A Contextual Multipartite Network Approach to Comprehending the Structure of Naval Design

by

Morgan C. Parker

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Naval Architecture and Marine Engineering)
in The University of Michigan
2014

Doctoral Committee:

Associate Professor David J. Singer, Chair
Associate Professor Matthew D. Collette
Professor Mark E. Newman
Professor of Practice Donald C. Winter

An essential concept from the work of the late author and naval architect David K. Brown (Brown, 1997):

The Stages of a Design Project:

1. Enthusiasm
2. Disillusionment
3. Panic
4. The Search for the Guilty
5. The Punishment of the Innocent
6. Praise and Honor for the Non-Participants

To William and Mary, for building the character of the man who built mine.

ACKNOWLEDGEMENTS

Most readers skip the acknowledgments section — seemingly for a good reason. It is usually either sappy gibberish or a mundane listing of everyone the author ever knew. If you dear reader (reading this you have now become one of a very few) are a friend of mine or the NA&ME department at Michigan, this section may be worth your while. Otherwise be forewarned, a sappy listing follows.

Firstly I would like to acknowledge my adviser, mentor, and friend, David Singer. I owe my Ph.D. to Dave's vision, but I am most thankful to him for giving me the opportunity to be myself, and then trusting me to take it. My road lead to some interesting places, including Bulgaria, but I could not have gotten a better education.

That brings me to my second acknowledgment, Donald C. Winter. A man in his position could choose to do almost anything, but he chose to share his experience with students. Any design I am ever a part of will be safer because of Professor Winter.

All of the faculty and staff in the NA&ME department are deserving of thanks. Of the faculty Professors Troesch and Collette deserve special treatment. They are among the best teachers I have ever had. Of the staff I would like to single out Kathy Stolaruk, Kay Drake, Sue Taylor and Kathy Brotchner. As an undergraduate the department felt like a family, and that was because of them.

Dr. Thomas McKenney has been a good man and friend since sophomore year, without the camaraderie I don't think I would have stayed. I owe a great debt to Anthony Daniels, who welcomed me to the world of research and was gracious with is time as I learned the ropes. All the members of the ANCR lab were of immense help, including Dr. Alex Gray, Dr. Nathan Niese, Dr. Justin Gillespie, Dr. Fang Dong, Dr. Brian Cuneo, Dr. Joshua Knight, Dr. Douglas Rigterink, Jason Strickland, Colin Shields, Austin Kana and Michael Sypniewski. Though the work is individual, the inspiration for a Ph.D. is a team effort.

Outside of the NA&ME department I owe special thanks to David Andrews, Rachel Pawling, Alan Brown, Chris McKesson and Kelly Cooper. Professor Andrews and Dr. Pawling provided their experience of design outside of the U.S. Navy, and a great number of discussions besides. Professor Brown has been my mentor whenever and wherever I have been on distant shores. The portion of my thesis that deals with the Taylor series I owe to Professor McKesson, his inspiration carried me a long way. Last but not least Kelly Cooper, her personal contributions to my development are too numerous to list, but foremost she has been the one responsible for bringing all these great people together. She was also the first one to fund me on my journey.

On a personal note I would like to thank my parents, Robert and Josie, and my wife Carly. Their encouragement, support and understanding were a prerequisite for all of this. Finally to my dog Dan-0, though I doubt she'll read this, there is no way to be unhappy around a happy dog.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

LIST OF APPENDICES

Appendix

CHAPTER I

Introduction

1.1 Background and Motivation

Common wisdom in the defense acquisition community states that an acquisition program should be structured around the following three questions, related to product, process and organization (Winter, 2014):

1. What to buy?
2. How to buy it?
3. Who to buy it from?

However, if current programs are an indicator, then answering the fundamental questions often digresses, things do not go according to plan and unexpected or undesirable outcomes result (Work, 2013; GAO, 2002). The structure and challenges of design are a microcosm of acquisition. Leopold's (1975) paper, "Should the Navy Design its own Ships?", discusses historical programmatic shortcomings and highlights a critical point, "Clearly, proper organizational arrangements and communication among organizations involved in the design process are crucial factors which influence strongly the resulting product." The salient observation is that product, process and organization are inextricably linked, and together determine the outcome of a naval design.

The ability of an individual naval engineer to wholly comprehend and manage this combined system was surpassed by 1865 (Brown, 1997). This has not stopped most naval architects from formulating their own favored version of a technical design process, and though there has always been a grudging recognition that external factors play a deterministic role few have attempted to codify them. Andrews (1981) did incorporate them in his variation on the design spiral, a cone, with the addition of constraints pushing on the design from the outside, including "...wider constraints on the environment in which the designer functions." These constraints have always been ill defined, yet "Any discussion of the ship design process which neglects the limitations imposed by the constraints on the designer is unlikely to provide a real framework for designing ships in the future" (Andrews, 1981). A 19^{th} century Controller of the Royal Navy said, "I hardly know of a case in which we have built a ship in the manner we should liked to have built it," and not much has changed (Brown, 1997). This problem is not limited to naval design, "Large-scale engineered systems are more than just a collection of technological artifacts: They are a reflection of the structure, management, procedures, and culture of the engineering organization that created them, and they are also, usually, a reflection of the society in which they were created"(Leveson, 2002).

The discipline that bears the standard for this viewpoint is Systems Engineering (SE), "an approach to creating executable solutions to complex real world problems" (Winter, 2014). Systems Engineering relies heavily on the principles of systems thinking, mainly focusing "...on systems taken as a whole, not on the parts taken separately. It assumes that some properties of systems can only be treated adequately in their entirety, taking into account all facets relating the social to the technical aspects"(Leveson, 2002). Good systems engineering is considered key to success, but it is not a guarantee. Accounting for the complex interactions inherent in systems design does not necessitate understanding them or their implications, either at a micro or emergent level. Systems engineers have framed the problem, but lack the tools to represent and analyze the entire combined system of product, process and organization. The Systems Engineering V is a process that has been developed through experience (i.e. trial and error); when followed the chances of success are increased but not predictable. Experience is highly valued, but avoiding the failures of the past is not the same as predicting the failures of the future.

What is needed is a mathematical framework capable of representing the many different domains of design in context, providing an opportunity for analysis and understanding before commitment. Such a framework must function with only low fidelity inputs, all that are available in early-stage design. A focus on the early-stages is critical because the "wicked" problem in design is requirements elucidation, finding the set of requirements that can yield a materially feasible and affordable solution (Andrews, 2012). This is the most influential stage of design, and understanding the complete problem is necessary for proper requirements elucidation (McKenney, 2013).

Design can be viewed as the act of generating information for decision making. It consists of people using methods and tools to generate and exchange specific pieces of information in a process. The process is part of an approach, such that information collects in context to form sets, from which decisions are made. McKenney (2013) provided much needed clarification to the terminology of design:

- *Design Approach: The overarching guiding principles of a design effort*
- *Design Process: A series of structured steps to implement the design approach*
- *Design Method: The way in which design alternatives are understood, analyzed, and selected for a particular approach and process*
- *Design Tool:* [Supports design methods by providing information for designer decision making, typically by automating mundane tasks]

McKenney notes that over reliance on design tools can lead to misunderstanding the complete design problem. It is a short logical jump to say that understanding design tools can predicate the outcome of requirements elucidation, which predicates the solutions generated, which predicate the success or failure of the design effort, which predicates the success or failure of an acquisition program as a whole. The lowly design variable is transformed into a monumental acquisition decision point by the context of the intermediate process and organizational structure. Comprehending and understanding these contextual relationships is only of value a priori, as lead indicators.

1.2 The Present Study

1.2.1 An Application for Network Theory

It is not a new thought that certain elements of design have more in common with the social sciences than engineering, yet a common framework must adequately represent the social, technical and temporal elements together in a mathematically rigorous way (Andrews, 1981). Network theory, a field largely developed within and for the social sciences featuring current developments from physicists focusing on the temporal and mathematical roots resting with Euler (1741) provides such a basis (Holme and Saramäki, 2012; Euler, 1956). It is of lesser value to the practicing engineer to represent and analyze the conglomerate of product, process and organization when a design has concluded, as the results are already plain to see. Any new representation must be constructable prior to beginning a design effort, meaning only the most basic information, the existence of relationships between problem components, can be counted on. Minimally a network is a set of points joined together by lines, and its mathematical treatment is irrespective of what it represents. The fidelity required to use networks matches that available in early-stage design.

1.2.2 Current Research Scope

Three broad problems have been identified:

- The complexity of naval design has gone beyond the limit of individual comprehension. It spans social and technical domains which must be represented together for context.

- It is ineffective to judge and analyze success at the end of design. Early-stage design is the most influential, and must be evaluated in advance.

- Many failures occur because incomplete information is available when decisions are made. Reliable outcomes depend on knowing whether sufficient information will be available when decisions must be made.

The problems can be restated as three research questions:

1. Can the structure of design (approach, process, methods, tools and organization) be accounted for?

2. Can a design be understood without designing anything?

3. Can the impact and timing of information be understood in advance?

The scope of this thesis is to address these questions by introducing new methods rooted in network theory — a simple and easily comprehended mathematical framework capable of representing design in all its esoteric forms. Addressing these questions represents a paradigm shift away from the classic a posteriori "How do we make this structure better?" to an a priori view, "How will this structure function?" The paradigm shift requires prognostic methods operating with only the most basic information to produce lead indicators. Approximate information becomes available where no information existed before, and consequences can be predicted rather than suffered.

1.2.3 Contributions

This thesis broadens the application of network theory for naval design from the analysis of physical systems to the general structure of design. It is the author's belief that this broader view is only valuable if the multitude of domains in play are represented simultaneously. The local and emergent behavior of design is a function of the context provided by the broader environment. The primary contribution of this thesis is a network structure to represent this context, with supporting contributions of methods for analysis and verification. The specific contributions addressing the research questions are:

1. Can the structure of design (approach, process, methods, tools and organization) be accounted for?
   - Introduction of a contextual multipartite network approach to represent the structure of naval design
   - Application and extension of existing network mathematics to provide meaningful predictive insight using multipartite design networks as inputs
2. Can a design be understood without designing anything?
   - Recognition that algorithms for finding path lengths can be used to quantitatively capture all node to node influence across multipartite design networks
     - Formulation of path influence algorithms and network weighting schemes, showing equivalency with first order Taylor series expansions
     - Introduction of interpretations for path influence results, comparable with a full factorial design of experiments

    - Development of a new metric, Winston centrality, enabling comparisons between path influence and other metrics

3. Can the impact and timing of information be understood in advance?

    - Application of network diffusion to model continuous information flow across a multipartite design network, effectively capturing classic flow problems in a closed form solution
        - Development of metrics to quantify and interpret continuous information flow across multipartite design networks
    - Introduction of a discrete information flow equivalent of the path influence algorithm, with requisite network weighting schemes
        - Verification of the path influence algorithm using discrete event simulation
        - Extension of diffusion metrics for discrete information flows
        - Testing of path influence and other metrics against more realistic discrete event simulations

1.3 Organization of the Thesis

Chapter II discusses the areas of research that are similar or contribute to the present study. Chapter III introduces basic network mathematics used throughout the thesis, and introduces multipartite networks as a structure to represent multiple design domains in context. The construction of two networks used as case studies throughout the thesis are presented as examples. Chapter IV demonstrates that the structure of design formulations alone provides information through network analysis to be useful for the engineer in a new way. The results of analyzing two multipartite networks are discussed, demonstrating not only their coherency but revealing the intent implicit in design formulations. Chapter V introduces network methods for predicting the explicit behavior of design formulations, dynamic network behavior, using fewer function calls than comparable non-network methods. A metric is also introduced to compare static and dynamic methods. Chapter VI is the first step in extending network methods to account for the temporal nature of design. This chapter approaches the problem by modeling the flow of information across a design network using an abstraction of Fick's second law of diffusion, which assumes continuous information flow. Chapter VII uses network methods and discrete event simulation to model the flow of information across a design network, meaning information transfer at discrete points in time. General concepts and metrics from Chapters V and VI are adapted for the discrete case. The results of previous chapters are then tested against those for discrete time. The importance of hierarchy and the multipartite structure for capturing discrete flow behavior is discussed.

CHAPTER II

Related Research

The purpose of this chapter is to present similar and overlapping research to this thesis. This allows the current research to be placed into context and its unique attributes to be better appreciated. The difficulty of searching for similar work in network theory and design is greatly exacerbated by the number of unique fields of study utilizing both, wittingly or unwittingly, with their own terminology and specific applications. This section represents the author's best effort to find and present those topics and references of most import. No synopsis of the general nature and approaches to naval design is provided, but an excellent one can be found in McKenney (2013). Readers unfamiliar with network theory may wish to read Section 3.1, defining the basic terminology, before this chapter.

The first subject of discussion is Design Structure Matrix (DSM) methods, Section 2.1, because they are the closest parallel to the current research. The DSM in its simplest form is actually the adjacency matrix of a network. The sections following will reference DSM methods because many of the fundamental concepts are similar. Section 2.2 introduces the use of networks in ship design, highlighting how ship designers have thought in network terms for a long time but have only recently discovered the underlying powerful mathematics. Section 2.3 then introduces the many varied applications of network like concepts to general design. Section 2.4 discusses design space exploration and design of experiments, highlighting the difference between proactively comprehending structure and retroactively measuring outcomes. A brief summary concludes the chapter.

2.1 Design Structure Matrix Methods

A popular commonality between network theory and the design world are the Design Structure Matrix (DSM) methods pioneered by Steward in the 1960's. His original work appears in a General Electric company document which does not seem to be publicly available, but his subsequent book or IEEE paper are often cited as the origins of DSM, which he expanded upon in later publications (Steward, 1981b,a, 1991, 1993). A symmetric DSM implies either an undirected or bi-directional network, whereas an asymmetric DSM implies a directed network. Similarly, the operations performed on DSMs such as clustering have network equivalents as well (Browning, 2001). Interestingly, little visualization beyond the matrix representation of DSMs appears in the literature until recently, though Steward actually drew networks in his original paper as shunt diagrams when describing the procedure for tearing (Steward, 1981b). The recognition that networks are the foundation of DSMs has only recently been emphasized in the literature. Eppinger and Browning (2012) state that "The DSM is a network modeling tool used to represent the elements comprising a system and their interactions, thereby highlighting the system's architecture (or designed structure)."

Browning identifies four main types of DSMs: system architecture, engineering organization, scheduling and parameter-based. The parameter-based DSM is of particular interest to the current research because it is "Used for modeling low-level relationships between design decisions and parameters, systems of equations, subroutine parameter exchanges, etc"(Browning, 2001). A narrow application of the methods of Chapters III and IV could be viewed as expanding the idea of parameter-based DSMs to entire design formulations, revealing the designer intent which predicates the solutions generated. This was published in Parker and Singer (2013), but the structure is substantially different. DSM research has focused almost exclusively on system architecture, organization and scheduling DSMs, leaving parameter-based DSM methods an attractive area for new research (Yassine and Braha, 2003; Yassine et al., 1999; Sosa et al., 2007b; Kerns, 2011; Guenov and Barker, 2005; Eppinger and Rowles, 2000; Doerry, 2009; Cooper et al., 2011; Black et al., 1990). If network methods were limited exclusively to design tools, then parameter-based DSMs are the closest comparison.

A more recent definition of DSMs reduces their primary types to Product Architecture, Process Architecture, Organization Architecture and combinations of them, termed Multidomain Architectures. Parameter-based DSMs are demoted to a subtype of process architecture (Eppinger and Browning, 2012). The new Multidomain Architecture definition is an integration of the other three types into a larger structure through some variation of "domain mapping," yielding work like that of Bartolomei (2007); Maurer (2007); Danilovic and Browning (2007); Eppinger and Browning (2012). Bartolomei's Engineering Systems Matrix (ESM) relates objects or physical artifacts of the system to other levels, such as political requirements, which are originally represented in separate matrices. The ESM requires an existing or template design environment from which to build. In his thesis, Bartolomei's ESM was constructed after the design process was complete, rather than being predictive. The main contributions were to provide a structure for representing an entire engineering system in matrix form, and a methodology to populate the matrix. Maurer has a similar thesis that predates Bartolomei's, but is focused specifically on managing complexity in design, describing a multiple-domain matrix (MDM) which is similar in function to Bartolomei's ESM (Maurer, 2007). Maurer focuses much more on analyzing the structure of the dependencies modeled, noting that, "...many approaches face the challenge of managing complexity in product development, but only a few focus directly on the structures implied in the system considered" and "A methodology focusing on the consideration of structures in product development seems to be promising as an approach for enhancing the possibilities of the analysis, control, and optimization of complex design"(Maurer, 2007). This thesis is in agreement with the first statement, and largely with the second. However, enhancing the control and optimization of complex design should only be attempted if it is well understood in the first place. DSM research and methods are largely motivated under the old paradigm, "How do we make this structure better," rather than the paradigm this thesis promotes, "How will this structure function."

A pure DSM usually relates the elements of one domain to itself. The evolution of the MDM or ESM allows the representation of links between different domains, but work in this area is still tied to a matrix representation and matrix mathematics. DSM researchers have tried to incorporate more information into a visual matrix structure, the result is matrices that may not be amenable to mathematical analysis because multiple pieces of information reside within a cell (Kreimeyer et al., 2008; Yassine et al., 2003a). To address this issue Kreimeyer et al. (2008) proposed another modification to the MDM, demonstrating in matrix form a relationship between elements in domain A & B through C, where C is conceivably a type of relationship. This splits what was once two pieces of information in one cell into a matrix math capable representation. This came at the cost of adding a new domain containing C to the MDM. However, the structure of this expanded MDM inadvertently created relationships that did not exist. The solution to this problem was to add yet another domain to the MDM, further complicating the interpretation. The DSM approach to handling multiple domains is motivated by a belief that only homogenous matrices are coherent. "From a structural point of view, a system can be disentangled down to a network-like model of entities and their relations. These entities can be of different kinds, e.g. some entities in a process can be documents, while others can be oral information, and other again can be work packages. However, if many such kinds are mixed, the network is incoherent"(Biedermann et al., 2013). The major divergence of this thesis from DSM work is that multiple domains are represented in a single network, and that this is, in fact, critical to understanding design as a whole. This thought is developed in Chapter III as multipartite networks, a much simpler and intuitive multidomain representation. This thesis can be considered an independent response to Bartolomei's (2007)'s question, "to what extent [are] existing social network measures applicable when analyzing a heterogeneous network with components from multiple domains?"

Applying MDMs of Kreimeyer et al.'s (2008)'s variety to larger systems yields an exponential growth in the size of the matrix; common sense suggests there is a limit to the amount of information that can be legibly presented or interpreted in matrix form. Kreimeyer et al. (2008) also identified this shortcoming anecdotally, but experimental results are mixed depending on which type of information is being sought. Generally networks that are dense (meaning many edges) have been found easier to understand with DSMs, while sparser networks are better represented as graphs (a synonym for network) (Keller et al., 2006). Situations where the relative position of nodes, or where path tracing is important are better represented with graphs (Keller et al., 2006). In terms of populating the nodes and edges/arcs of a network, the experience of ship designers leans towards a graph representation (Cooper et al., 2011). As mentioned previously, a DSM is actually the adjacency matrix of a network, so either representation can be converted to the other as the need arises.

In summary, DSM methods can be considered a design-specific mathematical subset of existing network theory. DSMs have not been found that metricize the role of a complete design formulation's internal structure in affecting the outcome of product development, either narrowly as parameter-based or more broadly. They have typically been applied to model product, process, or organization separately, though some success has been found through domain mapping to merge different types of DSMs into one coherent system model. Other matrix-related design tools such as the House of Quality and N^2 diagrams share similar bounds to those discussed for DSMs, and will not be specifically addressed. An excellent overview of current matrix methods to manage design complexity and their relationship to graph theory can be found in Lindemann et al. (2009).

2.2 Networks in Ship Design

The idea of representing a ship design through networks is not new. In many cases the representation of the design is in the form of constraints or variable interactions, which are shown as networks, though networks are not mentioned explicitly (Brown, 1986; Watson, 1962; Watson and Gilfillan, 1977). A good example is that of Brown (1993b), who used the term "mesh" to describe ship design in general. There are cases where networks are mentioned directly, but more often the term is used in association with design activities, in some way relating or contrasting with the design spiral (Laverghetta and Brown, 1999; Cooper et al., 2011). Cooper et al. (2011) describe a large Navy effort to capture the ship design process, primarily using the commercial software *Plexus*. They started their effort using DSMs, but found network representations easier to populate. Practitioners described the naval ship design process in several workshops, creating the network and making it the only (and therefore most accurate) representation available. Their nodes are design activities, whose basic relations are defined by a product model. In essence, their model starts with a notion of the product and works backwards. The current research starts with fundamental relationships within design formulations before any notion of the product exists, more amenable to revolutionary design concepts. Similar in nature to Cooper et al.'s (2011) work is the Design Building Block (DBB) approach introduced at IMDC in 1997 and championed in many other works by Andrews (Andrews, 2006). It is an academic vision of an ideal preliminary ship design process, although he does not use the term Andrews' illustration of the approach can be interpreted as a directed network of design activities. The DBB approach has been implemented in software form as a module of *Paramarine*, with each of the design activities corresponding to analysis tools, but novice users of the software lack an understanding of the fundamental impact of model structure on the outputs. In summary, beyond the occasional graphical representation of a network or a lead in to the DSM methods introduced in Section 2.1, network concepts in ship design have not been further developed until recently with one outlying exception.

MacCallum (1982) produced a paper for the first IMDC which could be considered a seminal work, though it has received little attention. With regards to design tools, "The designer is restricted to the methods used by available programs and very limited facilities are given to the designer to allow him to set up his own tests and evaluations. Thus one of the key features of creative design is lost"(MacCallum, 1982). In other words, the choice of a formulation predicates an outcome. The title of the paper "Understanding Relationships in Marine Design" hints at the intent of MacCallum to better inform the designer about the formulations they use rather than make a better tool. The dependencies between the parameters of a ship design method were drawn as a directed network, where the arcs connecting them represented quantitative functional relationships. This enabled the network to be coded as a design space exploration tool, where the "strength" of the arcs were generated to represent the influence of characteristics over one another. The directed network was fully recognized for what it was, but only as a visual. The underlying mathematics were not network based. The paper was expanded upon by Whitfield et al. (2003), adding a parameter-based DSM with the added wrinkle that both indirect and direct dependencies could be represented. A genetic algorithm (an optimizer) was run on the DSM to minimize the feedback loops. This is more like a process DSM and diverges from the original paper and the current research by seeking to improve rather than fundamentally understand the formulation. Chapter V of this thesis introduces a concept similar to MacCallum's arc "strength," though a network algorithm is at the core.

Justin Gillespie proved the applicability of networks to ship design layout problems in his thesis and other works by adapting the extensive constraint library of the Intelligent Ship Arrangements (ISA) software program into a network representation. ISA is a ship general arrangement tool that produces rational space allocation and arrangements for designer review (Parsons et al., 2008). Explicitly demonstrated in Gillespie's work is how a network approach to a product layout yielded innovative results not seen using other methods. He used computationally simple network mathematics to generate significant designer insight in a way ISA was not enabled to do, and generated rational layouts in a fraction of the time (Gillespie et al., 2010, 2011; Gillespie and Singer, 2011; Gillespie, 2012). The network methods provided different solution mechanisms to the same problem while generating insight into how the tool functioned, similar in concept to MacCallum (1982).

Of particular relevance to this research, Gillespie's network generated designs were baselined against ISA. In the process of doing so, he discovered designer intent implicit within ISA. This discovery demonstrates that tool structure can have an affect on the outcome of product development, and network theory can identify it. The current research diverges from Gillespie's work in that he focused on product structure (creating designs) rather than formulation structure (creating information). Network research into the development and understanding of physical product structures has continued with other affiliated researchers (Rigterink et al., 2013).

Capitalizing on Gillespie's insights, Parker and Singer (2013) conducted a case study verifying that a network representation of a ship design tool was feasible, and introduced a directed multipartite formulation suitable for representing such tools.

Standard network metrics were applied to the case study, concluding that network analysis can correctly identify what naval architects should intuitively understand about a design tool, identifying design drivers, constraints and other structural features. This work is included in Chapter IV, but with a broader focus on formulations rather than tools.

2.3 Networks in Design Generally

Network drawings and the mathematics underpinning them have no specific tailoring to any particular field or application. Researchers and practitioners have been using them under many different names for quite some time. The following sections highlight some of the more relevant examples.

2.3.1 Design Information Flow and Timing

Steward's (1981b) original Design Structure Matrix work aimed to show how information flowed during design among other things. Design can be defined as the act of generating information used for decision making, so almost every topic in this chapter relates in some way to information flow. Chapter VI introduces a network model of continuous design information flow as function of time that has not been found elsewhere, but discrete information flow is more realistic and much more complicated to represent. Baldwin et al. (1999) created an extensive data flow diagram of a construction project that was transformed into a process DSM for standard analysis, but the real contribution was a three-phase discrete event simulation. The simulation was motivated by the need to measure the impact of missing information, assumed information, and the timing of information availability. Baldwin et al. (1999) claims that techniques other than discrete event simulation are incapable of producing this information, and in the narrowest temporal sense they may be right. However, there are researchers that have produced similar results without discrete event simulation.

Yassine et al. (2003b) provides a good example of this, using a combination of DSMs and state space models to account for the asynchronous timing that results in "design churn." Smith and Eppinger (1997a) presents a model to account for and estimate the duration of iterations within coupled design tasks using a weighted abstraction of a DSM, a dynamical system analogy where the Eigen values and vectors are the primary output.

Research into the timing of design information such as Baldwin et al.'s (1999), Yassine et al.'s (2003b) or Smith and Eppinger's (1997a) are worthy of their own, if not multiple theses. A member of the present committee cautioned about over ambition on this front, a caution well justified. Baldwin et al. (1999) sought to create an extensive generic process model for construction projects, but recognized that every project would require changes to the model. This is representative of all three methods — they are homogenous in nature and require significant specialized definition and setup. This thesis advocates heterogeneous networks and lead indicators, meaning a minimalist setup, and no improved process or structure is advocated. Chapter VI introduces a continuous information flow model, and Chapter VII a discrete one. The distinguishing point about these methods is not their capability to measure the effect of timing in design, but that they operate on the same heterogeneous network structure that is the basis of all the methods in this thesis. A representative network is created one time, and built upon to produce multiple different lead indicators.

2.3.2 Information Flow in Other Fields

Information flow means many different things, and network methods have been developed in other fields, especially software and telecommunications, to solve specific problems. A good example is Ahlswede et al. (2000), who used the concepts of maximum flow and minimum cut set to determine the admissible coding rate region between multiple independent sources and a receiver. Daly and Haahr (2009) used network measures to identify improved data routing schemes for mobile networks where nodes were not continuously connected. Both papers used homogenous networks, but the mathematics presented could provide useful analogies in the future study of design networks.

2.3.3 Statistical Properties of Design Networks

The statistical properties of large scale product development networks have been studied using network theory. Braha and Bar-Yam (2004b) used large real world homogenous product development networks to develop insights about degree distributions, number of connections a node has, compared with other product development networks. Though they studied homogenous networks the potential importance of interplay between domains was recognized, "...there is a strong association between the information flows underlying the PD task network and the design network composed of the physical (or logical) components of the product and interfaces between them." The statistical properties of networks are not in the scope of this thesis, but larger heterogeneous networks could be studied in the same way. Batallas and Yassine (2006) built on the work of Braha and Bar-Yam (2004b,a) by adding DSM methods and applying other network metrics to identify critical nodes that broker information in the process. A mega team of these highly connected nodes is recommended to be formed to improve information exchange and knowledge retention over multiple projects. The applicability of network theory to study large organizations is supported, noting that the number of people involved in the development of the Boeing 777 was on the order of 17,000. The work is insightful, but like most DSM applications it uses a homogenous network. The concept of brokerage is represented differently in Chapter III using nodes providing context between different levels of a heterogeneous network.

2.3.4 Change Propagation

Change propagation data from large design activities has been studied extensively using network methods. Pasqual and de Weck (2012) and Pasqual (2010) analyzed the relationships in an organization between people, design teams and design artifacts to metricize those that absorb change, multiply change, carry change, are receptive to change, resist change, etc. Their data set included the outcome of engineers' individual change requests, so they then inferred a metric which evaluated an individual engineers' innovative capability. "By contrast, the 10% of engineers with $R_{PAR} < 1$ struggled to get changes accepted by relatively receptive areas. These engineers may not be quite as innovative or systems savvy, and might benefit from additional training"(Pasqual and de Weck, 2012). In this author's opinion, it is not good science to infer a specific individual's complex sociological status from a metric derived statistically from a large population, especially when the data set involved was not collecting sociological information. This highlights the danger in applying what is commonly thought of as a social science to engineering applications; the mathematics translate easily but the meaning of the results do not. A network measure must be put into context with the network it is used to describe. This being excepted, their work is very insightful in how networks can be used to analyze change propagation, and many of their created metrics are promising. Change propagation research helps inform the current research by demonstrating the application of networks to new fields, and provides examples for deriving custom metrics. However, it is essentially product or organization focused and is not predictive. "...It is unclear (and not within the scope of this paper) whether sufficient data would have been available to reveal any actionable trends in real time"(Pasqual and de Weck, 2012). Where change propagation research analyzes static product or organization networks after design completion, the current research analyzes networks to understand how technical, process and organizational structure together affect future designs.

2.3.5 Modularity

Modularity is actually a network metric itself, being used to describe the extent to which a network is divided into minimally interacting groups of like nodes. It is only natural then to try and apply network concepts to the growing interest in product modularity and architecture. Sosa et al. (2007a) provide a very good discussion of this subject in terms of component modularity, with great emphasis placed on the varying dependencies between different physical components. They also focus on the cascading of dependencies, which shares roots with the change propagation research mentioned earlier. To capture the impact of dependencies on the product level, they "...embed product-level requirements within "virtual" physical elements of the product and treat these as any other physical product components"(Sosa et al., 2007a). They include multiple types of dependencies (spatial, structural, material, energy, information) by creating a component DSM for each one, and apply a variety of modularity metrics to each dependency type. Their work again highlights the primary difference between DSM and the current research, homogenous vs. multipartite networks. They create different networks for different dependency types, but all the nodes in these networks are of the same type, a physical component.

2.3.6 Product Architecture

Wyatt et al. (2011) gave a more general discussion of product architecture, once again referencing physical system layouts, and use some network concepts to create a computational tool to aid in architecture design. Both Wyatt et al.'s (2011) and Sosa et al.'s (2007a) work provides meaningful demonstrations of network applicability to design generally, but both are limited to the study of physical products . In that sense, there is much in common with Gillespie's work. The current research diverges primarily on this point, as no physical product is being modeled or analyzed.

2.3.7 Bayesian & Neural Networks

When searching for literature involving networks and design, Bayesian and neural network literature dominates the results. A concise overview of both with applications to ship design was written by Clausen et al. (2001). The paper compares regression, Bayesian and neural network models for the determination of principal dimensions. The major distinction between Bayesian and neural network methods and the current research is that they are primarily concerned with taking a valued data set and predicting or calculating a valued output through the construction and tuning of a network. In Clausen et al. (2001), the edges and arcs were statistically discovered from a large as-built data set, rather than being specified ahead of time. A more rigorous mathematical description of the capabilities of Bayesian networks was shown by Shahan and Seepersad (2012), who used simulation rather than historical data to tune their network. Their work further highlights that Bayesian network methods are envisioned to be used as a design process, by demonstrating how the work can satisfy some of the principles of set-based design as defined by Sobek et al. (1999).

Neural and Bayesian network methods are often used to mathematically construct a design formulation rather than decode one. To construct such networks, a designer must select appropriate inputs and outputs, implying a priori knowledge about the existence and appropriateness of links between them. The current research is concerned with generating (and predicting) that knowledge based on fundamental relationships within existing formulations. An example of how the two methods could work together would be to use the current research to better formulate the inputs and outputs to a Bayesian or neural network using a common set of nodes, noting the existence of particularly strong relationships. Once the Bayesian or neural network model is created, its statistically derived relationships could be verified against those noted previously. If the results agreed, the designer has more confidence that the formulation will utilize relevant information and behave as expected.

2.3.8 Product Optimization & Design Problems

So far, it has been shown that network analysis has been used very broadly in the description and analysis of design processes, organizations and products. There is another area also being explored, where network analysis is a major part of solving well formulated design problems. Devendorf et al. (2010) used a 16 variable continuous multi-objective optimization problem for a case study . The variables were divided among five designers with individual objective functions. The convergence time, or transient response, for solving each variable was recorded. Then network analysis, mostly centrality, was used to identify and rearrange the solution process to reduce the overall convergence time. DSM's are used as the first visualization, with actual network representations also being shown. This work could be considered an example of a parameter-based DSM, with some network metrics being used to further optimize the solution process. Subsequent work by the same authors on the same subject (distributed design process architecture) dropped all mention of network theory (Devendorf and Lewis, 2011).

The concept of using networks to visualize and aid in problem solving is not new, networks being a common representation of linear programming problems (Bertsimas and Tsitsiklis, 1997). Michelena and Papalambros (1995) used some network reliability concepts to optimally decompose multi-objective structured partitioned optimization problems. Along similar lines, Shai (2003) transferred an engineering problem into a graph theory representation, which can then either be solved directly or transformed into a form with a known solution process in another engineering discipline.

These works are used different types of network theory to directly solve product focused engineering design problems, or aid the process of doing so. The current research is operating on a macro level comparatively, analyzing the structure of the formulations used to generate these specific problems.

2.4 Design Space Exploration and Design of Experiments

Design space exploration is similar to Design of Experiments in that the primary purpose is to correlate valued inputs to valued outputs. When studying well posed design problems, design space exploration is often used to correlate an outputted feasible region to inputs, without necessarily understanding the "how" of the link between the two. This is evident from the use of Pareto fronts, meta-models, response surfaces, etc. These results are lag indicators, meaning that they are generated post process. An appropriate analogy is the marionette. Design space exploration correlates the movements of the puppet with the position of the control bar, while the strings between puppet and control bar are left unresolved. Deb et al. (2014) diverges slightly from this definition by trying to reverse engineer the relationships between problem components, solving for a Pareto front and then creatively examining the variables that define the optimums. This approach still qualifies as a lag indicator, watching the puppet show and then trying to figure out how the strings are connected. Design of Experiments (DOE) is a set of statistical methods in which an engineer or scientist can quantify and understand the errors and inferences resulting from an experiment. A DOE can be used to quantify and increase the accuracy of the correlations found in design space exploration by guiding the sampling of the design space. An example of a more typical implementation that mixes design space exploration, DOEs, optimization and meta-models can be found in Diez et al. (2013). The current research is not focused on lag indicators, but using fundamental structure to generate preprocess or lead indicators. The analogy is to look at the strings before the show, knowing in advance what the marionette can and cannot do. Chapter V introduces a network version of design space exploration, but utilizing the fundamental structure. A DOE is used to validate the lead indicators produced by the network method, but overall the current research is inherently preprocess or lead indicator focused.

2.5 Summary

The purpose of this chapter was to give a general sense of similar and overlapping research to that of this thesis. This allows the current research to be placed into context and its unique attributes better appreciated. There has been significant research, some of it network related, into the structure of design products, processes, and organizations, as well as analogous network research in other fields. The distinguishing characteristics of this thesis include a focus on understanding rather than improvement, recognizing that the structure of a formulation will have an impact on the outcome. The second distinguishing feature is that a design's logical structure, process and organization can be represented and analyzed together in a heterogeneous network. Symbolically, if the goal of design is to put an arrow in the bullseye, then process research investigates the ballistics of the arrow through its flight, accounting for wind, range, angle and power. Organizational research tries to place the archer in the best location to make the shot and product research has made the bullseye as large as possible and optimized the bow and arrow themselves. But what has gone largely unaddressed is that it is the complete integration of archer, bow, arrow and environment which ultimately predicates the possibility of success or failure.

CHAPTER III

Network Methods & Construction

This chapter introduces network terminology and mathematics common to the remaining chapters with Section 3.1, and then discusses the creation of multipartite ship design networks in Section 3.2. Representing design and acquisition structures as context based multipartite networks is one of the unique contributions of this work, and is fundamental to the analysis methods discussed in subsequent chapters. The specific construction of two multipartite ship design networks is discussed in Sections 3.3 and 3.4, both of which are used as test cases in later analysis.

3.1 Terminology and Essential Mathematics

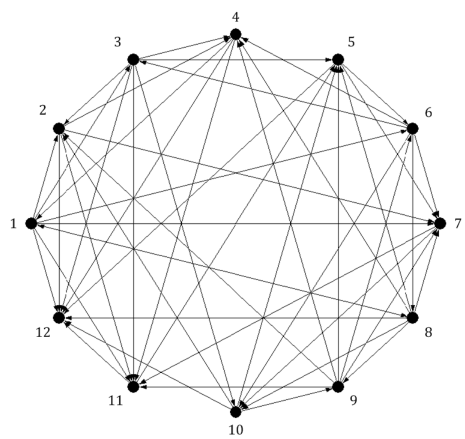
Network theory and graph theory are essentially the same, with mathematicians typically preferring graph theory terminology and the social sciences preferring the network equivalents. A network is defined as a finite set of n elements called *nodes* and a set of m lines that connect pairs of nodes. If a line has a direction, it is referred to as an *arc*, if it is directionless, or bidirectional, it can be referred to as an *edge*. Typically, a network containing edges contains no arcs, and vice versa. A network containing only arcs is called a directed graph, digraph, or *directed network*. This research makes extensive use of directed networks, and unless otherwise noted a directed network is assumed.

3.1.1 The Adjacency Matrix

The *adjacency matrix*, $\mathbf{A}$, is an $n \times n$ matrix representing the m arcs in the network. Entry $\mathbf{A}_{ij}$ of the adjacency matrix represents the arc running from node i to node j. This notation is not consistent across the literature. The notation adopted here is consistent with very common Design Structure Matrix (DSM) literature (Browning, 2001), but inverted from the main network references this research draws upon (Gillespie, 2012; Newman, 2010). It is more natural to the naval designer to say i influences j, rather than j is influenced by i. Mathematically there is no difference; converting between notations requires merely the transpose of the adjacency matrix. In an undirected network, the adjacency matrix is symmetric as each edge is bidirectional thus $\mathbf{A}_{ij} = \mathbf{A}_{ji}$. In an unweighted network, i.e. each arc is of equal importance, the existence of an arc from i to j is denoted by a 1 in the adjacency matrix. In a weighted network, the existence of an arc is signified by a non-zero value denoting the arc weight. Some types of networks allow for a self edge or arc, $\mathbf{A}_{ii} \neq 0$. This research does not require the existence of self arcs so the entries along the diagonal are equal to 0. In DSM visualizations, the diagonal elements are often presented with no value, but are shaded in. A simple directed network and its adjacency matrix are shown in Figure 3.1. This is a digraph of the 2011 NCAA college football schedule within the Big 10 conference. Each arc is a game played, with the arrow pointing towards the loser, i.e. Michigan was victorious over Ohio State.

3.1.2 Centrality

The most basic network metric is *degree centrality*. The degree of a node is the total number of edges or arcs connected to it. A directed network requires more specificity, where *in-degree* is the total number of arcs directed at the node, and *out-degree* is the total number of arcs the node is directing. These metrics can be calculated by summing over the columns or rows of the adjacency matrix respectively,



(a) 2011 Big 10 Football Network

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

(b) 2011 Big 10 Football Adjacency Matrix

Figure 3.1: 2011 Big 10 Football

yielding vectors $\mathbf{k}_{in}$ and $\mathbf{k}_{out}$ containing an entry for each node.

Degree centrality evaluates nodes as if they exist in isolation, or can be decoupled somehow from the network. Though informative, a lot of information contained in the network is not represented using only degree centrality. In naval design, considering only degree centrality might show the direct importance of one aspect of design, but neglects the indirect influences that cause cascading changes. Park and Newman developed a relatively new measure to college football, referred to herein as *Park centrality*, which takes into account the relationship of each node to every other node (Park and Newman, 2005). This is one creative way of addressing the limitations of degree centrality. The idea is that a node's ranking is increased from each node it directly influences (out-degree), and a discounted increase for each node that the influenced node influences and so on. At the same time, a node receives a decrease in rank for each node that influences it (in-degree), and a discounted decrease for each node that influences the influencing node and so on. Though the application is new, Park centrality is actually a generalization of *Katz centrality* as shown in Eq. 3.1 where $\mathbf{w}$ and $\mathbf{l}$ are the win (influencing) and loss (influenced) ranking vectors respectively. Subtracting the loss ranking from the win ranking yields overall Park centrality as shown in Eq. 3.2.

$$\mathbf{w} = \mathbf{k}_{out} + \alpha \mathbf{A}\mathbf{w} \quad \rightarrow \quad \mathbf{w} = (\mathbf{I} - \alpha \mathbf{A})^{-1}\mathbf{k}_{out} \tag{3.1}$$
$$\mathbf{l} = \mathbf{k}_{in} + \alpha \mathbf{A}^T\mathbf{l} \quad \rightarrow \quad \mathbf{l} = (\mathbf{I} - \alpha \mathbf{A}^T)^{-1}\mathbf{k}_{in}$$

$$\text{Park Centrality} = \mathbf{w} - \mathbf{l} \tag{3.2}$$

The parameter α is the discount factor, the weighting desired for indirect wins/losses. However, α is limited to $\alpha < \lambda_{max}^{-1}$ if the result is to converge, where λ_{max} is the largest eigenvalue of $\mathbf{A}$. In cases where the exact final eigenvalues are not fully known, i.e. halfway through a season, a reasonable bound for α can be derived from an equivalent randomly generated network. This yields the expression for α shown in Eq. 3.3.

$$\alpha = \frac{\langle 2k \rangle}{\langle k^2 \rangle - \langle k \rangle} \tag{3.3}$$

The full derivation can be found in Park and Newman's work, though their notation is different, requiring an opposite placement of the matrix transposition (Park and Newman, 2005). As shown in Table 3.1, this ranking scheme accurately reflects what was generally perceived as the Big 10 standings at the end of the 2011 season. In analysis of the entire college football season for the same year, this simple ranking scheme had the same accuracy as the AP Top 25 (88%), and took only seconds to calculate. This accuracy is on par with Park and Newman's results from an earlier season (Park and Newman, 2005).

Another common measure of centrality is *betweenness centrality*. Degree centrality and Katz centrality are measures of a nodes direct and indirect impacts on a network from the standpoint that flow from the network is either emanating from, or terminating at the node being evaluated. Consider these egotistical measures of centrality.

Betweenness centrality is a measure of a nodes impact on flow between other nodes in the network, hence betweenness. In naval design formulations, betweenness is one way of representing how important a node is for transferring information between other aspects which are not directly connected. As an example, engine rpm is only connected to speed by acting through gear ratios in most naval vessels, showing the importance of gear ratios in the selection of a prime mover. A design model which yields no betweenness for gear ratios might be assuming a directly connected two-stroke diesel. A frigate designer would want to follow up on that. Betweenness can be calculated by quantifying the number of geodesic paths (shortest paths) between all other nodes in the network that pass through the node of interest. Newman's general definition of betweenness centrality is shown in Eq. 3.4 (Newman, 2010).

$$x_i = \sum_{st} \frac{n^i_{st}}{g_{st}} \tag{3.4}$$

In Eq. 3.4, x_i is the betweenness score of node i, s and t are the index values for all other nodes in the network, n^i_{st} is the number of geodesic paths from s to t that pass through i, and g_{st} is the total number of geodesic paths between s and t.

