 and 6.2 – average EVOI-*loss* of query selected as a function of the askable decision set, where the size of the doable decision set is fixed at 15 and uniformly sampled from the complete Boston housing decision set, and where a) the askable decision set is uniformly sampled from the complete Boston housing decision set (Experiment 6.1); or b) the askable decision set is sampled uniformly from the doable decision set (Experiment 6.2).

CHAPTER 7

Conclusions

A variety of research communities are concerned with the problem of how an agent should select which query to ask its human user from an available set in order to improve future decision-making under uncertainty, both from a theoretical perspective (e.g., preference elicitation) and from a practical perspective (e.g., human/agent interaction). Among these communities, an oft-used criterion (and the one used in this dissertation) for measuring the value of asking a query is its associated Expected Value of Information (EVOI), which measures the expected impact of a query's response on the agent's policy and hence value.

EVOI-based query selection presents significant challenges in settings where optimal planning under uncertainty is computationally expensive, since EVOI measures the expected impact of the query on the agent's policy, especially when the set of queries the agent considers is large. In such cases any benefits brought about by considering queries to ask may be outweighed by the additional computational expense required to reason over them, limiting the scope of potential practical applications of querying in agents. The work in this disseration makes theoretical and practical contributions to overcoming these challenges in EVOI-based query selection, focusing on settings where incorporating query responses to update the agent's uncertainty is much less computationally demanding than optimal planning computations.

7.1 Summary of Contributions

I now enumerate the main contributions I make in this dissertation, ordered by the chapters in which they appear.

1) Hybrid Algorithm for Action Query Selection in Sequential Decision-Making (Chapter 3). The first contribution is the Hybrid algorithm, which is designed for action query selection in sequential decision-making settings. Hybrid utilizes the computational effi-

ciency of uncertainty reduction to select a promising subset of queries on which to perform computationally expensive EVOI analysis. The efficacy of this Hybrid algorithm is demonstrated in the empirical investigations, where it displays clear advantages over pure uncertainty reduction and pure EVOI maximization in a setting considered in related work, supporting the hypothesis that hybrid uncertainty-based and EVOI-based techniques can be applied to feasibly perform EVOI-based query selection in problems of interest to the community.

2) EVOI-Sufficiency of k-Response Decision Query Set (Chapter 4). Another contribution is a theoretical result that narrows the space of queries that need even be considered, provided maximizing EVOI is the explicit target and all other factors such as their human understandability/answerability are ignored. Namely, I show that given the only requirement is that queries must be constrained to having k possible responses, the set of k-response decision queries is sufficiently general in that there is no benefit in considering any additional k-response queries. Practitioners can use this contribution to inform their design of query sets by restricting attention to decision queries, where efficient EVOI-based query selection algorithms exploiting submodularity have been recently developed. Of course, this contribution is directly useful only when decision queries can be sensibly asked and easily answered by humans in the target setting; however, even in such cases, the community can design query sets in a principled manner by designing them to be as informative about decision queries as possible, which connects with the next (third) contribution. In addition, this contribution forms the foundation for the fourth contribution below.

3) Response-Entropy Bound for EVOI-loss in Decision Query Projection (Chapter 5). The next contribution establishes a means to characterize the worst-case loss in EVOI when using some k-response query in place of a decision query. Intuitively, the more a k-response query reduces uncertainty (in expectation) in what the response to a locally optimal decision query would be, the smaller the loss in using the k-response query as a substitute for that k-response decision query. Researchers can use this contribution to assess the loss in EVOI associated with using some k-response query set that is suited to their target setting according to factors independent of EVOI (such as costs in developing interfaces for accurately presenting queries to humans and cognitive burdens imposed on humans to answer them), compared to if they were to use the k-response decision query set. In addition, this contribution, combined with the previous contribution, provides theoretical justification for the next contribution.