There are a few variations on how to calculate betweenness which may change the magnitude of the metric, but not the ranking of nodes relative to one another. Some measures include the reference node in the summation of geodesic paths, meaning a connected node will never have zero betweenness. This work does not use this definition, meaning s or t cannot be equal to i, and connected nodes can have zero betweenness in certain circumstances. Several examples will be discussed in later sections. The betweenness values shown in this work are computed by the software *Pajek* , and most are then normalized over the lowest nonzero result (de Nooy et al., 2005). Normalizing emphasizes that betweenness centrality is important as a comparison across nodes, not necessarily as an individual attribute which may be important for other applications. Though nodes are used as the example, betweenness can also be calculated for edges or arcs.

Betweenness results from the Big 10 2011 season show that Indiana has a score of zero, this is expected because Indiana did not win a single conference game. All arcs connected to Indiana terminate at Indiana so there are no paths connecting other teams running through Indiana. All of the centrality measures discussed for the 2011 Big 10 Conference network are shown in Table 3.1.

Table 3.1: 2011 Big 10 Football Centrality Measures

Node #	Team	Out-Degree	In-Degree	w	l	Park	Betweenness
1	Wisconsin	7	2	48.54	11.79	36.75	0.61
2	Purdue	4	4	13.85	20.28	-6.43	0.20
3	Penn St.	6	2	27.96	10.37	17.59	0.16
4	Ohio St.	3	5	18.40	24.78	-6.37	0.55
5	Northwestern	3	5	17.21	31.38	-14.17	0.82
6	Nebraska	5	3	40.85	18.29	22.56	1.00
7	Minnesota	2	6	10.19	34.91	-24.72	0.36
8	Michigan St.	7	2	49.56	10.37	39.18	0.33
9	Michigan	6	2	35.87	11.73	24.14	0.32
10	Iowa	4	4	22.64	24.59	-1.95	0.70
11	Indiana	0	8	0.00	55.68	-55.68	0.00
12	Illinois	2	6	6.79	37.70	-30.91	0.26

3.1.3 Similarity

Centrality measures can be somewhat anonymous. For instance, the fact that Penn State and Michigan have identical in-degree and out-degree values does not provide any information about which teams were played to yield this record. It is obvious that there were differences, as the Park and betweenness centrality measures are quite different. Direct observation of the adjacency matrix can reveal the specifics, but that information can also be encoded in a *similarity* measure. Structural equivalence is one type of similarity; two nodes are structurally equivalent if they have the exact same set of relationships to the exact same set of other nodes. For example, if Penn State and Michigan played the exact same teams during the season and the results of those games was the same they would be structurally equivalent. In naval design, structural equivalence could mean that two nodes are involved in the exact same set of calculations. This could signify a redundant unnecessary variable, or two variables which should belong to the same discipline etc. One way to calculate structural equivalence is *cosine similarity*. The principle is the same as that of a vector dot product. If the dot product between two rows or columns of the adjacency matrix is 1, then the corresponding nodes are structurally equivalent for an undirected network. For directed networks, the cosine similarity for in-degree and out-degree must be calculated separately as either the column or row dot products, respectively. For unweighted networks, all entries in **A** are either 1 or 0, out-degree cosine similarity can be computed as shown in Eq. 3.5.

$$\sigma_{out}^{ij} = \frac{\mathbf{A_i} \bullet \mathbf{A_j}}{\sqrt{k_{out}^{i} k_{out}^{j}}} \tag{3.5}$$

In Eq. 3.5, σ_{out}^{ij} is the out-degree similarity, $\mathbf{A_i}$ is the i^{th} row of $\mathbf{A}$ and k_{out}^{i} is the out-degree of node i. In-degree cosine similarity is computed in the same fashion using the columns of **A** rather than the rows.

3.2 Multipartite Networks

3.2.1 Design Context and a Multipartite Definition

This research uses a single network with multiple node types based on a hypothesis that in complex product design, elements of a domain do not directly influence one another, they must have context provided by another domain. As an example, variables within a design tool do not directly influence each other, they must have the context provided by a mathematical function. In naval architecture, length (L) alone has no bearing on longitudinal strength, it is its relation to depth (D) that is commonly used in early design. The L/D ratio is a function through which length and depth relate. Again, when formulating a network an engineer might conclude that length influences beam (B) for powering reasons, but that influence is routed through the L/B ratio, a function which provides context. This insight leads one toward a specific network structure, a *multipartite* network. A network is called r-partite (or multipartite) if it is partitioned into r classes such that every arc or edge has its ends in different classes: nodes in the same partition cannot be adjacent (Diestel, 2005). Multipartite thinking is not new to engineering, but has been ill defined and narrowly applied (Kreimeyer et al., 2008). The multipartite networks presented in Sections 3.3 and 3.4 accurately represent ship design methods, and as used in later chapters they can also represent processes and organizations, in hierarchy, context and fidelity. The author argues that the multipartite representation is an accurate reflection of how designers think. Thus, a multipartite representation is one possible way of increasing a designer's understanding of the methods they use and the processes and organizations they are a part of.

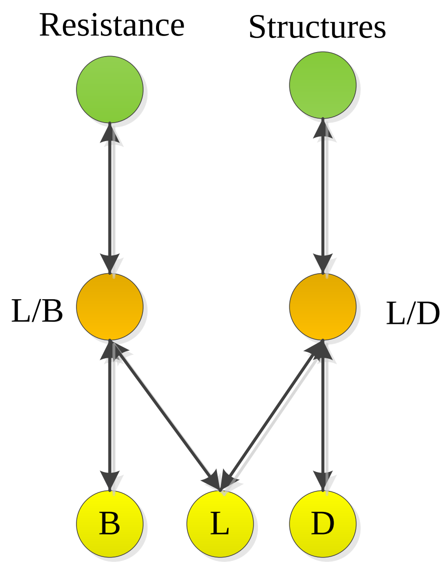
3.2.2 Contrast with Similar Methods

A multipartite structure can be projected into a homogeneous network, called a one mode projection. In standard network theory multipartite networks are typically used to show a node's membership in a group. All nodes belonging to the same group will have an edge to the same node of another type which defines the group. In this form, an incidence matrix substitutes for the role of an adjacency matrix, and is defined with dimension $g \times n$ in Newman's notation, where g is the number of groups and n the number of nodes belonging to groups (Newman, 2010). One mode projections are easier to analyze, so methods of weighting the edges to retain some of the information of the full network have been developed (Newman, 2010; Zhou et al., 2007). However, it is not possible to recreate the full multipartite network using only the information available in the one mode projection, and representing grouping is not why multipartite methods were chosen for this research. As a result, this thesis does not make use of the incidence matrix or the one mode projection techniques common to existing theory.

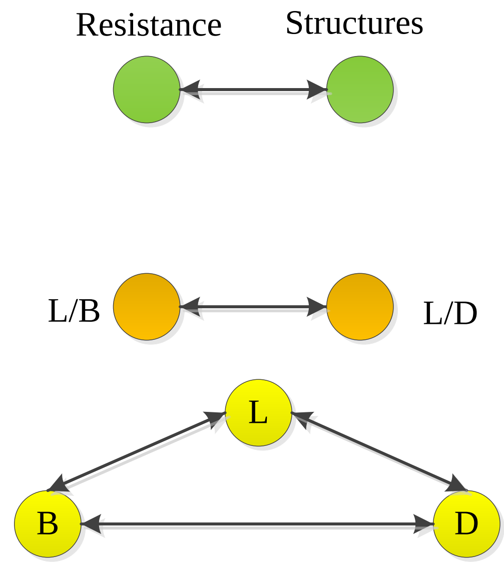
The reason that Multi-Domain Matrices of the Design Structure Matrix world have such complicated transition rules is that they essentially start from a one mode projection and then attempt to add context to build up a larger structure. As DSM researchers have attempted to correctly map these different homogeneous domains to one another, the resulting matrix representations have been increasingly cumbersome and complicated as described in Kreimeyer et al. (2008). Avoiding this mistake, a fundamental property of the current research is that multipartite networks are created with all the a priori nodes and context available from the start. Separate homogeneous networks cannot be used independently, or easily, to capture the context provided by a multipartite network. Fig. 3.2 demonstrates this visually using the basic naval architecture relationships described earlier. The multipartite network on the left naturally provides context, while its associated one mode projections on the right do not.

3.2.3 Application to Design Networks

Design tools in the classic sense are simply design formulations or methods that have been automated to remove tedium from the human role, but their underlying logical structure is identical in nature to that of design and acquisition. The name of nodes differs but not their context or the interpretation of their function. The common structural makeup is important, as it allows concepts to be developed and tested on smaller formulations that remain applicable to larger ones. This thesis uses design methods as case studies to test the contributions, but is not a specifically method or tool focused thesis. The literature supporting the methods and their quantitative



(a) Multipartite Network



(b) One Mode Projections

Figure 3.2: Multipartite Network Example and Associated One Mode Projections

outputs allows the network metrics to be compared with reality, meaning their value as predictors of formulation behavior can be shown. This would not be possible for case studies derived from design or acquisition programs at large.

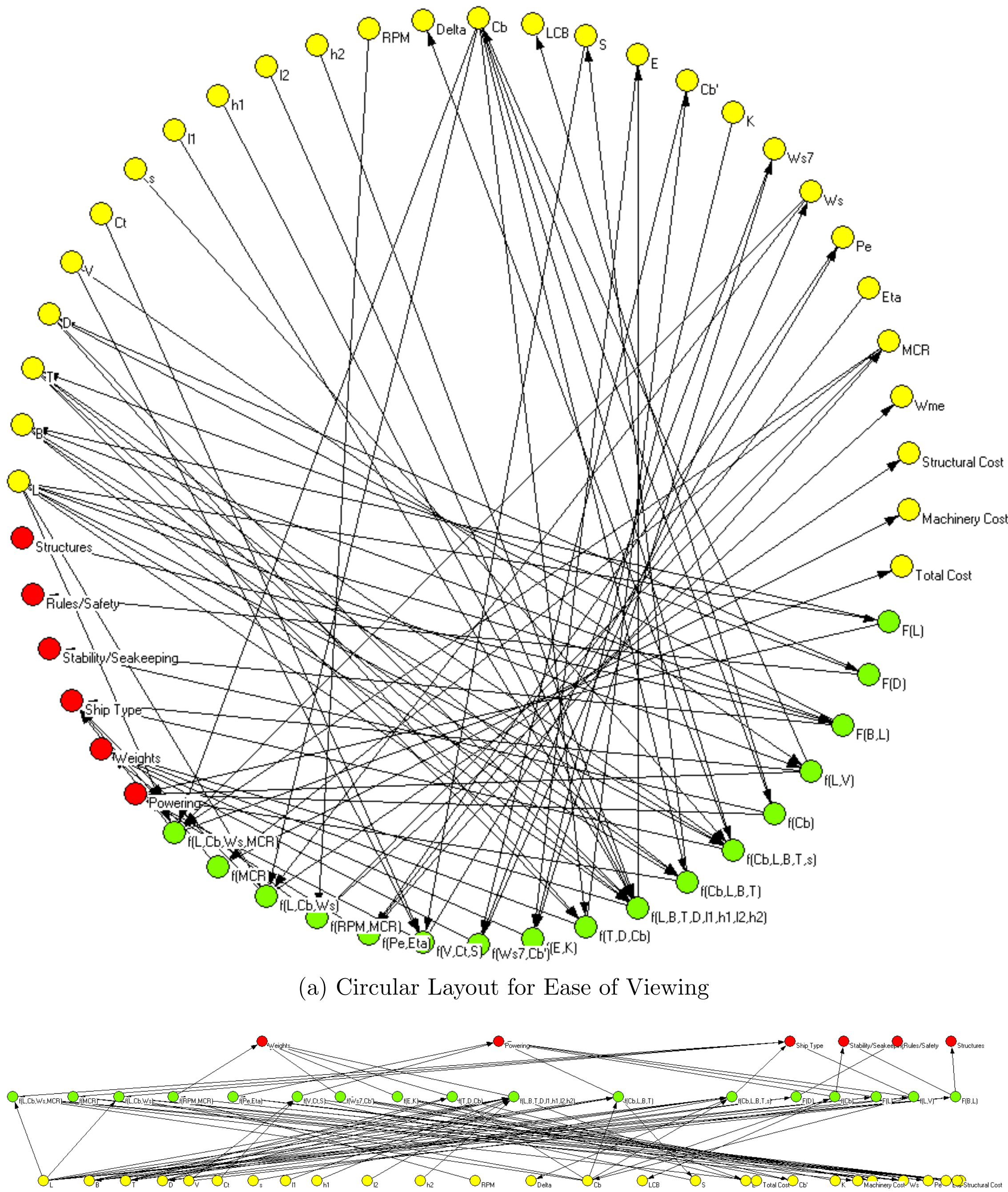
Converting a design tool or method to a multipartite network is relatively simple. For the purposes of this research a design tool or method is any systematic formulation an engineer might use to manipulate or analyze a design. With this definition, a physics based model, an empirical model, or a black box piece of software all equally qualify. At the earliest stages, an engineer might have a simple regression model to determine basic parameter ranges. This model might have a list of variables, requirements, functions and design disciplines with which to group them. Design Structure Matrix methods would typically represent each of these in a separate matrix, meaning a separate network. These separate networks are in effect one mode projections of a larger multipartite network. The following two sections document the creation of such a multipartite network from two different design formulations. This allows designers to think about different problem components in context and with all the a priori information available, avoiding the decoupling that results in a loss of information.

3.3 Watson & Gilfillan Tripartite Network Construction

The metrics presented in Section 3.1 and the context based multipartite network formulation of Section 3.2 were first tested on an abbreviated network representation of the classic Watson & Gilfillan ship design method, the complete results of which are presented and discussed in Chapter IV (Watson and Gilfillan, 1977). The word method is used as opposed to the word tool because the paper described a design approach based on regression equations and first principles that was not automated. "Methods" also appeared in the title of the paper. To construct the network for analysis a set of relationships were modeled from formulae or presumed functions of the paper, either as printed or derived from printed charts. Terms involving cost were not available from Watson & Gilfillan, and were taken from the NA 470 cost spreadsheet used for instruction at the University of Michigan. The three node types of the network are variables, functions and disciplines. The relationships between variables, their defining functions and the disciplines involved are shown in Table 3.2. There are a total of 51 nodes in this network, 28 variables, 17 functions and 6 disciplines. The function to variable relationships are well defined in the source paper. The discipline groupings are the judgment of the author, with guidance from the organization of the source paper. Since there are three different node types being represented, the network is tripartite. Once the relationships between nodes were assigned, each node was given a number. An arc list, containing the ordered pairs of nodes for each arc was then constructed from the node numbers. With software (it can be done by hand) this arc list was used to create the adjacency matrix. In this case, the arc list was input directly into the freeware network analysis software *Pajek* for visualization and basic metric calculation. The Watson & Gilfillan ship design network as visualized by *Pajek* is shown in Fig. 3.3. Both views are the exact same network, just displayed in different layouts. Of primary importance is that the formulae of the Watson & Gilfillan method naturally form a multipartite network, the network structure was not forced upon the method. This lends support to the idea that designers think, and later write about their methods, in a multipartite manner.

Table 3.2: Watson & Gilfillan Network Nodes

Variable	Defining Function	Discipline
L		
B	$f(L)$	Powering
T	$f(D)$	Rules/Safety Freeboard
D	$f(B, L)$	Stability/Seakeeping & Structures
V		
C_t		
s		
l_1		
h_1		
l_2		
h_2		
RPM		
Δ	$f(Cb, L, B, T, s)$	Ship Type
C_b	$f(L, V)$	Powering & Ship Type
LCB	$f(Cb)$	Powering & Stability/Seakeeping
S	$f(Cb, L, B, T)$	Powering
E	$f(L, B, T, D, l_1, h_1, l_2, h_2)$	Weights
C_b'	$f(T, D, C_b)$	Weights
K		
W_{s7}	$f(E, K)$	Weights
W_s	$f(W_{s7}, C_b')$	Weights
Pe	$f(V, C_t, S)$	Powering
η		
MCR	$f(P_e, \eta)$	Powering
W_{me}	$f(RPM, MCR)$	Weights
Structural Cost	$f(L, C_b, W_s)$	Ship Type
Machinery Cost	$f(MCR)$	Ship Type
Total Cost	$f(L, C_b, W_s, MCR)$	Ship Type



(a) Circular Layout for Ease of Viewing

(b) Typical Tripartite Layout

Figure 3.3: Watson & Gilfillan Ship Design Equation Network

3.4 Sen Bulker Problem Multipartite Network Construction

The dynamic network structural analysis methods discussed in Chapter V were best demonstrated on a test network representing a complete ship design problem amenable to optimization. The design formulation selected to create the test network is a principal dimension evaluation for bulk carriers developed by Yang and Sen (1996) and expanded upon by Sen and Yang (1998). The case study, referred to as the Sen Bulker problem, has three objectives. The objectives are minimization of transportation cost (f_1), minimization of light ship mass (f_2), and maximization of annual cargo (f_3), dependent on a set of equations which can be organized into 11 levels that build upon each other as shown in Eqs. B.1 to B.30, and a set of constraints Eqs. B.31 to B.43. Six variables, six parameters and six constants are shown in Tables B.1 to B.3 which complete the model definition.

To construct a network from the Sen Bulker problem, each parameter, variable, function and constraint becomes an individual node. There is an arc between each variable/parameter node and the function(s) in which it is a term. Similarly, functions can be terms of other functions, which are also represented as an arc. Once this base network was constructed, it was partitioned into a multipartite network. This was accomplished by ordering the nodes such that all arcs pointed in one direction, forming the 11 levels of functions with each level dependent only on the levels preceding it as shown in Appendix B. Each level is naturally a layer in a multipartite network, as there are no arcs within a level. This is a further demonstration that multipartite networks are amenable to how engineer's structure design problems, and thus the methods to analyze them, reinforcing the conclusions of Parker and Singer (2013). The network is shown in Fig. 3.4 and consists of 59 nodes and 95 arcs. Each color represents a unique level, 13 in total when the variable and constant/parameters levels are included. The arc list for this network can be found in Tables B.4 and B.5.

Though constraints are dependent on variables, parameters and functions, all represented by arcs toward constraints, the problem formulation contains no explicit feedback from the constraints to the rest of the problem. These interactions are optimizer rather than formulation dependent, an important distinction. A second version of the Sen Bulker network was created to represent the impact of constraints on the rest of the problem, meaning bi-directional arcs. The test case network with bidirectional arcs will be denoted as the ↔ network when discussed.

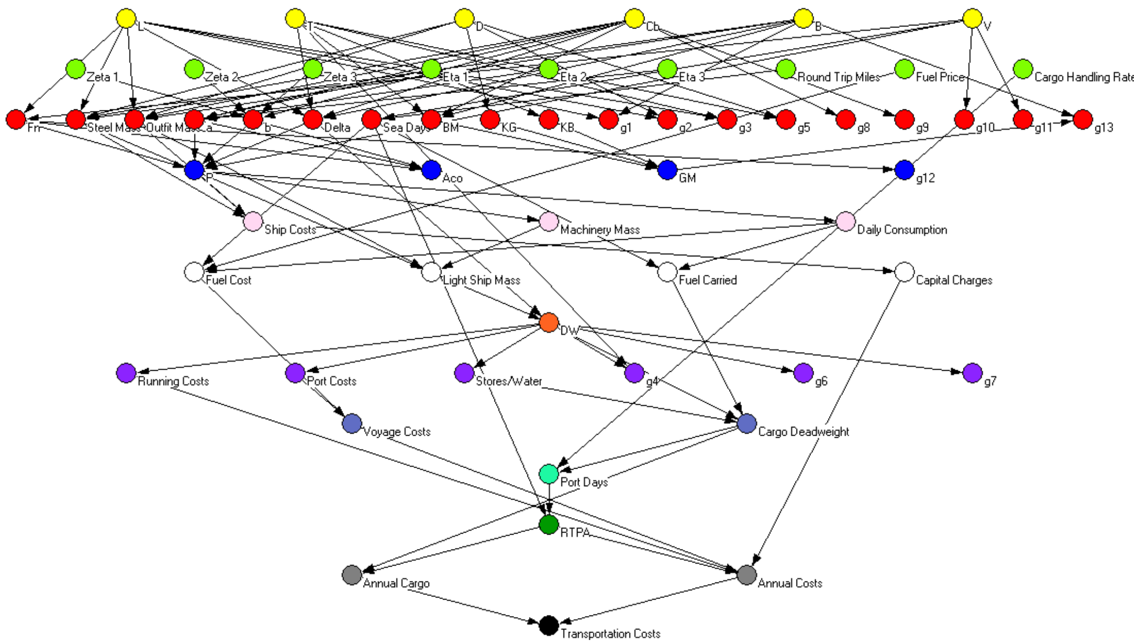


Figure 3.4: Sen Bulker Multipartite Network

Several other versions of a multipartite network were created from the Sen Bulker problem, the differences being the number of node types and the manner in which constraints were handled. The largest version included separate node types for the exponents and coefficients within functions and constraints, allowing for potential insight into the impacts beyond variable/function interaction. This larger network was not used in an effort to control the scope of the thesis, but it does show that there is great flexibility in how networks are created and their level of fidelity, even at the earliest stages of design. Though there are many possible network representations of the Sen Bulker problem, none is more or less correct than another. Every network derived as outlined is an accurate representation of the problem structure.

CHAPTER IV

Static Network Structural Analysis

The purpose of this chapter is to demonstrate that the structure of engineering formulations alone provides information through static network structural analysis that can be useful in a new way. One of the major novel contributions of this thesis is the introduction of multipartite networks to model design dependencies in context. Sections 3.2 to 3.4 explained the logic behind and construction of multipartite networks for design, but the proof is in the pudding. This chapter displays and discusses the results of analyzing multipartite networks, demonstrating not only coherency, but usefulness. These results contradict the claims of practitioners in other fields that advocate for homogenous networks (Biedermann et al., 2013; Maurer, 2007).

Static structural analysis means that only the structure of the network is being analyzed, not the information carried on the network. This type of analysis can be used on any network, meaning that what has been shown to work for design networks can be applied to acquisition structures at large. A benefit of using design formulations as test cases is that they exist prior to the formation of a process or organization, and long before a finished product emerges. For instance, if a ship is to be designed then a set of variables and the existence of relationships between them is known to exist simply by the existence of suitable design formulations. This is especially true if design organizations utilize tools or models like ASSET or Holtrop and Mennen, with well defined functions and variables (NSWCCD, 2005; Holtrop and Mennen, 1982). Without an explicit understanding of how these formulations function, how is the engineer to know how they will influence the final product?

A network representation of these design formulations, and other known or guessed relationships, can inform an engineer about design drivers, constraints, conflicts and general structure without performing a single design calculation, and without having to deeply study each one. A complete understanding of the minutia of every formulation used is ideal, gained through training, experience, and maybe even reading the user manual. Reality suggests that this is often not the case and is not likely to be. When new tools are under development, training, experience or manuals may not even exist. A network representation and analysis can provide an intermediate level of understanding when the ideal is not achievable.

Section 4.1 presents and discusses the results from analyzing the Watson & Gilfillan network described in Section 3.3, including perturbation analysis, and is an adaption of the PhD prospectus and a published conference paper (Parker, 2013; Parker and Singer, 2013). Static structural analysis was also conducted on the Sen Bulker problem network, primarily to verify the Watson & Gilfillan results, and is discussed in Section 4.2. Conclusions and contributions form Section 4.3.

4.1 Watson & Gilfillan Static Structural Analysis

4.1.1 Centrality Results

Four centrality measures were calculated for the Watson & Gilfillan network's nodes as outlined in Section 3.1, Park, out-degree, in-degree and betweenness. The results, sorted by node number are shown in Tables 4.1 to 4.3.

Significant insight can be gleaned from centrality, and example of which can be found by studying the results for length (L). It can be concluded that length is

Table 4.1: Watson & Gilfillan Variable Centrality Results

Node #	Node	Park	Out-Degree	In-Degree	Betweenenss
1	L	30.64	8	0	0.00
2	B	12.18	4	1	0.31
3	T	8.70	4	1	0.51
4	D	8.59	3	1	0.47
5	V	9.97	2	0	0.00
6	C_t	3.16	1	0	0.00
7	s	2.07	1	0	0.00
8	l_1	2.92	1	0	0.00
9	h_1	2.92	1	0	0.00
10	l_2	2.92	1	0	0.00
11	h_2	2.92	1	0	0.00
12	RPM	2.07	1	0	0.00
13	Δ	-5.34	0	1	0.00
14	C_b	12.70	6	1	0.41
15	LCB	-2.12	0	1	0.00
16	S	-1.64	1	1	0.80
17	E	-4.07	1	1	0.61
18	C_b'	-1.30	1	1	0.41
19	K	2.99	1	0	0.00
20	W_{s7}	-0.83	1	1	0.58
21	W_s	-0.39	2	1	0.77
22	Pe	-0.13	1	1	0.94
23	η	3.83	1	0	0.00
24	MCR	3.00	3	1	0.91
25	W_{me}	-2.98	0	1	0.00
26	Structural Cost	-4.48	0	1	0.00
27	Machinery Cost	-2.44	0	1	0.00
28	Total Cost	-5.92	0	1	0.00

Table 4.2: Watson & Gilfillan Function Centrality Results

Node #	Node	Park	Out-Degree	In-Degree	Betweenenss
29	$f(L)$	8.32	2	1	0.01
30	$f(D)$	5.51	2	1	0.35
31	$f(B, L)$	6.11	3	2	0.27
32	$f(L, V)$	8.88	3	2	0.22
33	$f(Cb)$	0.90	3	1	0.07
34	$f(Cb, L, B, T, s)$	-6.12	2	5	0.17
35	$f(Cb, L, B, T)$	-3.44	2	4	0.83
36	$f(L, B, T, D, l_1, h_1, l_2, h_2)$	-7.76	2	8	0.71
37	$f(T, D, C_b)$	-2.91	2	3	0.48
38	$f(E, K)$	-2.04	2	2	0.66
39	$f(W_{s7}, C_b')$	-2.40	2	2	0.94
40	$f(V, C_t, S)$	-1.52	2	3	1.00
41	$f(P_e, \eta)$	1.19	2	2	1.00
42	$f(RPM, MCR)$	-1.71	2	2	0.26
43	$f(L, C_b, W_s)$	-4.52	2	3	0.28
44	$f(MCR)$	-0.71	2	1	0.20
45	$f(L, C_b, W_s, MCR)$	-7.23	2	4	0.39

Table 4.3: Watson & Gilfillan Discipline Centrality Results

Node #	Node	Park	Out-Degree	In-Degree	Betweenenss
46	Powering	-17.69	0	6	0.00
47	Weights	-23.19	0	5	0.00
48	Ship Type	-20.25	0	5	0.00
49	Stability/Seakeeping	-4.63	0	2	0.00
50	Rules/Safety	-2.25	0	1	0.00
51	Structures	-2.50	0	1	0.00

a primary design driver for two reasons. First, it has the highest Park centrality of any node (functions and disciplines included) in the network. This means that the direct and indirect influence of length over other nodes in the network, minus the amount to which length itself is influenced, is far greater than any other node. Secondly, closer examination reveals that length itself is not influenced at all, its in-degree and thus "loss" term for Park centrality are zero. By contrast, the out-degree is eight, the highest of any node in the network. In terms of design, this means that a change in length will have more and farther reaching impacts on other parts of the design than any other change, whereas changes anywhere else will have no direct or indirect impact on length. If an optimization problem were to be formulated using this network, length could serve as an independent variable. What cannot be concluded from these metrics is the magnitude of impact a change in length will have on other nodes, only that an impact exists. Recalling that betweenness centrality is a measure of a nodes impact on flow between other nodes in the network, it becomes obvious that without non-zero in and out-degree centrality the betweenness score for a node will be zero, as is the case for length. This reinforces the concept that length is an independent variable, though with more nuance. Length is not required for coupling between other nodes, though its removal from the network can still isolate flow to nodes solely dependent on length, such as Node 29 ($f(L)$) and thus Node 2 (B, beam).

There are multiple ways in which a design can be "driven", such as that shown by length or the opposite, by constraining a design. By Park centrality, the most influenced variable in the network was Total Cost, Node 28. It has no influence over other variables with an out-degree of zero, and thus a betweenness of zero. If there were constraints on cost, the network indicates the design could be highly sensitive. In practice, this has proved to be exactly the case. The NA 470 Weights I spreadsheet uses the same basic Watson & Gilfillan formulation. There are no cost inputs (i.e. zero out-degree and betweenness) when students formulate their principal ship parameters using the Weights I spreadsheet. It is only when those parameters have stabilized that cost is checked. If their cost value is deemed too high, they must restart the entire process or fudge the cost number. The network indicates the addition of a function that relates cost back to length within the Weights I tool might alleviate the risk of this occurrence. This example is anecdotal, but changes in network structure to account for such feedback have been shown to have major influence as discussed later in Section 4.2.2.

The previous two examples of structural insight focus primarily on variables. Looking beyond variables, Park centrality results show that the weights discipline, defined by the author in the tripartite structure, is the most influenced node of the entire network. By contrast, the structural and stability/seakeeping disciplines were not as influenced as several functions, and even variables. This indicates that the formulation is focused more heavily on the weight related aspects of design rather than structural. This is a potential shortcoming. This was in fact a complaint noted in the discussion section of the original paper, "At the technical level the paper has main sections devoted to dimensions, displacement, form, powering, and so on, but nowhere is there even a sub-heading for ship strength, much less structural materials"(Watson and Gilfillan, 1977).

Comparative analysis across different node types should be undertaken carefully, as there are complex interactions taking place. However, standard parameter-based DSM methods would not have been able to characterize any of these cross node type interactions because by definition they define homogeneous networks. This is important, because to accurately model a complete design evolution multiple node types must be considered as part of the total network. For comparison, a one mode projection of the variable network is shown in Fig. A.1 found in Appendix A. Recreating the information inherent in the full tripartite representation shown in Fig. 3.3 is not possible without adding significant contextual information from an outside source.

Given that every arc in the one mode projection might represent several paths in the tripartite representation, this becomes a very tedious process even for the small test case network. This demonstrates the value of starting with a network containing as much information as possible in a multipartite representation, especially for larger networks.

4.1.2 Similarity Results

Cosine similarity measures were also calculated for the Watson & Gilfillan network. Given that the similarity between node i and node j is commutative, a matrix showing the results is symmetric, with ones along the diagonal as a node is similar with itself. As a result, the cosine similarity for a directed network can be displayed as a square table, the upper triangular portion being either in or out-degree, with the lower triangular portion being the other. Despite this compact representation, the Watson & Gilfillan similarity table is too large for display in this document. The network was simplified into an undirected tripartite network for simplicity and the single cosine similarity measure calculated. This does not alter the structure of the network with respect to the existence of arcs, but does cast out their directed nature for the benefit of a smaller set of results.

Of interest, eight pairs of nodes were perfectly similar, all variables as shown in Table 4.4. The first two entries in the table do not correlate to the directed equivalents, but are present because each node has only one edge, that to a common function which relates the two. The other six entries do correlate to the out-degree similarity measures. The in-degree similarity measure is zero for these nodes, as these particular variables have zero in-degree. The l_1, l_2, h_1, and h_2 terms all feed into the equipment number function, and since the edges point in the same direction (from variable to function) the similarity is the same in the undirected and out-degree directed case. This is an example of a set of variables that might be merged. In the paper they represent the dimensions of either full width deck erections (superstructures) or deck houses. These two types of structures typically being mutually exclusive or combined in modern ships, only one set of these variables is generally needed. This illustrates the application of similarity to identify a source of unnecessary complexity in a ship design formulation. Given that an acquisition program might involve tens of thousands of engineers using hundreds of different formulations, the identification of redundancy to reduce complexity could have a huge impact on program structure. This is especially true in a temporal network, where it takes time for information to propagate. Waiting for redundant information is a wasted opportunity.

Table 4.4: Watson & Gilfillan Undirected Perfect Similarity

Node 1	Node 2
s	Δ
RPM	W_{me}
l_1	h_1
l_1	l_2
l_1	h_2
h_1	l_2
h_1	h_2
l_2	h_2

There are a variety of possible interpretations for similarity measures in a ship design formulation. Redundancy has been demonstrated, but cohesive groups of nodes might also be identified. These might then be merged, better organized under one discipline, or perhaps divided across disciplines for parallel design activities. Conversely, two nodes which are perceived as similar to the designer may not be similar within the formulation, requiring a check to see if the formulation is using the variable, function or discipline as expected.

4.1.3 Perturbation Analysis Results

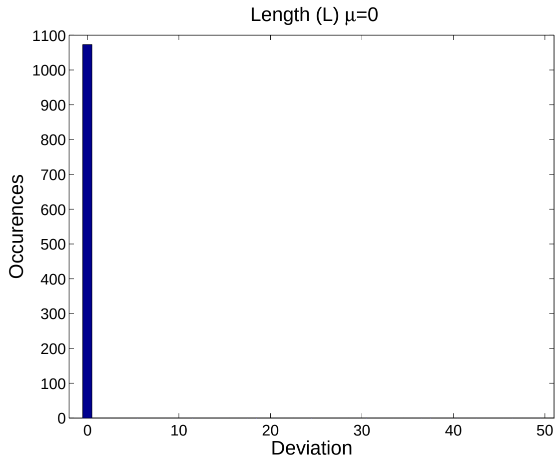
Network perturbation analysis can help identify areas where a false or missing basic assumption in the creation of a network can make a significant structural difference. For a design tool, a missing assumption can be thought of as the existence of a relationship between nodes that is not present in the formulation, but that will be a factor in the resulting design. The impact of such a missing assumption can be measured by adding an arc to the network and observing the overall change to the network's structure.

Perturbation analysis is particularly useful for formulations such as Watson & Gilfillan which are used in the early stages of design when very little information is available. In early design there is still a great deal of uncertainty in both the inputs to the formulation and the resulting outputs. As has been described in literature regarding Set-Based Design (SBD), and is common knowledge to practicing designers, an inaccurate assumption or mistake in early stage design can be quite costly to remedy later on (McKenney et al., 2012; Singer et al., 2009). The Watson & Gilfillan unweighted directed network structure allows for 1073 possible arcs that can be added to the network while maintaining the multipartite definition. Said another way, there are 1073 potential missing relationships between the existing nodes.

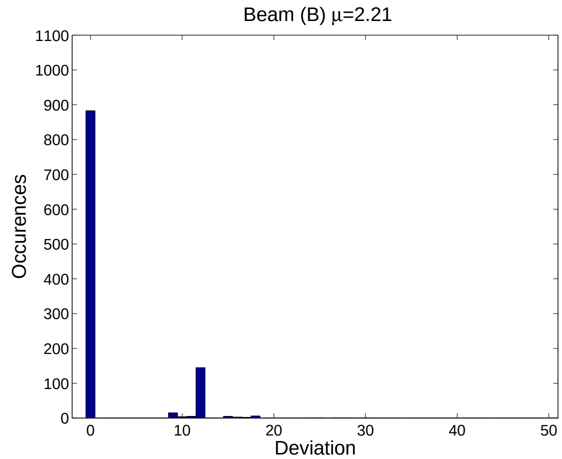
It is useful to the engineer to think about the general stability of the network structure via risk. Risk in this case can be defined as the likelihood that a change will occur along with the magnitude of that change. To measure the risk of a missing assumption each node was ranked by its Park centrality, and then the deviations between the initial ranking and the ranking created after the addition of a single arc was recorded. One visual representation of this risk is a histogram, as shown in Fig. 4.1. The abscissas of the plots show the deviation from the initial ranking, while the ordinate shows the total number of times that deviation occurred over all 1073 new arcs added separately. If a plot peaks at zero deviation with a sharp drop off, it is unlikely a deviation will occur, and if one does occur it is likely to be small. This means low risk, i.e. a generally stable network structure. The mean deviation alone, μ in Fig. 4.1, does not provide a good estimate of network stability.

The most significant result of this analysis is that length (L), already identified as the most influential node in the Watson & Gilfillan network, faces zero risk in losing this distinction with the addition of any single arc to the network. Showing the same trend are other variables classically perceived to be the principal dimensions of a vessel. The mode of deviation for each principal dimension in all cases is zero with a sharp drop off. The bottom right plot in Fig. 4.1 shows the additive result for all nodes in the network, once again indicating low risk.

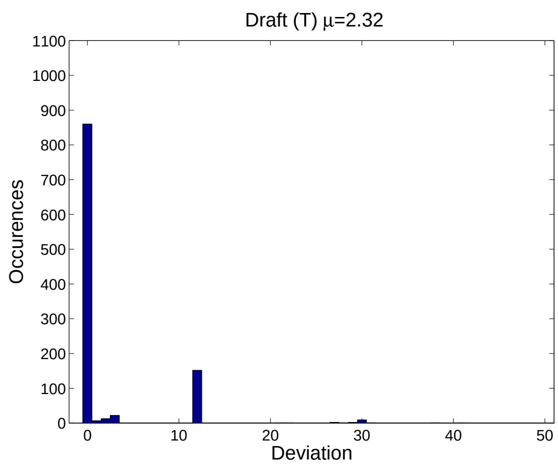
The perturbation analysis conducted was the addition of any single arc to the existing set of nodes. Perturbation analysis can also be done for the addition or deletion of multiple arcs, though computational time will increase geometrically with number of simultaneous changes being evaluated. It is also within reason to randomly add or remove nodes, measuring the structural impacts from missing or incorrect variables, functions, disciplines etc. The basic principle of perturbation analysis having been defined for design networks, these studies are left for future work.



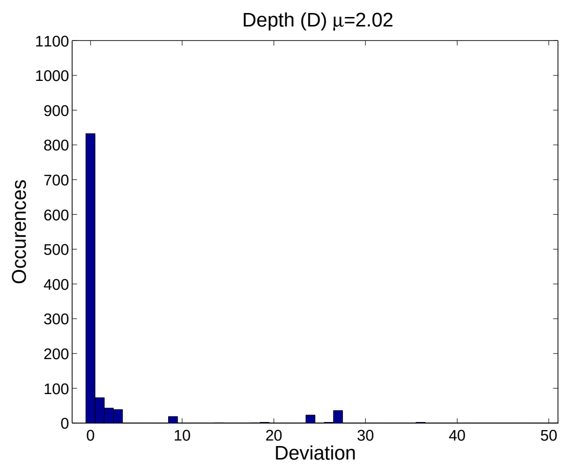
(a) Length Park Rank Deviation



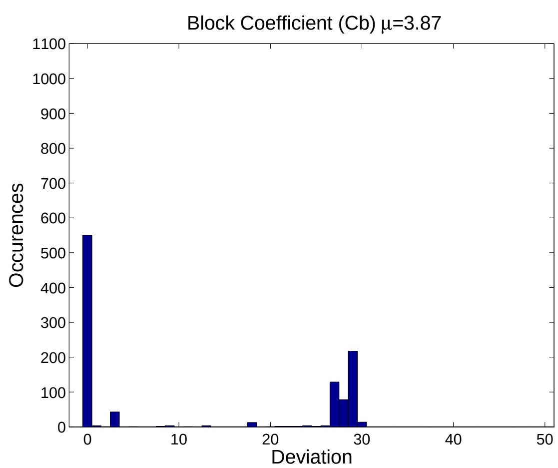
(b) Beam Park Rank Deviation



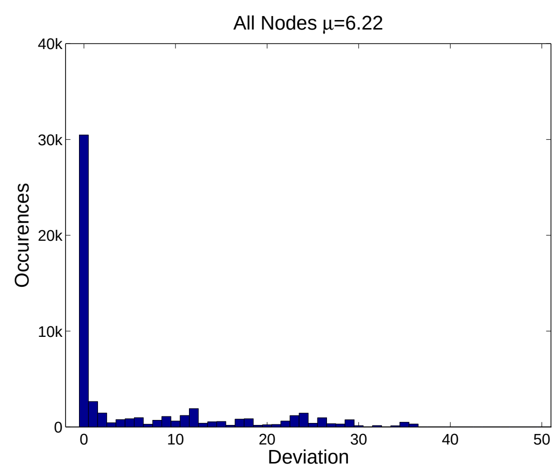
(c) Draft Park Rank Deviation



(d) Depth Park Rank Deviation



(e) Block Coefficient Park Rank Deviation



(f) All Nodes Park Rank Deviation

Figure 4.1: Perturbation Results

4.2 Sen Bulker Static Structural Analysis

In-degree, out-degree, Park and betweenness centrality were calculated for the Sen Bulker test case networks, the results for the $\leftrightarrow$ network are shown in Tables 4.5 and 4.6, meaning the structural impact of feedback from constraints is included. This network with bidirectional arcs is shown in Fig. 4.2, and is only distinguishable from Fig. 3.4 by the fact that arcs connected to constraints have arrows pointing in the upward and downward directions. The $\leftrightarrow$ results are discussed in the following subsections.

4.2.1 Degree Centrality Results

The network generally resembles a funnel, logical since it is many variables and functions inputting to a few objectives. The node identifier numbers 1 through 33 are variables and functions, roughly ordered in increasing levels through the network, i.e. variables are on level one (Nodes 1-6), functions of only variables are on level two (Nodes 7-13) etc. Nodes 34-59 are functions that input only to constraints, are formulation parameters or constraints themselves. If Nodes 1-33 actually behave like a funnel then one would expect the out-degree to be roughly inversely proportional to node number, which a quick glance reveals to be true. Thus the visual and mathematical representation of a number of variables funneling into a smaller number of objectives is verified. From an out-degree perspective then, variables are the most influential nodes in the network with the exception of the function node for deadweight (Δ_{DW}, Node 23), which has an out-degree equal to the highest in the network, belonging to L. Interestingly similar to the Watson & Gilfillan network, the Sen Bulker problem also appears to be driven by weight.