4) DEER Algorithm for Query Selection (Chapters 5 and 6). A fourth contribution draws on the previous two contributions to implement the Wishful Query Projection (WQP) approach for query selection, which is hypothesized in this dissertation as a means for tractably finding queries with high EVOI in settings where repeated optimal planning computations would be computationally prohibitive, in the form of a concrete query selection algorithm called Directed Expected Entropy Reduction (DEER). DEER begins by selecting an EVOI-optimal k-response decision query, which, according to the second contribution above, is the ideal k-response query to ask purely in terms of EVOI. However, this ideal query cannot be asked directly unless it is included by the target query set, and so the DEER algorithm selects the query from the target query set serving as the best substitute according to the third contribution above, and hence approximates EVOI-based query selection by restricting EVOI evaluations to an efficiently searchable (decision) query space and then finding a suitable match in the askable query set.

Applying the second and third contributions described above, DEER is shown to be principled in that it is guaranteed to select a query with EVOI close to the best query in the set, as a function of two different measures of the query set's similarity to the k-response decision query set. DEER is also empirically evaluated by comparing it with baseline algorithms and several other approximate query selection algorithms, profiling its computational costs and performance abilities to delineate the contexts in which it provides an effective tradeoff between good query choices and computational costs, as a function of key properties of the query set to be selected from and the decision problem the agent faces, confirming the hypothesis that WQP can be implemented to produce an efficient and approximate query selection algorithm both from a theroetical and empirical perspective. DEER thus contributes to the community a new query selection algorithm with well-understood strengths and limitations that can be particularly effective when (1) the computational cost of updating the agent's uncertainty in light of query responses is small compared to evaluating the EVOI of a query, (2) the agent's query set is balanced in the extent to which it contains queries similar to the range of k-response decision queries; and (3) the agent's query set is rich in the extent to which it contains a query with similar EVOI to that of the EVOI-optimal k-response decision query.

7.2 Future Work

I close this dissertation with an informal discussion of a selection of opportunities for future work, organized by topic.

7.2.1 Query Selection in Sequential Decision-Making

In Chapter 3 I studied three instances of query selection problems defined in sequential decision-making settings, for which I provided some basic uncertainty representations and computational tools allowing EVOI-based query selection to be feasibly approximated. However, I did not consider implementing Wishful Query Selection (WQP) as a means for approximating EVOI-based query selection in these settings, and I discuss this topic below, focusing on the computational challenges associated with WQP selection in these settings. Then, as a separate topic I revisit the Ask When Impacts Next Action (AWINA) heuristic discussed briefly in Chapter 3 as a means to address the question of *when* the agent should query in addition to the question of focus in this dissertation of *what* the agent should query.

7.2.1.1 Implementing k-Response Decision Query Selection

Recall that this dissertation considers EVOI-based query selection in settings where optimal planning computations are the primary computational concern. However, in Chapter 6 I only evaluated WQP in a setting where optimal planning could be feasibly accomplished by enumerating the entire decision set, and focused on understanding the key factors influencing the EVOI-maximizing effectiveness of WQP. However, for sequential decision-making settings like those studied in Chapter 3, the decision set (policy set) is far too large to enumerate, preventing even the greedy construction procedure for *approximating* WQP's first step, wishful query selection, from being feasibly applied, since its complexity is $\mathcal{O}(k^2|U|B)$. For WQP to be feasibly applied to query selection in sequential decision making problems, wishful query selection would need to take advantage of structure to compute or approximate the EVOI-optimal k-response decision query.

In constructing the approximately EVOI-optimal k-response decision query, the step of the greedy algorithm causing the linear dependence on $|U|$ is the step of computing the best decision to add to the set given a subset of decisions added so far. Specifically, when adding the j^{th} decision to the set $\{u_1, u_2, \ldots, u_{j-1}\}$, the step needs to solve the following optimization problem:

$$u_j^* = \arg\max_{u_j \in U} \Big\{ \mathbb{E}_{\omega \sim \psi} \Big[\sum_{i=1}^{j} \delta(V_\omega^{u_i} > \max_{u \neq i} V_\omega^{u \neq i}) V_\omega^{u_i} \Big] \Big\}.$$

Consider applying the greedy construction procedure to k-response decision query selection in the reward-uncertain sequential decision-making setting studied in Chapter 3. Here, decisions correspond to policies that map state to action, so intuitively, the above optimization problem boils down to the following: suppose the agent currently occupies

state s, and consider a set of $j-1$ policies $\pi_1, \pi_2, \ldots, \pi_{j-1}$. Which j^{th} policy should it add to the set if it will be told which of the j policies has highest value, and then it must adopt that policy?