It might be expected that the objectives would have zero out-degree because the Sen Bulker networks were derived from an optimization problem independent of the optimizer and without recursive loops. This is true for the Transportation Cost

Table 4.5: Sen Bulker ↔ Centrality Results Part One

#	Node	In-degree	Out-degree	Park	Betweenness
1	L	3	7	292.00	0.33
2	T	3	6	209.22	0.65
3	D	2	5	173.67	0.12
4	C_b	2	8	104.37	0.11
5	B	2	6	148.27	0.26
6	V	2	5	24.54	0.11
7	F_n	3	3	-44.95	0.14
8	Steel Mass	4	2	-72.37	0.05
9	Outfit Mass	4	2	-72.37	0.05
10	a	4	2	2.57	0.16
11	b	4	2	2.57	0.16
12	Δ	4	2	-66.12	0.39
13	Sea Days	2	3	2.02	0.04
14	P	5	3	-73.29	0.72
15	A_{co}	3	0	-30.74	0.00
16	Ship Costs	3	1	-118.62	0.10
17	Machinery Mass	1	1	-11.84	0.44
18	Daily Consumption	1	2	-33.58	0.16
19	Fuel Cost	3	1	-18.10	0.03
20	Light Ship Mass	3	1	-39.49	0.57
21	Fuel Carried	2	1	-16.34	0.07
22	Capital Charges	1	1	-48.60	0.06
23	Δ_{DW}	5	7	-108.57	1.00
24	Running Costs	1	1	-102.54	0.03
25	Port Costs	1	1	-102.38	0.03
26	Stores&Water	1	1	-101.61	0.00
27	Voyage Costs	2	1	-51.00	0.01
28	Δ_{Cargo}	3	2	-152.67	0.17
29	Port Days	2	1	-63.44	0.06
30	RTPA	2	2	-27.44	0.03

Table 4.6: Sen Bulker $\leftrightarrow$ Centrality Results Part Two

#	Node	In-degree	Out-degree	Park	Betweenness
31	Annual Cargo	2	1	-76.93	0.02
32	Annual Costs	4	1	-99.48	0.05
33	Transportation Costs	2	0	-74.74	0.00
34	BM	3	1	-71.42	0.06
35	KG	1	1	-10.47	0.01
36	KB	1	1	-44.22	0.04
37	GM	4	1	-63.03	0.22
38	ζ_1	0	1	4.38	0.00
39	ζ_2	0	1	4.38	0.00
40	ζ_3	0	1	4.38	0.00
41	η_1	0	1	4.38	0.00
42	η_2	0	1	4.38	0.00
43	η_3	0	1	4.38	0.00
44	Round Trip Miles	0	1	3.34	0.00
45	Fuel Price	0	1	1.64	0.00
46	Cargo Handling Rate	0	1	1.88	0.00
47	g_1	2	2	179.52	0.10
48	g_2	2	2	189.88	0.04
49	g_3	2	2	204.37	0.21
50	g_4	2	2	41.04	0.60
51	g_5	2	2	156.12	0.13
52	g_6	1	1	-44.27	0.00
53	g_7	1	1	-44.27	0.00
54	g_8	1	1	42.56	0.00
55	g_9	1	1	42.56	0.00
56	g_{10}	1	1	10.01	0.00
57	g_{11}	1	1	10.01	0.00
58	g_{12}	1	1	-18.33	0.00
59	g_{13}	2	2	34.76	0.23

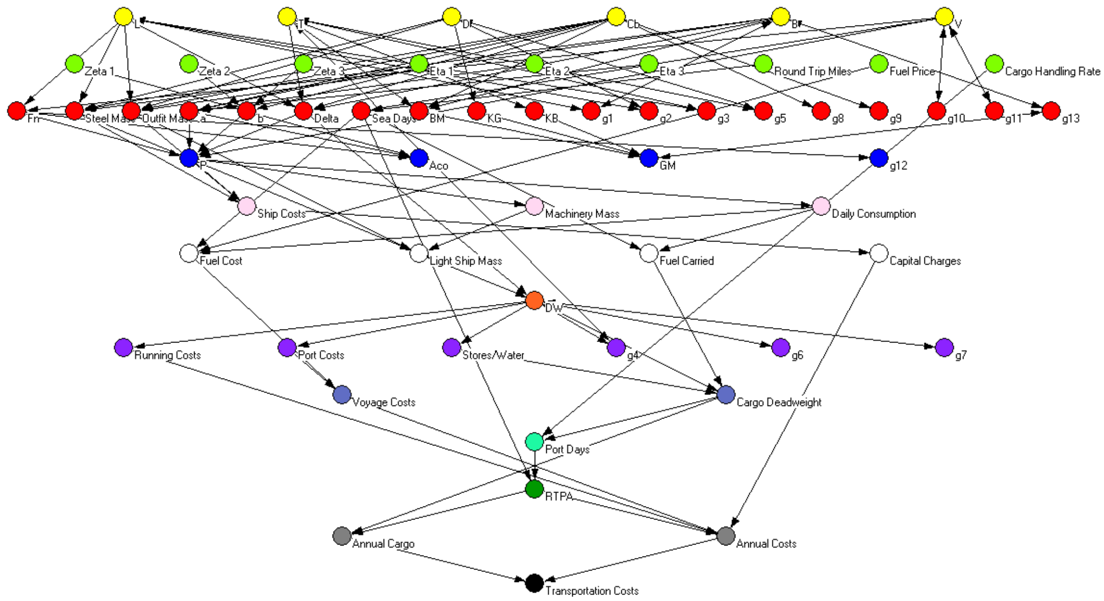


Figure 4.2: Sen Bulker Bidirectional Network

objective (Node 33), but the only other node with zero out-degree is the Admiralty Coefficient (A_{co}, Node 15). As the Admiralty Coefficient has zero out-degree and is not an objective, one must wonder then what its purpose is in the formulation. In name it is redundant and does not even appear in a *MATLAB* formulation of the problem, but in function is the denominator of function for power (Eq. B.11). This quirk in the network is a product of the fact that Sen and Yang (1998) used A_{co} to explain and develop a resistance regression curve using a, b and F_n, but did not use it by name in the formulation. This shows the human element of design, where a classically trained naval architect might gain value from seeing A_{co} even though it is an unnecessary abstraction for the problem formulation. The remaining two objectives, Light Ship Mass and Annual Cargo, have non-zero out-degree because they have indirect and direct influence on Transportation Cost respectively. The network identifies that the objectives are far from independent, and a more computationally efficient formulation may exist.

Gaining insight from in-degree is less clear cut, though as the in-degree of the variables is non-zero it is immediately obvious that results are for the $\leftrightarrow$network. Nodes 38-46 have zero in-degree, correctly identifying them as either variables, parameters or constants of the formulation. This assumption is reinforced by the corresponding out-degree values which are all one, with relatively low Park centrality and zero betweenness centrality (due to zero in-degree). Similarity results, though not displayed here show that Nodes 38-40 and 41-43 are structurally equivalent. Though they are independently important to the mathematical formulation, it may be redundant to display them separately in the network.

4.2.2 Park Centrality Results

The variables of the $\leftrightarrow$ network have some of the highest relative Park centrality values in the network, as expected from studying the Watson & Gilfillan network. L, as with the Watson & Gilfillan formulation was the highest ranking node in the network, followed by T. What is interesting is that the variable V is the 14$^{\text{th}}$ ranked node. The rankings between 1 and 14 not held by variables are held by constraints. The reason for the highly ranked constraints is that they influence what would otherwise be the most important nodes in the network (the variables), which in turn influence every other node. Thus their propagation "win" score is discounted by only α^{x+1} compared with α^{x} for the variables themselves, see Eq. 3.1. On the other hand, they are only influenced by those same variables (bidirectional arcs), which in turn are influenced by no other nodes but the constraints themselves. So the "lose" score is relatively low. In summation, the Park centrality reflect the potential importance of constraints on the problem. A visual representation of this is found in Tables 4.7 and 4.8, which compares the bidirectional network Park centralities with the unidirectional ones. The number 1 ranked node has the highest park centrality and is shaded dark green, with a continuum of decreasing rank to number 59 and dark red. The unidirectional rankings clearly reflect the funneling effect discussed earlier, Nodes 1-33 showing a rough continuum of decreasing rankings. More importantly, the comparison shows that using bidirectional arcs to represent constraint influence can have a major impact on network behavior.

Beyond variables and constraints, it would be expected that the objectives have some of the lowest relative Park centralities of the functions in the $\leftrightarrow$ network. Though they are in fact low, they are not the lowest. That distinction goes to Δ_{Cargo}, Ship Costs and Δ_{DW}. Interestingly, the objectives could be viewed as abstractions of the three lowest nodes. In fact, each objective is separated by at most three arcs from one of the three lowest nodes. Δ_{Cargo} and the objective Annual Cargo are are directly connected. Though high Park centralities indicate driving nodes by connection, low Park centralities indicate what is being driven toward. If the objectives did not have low Park centralities, or they were significantly separated from those nodes with the lowest Park centralities, it could indicate that the formulation is ill suited for the task at hand. In the case of the Sen Bulker problem, weight (as the name implies) is of primary importance. Both the network structure of the formulation and the choice of objectives reflects this, showing good agreement between intent and formulation structure. This structure could be collapsed into a set of variables, three objective functions and a set of constraints. If relatively low Park centrality objectives are the measure of a quality formulation, this collapsed network would show perfect agreement between formulation structure and objective choice.

4.2.3 Betweenness Centrality Results

Δ_{DW} has the highest betweenness score in the $\leftrightarrow$ network, scores shown in Tables 4.5 and 4.6 being normalized by the highest value. Δ_{DW} has already proven to be a node of distinction based on the other measures, and a high in and out-degree (5 & 7) explain the high betweenness score. Given prior emphasis on this node, it would be in the engineer's best interest to verify that whatever value Δ_{DW} holds is an accurate

Table 4.7: Park Centrality Comparison Part One

#	Node	Park Centrality ↔	Park Centrality →
1	L	1	2
2	T	2	5
3	D	6	6
4	C_b	9	1
5	B	8	3
6	V	14	4
7	F_n	41	8
8	Steel Mass	48	17
9	Outfit Mass	49	18
10	a	24	15
11	b	25	16
12	$Delta$	46	7
13	Sea Days	26	20
14	P	50	21
15	A_{co}	35	43
16	Ship Costs	58	49
17	Machinery Mass	30	24
18	Daily Consumption	36	39
19	Fuel Cost	32	44
20	Light Ship Mass	37	38
21	Fuel Carried	31	42
22	Capital Charges	42	47
23	Δ_{DW}	57	37
24	Running Costs	56	48
25	Port Costs	55	46
26	Stores&Water	54	45
27	Voyage Costs	43	53
28	Δ_{Cargo}	59	56
29	Port Days	45	55

Legend	
Maximum	Minimum

Table 4.8: Park Centrality Comparison Part Two

#	Node	Park Centrality ↔	Park Centrality →
30	RTPA	34	54
31	Annual Cargo	52	57
32	Annual Costs	53	58
33	Transportation Costs	51	59
34	BM	47	31
35	KG	29	25
36	KB	38	26
37	GM	44	40
38	ζ_1	17	9
39	ζ_2	18	10
40	ζ_3	19	11
41	η_1	20	12
42	η_2	21	13
43	η_3	22	14
44	Round Trip Miles	23	19
45	Fuel Price	28	23
46	Cargo Handling Rate	27	22
47	g_1	5	32
48	g_2	4	33
49	g_3	3	34
50	g_4	12	52
51	g_5	7	35
52	g_6	40	50
53	g_7	39	51
54	g_8	10	27
55	g_9	11	28
56	g_{10}	15	29
57	g_{11}	16	30
58	g_{12}	33	36
59	g_{13}	13	41

Legend	
Maximum	Minimum

reflection of the input variables, and a comparison of the Sen Bulker formulation to other bulker preliminary design formulations could usefully center around Δ_{DW}.

On the opposite end of things are the nodes with zero betweenness. Excepting A_{co}, already identified as a quirk of the formulation, these nodes fall into three categories. Parameters/constants, constraints of one variable and sideline functions. Parameters and constants, having zero in-degree by definition have zero betweenness. That constraints of one variable have zero betweenness despite non-zero in and out-degree is due to the bidirectional arc formulation. Constraints of one variable are directly connected to only one other node, thus all paths going through the constraint must pass through the other node twice. As no geodesic path will go through the same node twice (they are self-avoiding), the constraint can lay on no geodesic paths. This yields zero betweenness using Eq. 3.4. The Sen Bulker network contains one sideline function, Stores & Water (Eq. B.23), which takes Δ_{DW} as an input and outputs to Δ_{Cargo}. Unlike with bidirectional arcs, Stores & Water has no betweenness because the nodes it connects are already directly connected, meaning any path through Stores & Water would be shorter skipping it, meaning no geodesic paths. The sideline configuration is shown in Fig. 4.3. This is further evidence of a formulation structure based around designer intuition and intent, rather than computational efficiency.

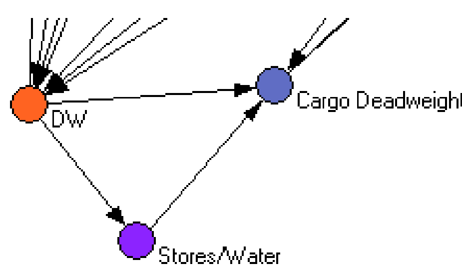


Figure 4.3: Sideline Node

4.3 Conclusions & Contributions

This chapter and the previous one are responsible for the first two novel contributions of this thesis:

- Introduction of a contextual multipartite network approach to represent the structure of naval design which enabled a new type of analysis and understanding
- Application and extension of existing network mathematics to provide meaningful predictive insight using multipartite design networks as inputs

Specifically, this chapter demonstrates that the structure of engineering formulations alone provides information through static network structural analysis that can be useful in a new way. Analysis of the Watson & Gilfillan and Sen Bulker problem multipartite networks yielded the following conclusions:

- A network representation of a ship design formulation is feasible and can generate lead indicators
- A multipartite network formulation can accurately reflect a ship design formulation and thus expose designer intent
- Analysis of multipartite ship design networks can correctly identify what naval architects intuitively understand about the formulation, correctly identifying design drivers, constraints and other features of model structure
- Many multipartite networks can be created for one design formulation, enabling the analysis of the formulation in different ways

The multipartite network structure now has a demonstrated ability to represent naval design, with corresponding analysis methods to better understand it. In the frame of design formulations, engineers now have the capability to better understand the impacts of the formulations they use on the products they will produce using a framework that naturally represents both formulations and thinking. Using static structural analysis, insight can be gained using the structure of formulations alone, meaning before design work begins. The successful construction and analysis of multipartite networks for design also contradicts the common practice of separating node types into separate homogenous networks or matrices. Taken as a whole, these conclusions support the hypothesis that in complex product design, elements of a domain do not directly influence one another, they must have context provided by another domain.

These results provide appropriate justification for further research into the subject. If analyzing the multipartite network structure of a simple ship design formulation verifies intuition, then analyzing network structures where no intuition is present, such as with very complex or new formulations, could prove highly valuable. The successful test of the multipartite formulation also validates its basis for extension to larger multipartite networks that include process and organizational elements.

The case studies presented in this chapter, though promising, were intentionally limited to static structural analysis. This means that only the structure of the networks were analyzed, not the information carried on them. Standard, yet unquantifiable, designer intuition was verified rather than reproducible experimental results. This research hypothesizes that multipartite network analysis can predict impacts of formulation structure on resulting designs. This chapter has shown this, but the next chapter describes and demonstrates a network method that produces verifiable results through dynamic structural analysis. This means the information carried on the network is represented, and formulation behavior can be predicted.

CHAPTER V

Dynamic Network Structural Analysis

Design formulations are increasingly becoming opaque, if not outright black boxes. Engineers often do not have the resources to intuitively understand the functioning of the tools they use, despite the common wisdom that this understanding is necessary. This opacity is not entirely driven by tool developers, but is also a reflection of the increasing complexity of vessels and the breadth and fidelity of analysis expected before fielding them. To cope with the challenge it is common to use a design of experiments or other meta-model that correlates the inputs of a formulation to the outputs. Thus an engineer has an idea in advance that changes in X will likely produce a change in Y. Such analysis is informative, but it does not provide information about the linkage between X and Y, that is how and why does X affect Y?

In Chapter IV network models of the Watson & Gilfillan ship design method and Sen Bulker problem were analyzed for the inherent properties in their static structure, answering the question of why X affects Y. This also proved informative for identifying design intent and design drivers/constraints, confirming the work of Parker and Singer (2013) and extending the work of Gillespie and Singer (2013). Though static structural analysis provides information about linkages, it does not explicitly resolve formulation behavior; i.e. how X affects Y. X may be identified as a design driver, but whether it increases or decreases Y has not been determined.

What is desired is a method that can both inform the engineer about the general nature and linkages within a formulation, while still resolving behavior.

This chapter addresses the need by introducing a network metric, termed path influence, creating information about the dynamic behavior of design problems using an identical network formulation to that for static analysis. Static analysis provides the engineer with a sense of whether a formulations's structure is representative of their design intent, whereas dynamic analysis provides quantitative information about how that intent will manifest in variables, functions, disciplines etc. when the formulation is actually used. The advantages of using network analysis to generate such information include an intuitive understanding of the interactions between problem components, not simply results. This means that a single network representation can now be used to answer questions of how and why inputs influence outputs. Secondly, network analysis can require significantly fewer function calls than comparable methods, especially when the structure of the problem is changing, requiring repetitive analysis. The third, and arguably most important advantage of network methods is that they can provide lead indicators. Its possible to generate design knowledge prior to fully exercising a formulation or beginning a design since only the most basic initial information is required, making network methods a candidate for use in early stage design when little is actually known. Design space exploration comparatively provides lag indicators because a tool or design must be fully exercised before results are available, often requiring a significant investment in time. The disadvantage of network analysis is that results are indicative, not exact, a trait shared with many competing methods. In reality this disadvantage may not exist, as the speed could be sufficient to apply network analysis to larger problems than possible with other methods, meaning that indicative information becomes a substitute for no information, which can only be advantageous.

Section 5.1 explains the mechanics of a Taylor series expansion, from which path influence is then derived. Section 5.2 displays and discusses the results of using path influence on the Sen Bulker problem, comparing partial and interpolated derivative weighting schemes with a full factorial design of experiments. Both of these sections are abstractions of Parker and Singer (2014). Section 5.3 discusses the differences between capturing tool formulation and optimization behavior, along with path influence results from the latter. Section 5.4 introduces a new metric to compare static and dynamic analysis, and Section 5.5 discusses further application of the metric. Section 5.6 concludes the chapter.

5.1 Capturing Formulation Behavior with Path Influence

5.1.1 Taylor Series Expansions and Paths

A Taylor series expansion provides a simple approximation of complex problem behavior by extrapolating around a baseline point using partial derivatives as a guide. This type of approximation is suitable for many design formulations and especially those encoded as continuous optimization problems. Many optimization algorithms rely on the Taylor series expansion or mathematically similar methods to guide the optimizer toward a local minimum (Bazaraa et al., 2006). The vector form of a Taylor series expansion of a real and differentiable function of multiple variables is shown in Eqs. 5.1 and 5.2.

$$f(\mathbf{r}) = \sum_{j=0}^{n} \left[\frac{1}{n!} \left((\mathbf{r} - \mathbf{a}) \cdot \nabla \right)^{n} f(\mathbf{a}) \right] + R_{n+1}(\mathbf{r}) \tag{5.1}$$

$$R_n(\mathbf{r}) = \frac{1}{n!} \left((\mathbf{r} - \mathbf{a}) \cdot \nabla \right)^{n} f(\zeta(\mathbf{r})) \tag{5.2}$$

In Eq. 5.1 $\mathbf{r}$ is the vector of variables and $\mathbf{a}$ is the point of expansion. $R_n(\mathbf{r})$ is the Lagrange form of the remainder (higher order terms). If f is continuous Eq. 5.2 is used to compute $R_n(\mathbf{r})$ where ζ is the point on the interval $[\mathbf{a}, \mathbf{r}]$ where the Lagrange form matches the actual remainder. Determining the correct value ζ is not always practical, but by sweeping over the interval the maximum and minimum of $R_n(\mathbf{r})$ can be found which provides error bounds on an expansion that neglects the higher order terms (Greenberg, 1998).

A complete expansion is shown in Eq. 5.3, but for engineering applications the first order terms are often sufficiently accurate to ignore the remainder, yielding Eq. 5.4.

$$f(\mathbf{r}) = f(\mathbf{a}) + (\mathbf{r} - \mathbf{a}) \cdot \nabla f(\mathbf{a}) + R_2(\mathbf{r}) \tag{5.3}$$

$$f(\mathbf{r}) \approx f(\mathbf{a}) + (\mathbf{r} - \mathbf{a}) \cdot \nabla f(\mathbf{a}) \tag{5.4}$$

A Taylor series expansion on the objective function(s) of a design tool that utilizes only the variables provides equivalent information to that gained from a design of experiments or other meta-model, excepting that the accuracy can vary between methods. However, in creating an objective function there are often many intermediate steps. The variables, objectives and intermediate functions can be represented as a network as shown in Fig. 5.1, where x and y are input variables, f_3 and f_4 the intermediate functions, and f_5 the objective. Including these intermediate functions in the expansions provides context to a variables influence on an objective. One way to think about the behavior of such problems is paths of influence, where a variable affects a function, which then has an effect upon another function and so on until the ultimate influence is on an objective. Problems become complicated when there are many paths of influence, often sharing component variables and functions. Mathematically f_5 could be stated in terms of x and y alone, that is $f_5(x, y)$ rather than $f_5(f_3, f_4)$, and the Taylor series expansion would be Eq. 5.5.

$$f_5 = f(x, y) \approx f_5^o + (x - x^o) \left.\frac{\partial f_5}{\partial x}\right|_{x^o, y^o} + (y - y^o) \left.\frac{\partial f_5}{\partial y}\right|_{x^o, y^o} \tag{5.5}$$

In Eq. 5.5 the contextual information provided by f_3 and f_4 is lost, i.e. the paths

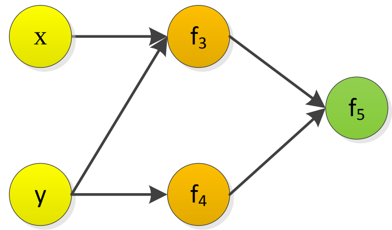


Figure 5.1: Network of Functions

that x and y take to get to f_5 are discarded. Maintaining this path information in the Taylor series expansion is a simple application of chain differentiation starting with the first order expansions of f_5, f_4, and f_3 written as they appear in Fig. 5.1 (Eqs. 5.6 to 5.8).

$$f_5 = f(f_3, f_4) \approx f_5^o + (f_3 - f_3^o) \left.\frac{\partial f_5}{\partial f_3}\right|_{f_3^o, f_4^o} + (f_4 - f_4^o) \left.\frac{\partial f_5}{\partial f_4}\right|_{f_3^o, f_4^o} \tag{5.6}$$

$$f_4 = f(y) \approx f_4^o + (y - y^o) \left.\frac{\partial f_4}{\partial y}\right|_{y^o} \tag{5.7}$$

$$f_3 = f(x, y) \approx f_3^o + (x - x^o) \left.\frac{\partial f_3}{\partial x}\right|_{x^o, y^o} + (y - y^o) \left.\frac{\partial f_3}{\partial y}\right|_{x^o, y^o} \tag{5.8}$$

Then, substituting the first order expansions of f_3 and f_4 into that for f_5 yields Eq. 5.9.

$$f_5 \approx f_5^o + (x - x^o) \left.\left(\frac{\partial f_3}{\partial x}\frac{\partial f_5}{\partial f_3}\right)\right|_{x^o, y^o} + (y - y^o) \left.\left(\frac{\partial f_3}{\partial y}\frac{\partial f_5}{\partial f_3} + \frac{\partial f_4}{\partial y}\frac{\partial f_5}{\partial f_4}\right)\right|_{x^o, y^o} \tag{5.9}$$

For the simple network in Fig. 5.1, computing a path conscious first order Talyor series expansion is relatively concise, as there are only three total paths in the network. For even the simplest design formulations there can be many hundreds of unique paths that link variables to objectives, requiring the path influence algorithm as described in the following section.

5.1.2 Path Influence

A *path* is defined as a sequence of connected nodes, but in the present context it is easier to think of path length as the number of arcs required to connect two nodes (Newman, 2010). A network representation such as Fig. 5.1 visualizes and allows the existence and length of paths to be computed using Eq. 5.10, where N is the number of paths of length r between i and j, computed using the adjacency matrix $\mathbf{A}$.

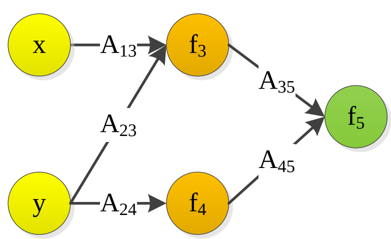
$$N_{ij}^{r} = [\mathbf{A}^{r}]_{ij} \tag{5.10}$$

This information by itself can be useful, i.e. counting the number of unique ways that x influences objective y, but if the influence of nodes over their neighbors can be quantified, then Eq. 5.10 can be applied to a weighted adjacency matrix, where N is no longer the number of paths of length r from i to j but the sum of the products of those paths' arc weights. A geodesic path is the shortest path between two nodes in a network. The diameter of a network is the longest geodesic path that exists, and the longest it can be is $n-1$ arcs since it takes $n-1$ arcs to connect n nodes in a chain. The weighted paths connecting each node can be computed by $\mathbf{A}^{r}\ \forall\, r \in [1, n-1]$ if they exist. It is often unnecessary to compute all the way until $r = n-1$, as once $\mathbf{A}^{r} = \mathbf{0}$ there is no reason to continue as a path of length $r+1$ cannot exist if there is no path of length r. In the worst case a total *path influence* matrix, $\mathbf{P}$, can be computed as shown in Eq. 5.11.

$$\mathbf{P} = \sum_{r=1}^{n-1} \mathbf{A}^{r} \tag{5.11}$$

There are much faster algorithms, running in $O(m+n)$ time or less, for finding path lengths (Newman, 2010). However, the networks analyzed in this thesis are small enough that the simplicity of Eqs. 5.10 and 5.11 outweigh the speed advantage of faster algorithms. More discussion of speed can be found in Section 5.2.5.

As stated earlier, a weighted adjacency matrix can be used where entry $\mathbf{A}_{ij}$ quantifies the influence of i over j. A mathematically elegant way to quantify "influence" is the partial derivative, where $\mathbf{A}_{ij} = \frac{\partial f_j}{\partial f_i}$. Instantiating this weighting scheme on the original network of Fig. 5.1 yields the weighted network and adjacency matrix of Fig. 5.2. Applying Eq. 5.11 to this matrix yields the the $\mathbf{P}$ matrix of Eq. 5.12. Though similar to a Jacobian matrix, terms on the diagonal are by network definition zero (no self arcs), and the functions and variables denoted in Fig. 5.1 are treated as separate scalar functions rather than as a single vector-valued function (which would not include the variables) as is the case of the Jacobian.



(a) Weighted Network

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & \frac{\partial f_3}{\partial x} & 0 & 0 \\ 0 & 0 & \frac{\partial f_3}{\partial y} & \frac{\partial f_4}{\partial y} & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial f_5}{\partial f_3} \\ 0 & 0 & 0 & 0 & \frac{\partial f_5}{\partial f_4} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(b) Weighted Adjacency Matrix

Figure 5.2: Partial Derivative Weighted Network

$$\mathbf{P} = \sum_{r=1}^{2} \mathbf{A}^r = \mathbf{A} + \mathbf{A}^2 = \begin{bmatrix} 0 & 0 & \frac{\partial f_3}{\partial x} & 0 & \frac{\partial f_3}{\partial x}\frac{\partial f_5}{\partial f_3} \\ 0 & 0 & \frac{\partial f_3}{\partial y} & \frac{\partial f_4}{\partial y} & \frac{\partial f_3}{\partial y}\frac{\partial f_5}{\partial f_3} + \frac{\partial f_4}{\partial y}\frac{\partial f_5}{\partial f_4} \\ 0 & 0 & 0 & 0 & \frac{\partial f_5}{\partial f_3} \\ 0 & 0 & 0 & 0 & \frac{\partial f_5}{\partial f_4} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.12}$$

Entry $\mathbf{P}_{15}$ is the influence term of x on the objective f_5, and this entry matches the x partial derivative terms of the network conscious Taylor series expansion of f_5 found in Eq. 5.9, the same being true for the y terms and $\mathbf{P}_{25}$. However, not all the terms in the fifth column belong in Eq. 5.9, only those corresponding to independent variables. In this instance the variables are known in advance, but this may not always be the case. Networks often contain subsets of nodes that can be classified in various ways. Directed networks contain in-components and out-components among others. The *in-component* of node i is the set of all other nodes that have a path to i, and includes i itself. An *out-component* is the opposite, being the set of nodes that can be reached from i, inclusive of i itself. In network terminology, the independent variables can be identified as the nodes with zero in-degree that also belong to the in-component of the objective node. This definition holds as an independent variable must have a path to the objective, and may not be influenced by another node.

A first order Taylor series expansion can be expressed using the path influence matrix as shown in Eq. 5.13, where $\mathbf{v}$ is an $1 \times n$ vector of the independent variables, with all other entries equal to 0, $\mathbf{v}^o$ the equivalent for the initial variable values, and $\mathbf{P}_{|i|_{\mathbf{v}^o}}$ the ith column of $\mathbf{P}$ evaluated at $\mathbf{v}^o$.

$$f_i = f_i(\mathbf{v}) \approx f_i^o + (\mathbf{v} - \mathbf{v}^o)^T \mathbf{P}_{|i|_{\mathbf{v}^o}} \tag{5.13}$$

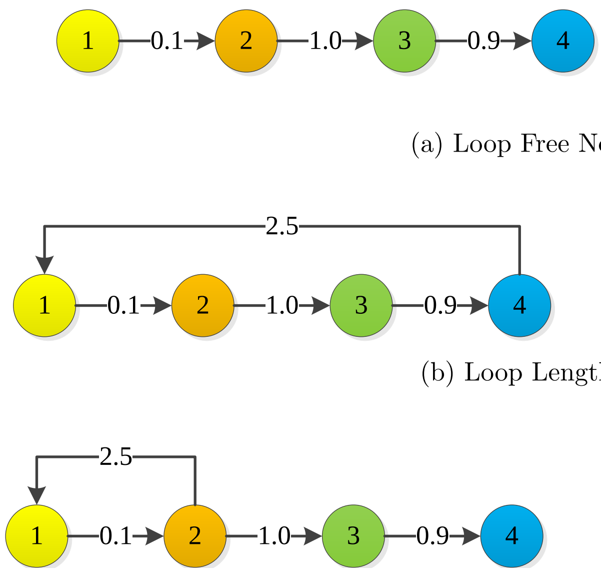
In an elegant fashion, the same network methods used to find paths, path lengths and path influence can be used to generate a first order Taylor series expansion. If first order Taylor series accuracy is acceptable, then by extension path influence can be an accurate predictor of formulation behavior that maintains the benefits of a network representation. In terms of design, an engineer can use their existing network representation to create path conscious first order Taylor series expansions of their objectives, meaning the context of all the intermediate functions used to create the objectives is still present. This means that the influence of variables on objectives is quantified, along with the way that influence is achieved.

5.1.3 The Influence of Loops

The Sen Bulker network and the example network shown in Fig. 5.2 do not contain loops, an important property when discussing path influence. A loop is still a path, so the the shortest length of a loop between i and itself is the minimum r for which $[\mathbf{A}^r]_{ii} \neq 0$. This means a path of length r exists between node i and itself. Similarly, the length of a geodesic path between nodes i and j (if one exists) is the minimum value of r such that $[\mathbf{A}^r]_{ij} \neq 0$. Thus any non-zero entry of $[\mathbf{A}^r]$ when $r > n-1$ signifies the existence of a loop, as any path longer than $n-1$ must contact the same node more than once, necessarily creating a loop. For path influence, any node involved in a loop shorter than $n-1$ will have a non-zero $\mathbf{P}_{ii}$ entry and Eq. 5.11 will not necessarily converge. Such a loop effectively creates a recursive relation, and makes path influence results suspect.

The characteristics of loops and path influence described above are demonstrated in Fig. 5.3. Fig. 5.3a shows a loop free network with its associated path influence matrix. Fig. 5.3b shows the same network where a loop with length n is added. In this case, the lower triangular portion of the $\mathbf{P}$ matrix reflects the influence of Node 4 on the other nodes of the network but the upper triangular entries of the $\mathbf{P}$ matrix are unchanged. The $\mathbf{P}_{ii}$ are all still zero, because the path influence algorithm stops at $n-1$, meaning the loop is not accounted for. Fig. 5.3c shows a network where the loop length is less than n. In this case the loop affects the upper triangular $\mathbf{P}$ values. $\mathbf{P}_{11}$ and $\mathbf{P}_{22}$ are non-zero, showing that these nodes are involved in a loop, though it does not necessarily show they are in the same loop.

The implication is that path influence is potentially ineffective for any network containing a loop with a minimum length less than n (there can be no loops with minimum length greater than n). Path influence was originally created to answer questions of how and why X affects Y within a design formulation. Design formulations often resemble the Sen Bulker problem, meaning no loops. However, some



$$\mathbf{P} = \begin{bmatrix} 0 & 0.1 & 0.1 & 0.09 \\ 0 & 0 & 1 & 0.9 \\ 0 & 0 & 0 & 0.9 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) Loop Free Network

$$\mathbf{P} = \begin{bmatrix} 0 & 0.1 & 0.1 & 0.09 \\ 2.25 & 0 & 1 & 0.9 \\ 2.25 & 0.225 & 0 & 0.9 \\ 2.5 & 0.25 & 0.25 & 0 \end{bmatrix}$$

(b) Loop Length $= n$

$$\mathbf{P} = \begin{bmatrix} .25 & 0.125 & 0.1 & 0.09 \\ 3.125 & 0.25 & 1.25 & 0.9 \\ 0 & 0 & 0 & 0.9 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(c) Loop Length $< n$

Figure 5.3: Path Influence with Loops

formulations are used to represent an iterative synthesis process, which by its very nature is a loop, and often contains many sub loops. However, an "iteration" is the repetition of a set of steps, and that set of steps can usually be traversed in a linear fashion. This means that the loop is created by linking the end of the process with the beginning, i.e. the design spiral. As long as the minimum loop length is greater than the longest geodesic path, the case of Fig. 5.3b, path influence can be used if the algorithm is stopped short of the loop length. Networks containing loops shorter or equal to the length of the longest geodesic path can be modified by removing an offending arc, or by consolidating the nodes of a loop into a single node. In the case of Fig. 5.3c this would mean merging nodes one and two. Path influence could then be applied with confidence.

5.1.4 Adjacency Matrix Normalization and Interpolated Derivatives

The form of **A** outlined in the previous section is necessary to derive Eq. 5.13, but is not the most useful form for understanding formulation behavior. In this case, it is preferable to know the relative influence of one node over another, not necessarily the magnitude of the partial derivative. This is important for problems where the magnitude of the various functions differ by orders of magnitude. As an example, the value of the partial derivative of the Froude number with respect to length is quite small compared with that of the Reynolds number, but the Froude number itself is also quite small comparatively. Thus, the form of **A** used in the remainder of this thesis normalizes the partial derivative relative to the original function value. Mathematically this is expressed as Eq. 5.14, where a unit change in i resulting in a unit change in j corresponds to an $\mathbf{A}_{ij}$ value of 1.

$$\mathbf{A}_{ij} = \frac{f_i^0 \left. \frac{\partial f_j}{\partial f_i} \right|_{f_i^o}}{|f_j^o|} \tag{5.14}$$

Computing the partial derivatives for use in Eq. 5.14 may not always be feasible, one reason being the often discrete nature of design formulations, which often lack locally differentiable functions. Though path influence was derived as a form of Taylor series, the overall path influence concept does not necessitate partial derivative **A** weightings. Alternative adjacency matrix weighting schemes are equally valid if they accurately reflect the influence of nodes over one another. A logical way around non-differentiable functions is to compute an interpolated derivative. One example is Eq. 5.15, where the inputs to each function are increased separately by 10%, and the resulting normalized change in the output becomes the arc weight. A 10% change in an input resulting in a 10% change in the output has an arc weight of 1, making this weighting scheme directly comparable to that of Eq. 5.14 and the matrix form of MacCallum's (1982) "strengths."

let f_j be a function dependent on inputs $k, l, m, ...$, then

$$\begin{aligned}
\mathbf{A}_{kj} &= \frac{f_j((k+|0.1k|, l, m, ...) - f_j}{|f_j|(0.1)} \\
\mathbf{A}_{lj} &= \frac{f_j(k, l+|0.1l|, m, ...) - f_j}{|f_j|(0.1)} \\
\mathbf{A}_{mj} &= \frac{f_j(k, l, m+|0.1m|, ...) - f_j}{|f_j|(0.1)} \\
&...
\end{aligned} \tag{5.15}$$

The discussion so far has focused on design formulations involving quantitative functions, as in design tools. However, the path influence algorithm is not limited to quantitative functions. Networks that represent more than just quantitative functions, such as processes, organizations or any combination thereof can utilize path influence if there is a suitable weighting scheme.

5.2 Path Influence Results

A case study was conducted to compare first order objective function Taylor series expansions and the two forms of path influence against a full factorial design of experiments. The Sen Bulker problem was specifically selected for this purpose, the network and formulation are defined in Chapter III and Appendix B respectively.

A design of experiments, Taylor series expansion and path influence all require an expansion point, Eq. 5.16 defines the expansion point used unless otherwise stated. Watson (1998) provides similar principal dimensions from the "Solidarnose", a representative bulk carrier built in 1991. His dimensions have been slightly modified such that Eq. 5.16 satisfies the constraints of the Sen Bulker problem. Length (L), draft (T), depth (D), block coefficient (C_b), beam (B) and speed (V) are the entries, the independent variables of the problem.

$$\mathbf{x}^0 = [L, T, D, C_b, B, V] = [225, 12.5, 19, 0.68, 32.2, 14.5] \tag{5.16}$$

5.2.1 Objective Function Taylor Series Expansion Results

It is possible to collapse the 11 function levels into the three non-linear objectives, becoming functions of the independent variables only. This form may be more suitable for classic optimization and direct Taylor series expansion. However, designer intuition provided by the intermediate functions would be nonexistent, and the potential insight gained from viewing the objective functions alone is limited. The collapsed objective function with the fewest terms, Annual Cargo (f_3), is still too large to display on the written page. This makes makes viewing it let alone interpreting it very difficult. As derived in Sections 5.1.1 and 5.1.2 the results of an objective function Taylor series expansion and that of the corresponding partial derivative path influence matrix entry are identical, meaning the normalized results presented in Table 5.1 for Transportation Costs, Annual Cargo and Light Ship Mass are the same for both forms, and are not restated here. Their equivalence was verified to check both the path influence algorithm and the translation of the Sen Bulker problem formulation to *MATLAB*.

5.2.2 Partial Derivative Path Influence Results

Transposed portions of the P matrix are shown in Tables 5.1 and 5.2 for the variable to function and variable to constraint path weights respectively. The **A** matrix weights were calculated using the normalized partial derivative weighting from Eq. 5.14. To clarify the weighting scheme, a unit change in L is approximated to result in a negative half unit change in F_n using the expansion, thus a normalized arc weight of -0.50 in the top left entry of Table 5.1. Bold entries in the tables denote non-zero values, necessary because there are several instances where the influence exists but is too small to show. Influence is shaded on a continuum from dark red for the most negative influence in a column, to dark green for the most positive influence in a column. The **P** matrix in this format allows an engineer to quickly:

- Determine if an interaction exists
- Determine the local magnitude and sign of an interaction
- Determine the relative importance of one interaction versus another
- Determine via summation the cumulative effect of interactions

Having an indication of problem behavior is helpful in two primary ways. First, for an engineer unfamiliar with the formulation there is an indication of where solutions or problems lay, and the general structure that produces them. An example can be seen by looking at the line for Transportation Costs. Recalling that lower costs are better, the strongest indicators are for a short, deep drafted, low freeboard and slow ship. Block Coefficient, and Beam are weak indicators, more liable to inaccurate trending (the actual block coefficient $\mathbf{P}$ value is 0.04). Using Power (P) as another example, any underwater dimension increases power, but beam, draft and block coefficient more than length. Above all however, speed increases required power. These results indicate that the formulation will behave as would be expected to optimize the dimensions of a bulk carrier. The second helpful contribution is that path influence analysis provides an easy verification for the engineer developing the formulation. If an error is made then it could show up as an odd or unexpected weighting scheme. Rather than going line by line through the code looking for errors, odd path weights can quickly be traced back until the error is found. This was experienced first hand.

Interpreting the path influence results for the constraints in Table 5.2 is similar to that for the functions in Table 5.1. However, the magnitude of several entries is significantly higher. A constraint that lies on its boundary has a value of zero, thus using the normalization scheme of Eq. 5.14 or Eq. 5.15 would result in an infinite arc weight. Similarly, constraints near their boundary have small values, potentially leading to high arc weights. Thus those constraints in Table 5.2 with highly influential inputs such as V's influence on g_{10}, could be examined for proximity to a constraint boundary. This is the case of g_{10}. If a variable is beneficial to an objective and highly detrimental to a constraint, or the inverse, this could be a lead indicator of constraint activity in an optimization problem. This line of reasoning is discussed in Section 5.3.

5.2.3 Interpolated Derivative Path Influence Results

The interpolated derivative path influence **P** matrix was formed from the **A** matrix weighting scheme of Eq. 5.15. To clarify the weighting scheme, a +10% change in L resulted in a -4.7% change in F_n, thus a normalized arc weight of -0.47 in the top left entry of Table 5.3. This weighting method allows the interpolated and partial derivative weighting schemes to be directly compared. Transposed portions of the **P** matrix are shown in Tables 5.3 and 5.4 for the variable to function and variable to constraint path weights respectively. The results of interpolated and partial derivative path influence are very consistent and the interpretation is mostly the same. The interpolated derivative weighting predicts the correct trend of Transportation Cost with Block Coefficient, but flips it for Beam (actual value -0.10). Again, these are the two weakest indicators.

5.2.4 Path Influence Accuracy

Like a Taylor series, path influence results are approximations of actual formulation behavior. The partial derivative weighting scheme will have the same error as a first order Taylor series by definition, and the interpolated derivative error also depends on the validity of a linear extrapolation around the baseline point. For path influence results to be useful the magnitudes of entries in the **P** matrix must be close enough to draw conclusions about the relative influence of one variable/function versus another, while the sign of entries is perhaps more important. A fully enumerated DOE using the six independent variables was conducted, allowing the error inherent in both path influence weighting schemes to be calculated. The DOE used zero and

Table 5.1: Transposed Partial Derivative **P** Matrix - Functions

#	Node	L	T	D	C_b	B	V
7	F_n	**-0.50**	0.00	0.00	0.00	0.00	**1.00**
8	Steel Mass	**1.70**	0.00	**0.40**	**0.50**	**0.70**	0.00
9	Outfit Mass	**0.80**	0.00	**0.30**	**0.10**	**0.60**	0.00
10	a	0.00	0.00	0.00	**-0.73**	0.00	0.00
11	b	0.00	0.00	0.00	**-0.40**	0.00	0.00
12	Δ	**1.00**	**1.00**	0.00	**1.00**	**1.00**	0.00
13	Sea Days	0.00	0.00	0.00	0.00	0.00	**-1.00**
14	P	**0.31**	**0.67**	0.00	**1.63**	**0.67**	**3.71**
15	A_{co}	**0.36**	0.00	0.00	**-0.96**	0.00	**-0.71**
16	Ship Costs	**0.94**	**0.11**	**0.25**	**0.49**	**0.58**	**0.63**
17	Machinery Mass	**0.28**	**0.60**	0.00	**1.46**	**0.60**	**3.34**
18	Daily Consumption	**0.31**	**0.66**	0.00	**1.62**	**0.66**	**3.69**
19	Fuel Cost	**0.31**	**0.66**	0.00	**1.62**	**0.66**	**2.69**
20	Light Ship Mass	**1.54**	**0.02**	**0.37**	**0.49**	**0.68**	**0.13**
21	Fuel Carried	**0.31**	**0.66**	0.00	**1.62**	**0.66**	**2.94**
22	Capital Charges	**0.94**	**0.11**	**0.25**	**0.49**	**0.58**	**0.63**
23	Δ_{DW}	**0.87**	**1.23**	**-0.09**	**1.12**	**1.08**	**-0.03**
24	Running Costs	**0.26**	**0.37**	**-0.03**	**0.34**	**0.32**	**-0.01**
25	Port Costs	**0.70**	**0.99**	**-0.07**	**0.90**	**0.86**	**-0.02**
26	Stores&Water	**0.44**	**0.62**	**-0.04**	**0.56**	**0.54**	**-0.02**
27	Voyage Costs	**0.48**	**0.81**	**-0.03**	**1.30**	**0.75**	**1.48**
28	Δ_{Cargo}	**0.88**	**1.25**	**-0.09**	**1.12**	**1.09**	**-0.07**
29	Port Days	**0.82**	**1.15**	**-0.08**	**1.04**	**1.00**	**-0.06**
30	RTPA	**-0.39**	**-0.56**	**0.04**	**-0.50**	**-0.49**	**0.55**
31	Annual Cargo	**0.49**	**0.69**	**-0.05**	**0.62**	**0.60**	**0.48**
32	Annual Costs	**0.65**	**0.19**	**0.15**	**0.52**	**0.47**	**0.78**
33	Transportation Costs	**0.16**	**-0.50**	**0.20**	**-0.10**	**-0.13**	**0.30**
34	BM	0.00	**-1.00**	0.00	**0.04**	**2.00**	0.00
35	KG	0.00	0.00	**0.91**	0.00	0.00	0.00
36	KB	0.00	**1.00**	0.00	0.00	0.00	0.00
37	GM	0.00	**-0.07**	**-3.87**	**0.10**	**5.34**	0.00

Legend

Maximum	Zero	**Minimum**

Table 5.2: Transposed Partial Derivative **P** Matrix - Constraints

#	Node	L	T	D	C_b	B	V
47	g_1	**-7.08**	0.00	0.00	0.00	**7.08**	0.00
48	g_2	**3.75**	0.00	**-3.75**	0.00	0.00	0.00
49	g_3	**18.00**	**-18.00**	0.00	0.00	0.00	0.00
50	g_4	**-7.73**	**16.69**	**0.79**	**-9.97**	**-9.55**	**0.27**
51	g_5	0.00	**8.33**	**-8.87**	0.00	0.00	0.00
52	g_6	**-0.92**	**-1.31**	**0.10**	**-1.19**	**-1.14**	**0.03**
53	g_7	**0.10**	**0.14**	**-0.01**	**0.13**	**0.12**	**0.00**
54	g_8	0.00	0.00	0.00	**-13.60**	0.00	0.00
55	g_9	0.00	0.00	0.00	**9.71**	0.00	0.00
56	g_{10}	0.00	0.00	0.00	0.00	0.00	**-29.00**
57	g_{11}	0.00	0.00	0.00	0.00	0.00	**4.14**
58	g_{12}	**-0.49**	0.00	0.00	0.00	0.00	**0.98**
59	g_{13}	0.00	**0.61**	**33.20**	**-0.82**	**-38.18**	0.00

Legend		
Maximum	Zero	**Minimum**

+10% as the two possible states for each variable. A DOE only links independent variable inputs to function outputs, meaning that only a subset of the full path influence matrix is comparable. For example, there are 6 variables and 44 functions in the Sen Bulker problem, so the DOE results form a 6×44 matrix for each DOE combination. Path influence produces a single 59×59 **P** matrix. The influence of multiple variable changes is computed with the same logic as Eq. 5.13, but as only the variable to function path influence results are comparable a 6×44 subset was used instead of the full 59×59 **P** matrix. In other words, a majority of the path influence results were not validated against a full factorial DOE, because the DOE did not produce comparable results.