The main factor that makes this problem nontrivial is that the probabilities for each policy being the best in the set change as a function of the j^{th} policy added to the set. Thus, a good policy to add is one that is different enough from the ones already added so that it would be a better choice than the rest under some subset of Ω, but different in a way that the subset of Ω on which it is better has reasonably high probability. It seems unlikely that solving this problem should require exhaustively searching the entire policy space, since the way the weights on each policy added so far change as a function of the j^{th} policy to add is directly related to how they compare in expected value. Thus, I conjecture that it can be framed as a convex optimization problem of some sort, if not a linear program. Ideally, solving this problem would have similar complexity to that of optimal planning in the same setting (ideally adding only a linear or quadratic dependence on k), making it tractable. Of course, even if this is the case, it is unclear whether k-response decision query selection could be solved in a computationally feasible manner for other forms of uncertainty besides reward uncertainty, as the case of reward uncertainty allows optimal planning under uncertainty to be reduced to a fully observable planning problem through the mean-MDP method discussed in Chapter 3, unlike other forms of uncertainty.

7.2.1.2 When to Query?

I assumed in Chapter 3 that the agent must ask its query while occupying its current state in order to focus on the question of *what* query the agent should ask, but developing an effective query-asking agent in a sequential decision-making setting in practice would surely require addressing the question of *when* the agent should query. While here I will not propose a way to extend the setting considered in Chapter 3 to fully capture this question, the Ask When Impacts Next Action (AWINA) heuristic briefly discussed in Section 3.2.3.5 suggests one way the agent can be intelligent about when it queries.

Recall that when using AWINA, the agent only asks the query it selects if the response has the potential to change its action at its current state. Otherwise, it takes an action and then selects a new query conditioned on its new state, and repeats the process. Supposing the agent can ask an unlimited number of queries before taking actions, the agent would only ask queries until no query would change its next action, which does limit the number of queries asked by the agent.

However, AWINA suffers from the limitation that the agent will query if just one of the possible responses to the query could influence its action. This could result in the agent

asking many queries before taking its action if there exist many factors that are unlikely to be true but would change the agent's action if true, supposing the askable query set contains queries asking about those factors. Since these queries would typically have neglible EVOI, one way to overcome this issue would be to impose a threshold, possibly expressed as a *cost of querying*, and unless the threshold is exceeded by the query's EVOI, the query would not be asked, and the agent would take an action instead. The main challenge would be defining costs that are appropriately callibrated with respect to values in the decision problem versus costs the user associates with answering a query.

7.2.2 Extending Wishful Query Projection

Recall that the WQP algorithms developed in Chapter 5 implement the first step of WQP (wishful query selection) by computing the EVOI-optimal k-response decision query d_k^*. Since I assumed that the agent's askable query set contains only k-response queries, d_k^* is guaranteed to have EVOI at least as high as the EVOI-optimal askable query since d_k^* is the EVOI-optimal k-response query (Theorem 4.2), and thus d_k^* represents an ideal query for the agent to ask given its limitation to k-response queries, which the second step of WQP, query projection, attempts to match as closely as possible with an askable query. However, as discussed above, a glaring limitation of WQP is that it selects the wishful query without considering the content of the askable query set (beyond the constraint that it contains only k-response queries). A variety of extensions are possible to address this problem, and I discuss two below.

7.2.2.1 Optimizing k

For cases where the askable query set is impoverished in terms of EVOI (F2 discussed in Chapter 6), one potential way to select a better wishful query to serve as the goal would be to choose a smaller k for the wishful query selection step, since it would typically have lower EVOI (never higher EVOI) than d_k^*. In fact, it is easy to construct examples where WQP – specifically, DEER – can select a better (k-response) askable query by using a smaller k for wishful query selection, and I give one such example next.

> *Consider a case where there are six possible decisions, where u_1, u_2, and u_3 each have $\frac{1}{3}$ probability of being optimal but $\frac{2}{3}$ probability of having extremely negative value, u_4 and u_5 each have $\frac{1}{2}$ probability of having second-best value but $\frac{1}{2}$ probability of having extremely negative value, and u_6 always has extremely negative value. Finally, suppose the askable query set contains two*

trinary-response queries (so here, k=3). The first askable query, q_{126}, is the decision query about u_1, u_2, and u_6. The second askable query, q_{456}, is the decision query about u_4, u_5, and u_6.

Here, q_{456} is the better of the two askable queries because it allows the agent to choose the better of u_4 and u_5 as a function of its response, but note that d_k^* is the decision query asking about u_1, u_2, and u_3, which projects to q_{126}, and as a result DEER selects the wrong askable query. However, if the k used to choose the wishful query is set to 2 instead, DEER selects the right askable query since d_2^* is q_{45}, which projects to q_{456}.

The example just shown illustrates a case where the knowledge obtainable by a k-response askable query set corresponds more closely to the $(k-1)$-response decision query set than the k-response query set, and thus choosing $k-1$ instead of k for the wishful query selection step is beneficial for this case. However, there are also cases where it can be better to choose a *larger* k for the wishful query selection step. For example, it could be that the closest match contained in the askable query set for d_k^* is not only a poor match for d_k^*, but also gives no information about other decisions that could be fruitfully queried about. In such a case, using $k = |U|$ would produce a wishful query asking about all decisions, and could conceivably project to a more valuable askable query that reveals information about decisions other than those queried by d_k^*.

In conclusion, extending WQP to intelligently choose a different k than the one that matches the number of responses for queries in the askable query set can result in fruitful EVOI gains in some cases. Futher investigation into what factors influence which k is best and/or identification of properties of the mapping between k and DEER's selected query EVOI may inform the design of methods, heuristic or exact, for choosing k.

7.2.2.2 Informed Wishful Query Selection

In order to extend DEER to be robust to cases where the askable query set contains no query similar to the decision query d_k^* selected in the first step, but contains queries very similar to other k-response decision queries (F1 in Chapter 6), one approach would be to prune the set of decisions that can be used to construct the wishful query. That is, prune the decision set to ones the askable query set contains informative queries about. In particular, decisions with very high value for some candidate models and very low value for others should be scrutinized, since they can be part of valuable decision queries that need very close matches in the askable query set to achieve similar EVOI.

One heuristic method for pruning such decisions could be to prune any decisions whose posterior expected value remains similar across all askable query responses, according to some measure of similarity, perhaps involving a threshold parameter. Computationally, this requires only value compuations as opposed to optimal planning computations, provided only a subset of decisions are scrutinized. One potentially promising subset of decisions to scrutinize would be those composing d_k^*, and wishful query selection could be repeated, at each step only considering decisions that have yet to be pruned, and terminated once none of the k decisions composing the current wishful query can be pruned.

7.2.3 EVOI-Sufficiency

In Chapter 4 I studied the problem of how the agent can narrow the space of queries it should consider from the general k-response query set to a subset, when the goal is to select a query on the basis of EVOI only. First I considered a myopic setting where the agent can only ask a single k-response query, and proved that the agent can restrict attention to k-response decision queries at no EVOI-loss, i.e., the set of k-response decision queries is EVOI-sufficient. Then I considered a nonmyopic setting where the agent can dynamically select a sequence of n k-response queries to ask, or equivalently a depth-n k-response query tree, and proved that the agent *cannot* consider only those query trees composed entirely of k-response decision queries without risking EVOI-loss. I also proved that the agent *can* consider only those query trees composed entirely of k-response decision-*set* queries without risking EVOI-loss. However, there exist a variety of open questions on the topic of EVOI-sufficiency, and I discuss some of them below.