5.2.4.1 Overall Path Influence Accuracy

For the results that are comparable, the network methods performed very well overall. Deviation is defined as the percentage difference between a path influence result and the comparable exact DOE result relative to the baseline function value.

Table 5.3: Transposed Interpolated Derivative **P** Matrix - Functions

#	Node	L	T	D	C_b	B	V
7	F_n	**-0.47**	0.00	0.00	0.00	0.00	**1.00**
8	Steel Mass	**1.76**	0.00	**0.39**	**0.49**	**0.69**	0.00
9	Outfit Mass	**0.79**	0.00	**0.29**	**0.10**	**0.59**	0.00
10	a	0.00	0.00	0.00	**-0.54**	0.00	0.00
11	b	0.00	0.00	0.00	**-0.56**	0.00	0.00
12	Δ	**1.00**	**1.00**	0.00	**1.00**	**1.00**	0.00
13	Sea Days	0.00	0.00	0.00	0.00	0.00	**-0.91**
14	P	**0.30**	**0.66**	0.00	**1.03**	**0.66**	**4.07**
15	A_{co}	**0.33**	0.00	0.00	**-0.54**	0.00	**-0.71**
16	Ship Costs	**1.05**	**0.33**	**0.24**	**0.72**	**0.79**	**2.06**
17	Machinery Mass	**0.27**	**0.59**	0.00	**0.92**	**0.59**	**3.65**
18	Daily Consumption	**0.30**	**0.65**	0.00	**1.02**	**0.65**	**4.05**
19	Fuel Cost	**0.30**	**0.65**	0.00	**1.02**	**0.65**	**3.14**
20	Light Ship Mass	**1.59**	**0.02**	**0.36**	**0.46**	**0.67**	**0.14**
21	Fuel Carried	**0.30**	**0.65**	0.00	**1.02**	**0.65**	**3.37**
22	Capital Charges	**1.05**	**0.33**	**0.24**	**0.72**	**0.79**	**2.06**
23	Δ_{DW}	**0.86**	**1.23**	**-0.09**	**1.13**	**1.08**	**-0.03**
24	Running Costs	**0.25**	**0.36**	**-0.03**	**0.33**	**0.31**	**-0.01**
25	Port Costs	**0.68**	**0.98**	**-0.07**	**0.90**	**0.85**	**-0.03**
26	Stores&Water	**0.42**	**0.60**	**-0.04**	**0.55**	**0.53**	**-0.02**
27	Voyage Costs	**0.47**	**0.80**	**-0.03**	**0.96**	**0.74**	**1.73**
28	Δ_{Cargo}	**0.87**	**1.25**	**-0.09**	**1.14**	**1.09**	**-0.07**
29	Port Days	**0.80**	**1.15**	**-0.08**	**1.05**	**1.01**	**-0.07**
30	RTPA	**-0.37**	**-0.53**	**0.04**	**-0.49**	**-0.46**	**0.48**
31	Annual Cargo	**0.50**	**0.71**	**-0.05**	**0.65**	**0.62**	**0.40**
32	Annual Costs	**0.72**	**0.32**	**0.15**	**0.60**	**0.60**	**1.70**
33	Transportation Costs	**0.26**	**-0.33**	**0.19**	**0.01**	**0.04**	**1.33**
34	BM	0.00	**-0.91**	0.00	**0.03**	**2.10**	0.00
35	KG	0.00	0.00	**0.91**	0.00	0.00	0.00
36	KB	0.00	**1.00**	0.00	0.00	0.00	0.00
37	GM	0.00	**0.17**	**-3.87**	**0.09**	**5.60**	0.00

Legend		
Maximum	Zero	**Minimum**

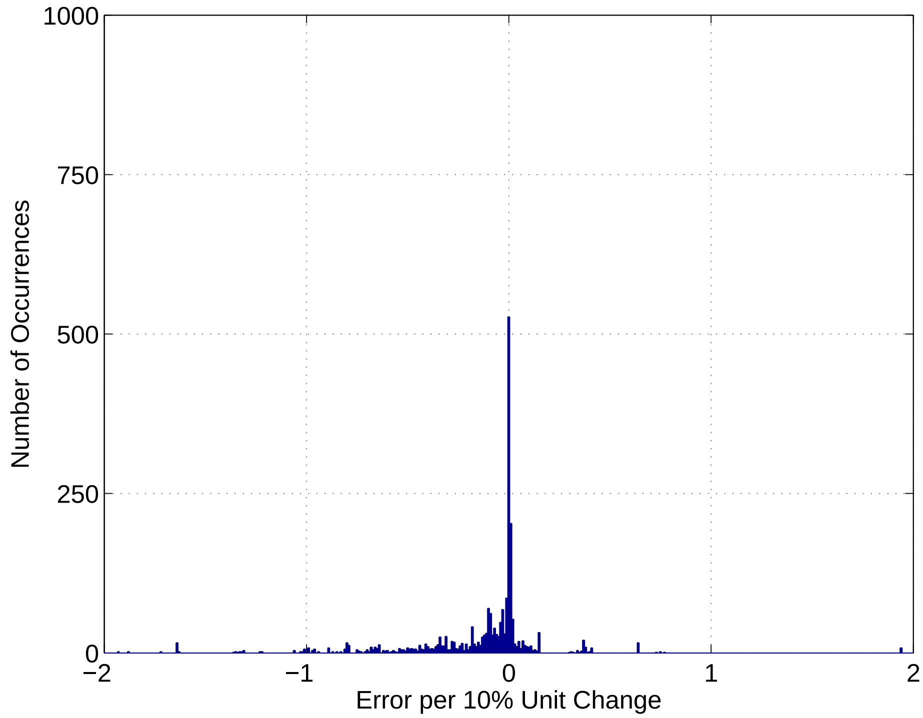
Table 5.4: Transposed Interpolated Derivative **P** Matrix - Constraints

#	Node	L	T	D	C_b	B	V
47	g_1	**-7.08**	0.00	0.00	0.00	**6.43**	0.00
48	g_2	**3.75**	0.00	**-3.41**	0.00	0.00	0.00
49	g_3	**18.00**	**-16.36**	0.00	0.00	0.00	0.00
50	g_4	**-7.38**	**17.05**	**0.75**	**-9.71**	**-9.26**	**0.29**
51	g_5	0.00	**8.33**	**-8.87**	0.00	0.00	0.00
52	g_6	**-0.91**	**-1.31**	**0.09**	**-1.20**	**-1.15**	**0.04**
53	g_7	**0.10**	**0.14**	**-0.01**	**0.13**	**0.12**	**0.00**
54	g_8	0.00	0.00	0.00	**-13.60**	0.00	0.00
55	g_9	0.00	0.00	0.00	**9.71**	0.00	0.00
56	g_{10}	0.00	0.00	0.00	0.00	0.00	**-29.00**
57	g_{11}	0.00	0.00	0.00	0.00	0.00	**4.14**
58	g_{12}	**-0.46**	0.00	0.00	0.00	0.00	**0.98**
59	g_{13}	0.00	**-1.47**	**33.20**	**-0.75**	**-40.46**	0.00

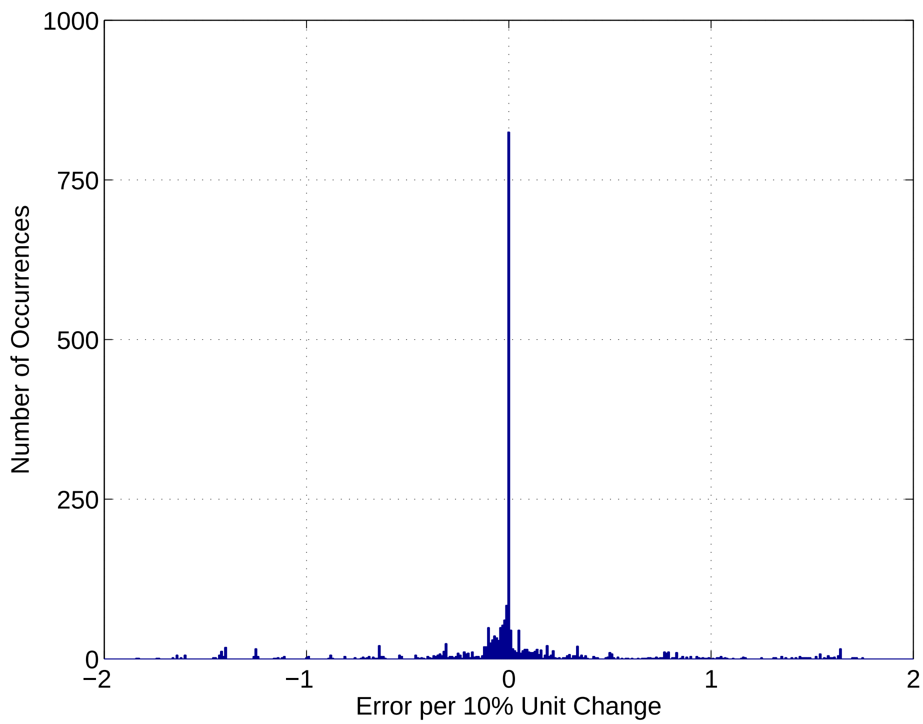
Legend		
Maximum	Zero	**Minimum**

Fig. 5.4 shows histograms of deviation for both path weighting schemes. The abscissa displays the magnitude and direction of deviation normalized by 10%. A deviation value of one means the path influence matrix predicted response is 10% higher than the exact DOE response relative to the baseline function value. Nodes which are not connected by any path were not included in the calculations, as neither a DOE or path influence evaluate the influence of disconnected nodes. In this case partial derivative path influence is +/-0.005 (+/-.05%) accurate 22.7% of the time, while interpolated derivative path influence is 34.8% accurate. More data on the distributions is shown in Table 5.5, and statistics can be found in Table 5.6, where μ and σ are the mean and standard deviation respectively.

Trend accuracy is defined as the percentage of occurrences that path influence correctly identified the sign of influence. Path influence predictions are 99.0% and 97.4% trend accurate over all 64 variable combinations of the DOE for partial derivative and interpolated derivative weighting schemes respectively. When only single variable changes occur, trend accuracy is 98.9% and 99.6% respectively.



(a) Partial Derivative Path Weighting



(b) Interpolated Derivative Path Weighting

Figure 5.4: Path Influence Deviation Histograms

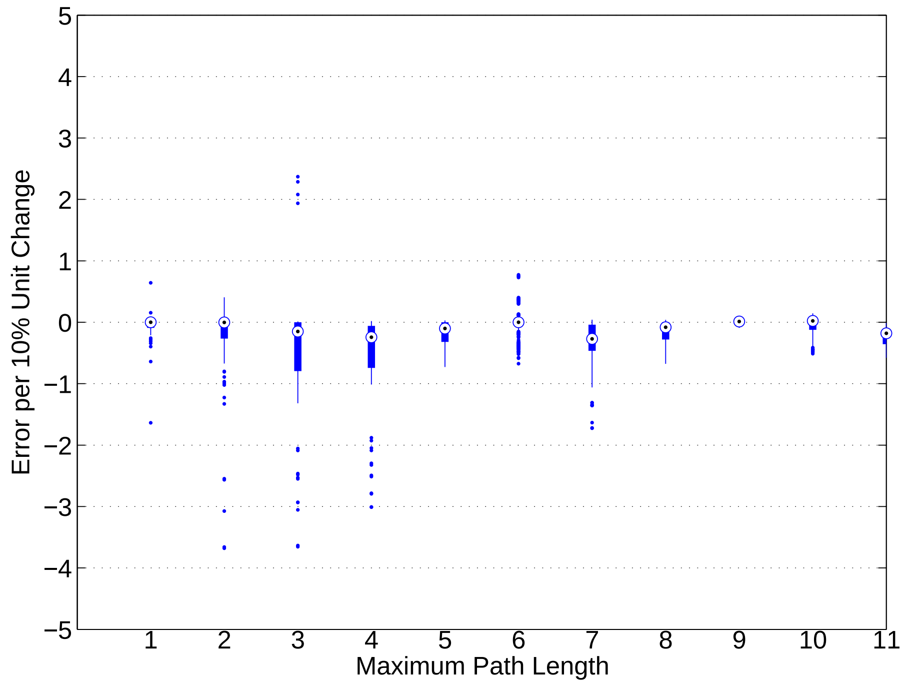
Table 5.5: Path Influence Error Distributions

Error Bounds +/-	Percentage within Error Bounds Partial Derivative	Interpolated Derivative
0.05%	22.9%	34.8%
1%	57.5%	60.1%
5%	84.8%	79.4%
10%	93.0%	87.3%
20%	95.7%	95.9%
30%	98.8%	98.0%
40%	100.0%	99.3%
50%	100.0%	100.0%

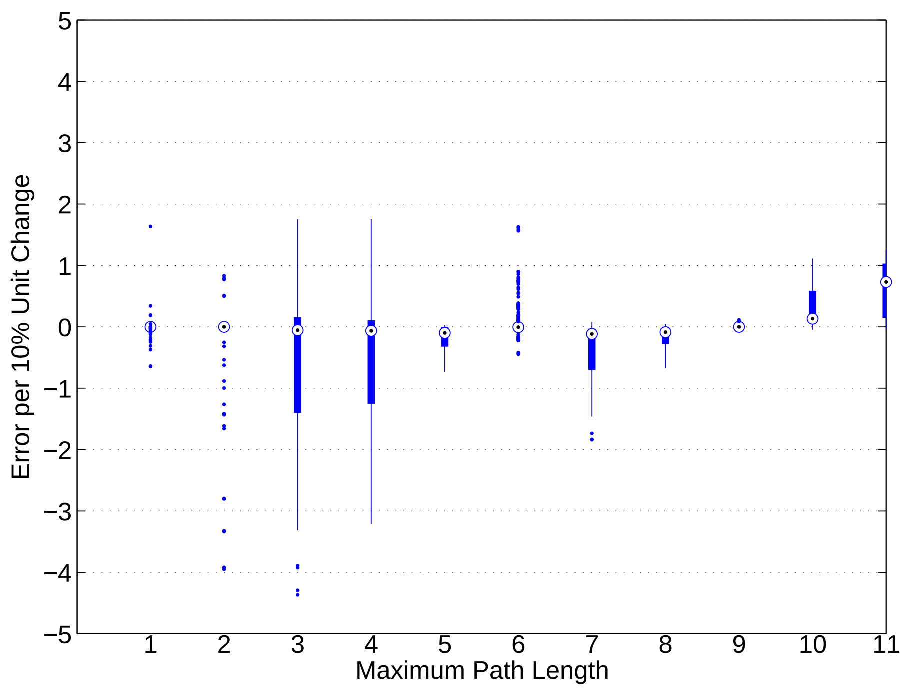
5.2.4.2 Accuracy as a Function of Path Length

A further question regarding path influence accuracy is whether it is path length dependent. Deviation box plots for the maximum and geodesic path lengths between nodes (nodes can be connected by self avoiding paths of multiple lengths) are shown in Figs. 5.5 and 5.6. The shape of the error distributions varies somewhat visually, but the mean, median and mode do not correlate with path length as evidenced by the distribution statistics for the maximum path lengths, Fig. 5.5 and Table 5.6.

However, three of the four box plots show a rough decrease of standard deviation as path length increases. The narrowing is likely due to the fact that there are many fewer paths of higher length, and they all point toward a smaller set of nodes. For example, there only 28 paths 11 arcs long and all point toward Transportation Cost, out of 1838 total paths in the network. The inverted triangular structure of the network visually shows the narrowing, Fig. 3.4. For path lengths 10 and 11 using the interpolated derivative weighting, the standard deviation and median increase. Paths of length 10 and 11 all go through the same node, RTPA. If RTPA is off, then every path of length 10 and 11 will be affected.

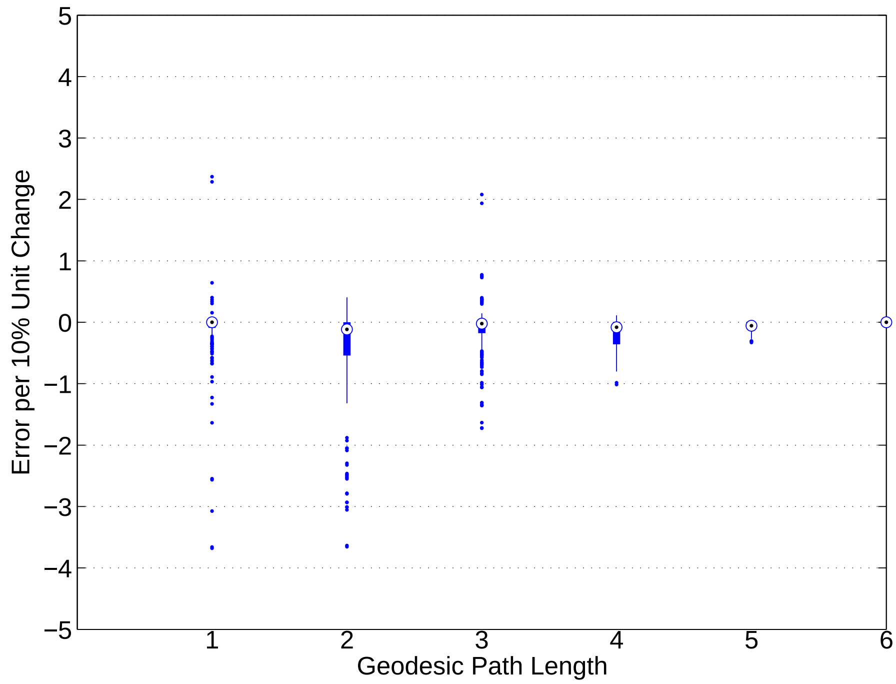


(a) Partial Derivative Path Weighting Box Plot

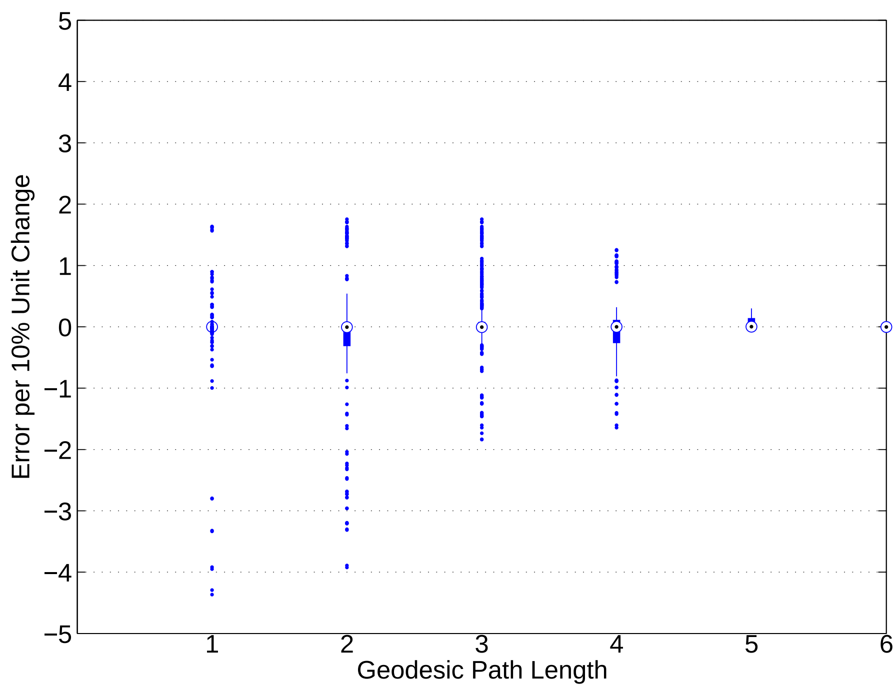


(b) Interpolated Derivative Path Weighting Box Plot

Figure 5.5: Path Influence Error Parsed by Maximum Path Length



(a) Partial Derivative Path Weighting Box Plot



(b) Interpolated Derivative Path Weighting Box Plot

Figure 5.6: Path Influence Error Parsed by Geodesic Path Length

Table 5.6: Deviation Distribution Statistics Parsed by Maximum Path Length

	Partial Derivative Weighting				Interpolated Derivative Weighting			
Max Length	μ	Median	Mode	σ	μ	Median	Mode	σ
All Non-Zero	-0.19	-0.03	-0.01	0.63	-0.11	0.00	-0.01	0.80
1	-0.06	0.00	-0.01	0.27	0.01	0.00	-0.01	0.27
2	-0.32	0.00	-0.01	0.85	-0.23	0.00	0.00	1.02
3	-0.34	-0.15	-0.01	1.31	-0.68	-0.06	-0.01	1.60
4	-0.56	-0.24	-0.02	0.77	-0.33	-0.06	-0.01	1.16
5	-0.16	-0.10	0.00	0.19	-0.16	-0.10	0.00	0.19
6	-0.04	0.00	0.01	0.19	0.06	-0.01	0.00	0.26
7	-0.36	-0.27	0.00	0.44	-0.39	-0.11	0.00	0.51
8	-0.14	-0.08	0.01	0.18	-0.14	-0.09	0.00	0.18
9	0.01	0.01	-0.05	0.06	0.01	0.00	-0.01	0.04
10	-0.06	0.02	0.07	0.17	0.29	0.13	0.08	0.34
11	-0.23	-0.18	-0.23	0.16	0.59	0.73	0.14	0.45

5.2.4.3 Accuracy Conclusions

The low deviation and more importantly high trend accuracy of both weighting methods renders reasonable confidence in the lead indicators drawn from path influence. This accuracy is no doubt partially due to the relatively simple behavior of the Sen Bulker problem, but it is representative in complexity of preliminary design tools which are used when predictive metrics can provide the most value. For networks that represent the context of process, organizations, etc., the weightings are likely to be less complex, and similar or higher accuracy should be expected. Chapter VII demonstrates examples of this.

For the Sen Bulker problem specifically, it is not surprising that the interpolated derivative weighting scheme would be slightly more accurate within the +/- 0.05% error bounds because the DOE used the same 10% multiplicative factor that was used to create the interpolated derivative weights. However, as shown in Table 5.5, over the remainder of the error bounds the partial derivative weighting scheme was on par or more accurate than the interpolated derivative weighting scheme. Furthermore, the partial derivative weighting scheme systematically under predicted the results, and with much less standard deviation. This is evident from the box plots and deviation distribution plot. Consistent, rather than more exact, predictions and a higher overall trend accuracy give the partial derivative weighting scheme an edge. Though the overall accuracy of the two weighting methods is similar, where error does occur can be very important.

5.2.5 Path Influence vs. DOE Computation

Path influence has a decided advantage over full factorial DOEs where function call count is concerned. Either the partial derivative or interpolated derivative weighting scheme requires a value for each node at the baseline and another for each input to that node, totaling $n + \sum \mathbf{k}_{in}$ function calls for an entire network if every node is a function. The Sen Bulker problem has 44 functions with a total in-degree of 95, summing to $44 + 95 = 139$ function calls. By comparison, the full factorial DOE consists of six variables with two states, $2^6 = 64$ total combinations, each requiring 44 function calls. This makes the total number of function calls $44 \times 64 = 2816$ for the DOE. This means that path influence requires less than 5% of the DOE function call count for the Sen Bulker problem, demonstrating that from a function standpoint path influence is much less computationally intensive. The 5% assumes that the DOE is computing the intermediate functions, trying to replicate (though not fully) the volume of information available from path influence. Even if only the three objectives are evaluated the DOE still requires $64 \times 3 = 192$ function calls which is more than path influence.

A fair comparison of actual run time would have to account for the matrix manipulation inherent with generating $\mathbf{P}$ and determining the partial derivatives if that weighting scheme is used. The Sen Bulker problem is too simple for such a comparison, computing in less than 1/10th of a second for both cases. Faster path length algorithms were discussed briefly in Section 5.1.2, but these and time studies are a matter for future research when the size of the networks demand it.

5.3 Capturing Optimization Behavior with Path Influence

It was originally envisioned that metrics like path influence could give insight into "optimization behavior" without having to exhaustively explore the objective space. The results of the previous sections show that this is achievable when the behavior of the formulation is under investigation. However, this is distinctly different from predicting the behavior of an optimization tool using that formulation. The static structural analysis results of the Sen Bulker network varied significantly when arcs representing constraint feedback were added, as shown in Section 4.2.2. From an optimization standpoint, constraints really only affect a problem when they are active, or nearly active. If an optimization is unconstrained, path influence results can already show how variables affect objectives and by how much. This means that predicting constraint activity is the area requiring focus if path influence is to be used for determining overall optimization behavior.

5.3.1 Optimization Verification

To determine overall constraint activity, it was necessary to run and verify the optimization problem, namely by comparing it with the published optimization results of Sen and Yang (1998). The formulation was encoded into *MATLAB*, each equation verified using an optimal point from the published results. Interestingly, Sen and Yang (1998)'s published formulation contains dimensional errors, specifically in Eq. B.11, where V should be in m/s but was left in kts while Fn remained non-dimensional. These errors were corrected for the analysis in all other sections, but not here because the purpose was to verify with known optimization behavior. A minimization problem was solved for each objective individually with a *MATLAB* standard Sequential Quadratic Programming implementation to identify the bounds of the Pareto front,

f_1^o, f_2^o, and f_3^o respectively. The front itself was resolved using a min max optimization scheme as outlined in Eqs. 5.17 to 5.19, consistent with Sen and Yang (1998)'s approach. If it is assumed that the objective scaling in Eq. 5.19 is perfect, then the optimizer should seek each objective's optimum with equal vigor, resulting in a balanced single solution. By varying the weighting on each objective, w_k of Eq. 5.18, different solutions are created which can be culled to form a non-dominated set. This set becomes the Pareto front for the problem. It was discovered that f_2^o and f_3^o match with Sen and Yang (1998), while f_1^o was slightly better. Each two-dimensional Pareto front was verified as shown in Figs. 5.7 and 5.8, where the solid line is Sen and Yang (1998)'s results and the asterisks are the bounds of the front. The minor discrepancies are due to the coarse nature of Sen and Yang (1998)'s printed results, evident from the fact that f_2^o and f_3^o are not always the endpoints of their respective fronts.

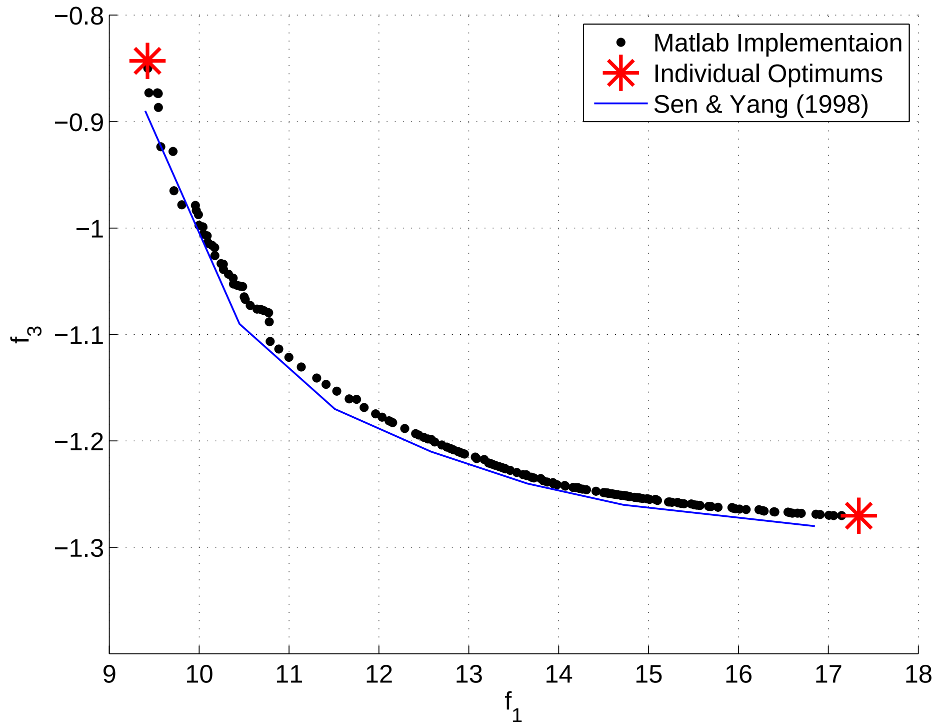
$$\begin{gathered}\min_x \max_k \left[w_k z_k(x)\right] \\ \text{s.t.} \quad g(x) \leq 0\end{gathered} \tag{5.17}$$

$$z_k(x) = \frac{|f_k(x) - f_k^o|}{|f_k^o|} \quad \text{and} \quad w_k = [0, 1] \tag{5.18}$$

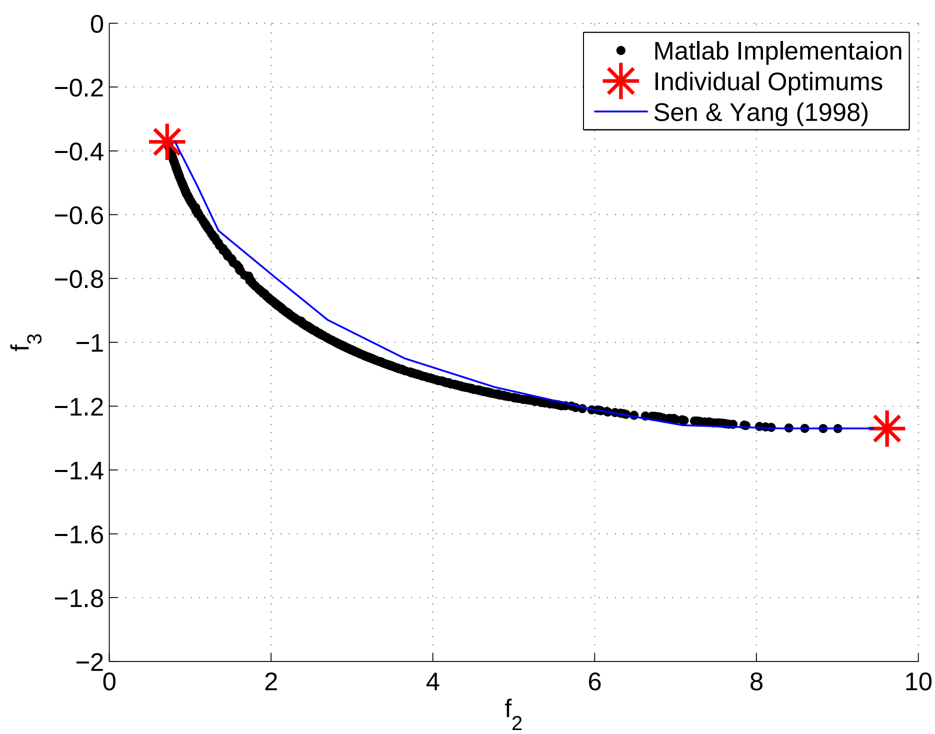
$$\begin{aligned} f_1(x) &= \text{Transportation Cost} \quad \left[\frac{\pounds}{tonne}\right] \\ f_2(x) &= \frac{\text{Light Ship Mass}}{10000} \quad \left[10^3\ tonnes\right] \\ f_3(x) &= -\frac{\text{Annual Cargo}}{1000000} \quad \left[10^6\ tonnes\right] \end{aligned} \tag{5.19}$$

5.3.2 Method for Predicting Constraint Activity with Path Influence

There is no single optimum for the Sen Bulker problem, but a three-dimensional Pareto front between the three objective optimums. Along this front there is no single set of active constraints, meaning to determine constraint activity a single solution point is required. This research used the three separate objective optimums (f_1^o, f_2^o, f_3^o) as points to determine constraint activity. This was thought a simpler

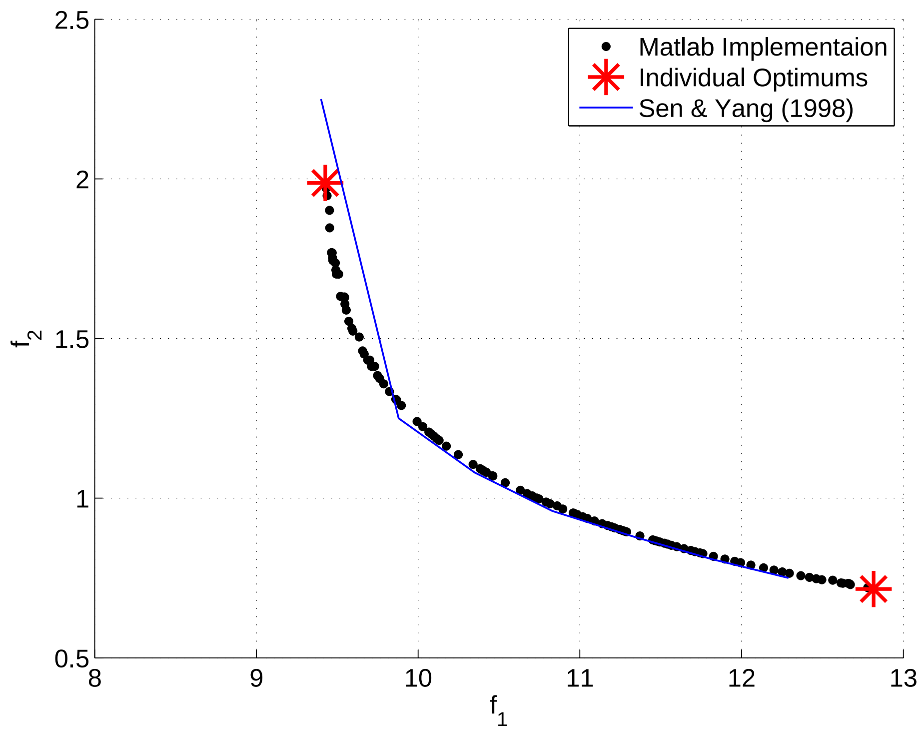


(a) f_3 vs. f_1



(b) f_3 vs. f_2

Figure 5.7: Pareto Front Verification Part One



(a) f_2 vs. f_1

Figure 5.8: Pareto Front Verification Part Two

scenario to test the concept than a multi-objective problem. To predict the constraint activity resulting from a single objective optimization, a sign adjusted column of **P** corresponding to the variable inputs of an objective was multiplied element by element to a corresponding constraint column of **P**. The resulting vector was summed to form a single number. The logic is that what is "good" for an objective in the path influence matrix is a positive value for maximization, while what is "bad" for a constraint is also a positive value (inactive constraints are less than zero). If the two are multiplied then a positive result indicates an active constraint, as what is driving a better objective is pushing toward a constraint boundary, or the opposite. Similarly, a negative value means that either the objective is being decreased, or the constraint is being decreased, either one of which would logically lead to an inactive constraint. See Table 5.7. This method is simply a selective extension of what was done to form the path influence matrix in the first place, which was demonstrated to be very effective at predicting trends.

Table 5.7: Constraint Trending for a Maximization Problem

Objective **P** Entry		Constraint **P** Entry		Result	
Good +	×	Good −	=	−	Inactive
Good +	×	Bad +	=	+	Active
Bad −	×	Good +	=	−	Inactive
Bad −	×	Bad −	=	+	Active

5.3.3 Constraint Activity Results & Conclusions

The results of predicting constraint activity for both weighting methods is shown in Table 5.8. The results are not promising for either method. Unfortunately, knowing that a constraint is likely to trend toward a boundary cannot predict activity, as there is no indication in the network for the proximity to the boundary. The exception is when the expansion point is very near a boundary, resulting in very high **P** matrix values as described in Section 5.2.2. However, high constraint **P** values are not necessarily linked to boundary proximity. Future work in this area should investigate alternative methods which may include representing the constraint constant in the network to locate the constraint boundary.

Table 5.8: Constraint Activity Prediction Results

Objective	Partial Derivative Weighting	Interpolated Derivative Weighting
f_1^0	69%	92%
f_2^o	69%	69%
f_3^o	38%	38%
Average	59%	67%

5.4 Comparing Static and Dynamic Analysis Methods

Rather than one network formulation or metric being better than another, they complement each other. The static network structural metrics provide a simpler and designer intent focused view, while the dynamic metrics provide a quantitative view of the implementation of that designer intent. Park centrality results were used to describe "impact" in Chapter IV, which was never defined. In this chapter, influence has been defined using the weighting schemes of Eqs. 5.14 and 5.15. In both cases, effects received by a node and transmitted by a node are measured. This makes Park centrality and path influence comparable, albeit with some manipulation and caveats.

5.4.1 Method

To compare path influence with Park centrality, a new metric termed *Winston centrality* was developed, Eq. 5.20. The sum of entries in the $\mathrm{i_{th}}$ column of the absolute value $\mathbf{P}$ matrix is subtracted from the sum of entries in the $\mathrm{i^{th}}$ row of the absolute value $\mathbf{P}$ matrix. The absolute values are necessary because influence can be positive or negative, equally analogous to impact.

$$p_i = \sum_{j=1}^{n} |\mathbf{P}_{ij}| - \sum_{j=1}^{n} |\mathbf{P}_{ji}| \tag{5.20}$$

Winston centrality is similar to Park centrality in that overall "losses" are subtracted from overall "wins," bearing in mind that that each entry in the **P** matrix already takes into account the indirect "wins" and "losses" accounted for in Park centrality. In form Winston centrality is also similar to in-degree subtracted from out-degree for a node in an unweighted network, Eq. 5.21.

$$\mathbf{k}^i_{out} - \mathbf{k}^i_{in} = \sum \mathbf{A}_i - \sum \mathbf{A}_{|i|} \tag{5.21}$$

5.4.2 Results

Park centrality and Winston centrality rankings are shown in Tables 5.9 and 5.10, where one and dark green is the highest ranked node, on a continuum to 59 and dark red for the lowest ranked node. Two major interpretations are that Park and Winston centrality rankings are quite similar, and that the two path influence weighting schemes show nearly identical results. The conclusion is that static network structural analysis is indicative of dynamic analysis, and can be used as a lead indicator of problem behavior.

There are three discrepancies between Park and Winston centrality rankings worth mentioning due to an interesting correlation with betweenness centrality. Power (P) is ranked near the top for Park centrality, but near the bottom for Winston centrality. This indicates that Power is not as influential as its place in the network structure would suggest, implying that its contributions to other functions, though structurally significant or numerous, are not nearly as important as the contributions it receives. What is interesting is that Power has the highest betweenness centrality in the network. Machinery Mass also has a ranking discrepancy, probably because its sole input is Power. However, Machinery Mass has the fourth highest betweenness in the network. Finally, deadweight (Δ_{DW}) has another discrepancy, mid range for Park centrality but near the top for Winston centrality. Deadweight has nearly balanced in-degree and out-degree, helping to explain the mid range Park Ranking. However, Deadweight has the second highest betweenness centrality. In summary, three major discrepancies between the two rankings are attached to three of the top four betweenness values in the network. As both path influence and betweenness rely on path calculations, this is a subject worthy of future investigation.

Table 5.9: Park and Path Influence Ranking Comparison Part One

#	Node	Park Centrality →	Partial Derivative	Interpolated Derivative
1	L	2	7	6
2	T	5	3	4
3	D	6	6	5
4	C_b	1	5	7
5	B	3	1	1
6	V	4	4	2
7	F_n	8	19	18
8	Steel Mass	17	22	22
9	Outfit Mass	18	25	25
10	a	15	21	20
11	b	16	28	26
12	*Delta*	7	13	13
13	Sea Days	20	23	23
14	P	21	51	50
15	A_{co}	43	50	51
16	Ship Costs	49	36	44
17	Machinery Mass	24	53	52
18	Daily Consumption	39	54	53
19	Fuel Cost	44	56	56
20	Light Ship Mass	38	27	28
21	Fuel Carried	42	55	55
22	Capital Charges	47	41	47
23	Δ_{DW}	37	17	17
24	Running Costs	48	29	29
25	Port Costs	46	35	35
26	Stores&Water	45	31	31
27	Voyage Costs	53	49	49
28	Δ_{Cargo}	56	34	34
29	Port Days	55	39	38

Legend

Maximum	Minimum

Table 5.10: Park and Path Influence Ranking Comparison Part Two

#	Node	Park Centrality →	Partial Derivative	Interpolated Derivative
30	RTPA	54	33	32
31	Annual Cargo	57	37	37
32	Annual Costs	58	43	45
33	Transportation Costs	59	47	48
34	BM	31	12	12
35	KG	25	9	9
36	KB	26	11	11
37	GM	40	44	41
38	ζ_1	9	16	16
39	ζ_2	10	14	14
40	ζ_3	11	15	15
41	η_1	12	10	10
42	η_2	13	2	3
43	η_3	14	8	8
44	Round Trip Miles	19	18	19
45	Fuel Price	23	24	24
46	Cargo Handling Rate	22	20	21
47	g_1	32	46	42
48	g_2	33	38	36
49	g_3	34	57	57
50	g_4	52	58	58
51	g_5	35	48	46
52	g_6	50	40	39
53	g_7	51	26	27
54	g_8	27	45	43
55	g_9	28	42	40
56	g_{10}	29	52	54
57	g_{11}	30	32	33
58	g_{12}	36	30	30
59	g_{13}	41	59	59

Legend

Maximum	Minimum

5.5 Further Applications of Winston Centrality

Winston centrality was used in the previous section to compare path influence with Park centrality, but this is by no means its only possible use. Pasqual and de Weck (2012) introduced several metrics derived from a design activity network to quantify change propagation characteristics for individual engineers. Their method has major drawbacks as discussed previously, but it does have parallels to influence propagation and possible interpretations for Winston centrality. Winston centrality results for the Sen Bulker problem are shown in Tables 5.11 and 5.12.