7.2.3.1 EVOI-Necessity

A natural question to ask in the myopic setting is whether the agent *needs* to consider all k-response decision queries to assuredly find the k-response query with maximum EVOI, provided no additional assumptions are made about the structure of the decision value function or the agent's uncertainty ψ. If the answer to this question turned out to be no, then the agent could consider a different k-response query set than decision queries, which might be easier to search and/or easier to convey to humans. However, I conjecture that the answer to this question is yes, and that it can be proven by showing that for any arbitrary k-response decision query d, there exists a decision set, model space, and form of uncertainty where d has strictly higher EVOI than any other k-response query.

Another question is whether depth-n k-response decision-set query trees are necessary in the same sense just described, but for the nonmyopic setting. I conjecture that the answer

to this question is yes as well, and that a similar proof technique to the one described above could be used to prove the claim.

7.2.3.2 EVOI-Sufficiency in Structured Settings

One or more of the agent's decision set, model space, or form of uncertainty may have potentially exploitable structure. As an example, the model space Ω could be over a space of weight vectors, and the decision value function for a given $\omega \in \Omega$ and decision u could take the form of the weighted combination (prescribed by ω) of features of u. In such a case, are there other EVOI-sufficient k-response query sets besides the ones proven to be EVOI-sufficient in their respective settings in Chapter 4? What about for other types of structure?

7.2.3.3 EVOI-Sufficiency for Selecting Query Sets

Consider a setting where the agent needs to select a *set* consisting of m k-response queries that it will select from, given the agent has limited knowledge of the uncertain decision problem that it will face. For concreteness, assume that the agent begins with a prior ξ over a set Ψ of possible priors ψ each supported by model space Ω, so that the agent explicily represents its uncertainty over the uncertain decision problem it will face. Also, assume that upon selecting a query set Q, the agent immediately observes the uncertain decision problem and can ask a single query from the query set Q it selected before it must make its decision (i.e., after choosing its query set the agent enters the myopic query selection setting). Which *subset* of size m of k-response queries should the agent consider as its query set, with the goal of maximizing the *expected* EVOI of the query it will ask? Solving this query set selection problem could be useful in situations where implementing human-agent interfaces for asking and answering queries is expensive and needs to be limited, or could be useful as a way to save computation in query selection by reducing the set of queries the agent considers.

Questions of EVOI-sufficiency extend to such a setting. Namely, can the agent restrict attention to subsets of k-response decision queries without risk of expected EVOI-loss? The answer would certainly be yes if $m \geq |D_k|$, since that would allow the agent to select any superset of D_k as its query set, which in turn would guarantee that the agent will be able to select the EVOI-optimal k-response query for any of the possible uncertain decision problems it turns out to face. Similarly, the answer would be yes if $m \geq |\Psi|$, since the agent could select the set of decision queries comprised of each of the EVOI-optimal decision queries for each $\psi \in \Psi$. For the general case, however, the answer is unclear. If the answer

is no, the natural question to ask next would be whether the agent can restrict attention to subsets of k-response decision-set queries without risk of expected EVOI-loss.

BIBLIOGRAPHY

Ali E. Abbas. Entropy methods for adaptive utility elicitation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 34(2):169–178, 2004.

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Machine Learning Conference (ICML)*, 2004.

Valentina Bayer-Zubek. Learning diagnostic policies from examples by systematic search. In *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 27–34, 2004.

Richard Bellman. A markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957. ISSN 0022-2518.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1st edition, 1996. ISBN 1886529108.

Edwin V. Bonilla, Shengbo Guo, and Scott Sanner. Gaussian process preference elicitation. In *Proceedings of the Twenty-Fourth Conference on Neural Information Processing Systems (NIPS)*, pages 262–270. 2010a.

Edwin V. Bonilla, Shengbo Guo, and Scott Sanner. Gaussian process preference elicitation. In *Proceedings of the Twenty-Fourth Conference on Neural Information Processing Systems (NIPS)*, pages 262–270, 2010b.

Craig Boutilier, Richard S. Zemel, and Benjamin Marlin. Active collaborative filtering. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 98–106, 2003.

Darius Braziunas. Minimax regret based elicitation of generalized additive utilities. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

Darius Braziunas and Craig Boutilier. Local utility elicitation in GAI models. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 42–49, 2005.

Darius Braziunas and Craig Boutilier. Elicitation of factored utilities. *AI Magazine*, 29(4): 79–92, 2008.

Urszula Chajewska, Daphne Koller, and Ronald Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI)*, pages 363–369, 2000.

Sonia Chernova and Manuela Veloso. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research (JAIR)*, 34:1–25, 2009.

David A. Cohen, Martin C. Cooper, Peter G. Jeavons, and Andrei A. Krokhin. The complexity of soft constraint satisfaction. *Artificial Intelligence*, 170(11):983 – 1016, 2006. ISSN 0004-3702.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal Of Artificial Intelligence Research (JAIR)*, 4:129–145, 1996.

Robert Cohn, Edmund Durfee, and Satinder Singh. Comparing action-query strategies in semi-autonomous agents. In *Proceedings of the Twenty-Fifth Conference on Artificial Intelligence (AAAI)*, pages 1102–1107, 2011.

Robert Cohn, Satinder P. Singh, and Edmund H. Durfee. Characterizing EVOI-sufficient k-response query sets in decision problems. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 131–139, 2014.

Sanjoy Dasgupta. Analysis of a greedy active learning strategy. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *Proceedings of the Seventeenth Conference on Neural Information Processing Systems (NIPS)*, pages 337–344. MIT Press, 2004.

Dearden, Friedman, and Andre. Model based Bayesian exploration. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 150–159, 1999.

Søren L. Dittmer and Finn V. Jensen. Myopic value of information in influence diagrams. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 142–149, 1997.

Michael O. Duff. Design for an optimal probe. In *ICML*, pages 131–138, 2003.

M. E. Dyer and A. M. Frieze. On the complexity of computing the volume of a polyhedron. *SIAM Journal on Computing*, 17(5):967–974, 1988.

Robert M. Fano and W. T. Wintringham. Transmission of information. *Physics Today*, 14: 56, 1961.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research (JAIR)*, 42:427–486, 2011.

Shengbo Guo and Scott Sanner. Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

Kshitij Judah, Alan Fern, and Thomas G. Dietterich. Active imitation learning via reduction to I.I.D. active learning. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 428–437, 2012.

Andreas Krause and Carlos Guestrin. Near-optimal value of information in graphical models. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, July 2005.

Andreas Krause and Carlos Guestrin. Near-optimal observation selection using submodular functions. In *National Conference on Artificial Intelligence (AAAI), Nectar track*, July 2007.

Andreas Krause, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *Journal of Water Resources Planning and Management*, 134(6):516–526, November 2008.

Marek Kuczma. *An Introduction to the Theory of Functional Equations and Inequalities: Cauchy's Equation and Jensen's Inequality*. Springer Science & Business Media, 2009.

Manuel Lopes, Francisco Melo, and Luis Montesano. Active learning for reward estimation in inverse reinforcement learning. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, pages 31–46, 2009.

Andrew McCallum and Kamal Nigam. Employing EM and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 350–358, 1998.

Francisco S. Melo and Manuel Lopes. Multi-class generalized binary search for active inverse reinforcement learning. *CoRR*, abs/1301.5488, 2013.

G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An analysis of approximations for maximizing submodular set functions–I. *Mathematical Programming*, 14(1):265–294, 1978.

Robert D. Nowak. The geometry of generalized binary search. *IEEE Transactions on Information Theory*, 57(12):7893–7906, 2011.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal Of Artificial Intelligence Research (JAIR)*, 2006.

Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2586–2591, 2007.

Burr Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin – Madison, 2009.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

Paolo Viappiani and Craig Boutilier. Optimal Bayesian recommendation sets and myopically optimal choice query sets. In *Proceedings of the Twenty-Fourth Conference on Neural Information Processing Systems (NIPS)*, pages 2352–2360, 2010.

Aaron Wilson, Alan Fern, and Prasad Tadepalli. A Bayesian approach for policy learning from trajectory preference queries. In *Proceedings of the Twenty-Sixth Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1142–1150, 2012.

Tong Zhang and Frank J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.