Zero Winston centrality indicates that a node is a receiver and transmitter of no influence, or it transmits exactly the same amount of influence as it receives. In either case it adds no new influence to the network, but might serve as an influence sorter. Outfit Mass is an example of an influence sorter, in that it has near zero Winston centrality but both receives and transmits influence. A node with negative Winston centrality receives more than it distributes, making it a damper or dead end for influence propagation through the network. The Admiralty coefficient (A_{co}) already identified as a dead end with other methods confirms this interpretation. A node with positive Winston centrality may be seen as a multiplier of influence, or highly influential alone. As would be expected, each of the variables has very high Winston centrality. Also of interest, the parameters η_2 and η_3 have very high Winston centrality, especially when compared with other constants and parameters such as the ζ values or Cargo Handling Rate. This indicates the sensitivity of the Sen Bulker problem to parameter changes.

Analyzing Winston centrality from an optimization viewpoint, the variables should be sources of influence and the objectives sinks of influence. This is unquestionably verified for the variables, and each of the the three objectives do have negative influence, though not by much for Light Ship Mass. Similar results were seen with Park centrality, and the Sen Bulker problem is unusual in that two of the three objectives

Table 5.11: Winston Centrality Results Part One

#	Node	Partial Derivative	Interpolated Derivative
1	L	53.50	53.38
2	T	60.01	59.90
3	D	53.96	53.49
4	C_b	56.77	53.14
5	B	78.13	80.20
6	V	58.69	65.50
7	F_n	4.74	5.95
8	Steel Mass	1.98	1.88
9	Outfit Mass	0.03	0.05
10	a	2.09	2.25
11	b	-1.49	-0.47
12	Δ	20.27	20.21
13	Sea Days	1.89	1.87
14	P	-26.73	-23.11
15	A_{co}	-26.51	-26.06
16	Ship Costs	-5.95	-15.79
17	Machinery Mass	-29.63	-27.22
18	Daily Consumption	-30.11	-27.58
19	Fuel Cost	-35.15	-32.72
20	Light Ship Mass	-1.17	-1.18
21	Fuel Carried	-34.62	-32.16
22	Capital Charges	-7.95	-17.79
23	Δ_{DW}	9.20	8.87
24	Running Costs	-1.82	-1.74
25	Port Costs	-5.26	-5.19
26	Stores		
	Water	-3.64	-3.54
27	Voyage Costs	-23.02	-21.65
28	Δ_{Cargo}	-5.16	-5.16
29	Port Days	-7.63	-7.62
30	RTPA	-4.41	-4.10

Table 5.12: Winston Centrality Results Part Two

#	Node	Partial Derivative	Interpolated Derivative
31	Annual Cargo	-7.27	-7.31
32	Annual Costs	-10.32	-16.13
33	Transportation Costs	-16.09	-20.45
34	BM	22.51	22.50
35	KG	39.92	39.92
36	KB	23.86	23.86
37	GM	-10.33	-10.69
38	ζ_1	9.63	11.44
39	ζ_2	16.73	19.87
40	ζ_3	13.36	15.87
41	η_1	25.23	25.19
42	η_2	60.43	60.33
43	η_3	48.86	48.78
44	Round Trip Miles	4.89	4.77
45	Fuel Price	1.77	1.77
46	Cargo Handling Rate	2.27	1.97
47	g_1	-14.15	-13.51
48	g_2	-7.50	-7.16
49	g_3	-36.00	-34.36
50	g_4	-70.98	-69.39
51	g_5	-17.20	-17.20
52	g_6	-7.81	-7.78
53	g_7	-0.83	-0.83
54	g_8	-13.60	-13.60
55	g_9	-9.71	-9.71
56	g_{10}	-29.00	-29.00
57	g_{11}	-4.14	-4.14
58	g_{12}	-2.46	-2.43
59	g_{13}	-163.09	-166.16

are actually used in the computation of other functions.

5.6 Conclusion & Contributions

Network representations of design formulations have been created and analyzed in previous chapters, providing static insight into designer intent. This chapter utilized the same network structure and a new network metric, termed path influence, to analyze the dynamic behavior of a representative preliminary design formulation, the Sen Bulker problem. This chapter comprises the third major contribution of this thesis:

- Recognition that algorithms for finding path lengths can be used to quantitatively capture all node to node influences across multipartite design networks
  - Formulation of path influence algorithms and network weighting schemes, showing equivalency with first order Taylor series expansions
  - Introduction of interpretations for path influence results, comparable with a full factorial design of experiments
  - Development of a new metric, Winston centrality, enabling comparisons between path influence and other metrics and the identification of potential influence multiplying, sorting and damping nodes

Specifically, path influence was used to measure the impact of variable changes on the entire formulation using two path weighting schemes, partial derivative and interpolated derivative. The partial derivative weighting scheme was demonstrated to be equivalent to a first order Taylor series expansion, while the interpolated derivative was developed for non-differentiable problems. Path influence results from each scheme were compared with a full factorial design of experiments, yielding acceptable levels of accuracy in magnitude prediction and high accuracy in predicting trends. These results indicate that path influence can confidently be used to:

- Determine the existence of interactions within design formulations

- Determine the local magnitude and sign of interactions

- Determine the relative importance of one interaction versus another

- Determine the cumulative effect of multiple interactions

Using path influence to generate such insight is advantageous due to the inherent network representation, allowing the intermediate functions used in formulations to be adequately represented and understood. Variable to objective influence can be traced through these intermediate functions, providing an intuitive understanding not necessarily possible from standard methods. Path influence can also require many fewer function calls than a DOE producing comparable results, though additional matrix manipulation is required.

A new metric, Winston centrality, was also introduced which is capable of comparing path influence results with Park or Katz centrality, making dynamic and static network structural analysis comparable. These two types of results for the Sen Bulker problem show general agreement, helping to verify the static analysis results and the overall network approach. Winston centrality is applicable beyond static and dynamic comparisons, specifically to the propagation of influence within a design formulation, and was discussed in this context.

CHAPTER VI

Network Diffusion of Design Information

Design can be defined as the act of generating information used for decision making. Generating information takes time, requiring that design and acquisition be viewed from a time-domain perspective. This chapter is the first step in extending network methods to account for the temporal nature of naval design, which remains a major research gap. Evidence of this gap can be found in research on Set-Based Design. The main theme of Set-Based Design is to delay critical decisions until the latest possible moment, a practice based on the relationship between cost, information, and influence. The claim is that one can improve a design by delaying the commitment of cost until later in the design process when investments can be backed by better information. By delaying cost commitment the time in which constituents can influence a design is also increased (Parker and Singer, 2012; Singer et al., 2009). The veracity of these claims has been demonstrated in practice, but researchers have been hard pressed to attach a mathematical underpinning to them (Bernstein, 1998; Liker, 2004; Mebane et al., 2011). The notable exceptions are those that have tried to account for the influence of time, using real options, Markov decision processes, etc. (Ford and Sobek, 2005; Knight and Singer, 2014; McKenney, 2013). The takeaway is that capturing temporal effects is critical to accurately modeling design or acquisition as a whole.

The previous chapters have shown that multipartite networks provide a generic structure that can produce lead indicators for design, but lead indicators that account for the temporal nature of design are still required. Design's temporal nature can be modeled using networks in several ways, this chapter approaches the problem by modeling the flow of information across a design network using an abstraction of Fick's second law of diffusion. Network diffusion uses the same common basis developed in Chapter III, meaning diffusion and the methods of Chapters IV and V can work in parallel. The simplicity of diffusion analysis means it can produce lead indicators for the early stages of design, but the continuous flow assumption is a limiting factor as design actually progresses discretely. Methods to capture discrete temporal effects are discussed in Chapter VII.

Section 6.1 introduces network diffusion and provides examples of how diffusion analysis can identify common problems designers encounter when working within processes or organizations. A diffusion model of a design organization based on the Sen Bulker problem is presented and analyzed in Section 6.2. Watson & Gilfillan network diffusion results are briefly presented in Section 6.3 to demonstrate that the Sen Bulker results are not atypical. Section 6.6 concludes the chapter.

6.1 Diffusion Modeling of Design Information Flow

The term diffusion has a variety of meanings, even within network science. Network models of diffusion have covered the fields of epidemiology, geography, economics, collective behavior, decay processes, interactive communication, etc., but primarily with empirical data analysis or static methods similar to those presented in Chapter IV (Valente, 1995). Some of these fields have strong dynamic models that do not rely on networks. In the case of epidemiology, these dynamic models can provide the time progression of a disease outbreak from a population standpoint, but they lack information about individuals (aka nodes) within the population (Strogatz, 1994). However, the status of individual nodes is necessary to realistically capture the spread of disease, as transmission often relies on individual contact (Newman, 2003). The complicating factor for epidemiological networks is that contact between individuals does not necessitate the transfer of a disease, i.e. an edge between two nodes does not guarantee transmission. Newman (2003) surveyed some methods which map between network and dynamic models, capturing the individual node characteristics necessary to determine the size of outbreaks based on the initial carrier, but what they lack "...is the time progression of a disease outbreak."

Capturing the temporal nature of a design network requires a different type of diffusion, because both a time progression and individual node characteristics are required. In the case of design networks such as Watson & Gilfillan or the Sen Bulker problem, arcs between nodes represent the transmission of information, not just the possibility of transmission. This makes design networks simpler than their epidemiological counterparts and a diffusion model based on Fick's second law is possible.

6.1.1 Fick's Second Law of Diffusion in Network Terms

Fick's first law of diffusion essentially states that a substance will flow between two locations at a rate proportional to the difference in the amount of the substance at each location, while Fick's second law expresses the amount of a substance at a location as a function of time. Fick's second law can be written as Eq. 6.1, where $\mathbf{\Psi}$ is the amount of substance at a location, C is a diffusion constant, and ∇^2 is the Laplace operator.

$$\frac{d\mathbf{\Psi}}{dt} = C\nabla^2\mathbf{\Psi} \tag{6.1}$$

Fick's second law can be expressed in network terms as shown in Newman (2010) and restated here in Eqs. 6.2 to 6.5. Nodes can be envisioned as locations that a substance might occupy, while the edges or arcs that connect nodes are the paths across which a substance can move. Using the adjacency matrix, Fick's second law can be rewritten as Eq. 6.2, where Ψ_i is the quantity of a substance at node i, and Ψ_j the quantity at node j. This form is valid for both directed and undirected networks.

$$\frac{d\Psi_i}{dt} = C \sum_j A_{ij}(\Psi_j - \Psi_i) \tag{6.2}$$

Algebraic manipulation and the key assumptions that the network is undirected, has at most a single edge between nodes, and no self edges yields Eq. 6.3. $\mathbf{D}$ is the diagonal matrix of the degree of each node as shown in Eq. 6.4.

$$\frac{d\mathbf{\Psi}}{dt} = C(\mathbf{A} - \mathbf{D})\mathbf{\Psi} \tag{6.3}$$

$$\mathbf{D} = \begin{bmatrix} k_1 & 0 & 0 & \dots \\ 0 & k_2 & 0 & \dots \\ 0 & 0 & k_3 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{6.4}$$

The matrix $\mathbf{D} - \mathbf{A}$ is known as the graph Laplacian ($\mathbf{L}$), with many other uses in network mathematics. Eq. 6.5 is the network form of Fick's second law, a sign change and substitution of $\mathbf{L}$ for ∇^2 being the only differences.

$$\frac{d\mathbf{\Psi}}{dt} = -C\mathbf{L}\mathbf{\Psi} \tag{6.5}$$

Eq. 6.5 is a first order differential equation with a solution of the form shown in Eq. 6.6. λ_i is the i^{th} Eigen value of $\mathbf{L}$ and $\mathbf{v_i}$ is the i^{th} Eigen vector of $\mathbf{L}$.

$$\mathbf{\Psi}(t) = \sum_i c_i e^{-C\lambda_i t}\mathbf{v_i} \tag{6.6}$$

6.1.2 The Analogy Between Design Information and Diffusion

Solving Eq. 6.5 provides an individual time series for the amount of a substance at every node in an undirected and simple network (no self-edges or multiedges). If information is the substance, then any question based around how much and when information is available at a given node can be answered.

A logical model of the flow of design information can be created using initial conditions. Any design process starts with assumptions or guesses about the values variables might take, and thus variables are logical nodes for positive initial information. At the same time, an organizational unit or discipline might have relevant design experience at the outset, also signified by a positive initial condition. The direction of information flow is governed by Fick's first law, i.e. flow rate is proportional to the difference in the information level between nodes. Information will flow from nodes that have it to nodes that don't until a steady state is reached assuming a closed system with conservation of information. Negative initial conditions are possible, thus creating a draw for information from one part of the network to another. This type of analysis is useful because the structure of the network can be evaluated for more efficient flows using only a set of initial conditions and provides a closed form solution. This is ideal for identifying classic problems within a design structure prior to implementation, when there is still time to modify it.

Fig. 6.1 is representative of an information level curve generated by diffusion analysis for a single node. Steady state is represented by the dashed line. When the information level is above steady state, information lead is present and the node possesses information to diffuse to other nodes. When below steady state, information lag is present and the node requires more information. A metric for lead and lag is the first moment of area above steady state or below steady state respectively. Steady state is determined by summing the initial conditions and dividing by the number of nodes, i.e. each node ends up with an equal amount of information (assuming a one component network). A high first moment of area below steady state indicates either a large amount of lag, lag whose centroid is late in the process, or both. These situations are undesirable. Separately, if (nearly) reaching steady state is used as a proxy for process completion, then the time to completion can be estimated for every node in the network. Relative completion between nodes at any given time can also be compared. It is important to distinguish between process completion and design completion, as one does not necessitate the other. The process will end when time or resources have run out, while there is never a guarantee of design completion.

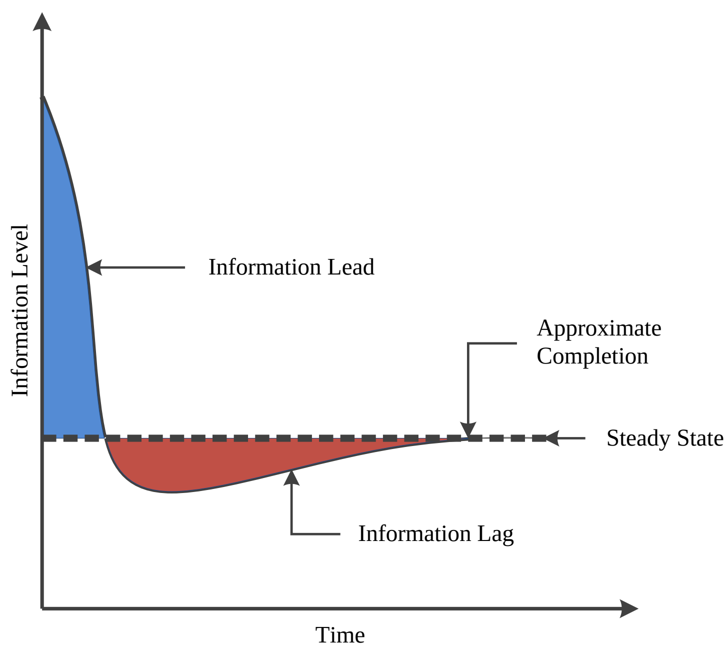
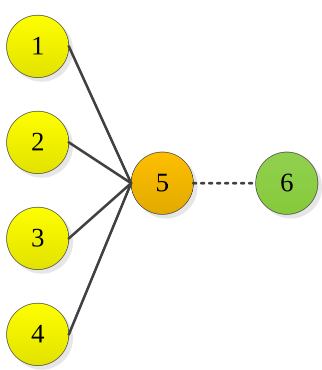


Figure 6.1: Information Levels as a Function of Time

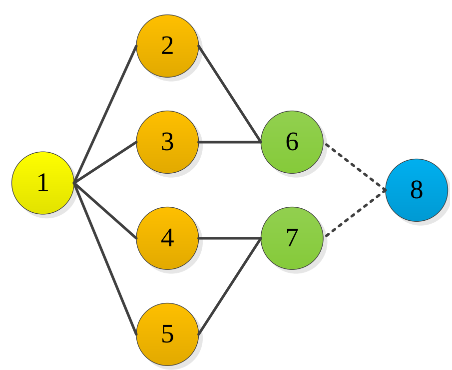
Fig. 6.1 is a time history for a single node, but the diffusion properties of an entire network can be evaluated in a similar fashion. Summing the absolute lead and lag over all nodes is a measure of how far (not in time but in information gap) a network is from steady state. This measure termed *Absolute Information Gap* varies with time, differentiating it from the total information level of the network which is constant. The absolute summation is necessary because information lead on one node does not necessarily counter lag at another node.

6.1.3 Canonical Design Problems in terms of Diffusion

One example of a problem within a design process is the bottleneck, information is available but reaches its destination slowly due to constrictions in the path it must take, creating information lead. Information lag is created when a node requires information but does not receive it in a timely manner. Curves such as Fig. 6.1 and first moment metrics can be used to identify bottlenecks and information lags, but the network structures that generate such conditions are not arbitrary. Fig. 6.2a displays a network that produces a bottleneck, while Fig. 6.2b produces excessive information lag. In both cases yellow or blue nodes represent variables and are initialized with a positive initial condition while other nodes start with zero or negative information as shown in Table 6.1. Dashed edges are modified for different cases.



(a) Design Bottleneck



(b) Design Lag

Figure 6.2: Canonical Design Networks

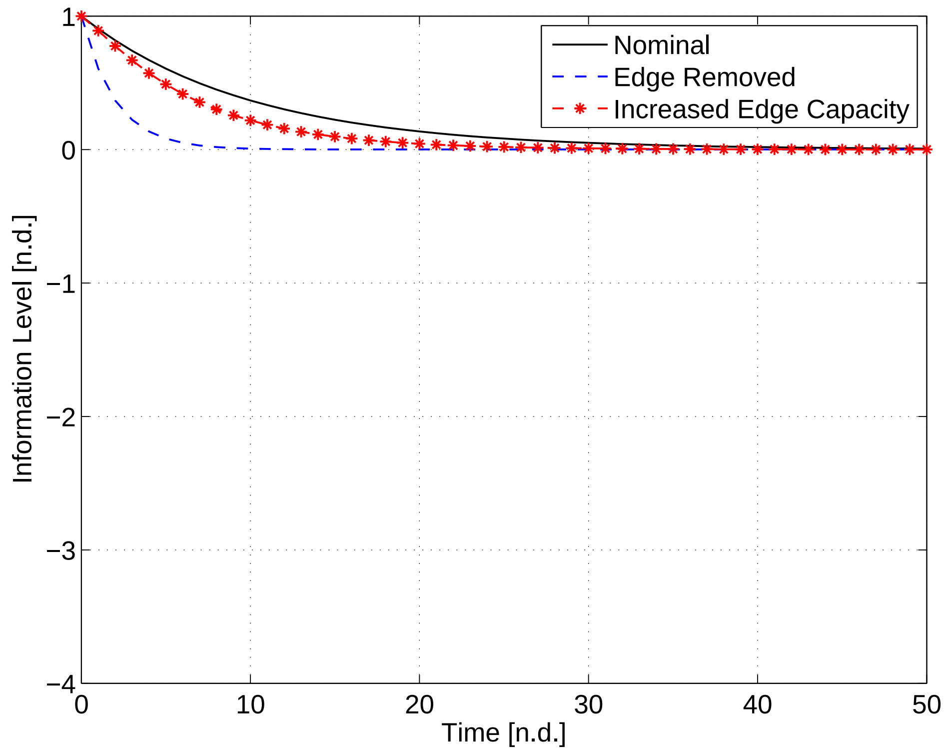
The shape of Fig. 6.2a obviously represents a bottleneck. In a design environment a

Table 6.1: Nominal Case Initial Conditions

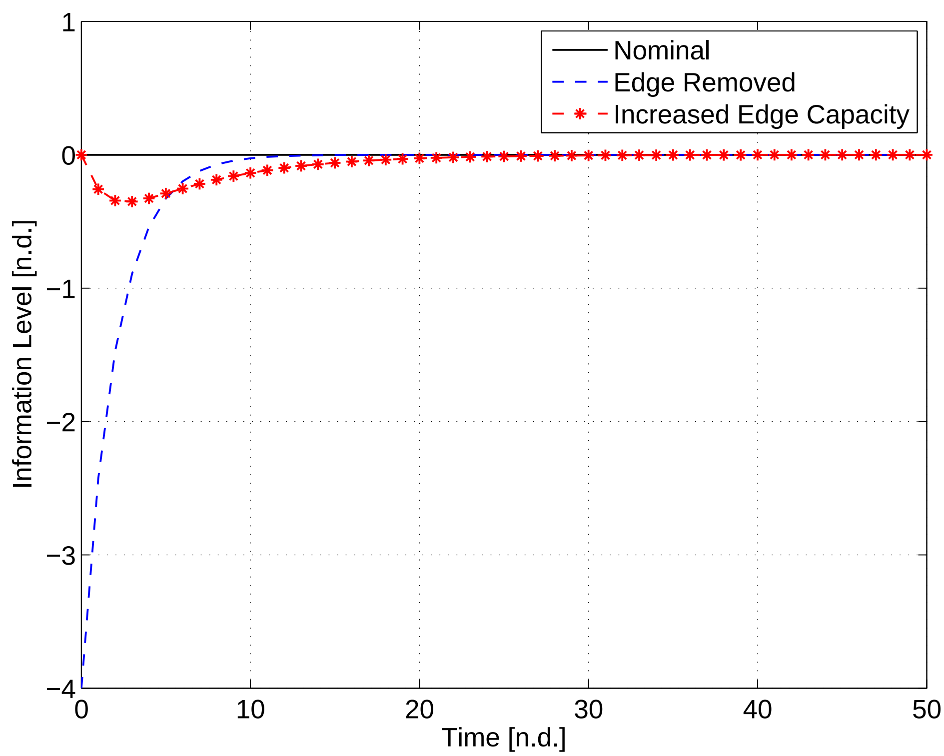
Node #	1	2	3	4	5	6	7	8	Steady State	C
Design Bottleneck	1	1	1	1	0	-4	N/A	N/A	0	0.1
Design Lag	4	0	0	0	0	0	0	4	1	0.1

bottleneck is created when information is inputted to a node faster than it can process it, creating a backup. This phenomenon is well enough documented in manufacturing to inspire fictional works (Goldratt and Cox, 2004). A commonly observed visible symptom of the design bottleneck is the collection of paperwork (or e-mails) on a design manager's desk, much of which must be sent on to oversight. The design manager in Fig. 6.2a is Node 5, while oversight is Node 6. The information level at Nodes 1 and 5 are shown for three different cases in Fig. 6.3. The first case is the nominal condition of Fig. 6.2a and initial conditions of Table 6.1. The second case removes the edge from Node 5 to Node 6, and puts a -4 initial condition on Node 5. The third case doubles the capacity of the edge between Node 5 and 6, changing the values of $\mathbf{A}_{56}$ and $\mathbf{A}_{65}$ from 1 to 2.

In the nominal case, Nodes 1-4 have information that must eventually diffuse to Node 6, but that information must pass through the single edge connecting Nodes 5 and 6 (aka a bottleneck), thus the information level at Nodes 1-4 is above steady state for a prolonged period. The steady state of the network in the all cases is zero, and nominally Node 5 never varies from this because any incoming information is immediately diffused to Node 6. If the edge connecting Node 6 is removed from the network and Node 5 assumes the -4 initial condition, then the information level of Nodes 1-4 reaches steady state much faster. Removing the oversight removes the bottleneck, as the information received by Node 5 need not be passed any further. Oversight is usually not negotiable, so another solution is to provide the design manager with greater capacity to process information. Doubling the capacity of the edge between Node 5 and 6 with nominal initial conditions also decreases the effects of



(a) Node 1



(b) Node 5

Figure 6.3: Bottleneck Network Information Levels

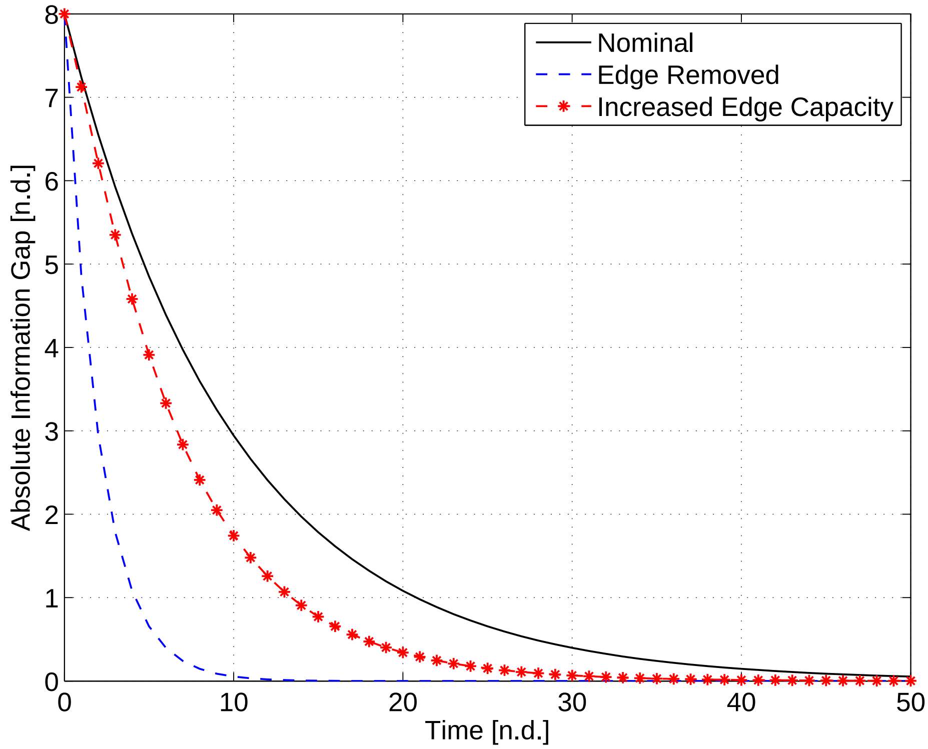
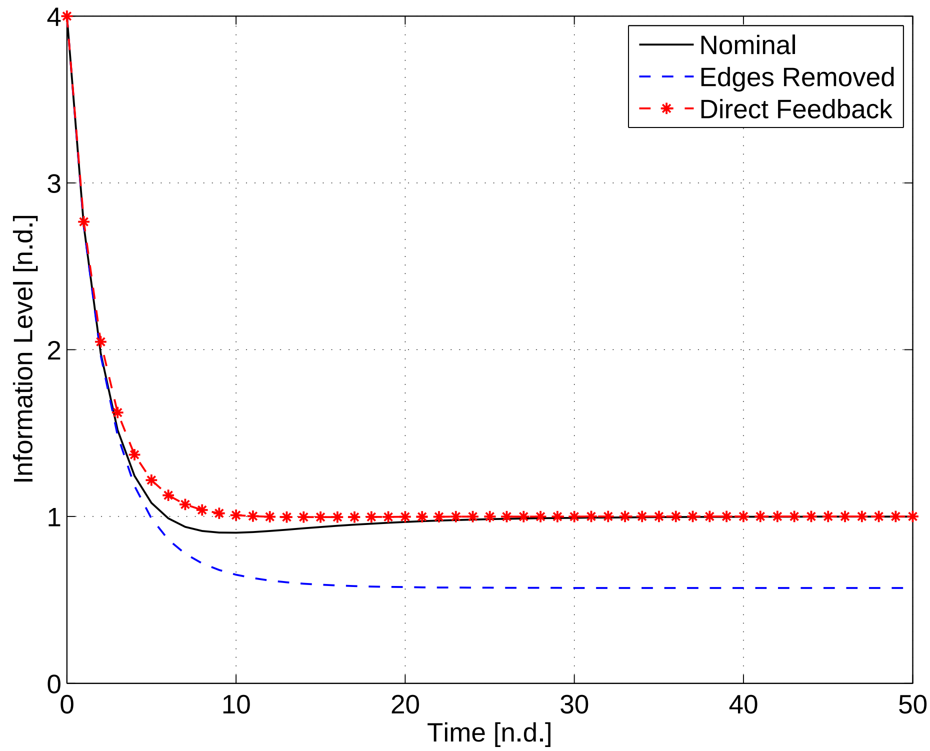


Figure 6.4: Bottleneck Network Absolute Information Gap Comparison

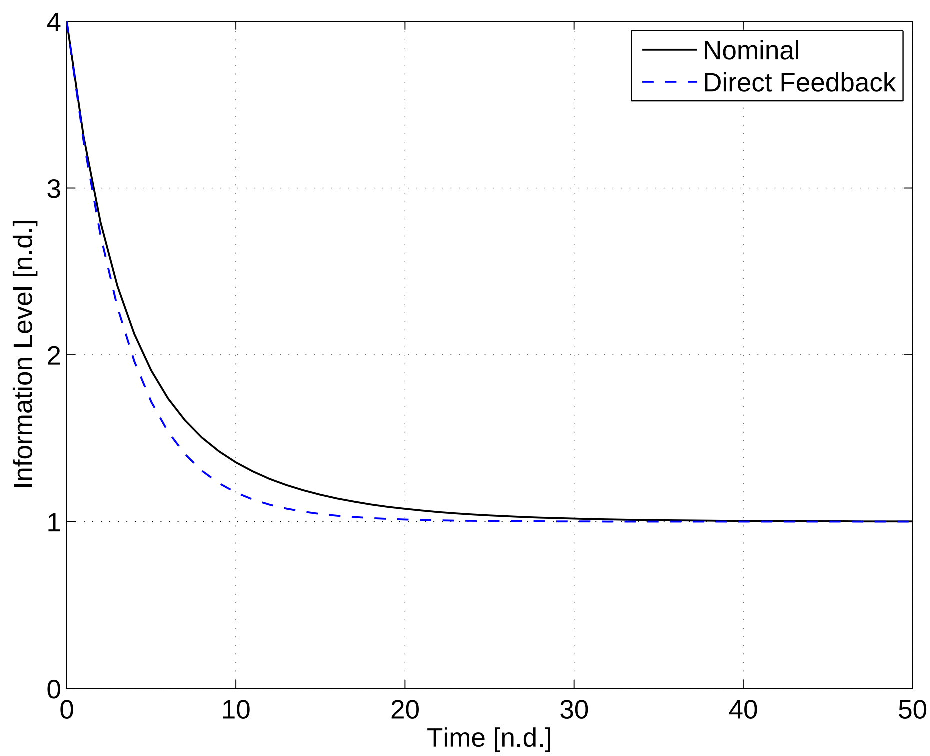
the bottleneck, though Node 5 experiences information lag not associated with its own initial condition. Taken as a whole, Fig. 6.3 demonstrates a nominal case that is analogous to an actual design problem, and two potential remedies.

Calculating the Absolute Information Gap for the three bottleneck cases provides a comparison of the total diffusion properties of the networks, as shown in Fig. 6.4. The conclusion is that removing Node 6 is the most effective option for reducing information lead and lag, but adjusting the capacity is also effective with the added advantage of not altering the fundamental structure of the network.

The lag condition of interest in Fig. 6.2b is harder to visually identify, but actually occurs at Node 1. Node 1 and 8 have equal amounts of information at $t = 0$, but the information at Node 1 diffuses faster than information from Node 8 in the nominal case because of the degree difference of the nodes. As a result, the information level of Node 1 drops below steady state before returning despite the fact that it started with an information level well above steady state. This situation is analogous to the assumptions made during design. Many variables have an assumed value at the beginning of a design process, which is later updated when more information becomes available. The updating information comes from nodes downstream in the process and must eventually flow back to reach the variable nodes. The variable in the lag network is Node 1, while the downstream information source is Node 8. Essentially Node 1 primes the early design nodes with an assumption which is then corrected later with information from Node 8. Disconnecting Node 8 drops the steady state level of the smaller connected network to ≈ 0.57 and eliminates the lag at Node 1, analogous to trusting the initial assumption. An alternative that keeps the steady state at 1 is to add an edge from Node 8 directly to Node 1, creating a direct feedback pathway. The added edge means information diffuses from Node 8 to Node 1, which has the added benefit of reducing Node 8's information lead. The information levels of Node 1 for the three cases are shown in Fig. 6.5a, and Node 8's two cases in Fig. 6.5b. The Absolute Information Gap plot for the three cases is shown in Fig. 6.6. Which solution is better is dependent on what the nodes represent, removing the edges results in a lower Absolute Information Gap overall, but the time to completion is about the same as direct feedback. Direct feedback maintains the existing steady state and eliminates the lag, but has a higher overall Absolute Information Gap.



(a) Node 1



(b) Node 8

Figure 6.5: Lag Network Information Levels

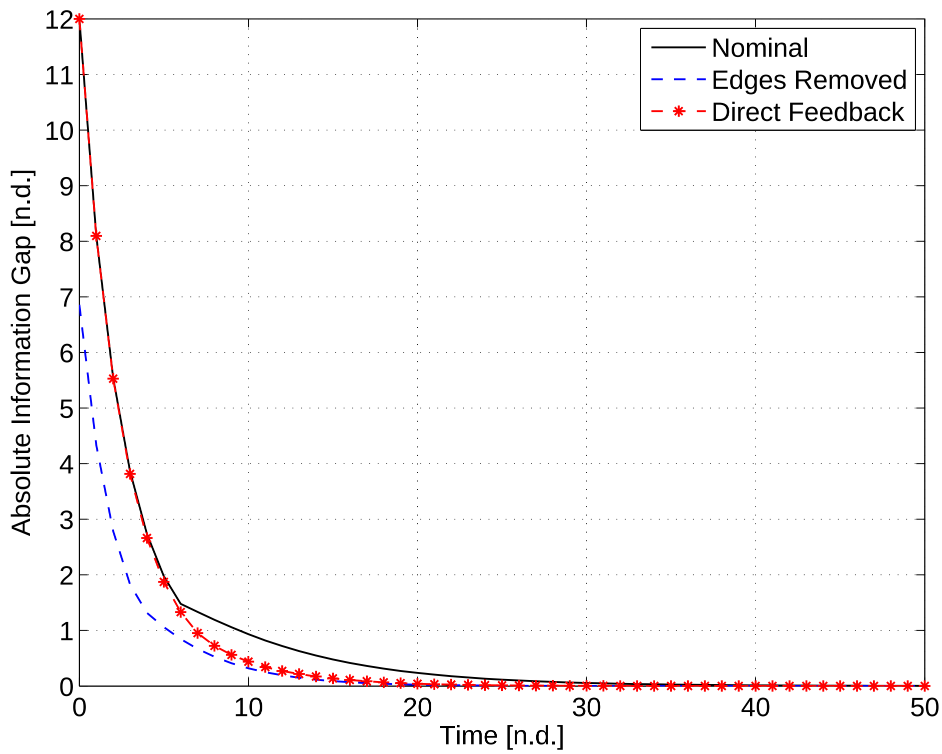


Figure 6.6: Lag Network Absolute Information Gap Comparison

6.2 Diffusion Analysis of the Sen Bulker Problem

The Sen Bulker network is a representation of a design problem formulation, but its basic structure could also represent a design process, organization, or combination. As an organization, the structure could represent the people who receive information from multiple sources upstream, combine it and then pass it on downstream. Removing the redundant (A_co) and optimizer specific portions of the formulation yields a structure that could easily be derived from an inverted organization chart, Fig. 6.7. This network, the Sen Bulker Organization, will be used for the remainder of the section.

6.2.1 Baseline Results

A baseline diffusion analysis was conducted on the Sen Bulker Organization network, with the six variable and nine parameter nodes having equal initial conditions such that the steady state is one. Eq. 6.5 was solved and plotted out to time $t = 100$.

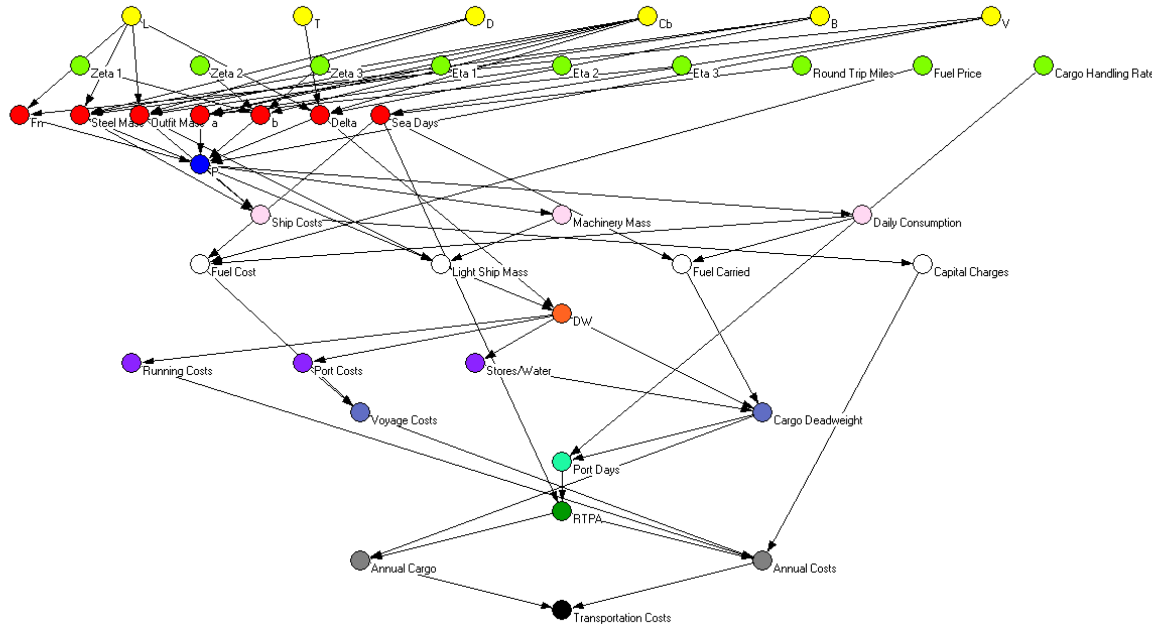
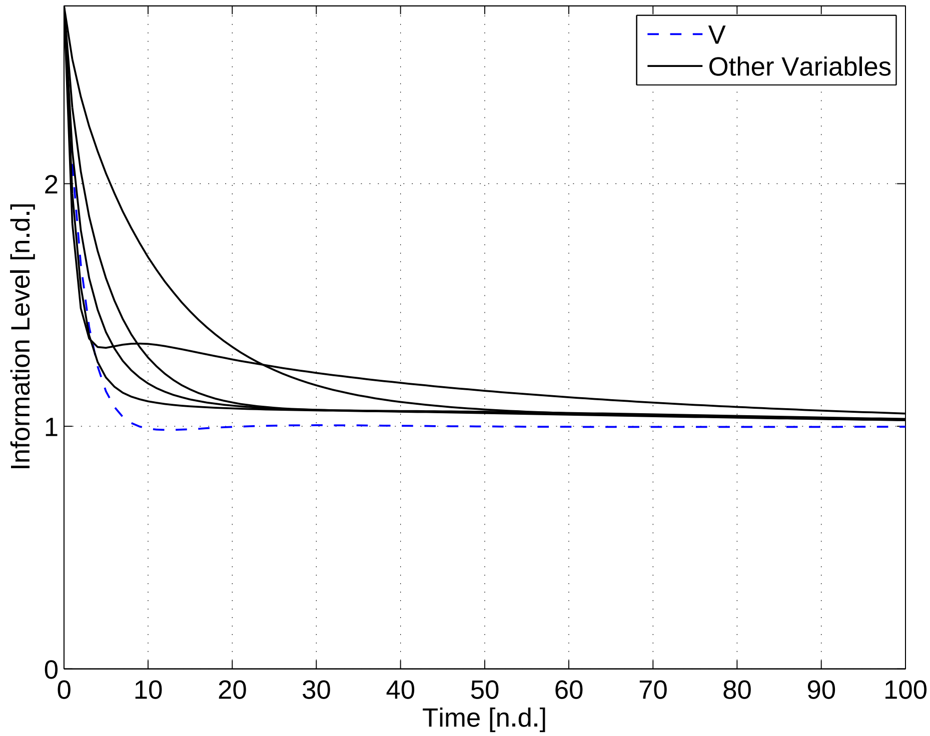


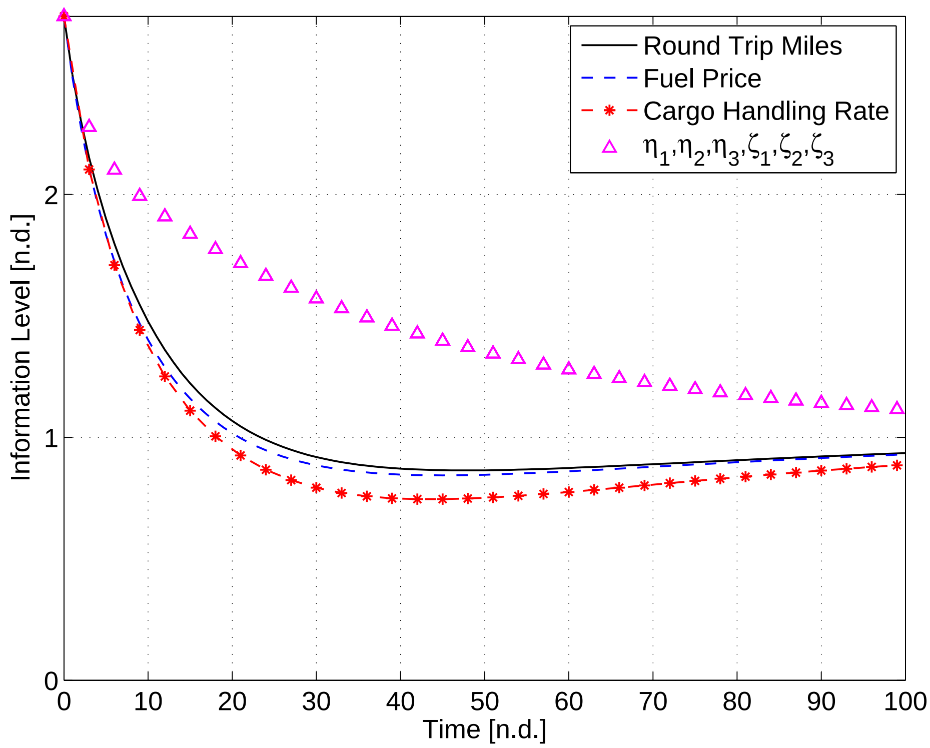
Figure 6.7: Sen Bulker Organization Network

The lead and lag first moments of area were numerically evaluated and are shown in Tables 6.2 and 6.3. Composite plots of the variable, parameter and objective information levels are shown in Figs. 6.8 and 6.9.

The hierarchical structure of the Sen Bulker Organization is clearly apparent in Fig. 6.7, and the diffusion plots and first moments of area confirm this. Generally, variables and parameters have lead with little to no lag, while the lag of functions increases with their level, i.e. functions furthest from the variables have more lag. Fig. 6.9a shows that Light Ship Mass experiences much less lag than Annual Cargo and Transportation Cost. Light Ship Mass is a fourth level function while the other two objectives are on levels ten and eleven. The benefit of diffusion is that intuition about the visual structure can be quantified for each node, and outlier behavior can be identified and potentially diagnosed. As an example, there are several notable exceptions to the general hierarchical trend. V, Round Trip Miles, Fuel Price and Cargo Handling Rate are variables/parameters that have a larger lag moment than lead, while the η's and ζ's have an exceptionally high lead moment.

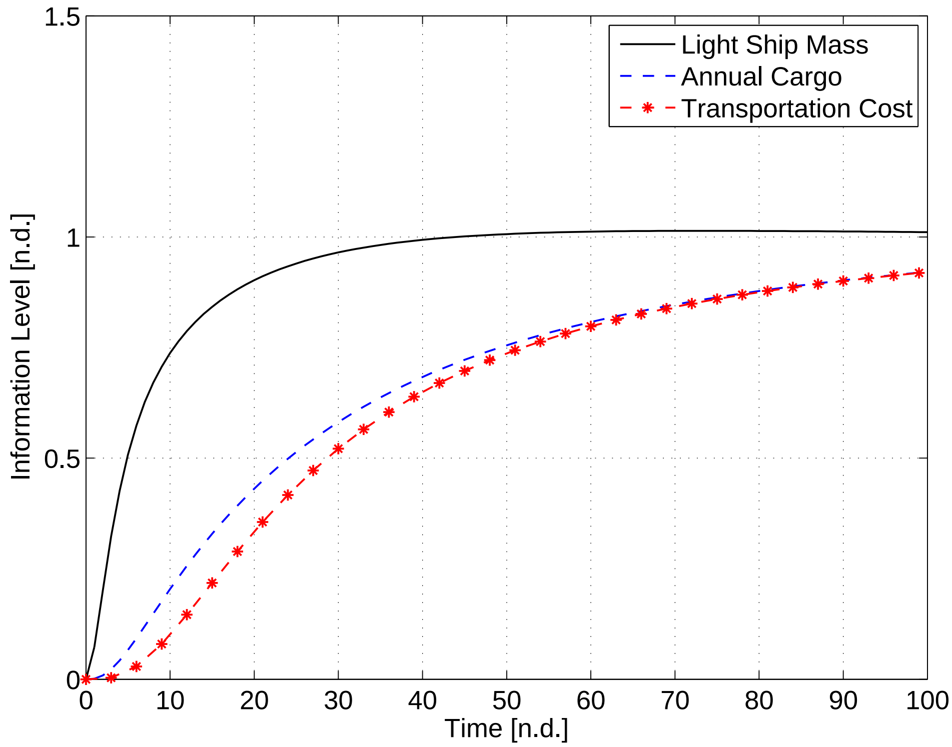


(a) Variables



(b) Parameters

Figure 6.8: Baseline Sen Bulker Organization Information Levels Part One



(a) Objectives

Figure 6.9: Baseline Sen Bulker Organization Information Levels Part Two

Table 6.2: Sen Bulker First Moments of Area Baseline Part One

Node #	Node	Lead First Moment	Lag First Moment
1	L	226	0
2	T	406	0
3	D	283	0
4	C_b	592	0
5	B	250	0
6	V	8	-11
7	F_n	176	-3
8	Steel Mass	232	0
9	Outfit Mass	232	0
10	a	1078	0
11	b	1078	0
12	Δ	209	0
13	Sea Days	0	-510
14	P	298	-4
15	A_{co}	–	–
16	Ship Costs	79	-38
17	Machinery Mass	152	-58
18	Daily Consumption	0	-333
19	Fuel Cost	0	-566
20	Light Ship Mass	48	-66
21	Fuel Carried	0	-610
22	Capital Charges	0	-529
23	Δ_{DW}	0	-548
24	Running Costs	0	-852
25	Port Costs	0	-817
26	Stores & Water	0	-805
27	Voyage Costs	0	-836
28	Δ_{Cargo}	0	-820
29	Port Days	0	-893
30	RTPA	0	-882
31	Annual Cargo	0	-1026
32	Annual Costs	0	-893
33	Transportation Costs	0	-1109

Table 6.3: Sen Bulker First Moments of Area Baseline Part Two

Node #	Node	Lead First Moment	Lag First Moment
38	ζ_1	1429	0
39	ζ_2	1429	0
40	ζ_3	1429	0
41	η_1	1429	0
42	η_2	1429	0
43	η_3	1429	0
44	Round Trip Miles	73	-475
45	Fuel Price	58	-538
46	Cargo Handling Rate	50	-882

The time plot in Fig. 6.8a of V mimics the same phenomenon found for Node 1 in the classic lag example, i.e. V supplies information to downstream nodes faster (due to the networks structure) than the other variables, thus its information level is below steady state (slightly) for a prolonged period. The effect is magnified for Round Trip Miles, Fuel Price and Cargo Handling rate as seen in Fig. 6.8b, as the nodes they input to are much further downstream in the organization.

The case of the ζ's and η's is a dramatic bottleneck. Recalling structural similarity from Sections 3.1.3 and 4.1.2, it is apparent that η_1, η_2 and η_3 are structurally equivalent, the same for the ζ's. This means that the three η's are all inputting to the same node, a, which itself only has one output. This single edge leaving a is one bottleneck. The identical situation exists for the ζ's and b. Compounding the bottleneck is that a and b only input to P, which has an in-degree of five but an out-degree of three, and the lead first moment for P also indicates a bottleneck.

6.2.2 Modified Network Results

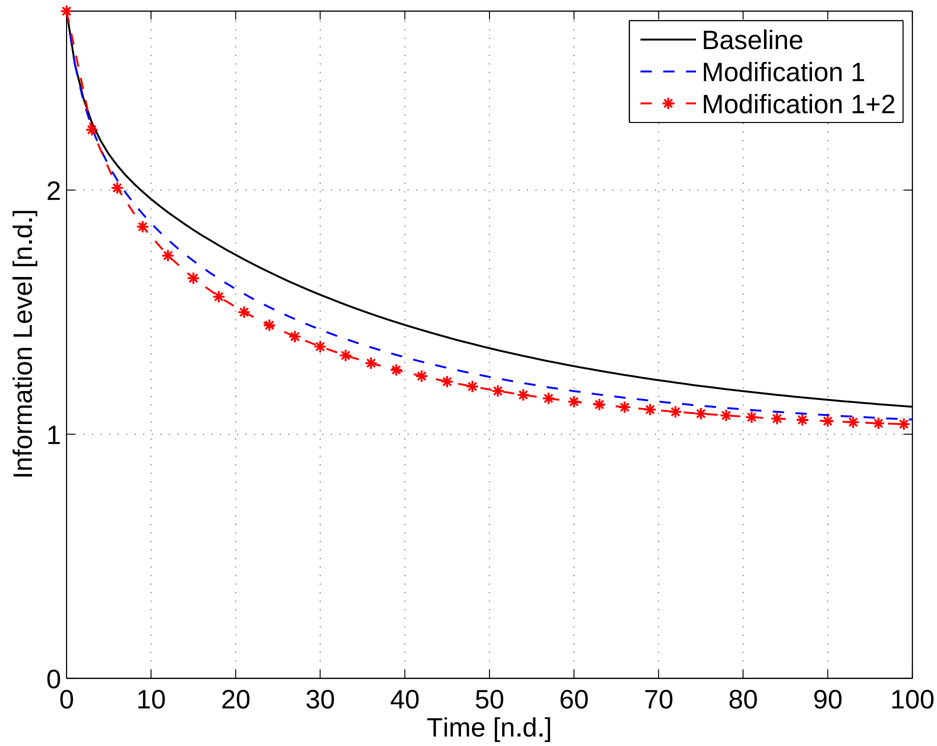
The previous chapters have not recommended "improvements" to network structure based on analysis results, but modifying the edge weights can provide for increased understanding when using diffusion. The obvious modification for the Sen Bulker Organization is to alleviate the bottleneck for the η's and ζ's. The edges linking a and b with P cannot be removed while preserving the basic formulation, but the capacity can be increased as done for the canonical bottleneck in Section 6.1.3. A second modification is to further alleviate the bottleneck by increasing the capacity of P's outgoing edges. The capacity of these edges was quadrupled, and the relevant plots are shown in Figs. 6.10 and 6.11.

Fig. 6.11a is the Absolute Information Gap plot for the three cases. It is clear that each modification reduces the gap slightly, but the curves' exponential shape hides the effect of the modifications. The curves never reach zero gap in real time because they are exponential, so a threshold value for process completion is necessary. This was set to five, and the time to completion and percent decrease relative to the baseline is shown in Table 6.4.

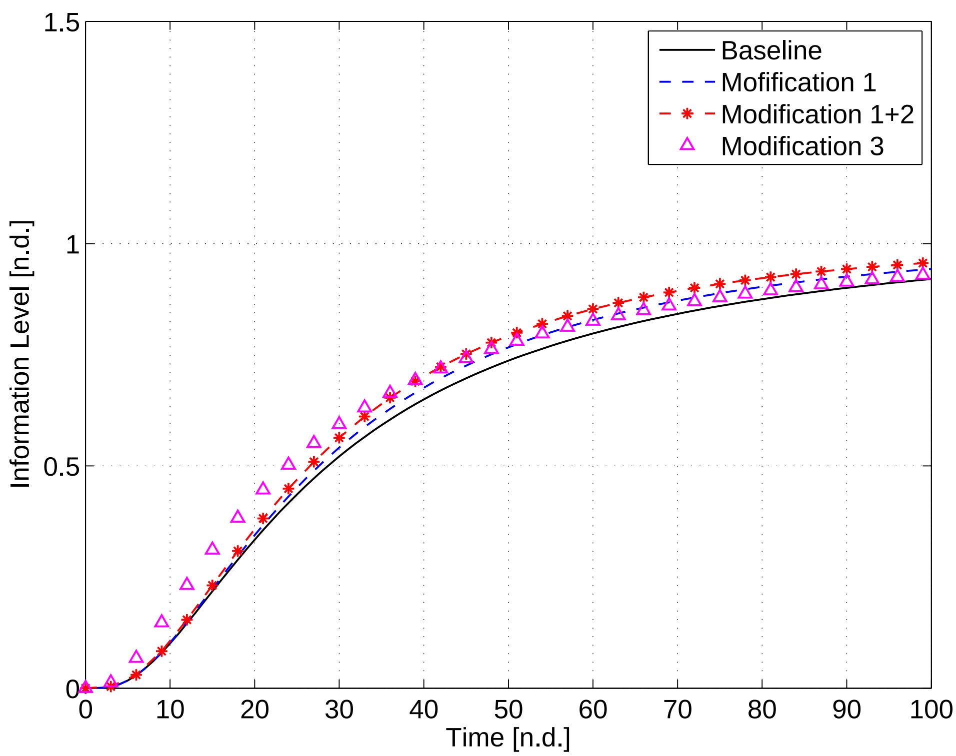
Table 6.4: Sen Bulker Time to Process Completion, Threshold=5

	Time to Completion	% Difference	Capacity	% Difference
Baseline	65	0	65	0
Modification 1	55	-15	71	9
Modification 1+2	47	-28	80	23
Modification 3	64	-2	80	23

The Sen Bulker Organization network has 65 arcs (edges for diffusion). These arcs define the capacity of the network to diffuse information. Capacity in a design network could signify personnel, computing resources etc. Weighting the edges increases the capacity, and diffusion allows this to be compared with reductions in process time. No arcs were added to the Sen Bulker Organization, but the weights were increased for the modified cases. It is apparent from Table 6.4 that intelligent additions of capacity can reduce time to completion significantly. To demonstrate the opposite, a third modification increased the weight of the arcs inputting to Transportation cost to 8.5, thereby matching the total capacity of Modification 1+2. As expected, this had minimal impact on time to threshold because those arcs are not involved in a bottleneck. The only noticeable difference was on the information level

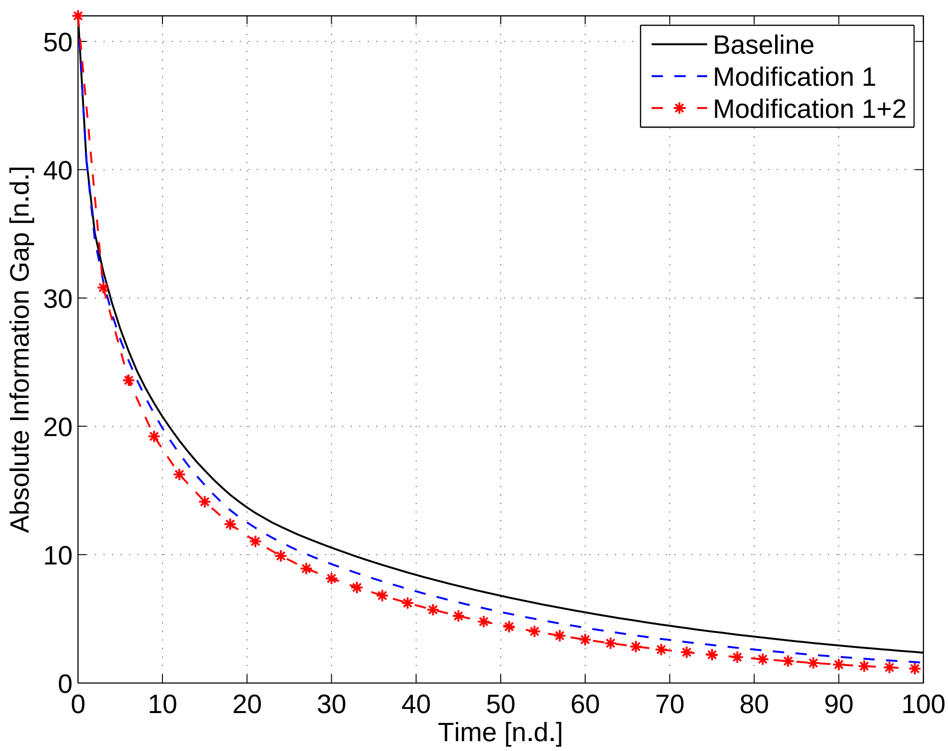


(a) η and ζ Parameters



(b) Transportation Cost Objective

Figure 6.10: Modified Sen Bulker Organization Information Levels



(a) Absolute Information Gap

Figure 6.11: Modified Sen Bulker Organization Absolute Information Gap

plot of Transportation Cost, outperforming the other modifications early but under performing them as time to completion approached as shown in Fig. 6.10b. The third modification is not shown on the other plots because it is indistinguishable from the baseline.

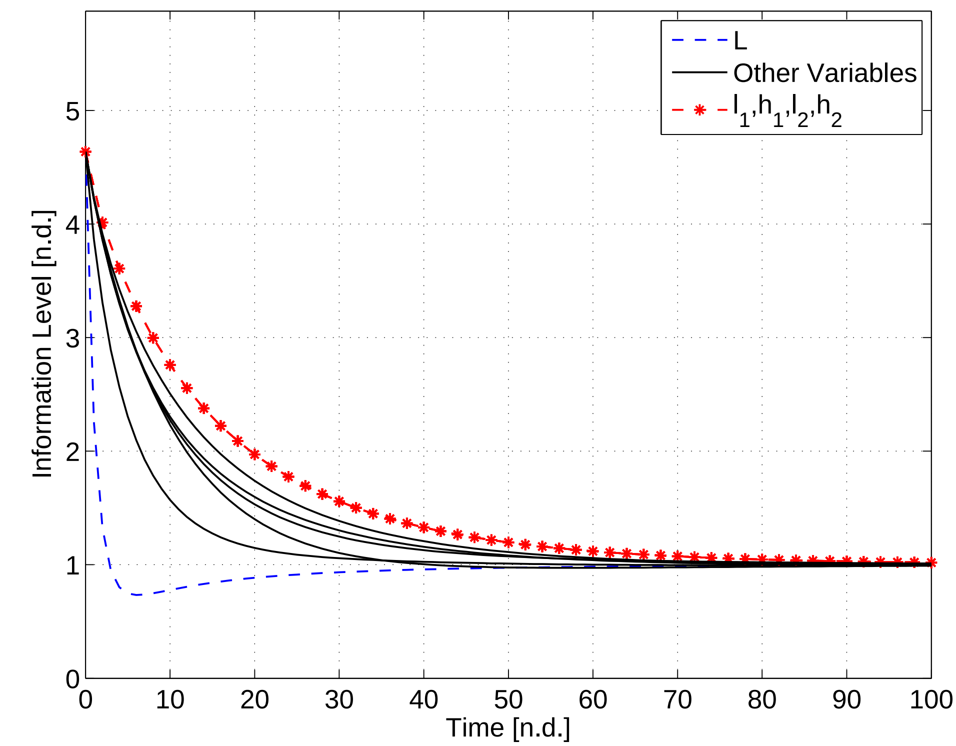
6.3 Diffusion Analysis of the Watson & Gilfillan Method

The Watson & Gilfillan network is suited for diffusion analysis without modification. The paper it is based on is titled "Some Ship Design Methods," and was published prior to the rise of the personal computer (Watson and Gilfillan, 1977). As a result the method is logically arranged as a workable process for an individual or team, and the network structure mirrors that process. From the organization standpoint, the disciplines that were added represent the collecting points for information prior to decision making, much like the objectives of the Sen Bulker network.

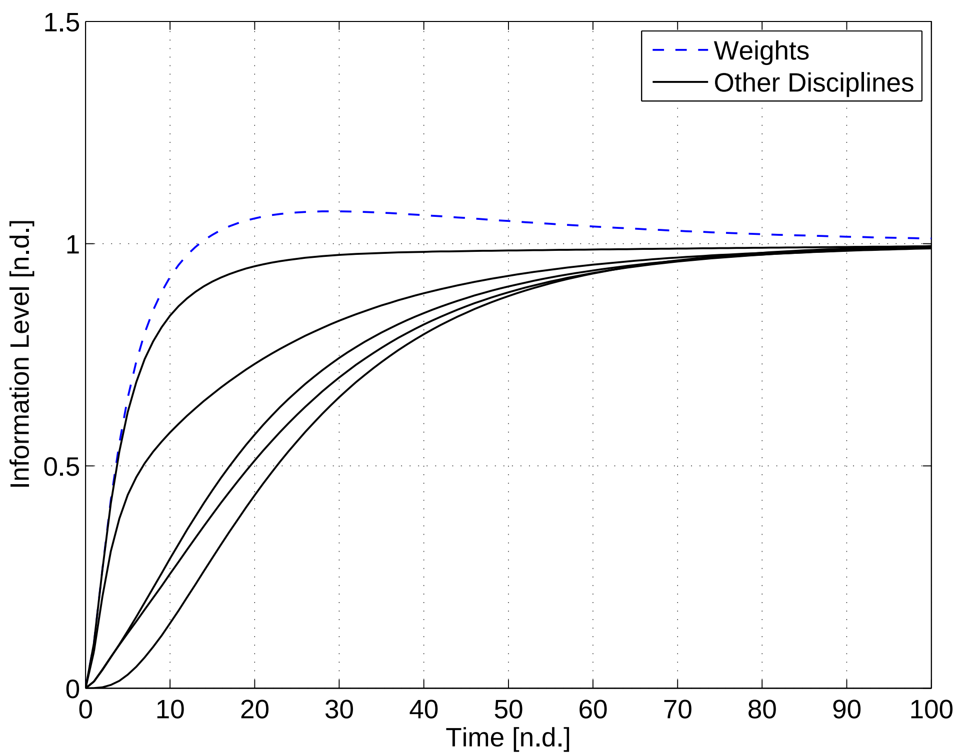
6.3.1 Baseline Results

A baseline diffusion analysis was conducted, with independent variable node (zero in-degree) initial conditions equally set so that the steady state is one. The relevant plots and first moment tables are shown in Fig. 6.12 and Tables 6.5 and 6.6.

Comparing Fig. 6.12 for the Watson & Gilfillan method with Figs. 6.8 and 6.9 for Sen Bulker Organization shows the same shapes of information level curves. This suggests that the diffusion results are not atypical, and the analysis of Section 6.2 is reasonable. There are several unique results from the Watson & Gilfillan network worth discussing. The first is the large lag first moment of area for Length (L), and the significant lead first moment of area for the Weights discipline. These are opposite of the expected trends. The lag of L can be explained by recalling that L has the highest out-degree and Park centrality of the network. This means that the information initially at L can disperse more quickly than its counterparts, dropping



(a) Independent Variables



(b) Disciplines

Figure 6.12: Baseline Watson & Gilfillan Information Levels

Table 6.5: Watson & Gilfillan First Moments of Area Part One

Node #	Node	Lead First Moment	Lag First Moment
1	L	2	-128
2	B	0	-43
3	T	1	-48
4	D	5	-78
5	V	143	-15
6	C_t	648	0
7	s	265	-76
8	l_1	991	0
9	h_1	991	0
10	l_2	991	0
11	h_2	991	0
12	RPM	509	0
13	Δ	0	-274
14	C_b	0	-245
15	LCB	0	-714
16	S	58	-33
17	E	467	-8
18	$C_b^{'}$	26	-224
19	K	812	0
20	W_{s7}	175	-77
21	W_s	0	-326
22	Pe	178	-17
23	η	513	-2
24	MCR	0	-169
25	W_{me}	117	-71
26	Structural Cost	0	-623
27	Machinery Cost	0	-807
28	Total Cost	0	-569
29	$f(L)$	0	-112
30	$f(D)$	0	-280

Table 6.6: Watson & Gilfillan First Moments of Area Part Two

Node #	Node	Lead First Moment	Lag First Moment
31	$f(B,L)$	0	-269
32	$f(L,V)$	0	-141
33	$f(Cb)$	0	-425
34	$f(Cb,L,B,T,s)$	0	-131
35	$f(Cb,L,B,T)$	0	-100
36	$f(L,B,T,D,l_1,h_1,l_2,h_2)$	448	0
37	$f(T,D,C_b)$	11	-117
38	$f(E,K)$	399	-2
39	$f(W_{s7},C_b')$	49	-155
40	$f(V,C_t,S)$	198	-5
41	$f(P_e,\eta)$	123	-10
42	$f(RPM,MCR)$	140	-3
43	$f(L,C_b,W_s)$	0	-362
44	$f(MCR)$	0	-504
45	$f(L,C_b,W_s,MCR)$	0	-319
46	Powering	0	-89
47	Weights	164	-14
48	Ship Type	0	-321
49	Stability/Seakeeping	0	-456
50	Rules/Safety	0	-547
51	Structures	0	-505

it below steady state. The Weights discipline is in the opposite situation, having the third highest in-degree and the lowest Park centrality. This means that Weights can draw information more quickly than its counterparts, raising it above steady state. The Watson & Gilfillan network is multipartite, so the information lag at L cannot be countered directly with information from another variable, it must flow from a function. The same can be said for the information lead at Weights, it cannot flow directly to another discipline. The fact that the information lead persists for so long at Weights is indicative that the disciplines are not tightly coupled. This is untrue of the Transportation Cost and Annual Cargo objectives of the Sen Bulker Organization. Those nodes are on different levels of the multipartite network, and are directly connected. As a result, their information level curves are nearly identical as seen in Fig. 6.9a.

The other interesting result worth mentioning can be seen in the lag first moments for Nodes 33-36 shown in Table 6.6. Node 33, $f(C_b)$, has a larger lag first moment than nodes 34 and 35 which are partly dependent on it. Nodes 34 and 35 have smaller moments because they are simultaneously receiving information from other nodes. That information arrives sooner than that for $f(C_b)$, reducing the lag. Node 36 is an extreme example of the same concept, though half of its inputs (L, B, T, D) have lag of their own, the other half (l_1, h_1, l_2, h_2) have enough lead to yield zero lag for Node 36.

6.3.2 Modified Network Results

Regarding bottlenecks, the situation of l_1, h_1, l_2 and h_2 in the Watson & Gilfillan network is the same as that of the η's and ζ's in the Sen Bulker organization. Both sets have the highest lead first moment as part of a classic bottleneck. The Watson & Gilfillan variables were identified as structurally equivalent in Section 4.1.2 and are used redundantly. Two modifications were tried to reduce the effect of the bottleneck.

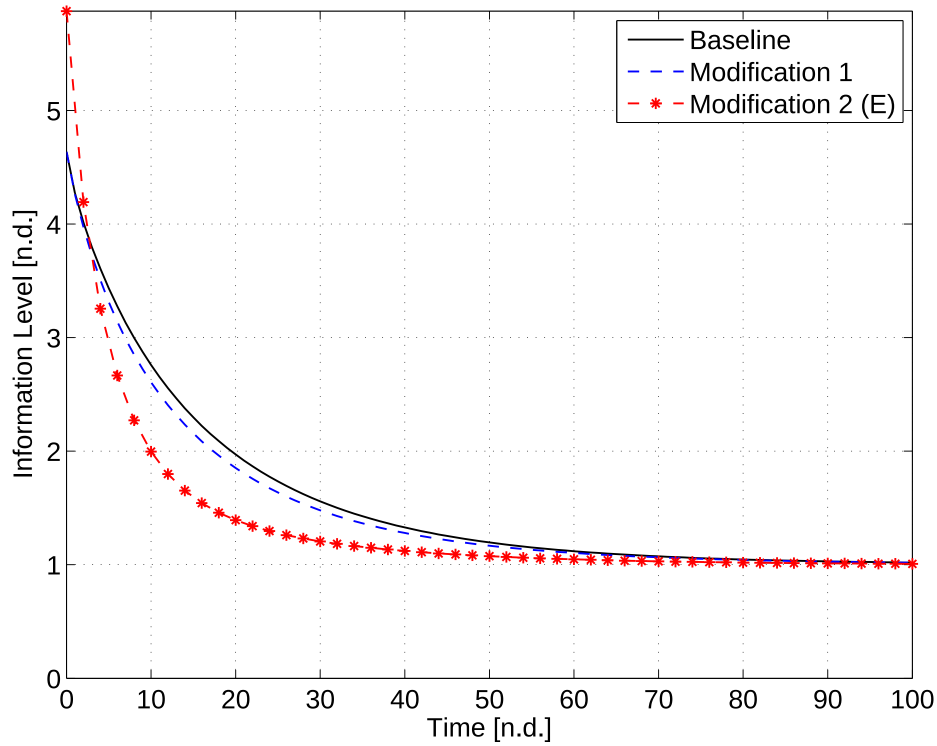
First, the two outgoing arc weights of the function they feed were quadrupled, the same approach taken for the Sen Bulker Organization bottleneck. The second modification was to delete the nodes entirely and place an initial condition on the function, Equipment Number (E), that they feed. The results are compared in Figs. 6.13 and 6.14.

The first modification did not have a noticeable impact on the Absolute Information Gap, as apparent in Fig. 6.14a. The lead of l_1, h_1, l_2 and h_2 was reduced, but this resulted in an increased lead for the Weights discipline. An explanation for this is that the variables in question have three directed paths to the Weights discipline, two to the Ship Type discipline but none to the remaining four disciplines. In other words, the modified arc weights increased the flow rate to the Weights discipline more than any other, trading lead at one location for lead at another. Diffusion operates on the undirected version of the network so there are paths connecting the variables to all disciplines, but comparing the first moments shows that lead was increased and lag reduced for Weights at the expense of the other disciplines. This is shown in Table 6.7. Static network analysis results already indicated that the Watson & Gilfillan method was weight based, and diffusion analysis indicates that a process or organization implementing it will be skewed towards weights as well.

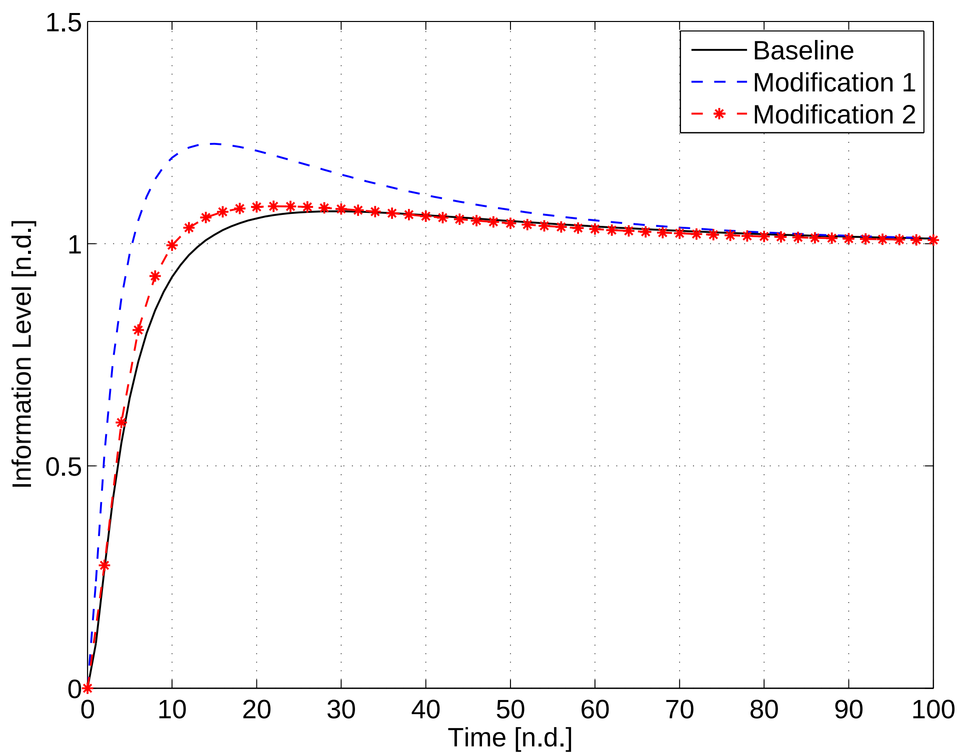
Table 6.7: Watson & Gilfillan Discipline Lead and Lag Comparison

	Baseline		Modification 1	
Node	Lead First Moment	Lag First Moment	Lead First Moment	Lag First Moment
Powering	0	-89	0	-113
Weights	164	-14	287	-3
Ship Type	0	-321	0	-325
Stability/Seakeeping	0	-456	0	-496
Rules/Safety	0	-547	0	-606
Structures	0	-505	0	-561

The second modification eliminates the redundant variables entirely, but places an information lead and different initial condition on E. This changes the structure

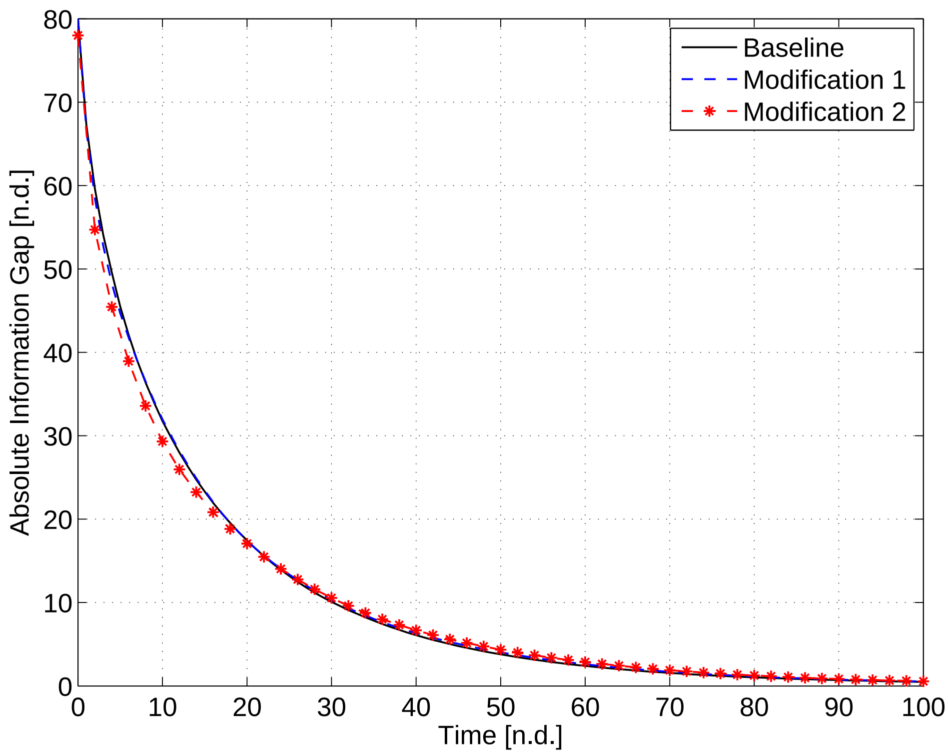


(a) l_1, h_1, l_2, h_2 or E



(b) Weights Discipline

Figure 6.13: Modified Watson & Gilfillan Information Levels



(a) Absolute Information Gap

Figure 6.14: Watson & Gilfillan Absolute Information Gap

of the network, but results in a lower first moment of area for both the node(s) with the initial condition and the Weights discipline than the first modification. The effect on the Absolute Information Gap remains minimal though, implying side effects elsewhere in the network. The implication of both modifications is that seemingly simple changes to a design process or organization can easily result in unintended consequences for information flow. Diffusion analysis provides a means to test changes and identify consequences in advance.

6.4 Comparing Diffusion and Static Analysis Methods

Betweenness centrality, introduced in Section 3.1, might seem to be a static network indicator for the first moments of lead or lag. Betweenness centrality is the ratio of geodesic paths that pass through a node (or arc) relative to the total number of geodesic paths in the network. A node with high betweenness lies on many such paths, and may be a candidate for a bottleneck or lag. However, there are a variety of conditions that can cause a node or arc to have high betweenness, whereas Section 6.1.3 and this section have shown that bottleneck and lag conditions come from particular circumstances. Furthermore, the nodes exhibiting the most lead and lag in the Sen Bulker Organization have zero betweenness as defined because their total degree is one, meaning no paths pass through them. The betweenness values of the Sen Bulker Organization network were checked against the diffusion results anyway, but no clear correlation was found.

Similarly, an anecdotal correlation between Park centrality and first moments of lead and lag area were discussed for nodes in the Watson & Gilfillan Network. The theme is that there are potential links between static network analysis metrics and diffusion. This is a promising area of future research, starting with the correlations between the metrics already outlined here and then moving to unexplored ones such as arc or edge betweenness.

6.5 Advantages & Limitations of the Diffusion Analogy

Diffusion introduces a new dimension, time, to the analysis of naval design networks. Diffusion is an analogy to the flow of information across a design network, answering questions about when and how much information is available at given node. Diffusion analysis is especially useful because it operates on the same type of network used for analysis in the previous chapters. This means that one network can be constructed and then evaluated in a variety of complimentary ways, maximizing the return on investment for network construction. The only requirements for diffusion analysis beyond the network are a set of initial conditions and selection of C. Diffusion requires many assumptions about the behavior of design processes and organizations identified in the next paragraph, but provides meaningful results in a closed form solution. These are good properties for producing lead indicators in a timely fashion.

Diffusion assumes that information flows across a network rather than taking discrete jumps. Implicit within this assumption is that design information is homogenous in content and source, when in fact design requires specific information from specific sources. In other words, diffusion can say that Node A received X amount of information from all other nodes by time t, but in reality Node A receives specification Q, R, and Z from Nodes B, C and D at times t_1, t_2, and t_3. The form of diffusion presented in Eq. 6.5 also requires an undirected network. Directed networks can easily be converted to undirected as was done for structural similarity analysis in Section 4.1.2, but this is in effect decreasing the fidelity of the network representation. The correct flow directions can be partially recreated with appropriate initial conditions, but their location and magnitude are also assumptions. Finally, time itself is only a relative quantity for diffusion analysis because the diffusion constant C is arbitrary and every edge in the network has equal capacity to carry information unless weighted edges are employed. The final limitation of Eq. 6.5 is that not all $\mathbf{L}$ matrices allow a solution, which must be checked on a network by network basis.

The assumptions listed above are for the specific interpretation of design information flow discussed in this chapter. Many other interpretations of diffusion on design networks are possible, meaning those methods outlined here open a set of possibilities rather than close one. Methods for capturing the discrete behavior of design information flow are discussed in the next chapter.

6.6 Conclusions & Contributions

The strength of analyzing design structures with network theory is that networks can represent design as it actually exists and the representations can be analyzed in a multitude of ways. Though the networks themselves are not models, the methods and metrics to analyze them are models or analogies. Diffusion is one such analogy, introducing time to the network study of naval design. This chapter comprises the fourth major contribution of this thesis:

- Application of network diffusion to model continuous information flow across a multipartite design network, effectively capturing classic flow problems in a closed form solution
  - Development of metrics to quantify and interpret continuous information flow across multipartite design networks

Specifically, the mathematical basis of Fick's second law of diffusion was presented and discussed in terms of design information flow. The analogy is formulated so that the primary input is a contextual multipartite design network, making it an easy extension to the methods and metrics of previous chapters. Representative bottleneck and lag networks were formulated as canonical design problems along with their diffusion analysis and potential solutions. The Watson & Gilfillan and Sen Bulker Organization networks were also analyzed. The results of all four examples were interpreted in accordance with the diffusion analogy, and shown to be insightful and congruent with the analysis methods presented in previous chapters. Absolute Information Gap, Information Level, and First Moments of Area for Lead and Lag were introduced as measures and metrics for continuous information flow.

The major limitation of diffusion in the present context is that it assumes information flows across a network rather than taking discrete jumps. Implicit within this assumption is that design information is homogenous in content and source, when in fact design requires specific information from specific sources. However, the work to develop diffusion for design networks did yield the idea for quantifying information level, gap and lag. The concept behind these metrics is of immense value well beyond the narrow application of diffusion. The next chapter develops a method to bring information level, gap and lag to the discrete case, overcoming the major limitations of diffusion.

CHAPTER VII

Discrete Information Flow & Verification

Chapter VI introduced a continuous design information flow analogy that is useful for identifying capacity issues that cause information bottlenecks and lag. The drawback of the analogy is that it treats information as continuous and uniform, when in fact design information is not the same and it moves along directed paths in a discrete fashion. As discussed in Section 2.3.1, extensive and broad ranging research has addressed this problem, the most complete and comprehensive solutions using Discrete Event Simulation (DES). Simulation requires a significant investment of time to collect data and construct models, and these high fidelity inputs make simulation less useful for predictive metrics unless the structure being evaluated is very similar to those of the past. Design methods, processes and approaches vary significantly from program to program in the naval field, not to mention a constantly changing organizational structure (Johnson, 1980). This chapter introduces a discrete information flow variant of the path influence algorithm, not displacing the comprehensive results from a simulation, but producing valuable lead indicators from only a low fidelity multipartite network structure.

The second purpose of this chapter is to compare the network metrics developed in this thesis with the results from a DES on the same network. Full scale testing of warships is not usually tractable, and neither is testing on their design structures. A tow tank is used as a substitute for reality in hydrodynamics, and simulation is the tow tank equivalent for temporal design research. The logic behind the path influence algorithm can be coded as a DES, and this has been done to verify the algorithm. As shown in Section 5.4, path influence can be compared with Park centrality. This makes DES results comparable across the board, allowing the efficacy of static metrics to be tested under the influence of time.

Section 7.1 introduces the information flow variant of path influence, Section 7.2 applies it to the Sen Bulker Organization network, and Section 7.3 verifies the results with a simulation. Section 7.4 introduces more realistic simulation models of design to further test path influence. Section 7.5 discusses the importance of network structure to the temporal results using comparisons with Park and Winston centrality. Section 7.6 concludes the chapter.

7.1 Path Influence for Information Flow

Information is built upon and transformed as it flows through a design network, context is provided with each level of the network it passes through. In Chapter VI the concept of the information level of a node was developed, but that definition only provided node to node or steady state comparisons. Another way to think about design information is in terms of completeness. *Information completeness* (I_C) is the fraction of required information available at a given node, optionally as a function of time. I_C of one means that a node has access to all its required information, I_C of zero means the node has no information. Viewed in this way, information completeness is the inverse of uncertainty, where uncertainty is the lack of information (Daft and Lengel, 1986).

7.1.1 Information Completeness with Equal Weightings

Two path influence weighting schemes were developed in Chapter V for networks that represent design methods with evaluable functions, but it was noted that path influence could accept any weighting scheme irrespective of what a multipartite network represents. Just as influence is built up along paths, so is information. All that is required is a weighting scheme for information, and the path influence algorithm can be used.

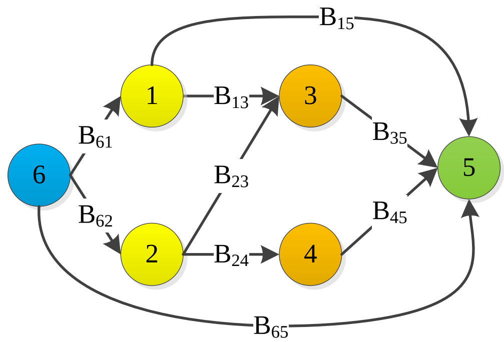
A node is information complete when it has received information from all of its inputs, so the information contribution of an arc can be computed using Eq. 7.1,

$$\mathbf{B}_{ij} = \frac{1}{k_{in}^{j}} \tag{7.1}$$

where $\mathbf{B}$ is a weighted adjacency matrix. This assumes that each input to a node contributes an equal amount of information and k_{in}^{j} is computed from the unweighted adjacency matrix. Computing the path influence matrix, $\mathbf{P}$, using Eq. 5.11 then provides an $n \times n$ matrix of the information contribution of every node to every other node. If the time it takes information to flow over an arc is equal across the network, then the information completeness of every node can be plotted as a function of discrete time using Eq. 7.2,

$$\mathbf{I_C}(t) = \mathbf{v}_o^T \sum_{t=0}^{t} \mathbf{B}^t \tag{7.2}$$

where $\mathbf{v_o}$ is a $n \times 1$ vector of the information completeness of the independent inputs at $t = 0$ with all other entries equal to 0. If the non-zero entries of $\mathbf{v}_o$ are equal to one, then the information content of every node at $t = n - 1$ will be one, meaning after all paths are accounted for every node will be information complete (input nodes' I_C does not change from $t = 0$ because they have zero in-degree). This formulation is also



(a) Weighted Network

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & \frac{1}{2} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

(b) Weighted Adjacency Matrix

Figure 7.1: Information Completeness Weighted Network Example

subject to the no loops condition discussed in Section 5.1.3, meaning iterations must be handled differently. Smith and Eppinger (1997a) use a form of Eq. 7.2 operating on a different matrix for a different purpose; handling the dynamics of iterations using an abstraction of DSMs. With some modification their method may prove promising if extended to multipartite networks.

To demonstrate Eqs. 7.1 and 7.2 a modified version of the example network from path influence and its I_C results are shown in Fig. 7.1 and Table 7.1. A sixth node and four arcs are added such that paths of length one, two and three connect the other nodes to Node 5. At $t = 1$, Nodes 1, 2, and 6 are information complete as the longest path to them is one arc long, while Node 5 has received a quarter of its information. At $t = 2$, Nodes 3 and 4 become information complete, while Node 5 receives another quarter of its information from a path two arcs long. At $t = 3$ all nodes are information complete because the longest paths in the network are three arcs long.

Table 7.1: Information Completeness in Discrete Time Example

	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
$t = 0$	0	0	0	0	0	1
$t = 1$	1	1	0	0	0.25	1
$t = 2$	1	1	1	1	0.5	1
$t = 3$	1	1	1	1	1	1

7.1.2 Information Completeness with Influence Weightings

Risk can be defined as probability×consequence, and there is an analogy to making decisions under uncertainty. The weightings of path influence can be thought of as consequences as well. More risk is accrued if an influential piece of information is missing versus an inconsequential one, greatly affecting subsequent design stages (GAO, 2002). To enact this concept, the adjacency matrix **B** is computed using the weighting scheme of Eq. 7.3,

$$\mathbf{B}_{ij} = \frac{|\mathbf{A}_{ij}|}{\sum |\mathbf{A}_{|j|}|} \tag{7.3}$$

noting that **A** has already been weighted for influence. Normalizing by the sum of the j^{th} column of **A** keeps the notation that information complete is signified by $\text{I}_C = 1$. The absolute values are used because negative influence does not counteract positive influence in this context.

7.1.3 Independent Flow Assumption

Path influence assumes that each path flows independently, when realistically information is held at a node until all inputs are received, context is added and then the information is released as a whole. Under this scenario, information completeness computed with the path influence algorithm is an upper bound, and its inverse is a lower bound on uncertainty. A pictorial example of this is shown in Fig. 7.2. At $t = 2$ the I_C of Node 6 would be 1/2, coming from input Node 2. But that information flows through Node 5, which is missing the contextual information provided by Node 1 which is four arcs away from Node 6. However, as shown in Section 7.4, the effect of this assumption on the relative completeness of nodes is minimal for largely hierarchical networks like the Sen Bulker Organization.

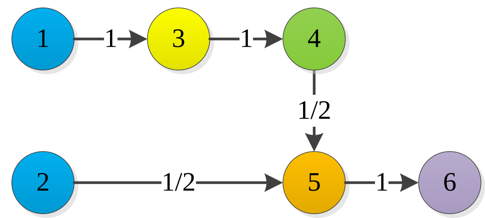


Figure 7.2: Information Completeness as an Upper Bound

7.2 Information Completeness Applied to the Sen Bulker Organization Network

Information completeness was computed for the Sen Bulker Organization network (described in Chapter VI) with both **B** weighting schemes using the path influence algorithm. The weightings for Eq. 7.3 are from the partial derivative weighting scheme of Chapter V, unchanged from the full network because the nodes that were removed do not influence any other nodes. However, the results do not directly correlate with those of Chapter V because of the absolute values used in Eq. 7.3.

7.2.1 Information Completeness with Equal Weighting Results

Table 7.2 and Figs. 7.3 and 7.4 show the I_C results of Node 33 (the Transportation Cost objective) from using the equal weightings of Eq. 7.1. Node 33 is at the end of the longest paths in the network, 11 arcs long, and therefore is not information complete until $t = 11$. Bar charts are used because information completeness changes in discrete jumps, not as a continuous function of time. The stacked bar charts in Figs. 7.3a and 7.3b show the contribution of each variable and parameter to the total information completeness respectively, highly informative figures. Engineers evaluating data at any point in time on the network structure have an indication of how much information they could have, and which inputs are driving that information.

As described in Section 7.1.3, the path influence algorithm provides a best case scenario for information completeness. For example, at time $t = 5$ the most information Node 33 could have is 56% of its total, meaning at a minimum the uncertainty at Node 33 is 44%. The inputs that contribute to the 56% I_C could be missing the context provided by inputs on paths more than four arcs long.

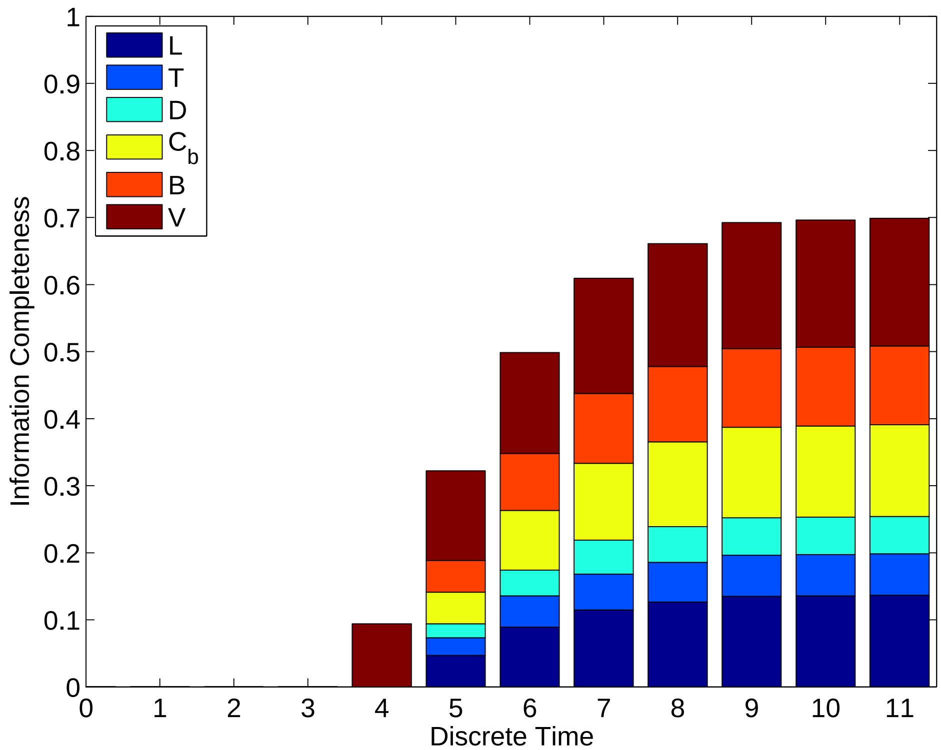
Table 7.2: Transportation Cost I_C with Equal Weightings Data

	Discrete Time							
Variable/Parameter	4	5	6	7	8	9	10	11
L	0	0.047	0.089	0.114	0.126	0.134	0.135	0.136
T	0	0.026	0.046	0.053	0.059	0.061	0.061	0.062
D	0	0.021	0.038	0.050	0.053	0.056	0.056	0.056
C_b	0	0.047	0.089	0.114	0.126	0.134	0.135	0.136
B	0	0.047	0.085	0.104	0.112	0.117	0.117	0.117
V	0.094	0.133	0.150	0.171	0.182	0.188	0.189	0.190
ζ_1	0	0	0.002	0.005	0.007	0.009	0.009	0.009
ζ_2	0	0	0.002	0.005	0.007	0.009	0.009	0.009
ζ_3	0	0	0.002	0.005	0.007	0.009	0.009	0.009
η_1	0	0	0.002	0.005	0.007	0.009	0.009	0.009
η_2	0	0	0.002	0.005	0.007	0.009	0.009	0.009
η_3	0	0	0.002	0.005	0.007	0.009	0.009	0.009
Round Trip Miles	0.094	0.125	0.125	0.133	0.133	0.133	0.133	0.133
Fuel Price	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021
Cargo Handling Rate	0.094	0.094	0.094	0.094	0.094	0.094	0.094	0.094
Total	0.302	0.560	0.749	0.885	0.947	0.991	0.996	1.000

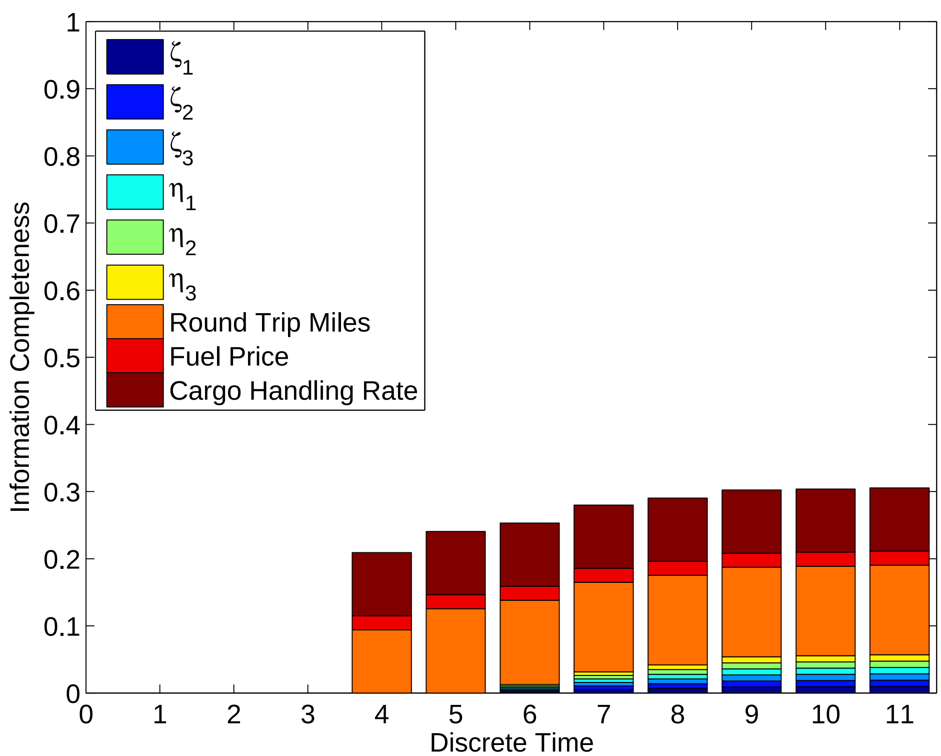
7.2.2 Information Completeness with Influence Weighting Results

The method and results described in Section 7.2.1 are based on the assumption that all information arriving at a node has the same value. This is definitely not the case in a design environment, some information is much more important. This is readily apparent by comparing Table 7.2 and Figs. 7.3 and 7.4 against Table 7.3 and Figs. 7.5 and 7.6 where influence is accounted for using the weighting scheme of Eq. 7.3. I_C approaches unity much faster and the variables contribute a greater portion to the total I_C.

The first result is expected because the vast majority of the partial derivative weights used to create **B** are less than one, meaning the more arcs traversed the

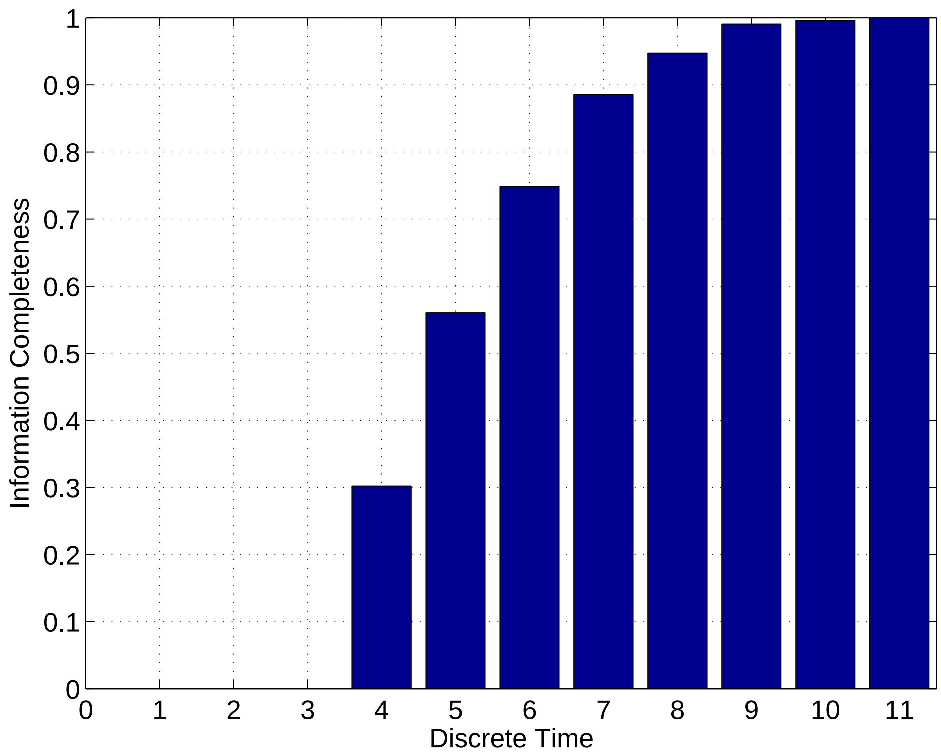


(a) Variable Inputs to Transportation Cost



(b) Parameter Inputs to Transportation Cost

Figure 7.3: Transportation Cost I_C with Equal Weightings Part One

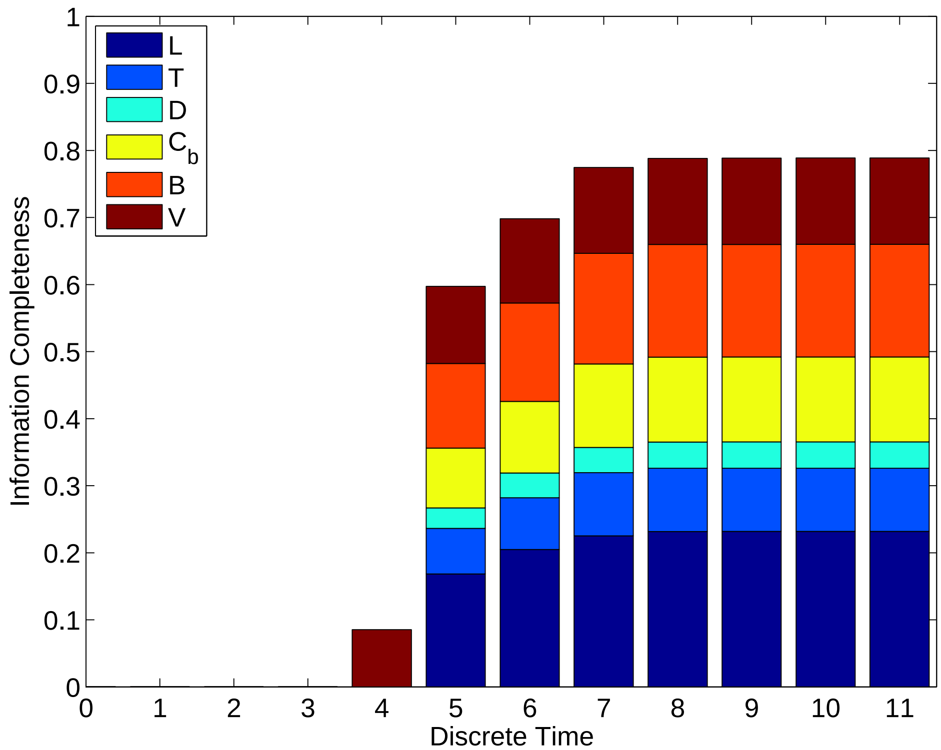


(a) Total Transportation Cost I_C

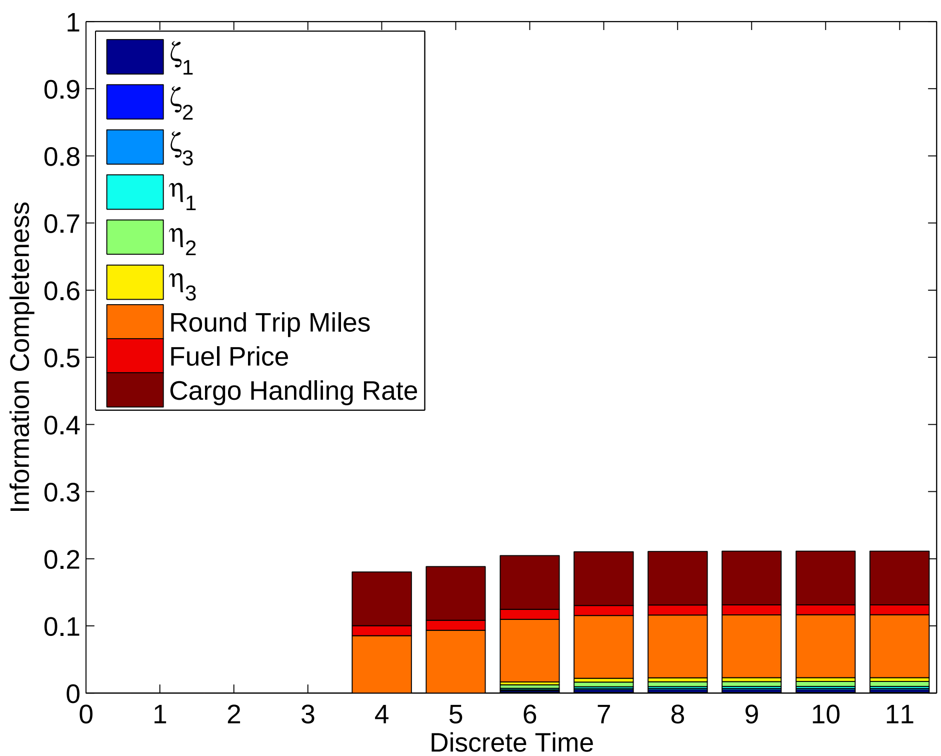
Figure 7.4: Transportation Cost I_C with Equal Weightings Part Two

Table 7.3: Transportation Cost I_C with Influence Weightings Data

	Discrete Time							
Variable/Parameter	4	5	6	7	8	9	10	11
L	0	0.168	0.205	0.225	0.232	0.232	0.232	0.232
T	0	0.068	0.077	0.094	0.094	0.094	0.094	0.094
D	0	0.031	0.037	0.038	0.039	0.039	0.039	0.039
C_b	0	0.089	0.107	0.125	0.127	0.127	0.127	0.127
B	0	0.126	0.147	0.165	0.168	0.168	0.168	0.168
V	0.085	0.115	0.126	0.128	0.128	0.129	0.129	0.129
ζ_1	0	0	0.001	0.002	0.002	0.002	0.002	0.002
ζ_2	0	0	0.002	0.003	0.003	0.003	0.003	0.003
ζ_3	0	0	0.002	0.002	0.002	0.002	0.002	0.002
η_1	0	0	0.002	0.003	0.003	0.003	0.003	0.003
η_2	0	0	0.005	0.007	0.007	0.007	0.007	0.007
η_3	0	0	0.004	0.006	0.006	0.006	0.006	0.006
Round Trip Miles	0.085	0.093	0.093	0.094	0.094	0.094	0.094	0.094
Fuel Price	0.015	0.015	0.015	0.015	0.015	0.015	0.015	0.015
Cargo Handling Rate	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080
Total	0.265	0.786	0.903	0.985	0.999	1.000	1.000	1.000

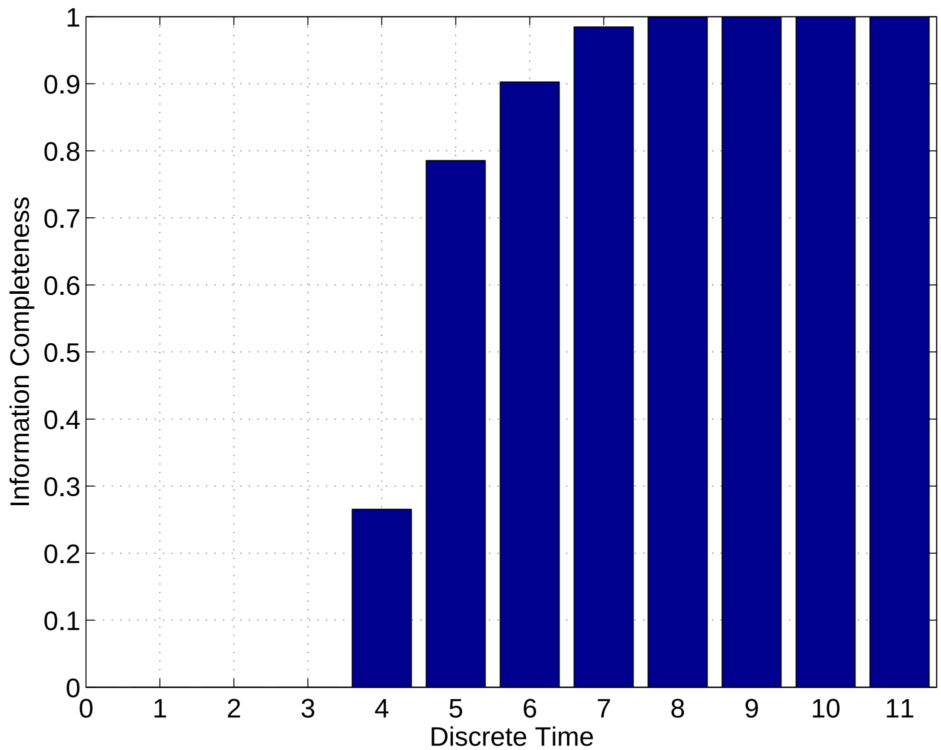


(a) Variable Inputs to Transportation Cost



(b) Parameter Inputs to Transportation Cost

Figure 7.5: Transportation Cost I_C with Influence Weightings Part One



(a) Total Transportation Cost I_C

Figure 7.6: Transportation Cost I_C with Influence Weightings Part Two

less the influence. As mentioned, the longest path in the network is 11 arcs long. Even if all the weights along such a path were 0.95, the contribution to Node 33 is $0.95^{11} = 0.57$, keeping in mind that number must also be normalized by the total contribution of all 459 other paths leading to Node 33. The I_C change of Node 33 from $t = 8$ to $t = 11$ is only 0.00068 to put things in perspective. If the weighting scheme accurately reflects influence, then the fact that variables contribute more to the total I_C than parameters is a good indication of a well formulated model. The objects under investigation are generally of most import. A case could be made for making Cargo Handling Rate and Round Trip Miles variables to quantify the sensitivity of the formulation to environmental factors beyond technical control.

7.3 Verification Using Discrete Event Simulation

The underlying mathematics of the path influence algorithm used in Eq. 5.11 and Information Completeness of Eq. 7.2 are the same. It is possible to verify both the mathematics and the code used to generate the results using a DES of an identical network structure and with the same assumptions. The two major assumptions are that information flow is independent, and the time to traverse each arc is equal. This section describes such a simulation using the Sen Bulker Organization network.

7.3.1 Simulation Logic for Information Completeness

Two basic components of a simulation are entities and locations (Banks et al., 2010). Entities move from location to location according to prescribed logic and timing. For a network, entities are individual pieces of information or influence while locations are the nodes and arcs. The assumption of path influence for discrete time is that it takes one unit of time to traverse any arc, and this was recreated in the simulation by having each piece of information wait for one unit of time at each node. Entities at a node are copied such that an identical version leaves along every

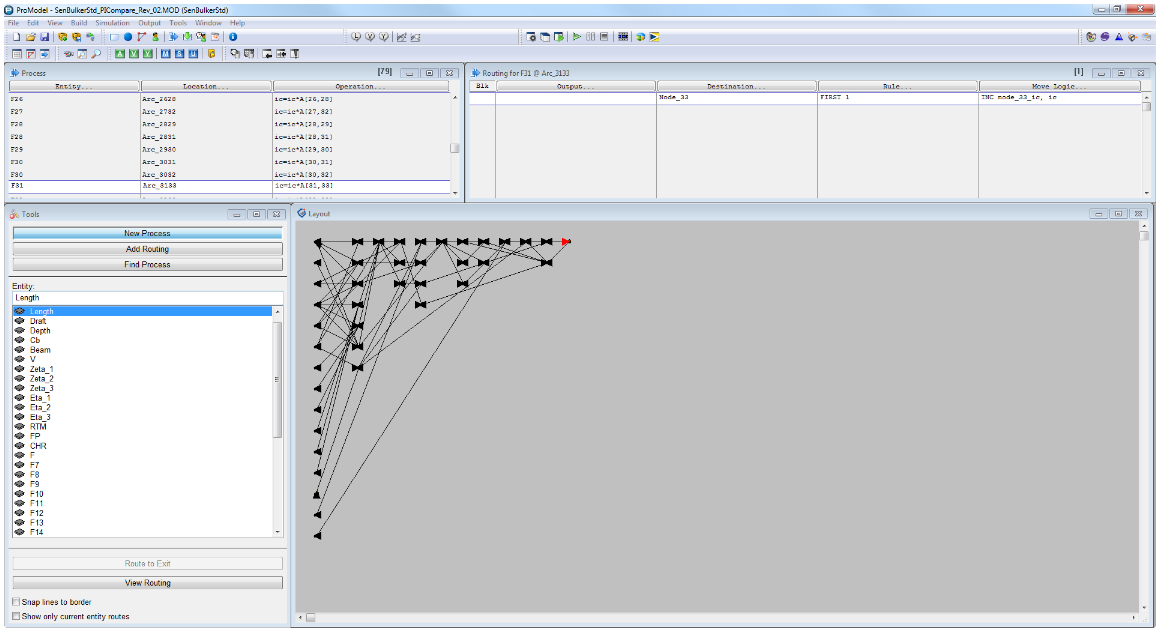


Figure 7.7: *ProModel* Interface for the Sen Bulker Organization

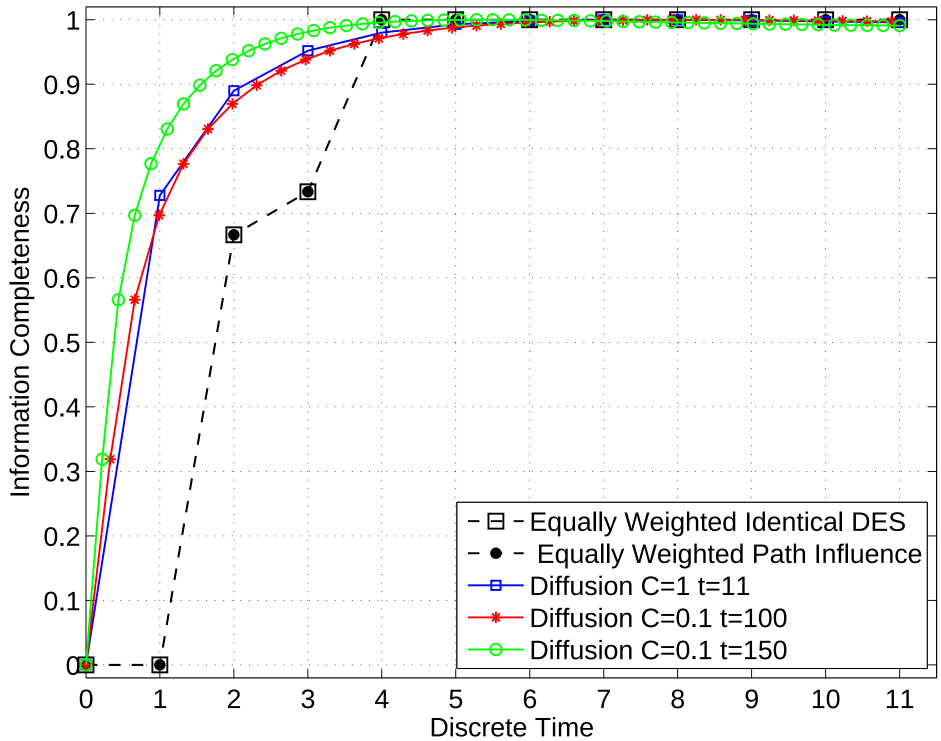
arc leaving the node. As entities move over an arc their contribution (influence or information) is multiplied by the weight that corresponds to the arc's entry in the **B** matrix. When an entity reaches a node, the node's I_C is incremented by the contribution of the entity. The simulation is initialized by having an entity enter each input node at time $t = 0$ with their contribution set to one, and letting the simulation run until all entities have exited the system. An exit occurs when the entity reaches a terminal node, in this case only Node 33 is a terminal node.

Simulation software is primarily written for manufacturing and supply chain management, *ProModel* was used in this case, requiring creative thinking to apply it to design network problems. It has a graphical interface, meaning a significant investment of time to construct the model, and can take even more time to debug. A screen shot of the basic interface with the Sen Bulker Organization network loaded is shown in Fig. 7.7, and the text version of the simulation described above is shown in Section B.4.

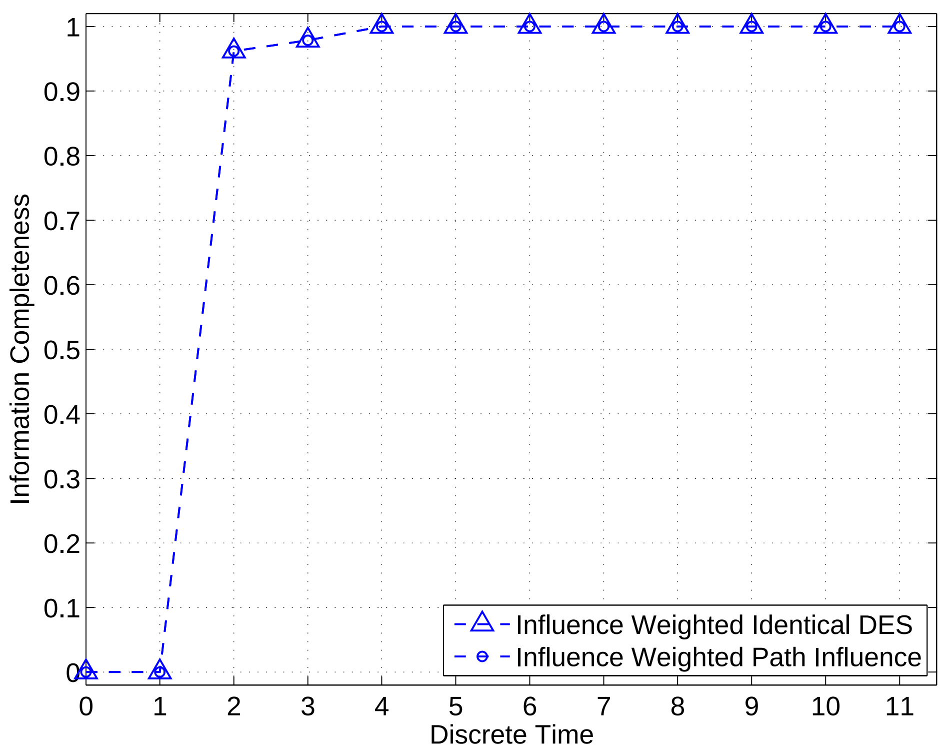
7.3.2 Verification Results

Figs. 7.8a, 7.9a and 7.10a plot the DES, equally weighted path influence and diffusion results for the three objectives of the Sen Bulker Organization network. Immediately apparent, and this is true of all nodes, is that the DES and path influence results are exactly the same. This verifies that the path influence algorithm of Eqs. 5.11 and 7.2 are correctly formulated and implemented. The purpose of including the diffusion results is to show that they can correlate with the more realistic discrete results. The diffusion results were calculated with three different parameter combinations as shown in the figures, and the total diffusion time was scaled to the discrete time. Information lead is a metric specific to diffusion, but the highest information lead nodes from Chapter VI are input nodes, yielding an I_C of one from time $t = 0$. As formulated I_C cannot exceed one. No scaling, parameters or initial conditions are required to use the path influence algorithm, it operates solely on the network's directed structure. Figs. 7.8b, 7.9b and 7.10b plot the DES and path influence results for the objectives when the influence weighting of Eq. 7.3 was used. Again, the results between simulation and path influence are exactly the same, further verifying the formulation and implementation. Overall the results demonstrate that path influence's precision is weighting scheme independent. Networks representing other types of design structures can use the path influence algorithm with equal confidence.

The lag first moment of area, developed in Chapter VI for information level curves, is a concise single number for each node that compares their overall behavior over the course of time. Hereafter it is referred to as the lag first moment, or lag FM. If anything the metric is more suitable for information completeness because it never decreases with time. In a single run information can only become more complete. Lag plots are normalized by the highest moment, such that the relative lag between nodes is easily distinguished. Absolute information gap curves are equally applicable,

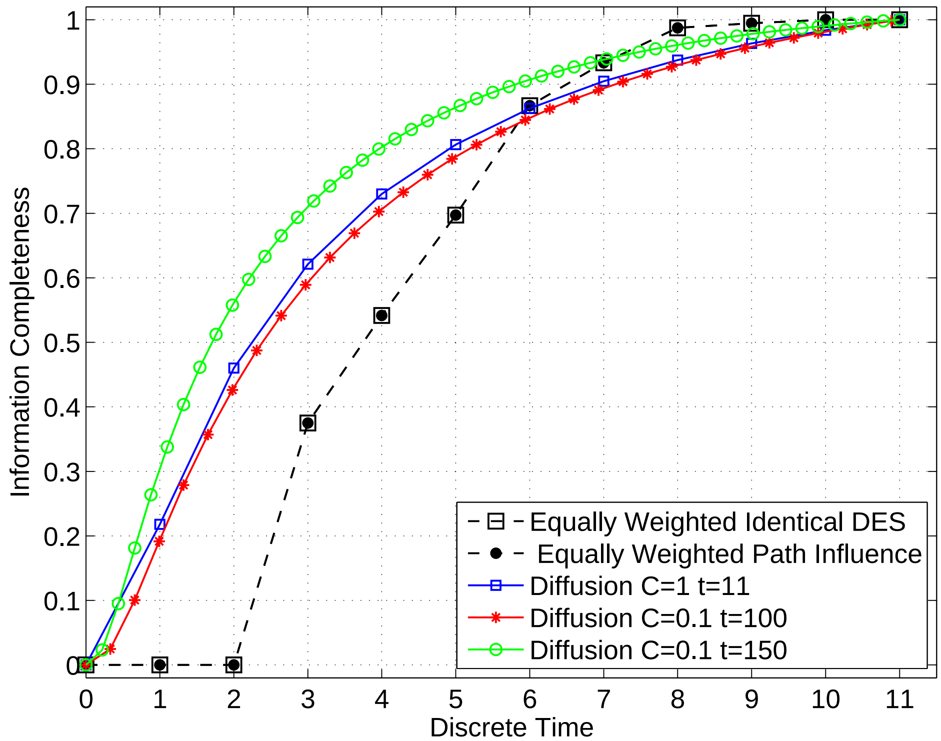


(a) Equally Weighted Light Ship Mass I_C

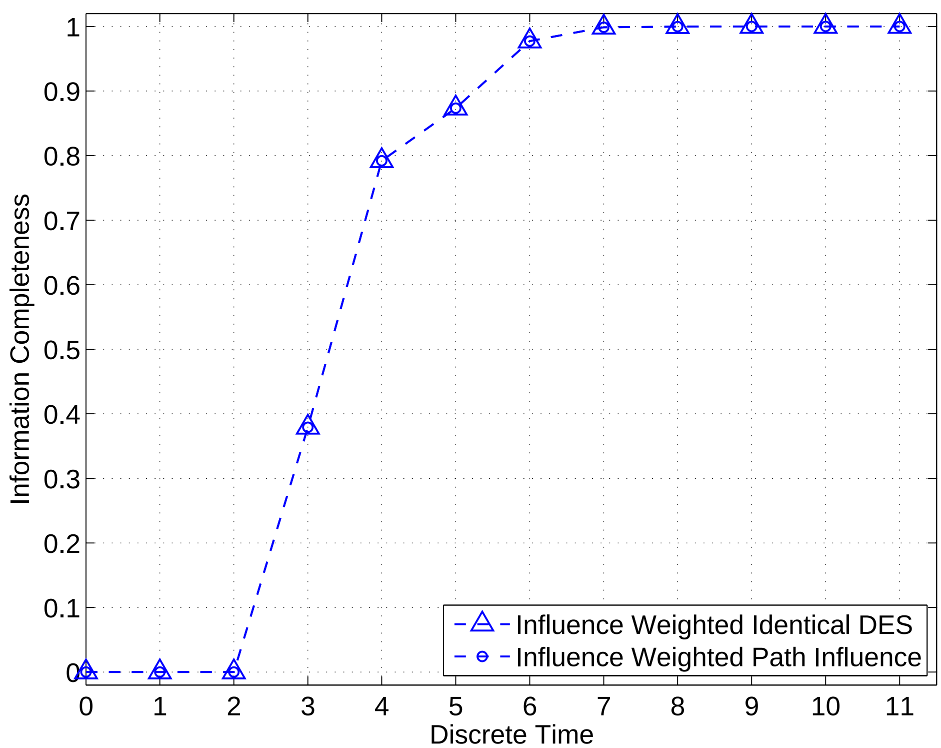


(b) Influence Weighted Light Ship Mass I_C

Figure 7.8: Light Ship Mass I_C Verification

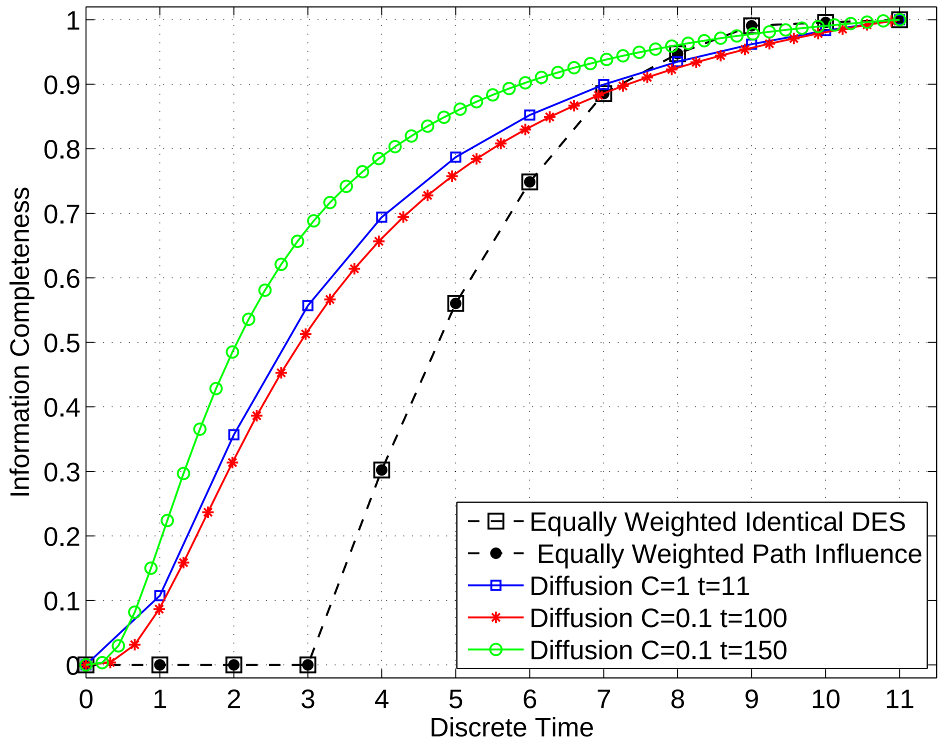


(a) Equally Weighted Annual Cargo I_C

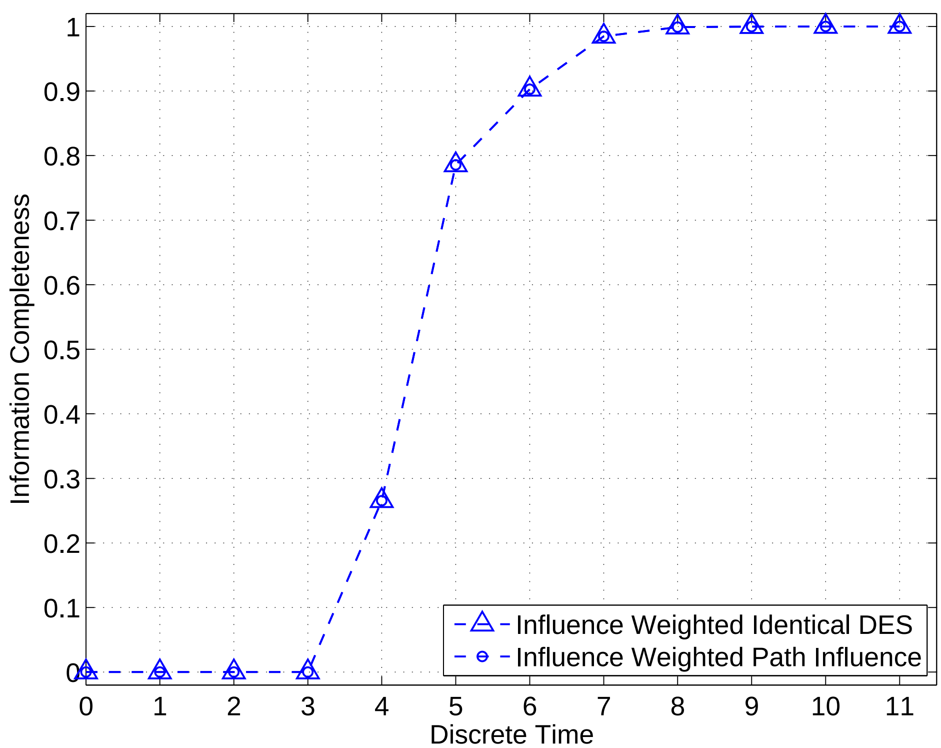


(b) Influence Weighted Annual Cargo I_C

Figure 7.9: Annual Cargo I_C Verification



(a) Equally Weighted Transportation Cost I_C



(b) Influence Weighted Transportation Cost I_C

Figure 7.10: Transportation Cost I_C Verification

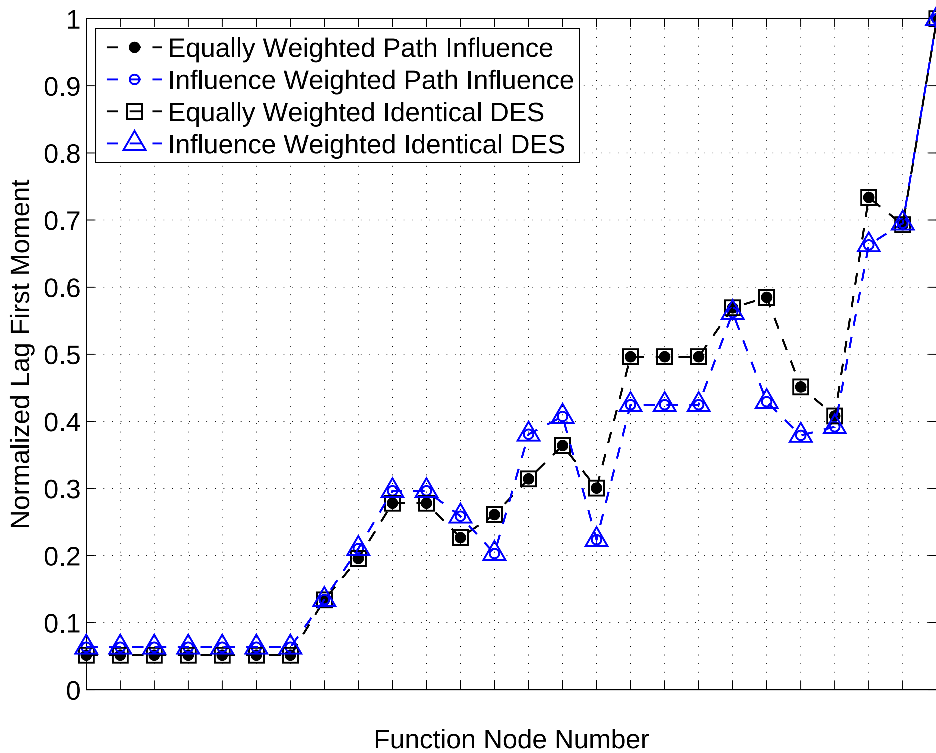


Figure 7.11: All Functions Lag First Moments

but require no absolute value as I_C only ranges between zero and one.

Fig. 7.11 shows the normalized lag first moments for all the functions in the Sen Bulker Organization network using both equal and influence weightings. Variables and parameters are not plotted because they have zero lag. Fig. 7.12 shows the information gap curves for the same cases. The path influence and DES results are again shown to be identical, expected as Figs. 7.11 and 7.12 are merely different views of the same results. Of note, Fig. 7.11 shows that the lag for both weighting schemes trends upward in the same manner with increasing node number. This is a product of the network's hierarchy, discussed further in Section 7.5. Also, the faster convergence for influence weighting mentioned in Section 7.2.2 is easily visible in Fig. 7.12.

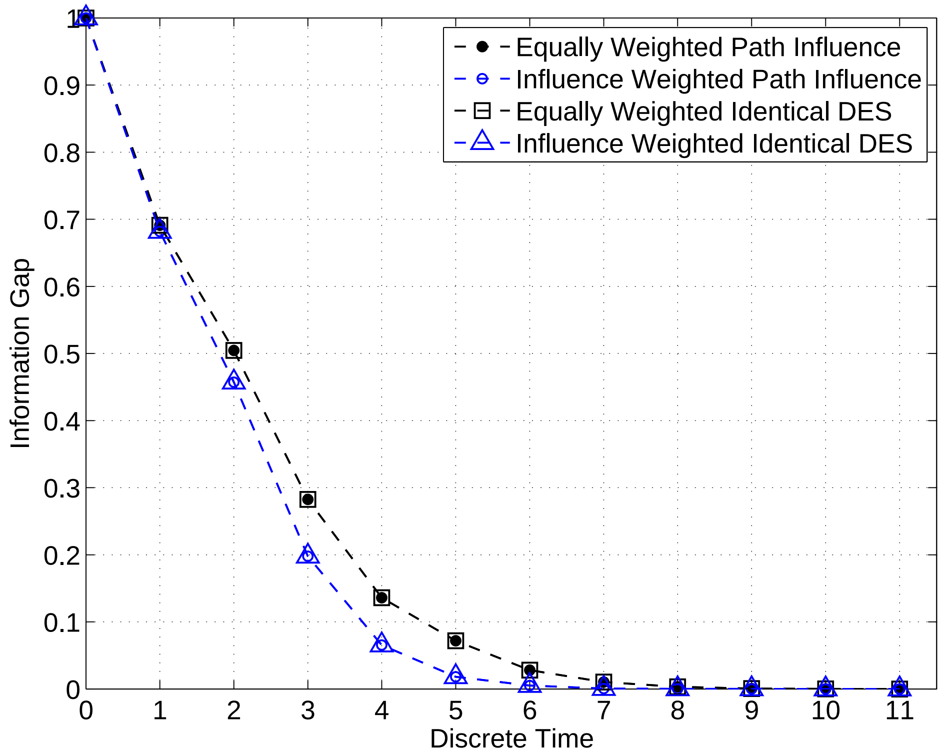


Figure 7.12: All Functions Information Gap

7.4 More Realistic Design Scenarios

A more realistic model of discrete design information flow is that information is held at a node until all inputs are received, context is added and then the information is released as a whole. I_C computed with the path influence algorithm is not capable of representing this scenario directly. However, as mentioned in Section 7.1.3 it does provide an upper bound on information completeness and thus lower bound on uncertainty for each node at each point in time. These bounds can be compared relative to one another, and a legitimate question is whether actual information flow trends in the same way as the bounds, or in a different fashion well below them. If information trends in a similar way, then I_C computed with the path influence algorithm remains a legitimate indicator of discrete design information flow.

Two different simulation scenarios of this more realistic model of information flow were tested against the path influence results using the Sen Bulker Organization network. The first simulation collates all required information at every node before releasing it to downstream nodes, but the time of travel on arcs is still set to one and is not stochastic. For real structures that are actually capable of adhering to rigid delivery schedules this may be a suitable model. The second simulation is more complicated, as the travel time over every arc is pulled from a triangular distribution. The minimum value of the distribution is the same as the mode, making a right triangle. This conforms to common experience that design will take up all the time allowed to it, never early, but often late. The minimum and mode are set to one time unit, while the maximum is set to five. In this scenario a perfectly on schedule (probabilistically impossible) simulation run would have identical results to the non-stochastic first scenario.

7.4.1 Non-Stochastic Move Time Results

Fig. 7.13a displays the first moments of lag of the non-stochastic move time simulation vs. those of the baseline DES and equally weighted path influence. The moments for the variables and parameters are not shown because they are zero. Of major importance is that with two exceptions, one minor, the trend of the nodes relative to one another between the non-stochastic move time simulation and path influence are the same. Though the magnitudes of the moments do not scale, the ranking of the magnitudes is the same.

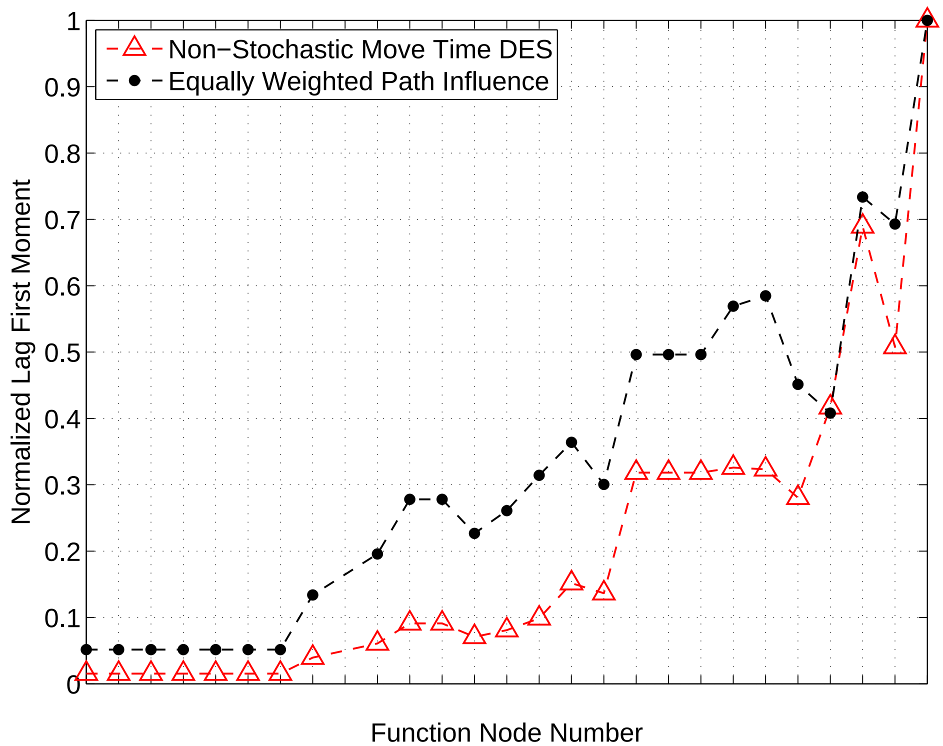
The major exception to this is Node 30. The explanation is that Node 30 is a 9^{th} level function yet is only two arcs away from Cargo Handling Rate, an input node. With path influence, after two time steps Node 30's information completeness goes up, as shown in Fig. 7.13b. With grouped information movement, the Cargo Handling Rate information must wait at an intermediate node until it becomes information complete before moving on. In this case, at least until time $t = 4$. This points to the importance of hierarchy in the network's structure. If the network is perfectly hierarchical, meaning information must pass through every level of the network, then independent and dependent information flow results would be the same. The Sen Bulker Organization network is mostly hierarchical, so the results match up fairly well. Where the hierarchy is broken, i.e. Node 30, is where the trends do not match.

Fig. 7.13a shows that the non-stochastic move time simulation has lower normalized first moments. The explanation for this is similar to that for Node 30's situation, but deals with the normalization. Fig. 7.4a of Node 33's I_C computed with the path influence algorithm shows that Node 33 starts receiving information at $t = 4$. With grouped information flow, Node 33 receives all of its information at one time, $t = 11$, making its lag moment comparatively astronomical. The longest paths from input nodes to the majority of other nodes in the network are much shorter, visibly evident from the networks shape. This skews the moment normalization because their moments are comparatively low. The correlation between the distribution of path lengths and properties of information completeness is an area for future research.

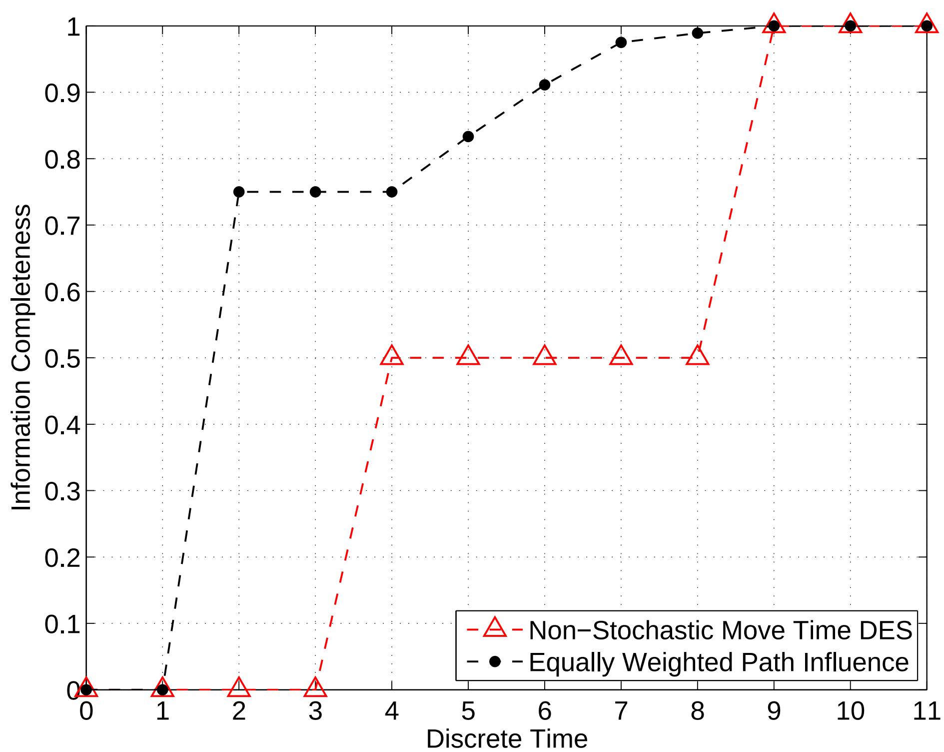
7.4.2 Stochastic Move Time Results

Fig. 7.14a shows a box plot of the lag first moments for the stochastic move time simulation. The simulation was run for 999 replications, the maximum allowable within *ProModel*. It is interesting that there are virtually no outliers below the median values, while there are very many above the median value for any particular node. This is likely a combination of the grouped information structure, all inputs must be early to proceed early versus only one input must be late to proceed late, and the right triangular distribution where the mean is much closer to the minimum value. Though not shown, the mean and medians for each lag distribution are nearly identical.

Fig. 7.14b shows the normalized lag first moment distribution extremes, forming a corridor, and the equally weighted path influence results for the functions nodes. It



(a) Normalized Lag First Moments

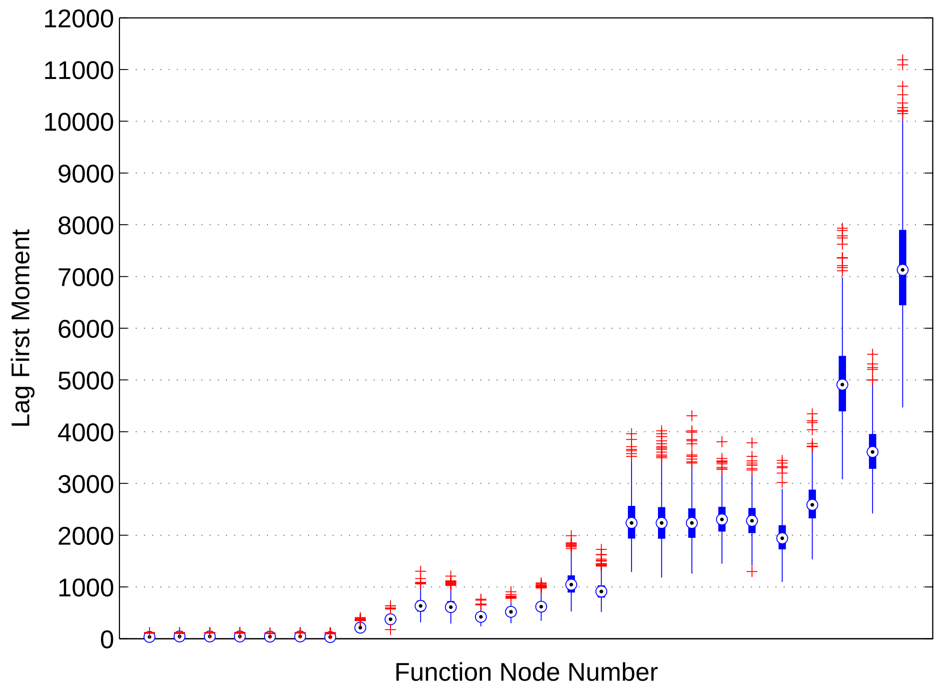


(b) Node 30 I_C

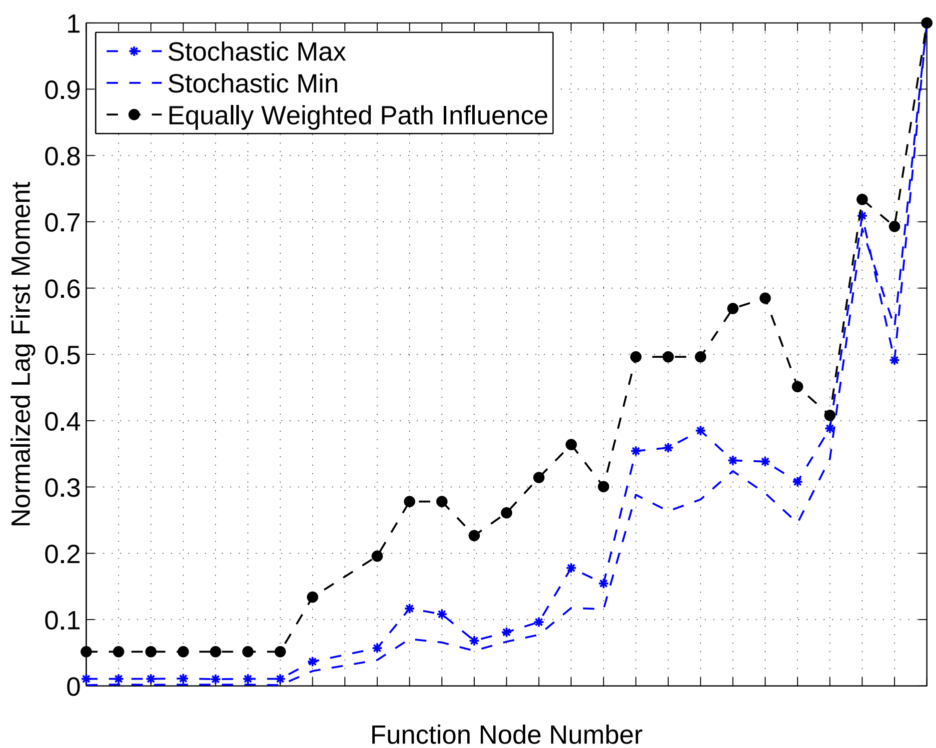
Figure 7.13: Non-Stochastic Move Time Lag First Moment Comparison

is apparent that all three trend quite similarly, meaning that even in the stochastic case the network's hierarchy is rather important. The means of each distribution are plotted along with the non-stochastic move time simulation and equally weighted path influence results in Fig. 7.15a, making the effect of hierarchy even more clear. Despite the large discrepancies in information gap, shown in Fig. 7.15b, the normalized lag first moment plots are nearly identical for the non-stochastic and stochastic move time simulations. Again, the exceptions are Node's 29 and 30. The implication is that if the network is sufficiently hierarchical the relative lag first moments may be move time independent, and the path influence algorithm can capture the trend.

Fig. 7.15b is an information gap plot comparing the mean gap of the stochastic simulation with the deterministic non-stochastic and path influence results. The plot verifies that the path influence algorithm provides an upper bound on information completeness (meaning lower bound on information gap) for the non-stochastic case. This is also true for the stochastic case because the minimum value of the triangular distribution was set to one. Despite this, path influence does not realistically indicate flow times for the stochastic case.

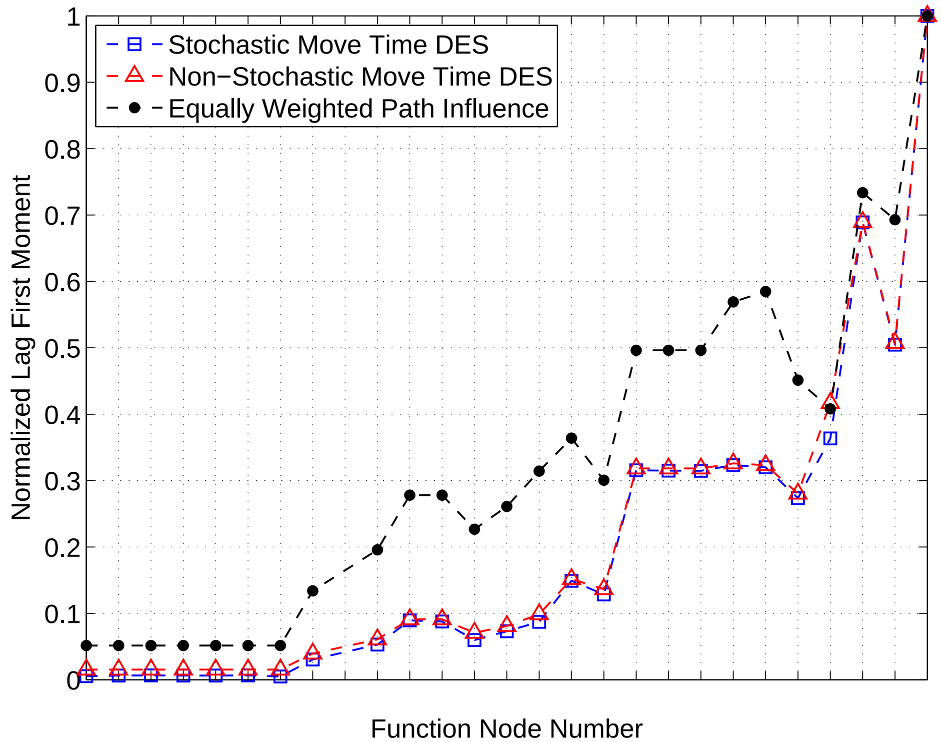


(a) Lag First Moment Distribution Box Plot

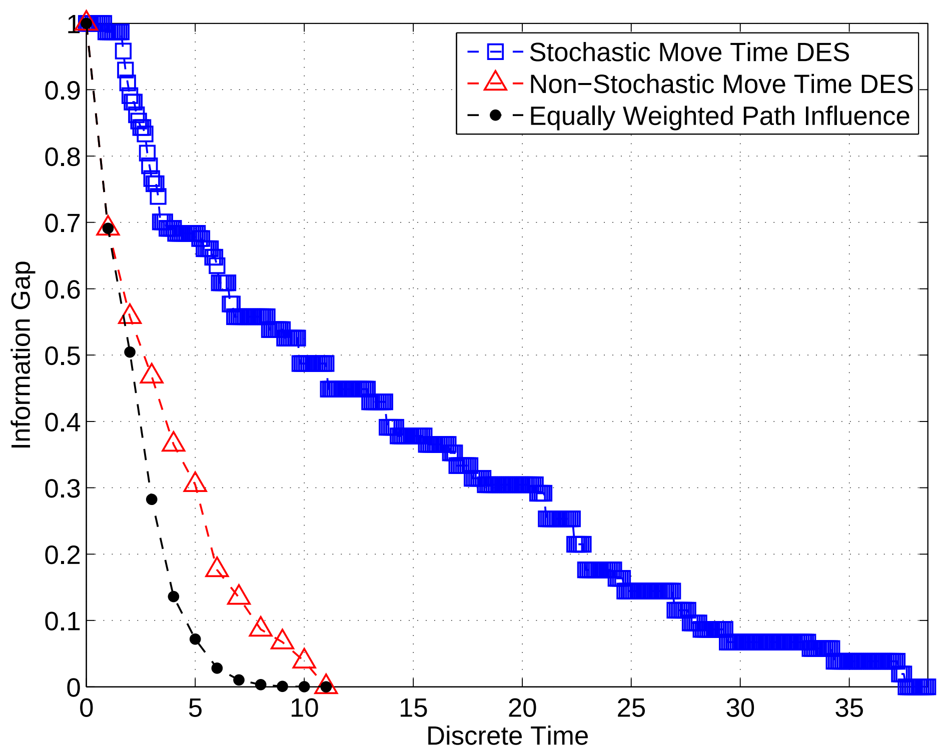


(b) Min Max Lag First Moment Corridor

Figure 7.14: Stochastic Move Time Lag Distribution



(a) Mean Lag First Moment



(b) Mean Information Gap

Figure 7.15: Stochastic Move Time Lag First Moment and Information Gap

7.5 Multipartite Significance to Temporal Results

The previous sections have demonstrated that the path influence algorithm can correctly identify the trending of discrete information flow across a largely hierarchical design network. This is an indication that fundamental structure, not the details of discrete information flow, is the deterministic factor in the overall behavior of a network. The fundamental structure of the Sen Bulker Organization network is hierarchical, but the ordering and inclusion of nodes was determined by the contextual multipartite definition. It is possible to create one mode projections of multipartite networks, and even to map them to each other, but neither of these approaches captures the information flow which is primarily governed by the full context of all nodes acting together.

Because the Sen Bulker Organization is largely hierarchical, Park centrality and Winston centrality, which are node centric measures of impact and influence respectively, should correlate with the simulation results as shown in Fig. 7.16. All metrics displayed in Fig. 7.16 have been normalized by their highest value, occurring at Node 33 in all cases. The Park centrality results are multiplied by negative one such that highly impacted nodes reflect a high value, consistent with the idea of lag. The plots show that the trending of the higher level nodes is well accounted for with all metrics. Across all function nodes, Park centrality better mirrors the trend than Winston centrality. This is expected, as Park centrality is a purely structural measure. Simply the function level provides a very accurate trend approximation with the exception of Nodes 29 and 30. The function level defines the hierarchy, increasing with path length from the input nodes, thus higher lag. As explained in Section 7.4.1, Nodes 29 and 30 are one and two arcs away from the input Cargo Handling Rate, thus their lag is much reduced. Both Park and Winston centrality capture this. The conclusion is that for sufficiently hierarchical structures, network methods can provide effective lead indicators for discrete temporal behavior.

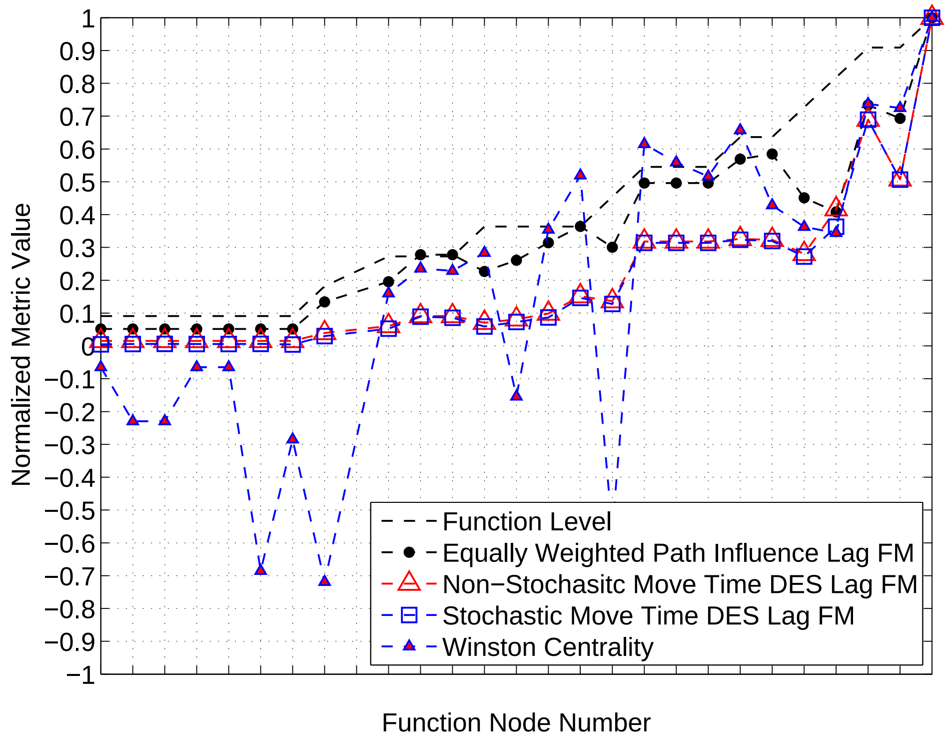
Table 7.4 compares the equally weighted path influence lag first moments with several other network metrics, using a color scale based on their absolute maximum value. The table reinforces the observations made about Fig. 7.16.

Table 7.4: Network Metric Comparisons

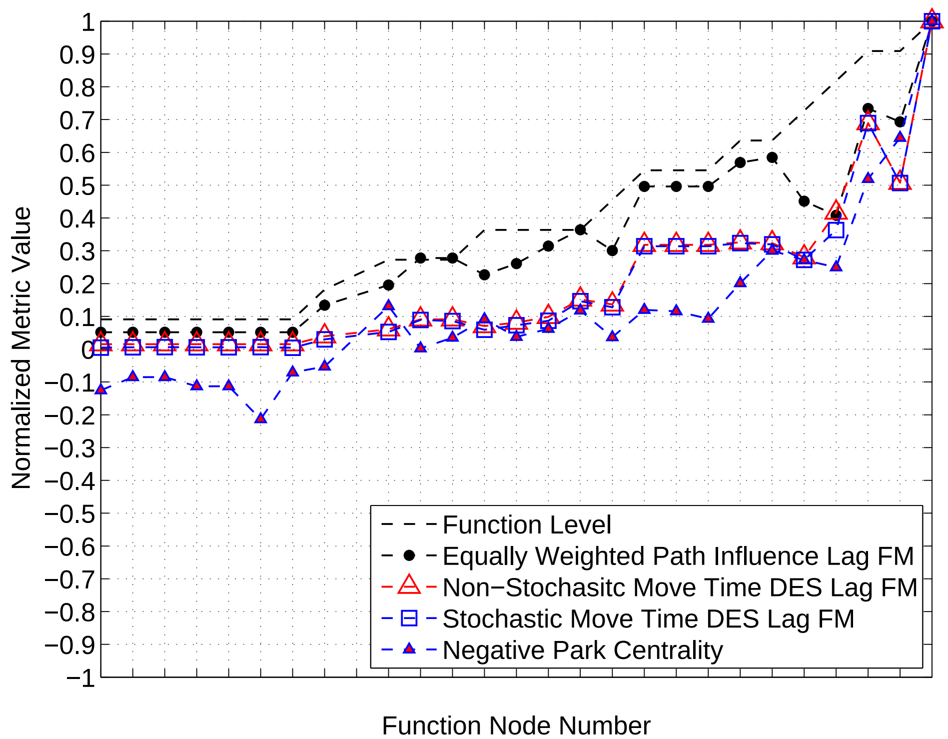
#	Node	Path Influence FM Lag	Winston Centrality	Park Centrality	Diffusion FM Lag
7	F_n	**1.00**	**0.36**	**20.33**	**2.96**
8	Steel Mass	**1.00**	**1.28**	**13.84**	**0.36**
9	Outfit Mass	**1.00**	**1.28**	**13.84**	**0.36**
10	a	**1.00**	**0.36**	**18.33**	**0.20**
11	b	**1.00**	**0.36**	**18.33**	**0.20**
12	Δ	**1.00**	**3.82**	**34.63**	**0.28**
13	Sea Days	**1.00**	**1.59**	**11.50**	**509.57**
14	P	**2.60**	**4.00**	**8.63**	**4.46**
15	A_{co}	N/A	N/A	N/A	N/A
16	Ship Costs	**3.80**	**-0.89**	**-21.25**	**37.74**
17	Machinery Mass	**5.40**	**-1.31**	**-0.38**	**58.46**
18	Daily Consumption	**5.40**	**-1.27**	**-5.67**	**333.25**
19	Fuel Cost	**4.40**	**-1.58**	**-14.80**	**566.35**
20	Light Ship Mass	**5.07**	**0.86**	**-6.08**	**66.41**
21	Fuel Carried	**6.10**	**-1.97**	**-10.07**	**610.14**
22	Capital Charges	**7.07**	**-2.89**	**-19.15**	**528.91**
23	Δ_{DW}	**5.83**	**3.12**	**-5.94**	**548.27**
24	Running Costs	**9.63**	**-3.43**	**-19.46**	**851.81**
25	Port Costs	**9.63**	**-3.11**	**-18.75**	**816.90**
26	Stores&Water	**9.63**	**-2.87**	**-15.03**	**805.32**
27	Voyage Costs	**11.05**	**-3.66**	**-32.62**	**835.90**
28	Δ_{Cargo}	**11.36**	**-2.39**	**-48.64**	**819.91**
29	Port Days	**8.76**	**-2.02**	**-44.04**	**892.87**
30	RTPA	**7.92**	**-1.92**	**-40.43**	**882.15**
31	Annual Cargo	**14.24**	**-4.10**	**-84.18**	**1025.75**
32	Annual Costs	**13.45**	**-4.04**	**-104.52**	**892.98**
33	Transportation Costs	**19.42**	**-5.57**	**-162.19**	**1108.87**

Legend

Absolute Maximum	**Absolute Minimum**



(a) Winston Centrality Comparison



(b) Park Centrality Comparison

Figure 7.16: Network Metric Comparisons

7.6 Conclusions and Contributions

The purpose of this chapter was to extend the path influence algorithm to temporal problems, and verify it with discrete event simulation, comprising the fourth major contribution of this thesis:

- Introduction of a discrete information flow equivalent of the path influence algorithm, with requisite network weighting schemes
  - Verification of the path influence algorithm using discrete event simulation
  - Extension of diffusion metrics for discrete information flows
  - Testing of path influence and other metrics against more realistic discrete event simulations

Specifically, the discrete event simulation was formulated using the same assumptions of path influence, and the results were an exact match. To test the algorithm against more realistic design scenarios two simulations of increasing complexity were run. Both with grouped information, differing by non-stochastic and stochastic information flow times. Common trends were revealed, indicating that the discrete behavior of information flow is primarily driven by a network's hierarchical structure, and thus the contextual multipartite definition. Given the importance of structure, the static metrics of previous chapters were compared with the simulation results, and also predicted the general trend. It has been said that "the truth is a function of time," and this chapter has demonstrated that contextual multipartite network structural indicators can capture the influence of time in certain circumstances (Winter, 2014). In the face of changing design structures, the simplicity of network methods makes refreshing results quick, a claim unmatched by high fidelity alternatives.

CHAPTER VIII

Conclusion

Acquisitions fail due to complex interactions between many domains, including social and technical. Typical research focuses on one domain or another, e.g. process, product or organization. Systems Engineering is responsible for the bigger picture, but lacks early stage predictive methods to comprehend the complete problem. The structure and challenges of design are a microcosm of acquisition; through multiple levels of context and increasing scale, fundamental relationships affect the outcome of a design, and thus acquisition as a whole. This thesis broadens the application of network theory for naval design from the analysis of physical systems to the general structure of design. This chapter restates the major novel contributions of the thesis outlined in Chapter I, and provides a detailed list of all contributions and the work to support them.

8.1 Major Novel Contributions

Three research questions were posed in Chapter I:

1. Can the structure of design (approach, process, methods, tools and organization) be accounted for?

2. Can a design be understood without designing anything?

3. Can the impact and timing of information be understood in advance?

The first major novel contribution addresses the first question, and was the introduction of a contextual multipartite network approach to represent the structure of naval design, first outlined in Chapter III. A single network with multiple node types is used based on a hypothesis that in complex product design, elements of a domain do not directly influence one another, they must have context provided by another domain. This insight leads one toward the multipartite network structure. The multipartite networks presented in Sections 3.3 and 3.4 accurately represent ship design methods, and reveal designer intent. As used in other chapters they also represent processes and organizations, in hierarchy, context and fidelity. The author argues that the multipartite representation is an accurate reflection of how designers think. Thus, a multipartite representation is a way of increasing a designer's understanding of the methods they use and the processes and organizations of which they are a part.

The second major novel contribution addresses the first question, and was the application and extension of existing network mathematics used on multipartite design networks, providing meaningful predictive insight. The majority of the supporting work is found in Chapter IV. It was demonstrated that the structure of engineering formulations alone provides information through static network structural analysis to be useful in a new way. Analysis of the Watson & Gilfillan and Sen Bulker problem multipartite networks proved that multipartite network representations of ship design formulations are feasible, and are accurate. Analyzing the networks correctly identified what naval architects intuitively understand the formulations, correctly identifying design drivers, constraints and other features of structure. The multipartite network structure now has a demonstrated ability to represent naval design, with corresponding analysis methods to better comprehend it. The successful construction and analysis of multipartite design networks also contradicts the common practice of separating node types into separate homogenous networks or matrices.

The third major novel contribution addresses the second question, and was the recognition that existing algorithms for finding path lengths in networks can be used to capture all node to node influences across multipartite design networks. An algorithm for this purpose termed path influence, and its supporting network weighting schemes were developed in Chapter V. When the network is weighted using partial derivatives, the path influence algorithm can be used to produce first order Taylor series expansions. Path influence was implemented on the Sen Bulker network, and results were compared with a full factorial design of experiments. Path influence produced highly accurate predictive metrics with a small fraction of the total function count required for the design of experiments. The output of path influence is an $n \times n$ matrix, so a new metric termed Winston centrality was developed that ranks nodes based on their total incoming and outgoing influence. Winston centrality allowed for comparisons with the static metrics of Chapter IV, the results were consistent verifying the earlier methods. Using path influence to generate insight is advantageous due to the inherent network representation, allowing the intermediate contextual nodes of a network to be adequately represented and understood. Input to output influence can be traced through these intermediate nodes, providing an intuitive understanding not necessarily possible from standard methods.

The fourth novel contribution addresses the third question, and was the application of network diffusion, based on Fick's second law, to model continuous information flow across design networks. This was described in Chapter VI. The diffusion analogy is formulated so that the primary input is a contextual multipartite design network, making it an easy extension to the methods and metrics of previous chapters. The primary advantages of the continuous information flow analogy are that it requires very few inputs and provides a closed form solution. Representative bottleneck and lag networks were formulated as canonical design problems along with their diffusion analysis and potential solutions. The Watson & Gilfillan and Sen Bulker Organization networks were also analyzed. The results of all four examples were interpreted in accordance with the diffusion analogy, and shown to be insightful and congruent with the analysis methods presented in previous chapters. Absolute Information Gap, Information Level, and First Moments of Area for Lead and Lag were introduced as measures and metrics for continuous information flow.

The fifth novel contribution addresses the third question, and was the introduction and verification of a discrete information flow equivalent of the path influence algorithm, Chapter VII. Verification was achieved by constructing a discrete event simulation of the Sen Bulker Organization network using the same assumptions as path influence. The results were an exact match, demonstrating that the algorithm and other results presented in this thesis are valid. In the process, many of the metrics and ideas generated from diffusion analysis were adapted to interpret discrete information flows. Two additional design scenarios of increased realism were simulated, where requisite information was grouped at each node before proceeding to the next node. The first simulation maintained a non-stochastic and equal travel time over all arcs, while the second simulation used stochastic travel times. The path influence algorithm still compared well, reaffirming its validity but also indicating that the hierarchical and multipartite network structure was the primary driver of discrete temporal flow behavior. This was a powerful conclusion because networks and methods that lack intermediate contextual nodes will not produce similar results, further supporting the multipartite approach. It also allowed the static network metrics of previous chapters to be compared, as they also reflect structural properties. Both Park and Winston centrality showed general agreement with path influence and simulation results, strengthening the conclusion on the importance of structure.

8.2 All Contributions in Detail

1. Can the structure of design (approach, process, methods, tools and organization) be accounted for?
    - Developed the contextual multipartite network approach to represent the structure of naval design
        - Constructed the Watson & Gilfillan tripartite network
        - Constructed the Sen Bulker formulation, optimization and organization multipartite networks
    - Applied and interpreted out-degree, in-degree, Park centrality, betweenness centrality and cosine similarity for multipartite design networks
        - Computed and interpreted out-degree, in-degree, Park centrality, betweenness centrality and cosine similarity for the Watson & Gilfillan network revealing design intent
        - Computed and interpreted out-degree, in-degree, Park centrality, and betweenness centrality for the Sen Bulker networks revealing design intent
        - Introduced perturbation analysis using arc addition and Park centrality for multipartite design networks
            * Conducted perturbation analysis of the Watson & Gilfillan network
2. Can a design be understood without designing anything?
    - Recognized that algorithms for finding path lengths can be adapted to quantitatively capture all node to node influences across multipartite design networks
    - Formulated the path influence algorithm

  - Introduced the partial derivative weighting scheme
  - Introduced the interpolated derivative weighting scheme
  - Demonstrated path influence equivalency with a first order Taylor series expansion
  - Applied path influence to the Sen Bulker formulation network
- Introduced interpretations for path influence results
  - Demonstrated trends and trend magnitudes using the P matrix
  - Developed logic for potential constraint activity prediction using P matrix trends
  - Generated and verified the Sen Bulker Pareto Front and constraint activity against published results
  - Tested path influence constraint prediction accuracy
  - Conducted a full factorial DOE using six variables and two states of the Sen Bulker formulation
  - Compared path influence results using the partial derivative weighting scheme with the DOE, including magnitude deviation and trend accuracy
  - Compared path influence results using the interpolated derivative weighting scheme with the DOE, including magnitude deviation and trend accuracy
  - Determined function call count upper bounds for path influence and computed them for the Sen Bulker formulation
  - Compared actual function call counts of path influence and the DOE for the Sen Bulker problem
- Developed a new metric, Winston centrality, to enable comparison between path influence and existing centrality metrics

  - Computed Winston centrality for the Sen Bulker formulation network and both path influence weighting schemes
  - Compared Winston and Park centrality results for the Sen Bulker formulation
  - Discussed other potential applications of Winston centrality, including the identification of influence multipliers, sorters and dampers

3. Can the impact and timing of information be understood in advance?

   - Introduced network diffusion to model continuous design information flow
     - Interpreted Ficks second law of diffusion for design information flow, including the meaning of initial conditions, the diffusion constant and arc weightings
     - Developed the information level curve, first moment of lag, first moment of lead and steady state concepts for diffusion based design information flow
     - Identified and discussed the limitations of the diffusion analogy
     - Demonstrated how design bottlenecks and lag are identified using diffusion
     - Applied and interpreted the results of diffusion on Watson & Gilfillan and Sen Bulker networks
     - Modified the Watson & Gilfillan and Sen Bulker networks to show different diffusion properties
   - Introduced a discrete information flow equivalent of the path influence algorithm
     - Developed the idea of information completeness using the path influence algorithm

  - Developed two weighting schemes for computing information completeness with the path influence algorithm
  - Reinterpreted the first moment of information lag for the discrete case
  - Demonstrated information completeness computed with path influence using the Sen Bulker network and interpreted the results

- Verified the mathematics of the path influence algorithm using a discrete event simulation of the Sen Bulker network with identical assumptions
- Ran a discrete event simulation with dependent information flow of the Sen Bulker network and compared the results with those of path influence
  - Identified network hierarchy as the cause of similar results
- Ran a discrete event simulation with dependent information flow and stochastic move times of the Sen Bulker network and compared the results with those of path influence
  - Discussed the difference between time and normalized first moment results
  - Further discussed the connection between hierarchy and the similar results
- Compared Park centrality, function level, and information completeness in terms of the Sen Bulker networks hierarchy

CHAPTER IX

Future Work

The contextual multipartite network approach, and the method and metrics to analyze it described in this thesis, are available both for extension to totally new directions and further in depth analysis for those directions already identified. This chapter is divided into several sections, the first describe the author's ideas concerning expanding existing directions, and the last some ideas for new directions entirely.

9.1 Extending to Acquisition

The contextual multipartite network approach was originally envisioned as a way to comprehend the structure of naval acquisition, but that problem is of a scale beyond the scope of a single thesis. However, design is a microcosm of acquisition, and the methods introduced in this thesis apply to both problems. Acquisition program failures usually result in restructuring. Major restructuring usually comes in one of two extremes, heavy government design involvement or near autonomy for private industry (Leopold, 1975). This constantly and widely swinging pendulum makes the predictive capability of network methods all the more valuable. A future research area with massive potential impact is to extend the multipartite structure to include the social and political domains of acquisition. After all, the undeniable yet often unaccounted for fact is that politics can be the conclusive factor in program failure or success (Brown, 1993a; Work, 2013; Scovel, 1975).

9.2 Statistical Network Analysis

Most of the research described in the literature, and the direction of temporal network research, deals with the statistical properties of large networks. The networks in this thesis were necessarily limited such that their known behavior and properties could be used as validation for the multipartite approach and metrics. However, future efforts like that of Cooper et al. (2011)'s to document the ship design process are an excellent opportunity to construct large and reality representative multipartite networks for statistical analysis. Networks on such a scale could be broken into smaller clusters or components for analysis on a smaller scale if necessary. If networks of scale cannot be captured, then new research should at least seek to use different networks as test cases, providing a broader pool on on which metrics can be tested and validated.

9.3 Continuation of Chapter IV

Section 4.1 presented perturbation analysis results from the addition of single arcs to the Watson & Gilfillan network. It would be of great value to test both the addition and removal of multiple arcs, as well as nodes as further tests of structural stability. No statistical analysis was done on the Watson & Gilfillan network because of its small size, but larger networks are commonly analyzed for their degree distribution and other properties which provide an indication of their robustness. This is currently being done by affiliated researchers for networks representing physical systems, but the statistical properties of multipartite design networks remain an area of curiosity.

9.4 Continuation of Chapter V

Three different weighting schemes were developed for path influence, the first two for the Sen Bulker problem and the third a combination of the first two and information completeness. It was noted several times that the path influence algorithm is independent of the weighting scheme used on the arcs, but the weightings for the Sen Bulker problem were based on its evaluable functions. For networks that are not based on design tools, or are a mix of technical and social nodes, new weighting methods are required to quantify influence.

Due to the small size of the Sen Bulker problem, time studies of path influence vs. comparable DOE methods were not of value, though one was tried. A much larger multipartite network(s) could be used to generate real time studies, but more efficient coding of path influence and the DOE would be required for a fair comparison. In the absence of real data, it is also likely possible to derive the order of magnitude of the number of operations required for path influence, based on the number of nodes and arcs on a network. This could then be compared with other network algorithms and existing results from DOEs.

Predicting constraint activity using path influence and resolving the link between betweenness centrality and the anomalous results from Winston centrality are two harder problems to solve, though both are potentially very valuable.

During the trial runs of path influence for the Sen Bulker problem, a mistake was made in the *MATLAB* code such that the maximum power **A** was raised to was the longest geodesic path length, 7, versus the 11 required before the matrix zeros out. This error was corrected, but it was noted that both the values in the **P** matrix and the overall accuracy were negligibly affected, and in some cases the higher level function nodes were more accurate in the erroneous run. As described in Chapter VII, the influence of nodes that are 11 arcs away can be quite negligible. A great expansion of path influence would be to quantify the relationship between accuracy and the longest accounted for path length.

Both Chapters V and VII rely on multipartite networks that contain no loops. This was not a problem, as both the Watson & Gilfillan and Sen Bulker networks naturally contained no cycles. However, full scale design networks will almost be guaranteed to contain loops, and a loop capable path influence algorithm is of high value. Smith and Eppinger (1997b) provided an equation that is very similar to Eq. 7.2 for identifying the controlling features of design iteration, but they rely more on the Eigen values and Eigen vectors, like dynamical system analysis, for their insights. More general graph theory algorithms are also available (Ponstein, 1966). With minor modification these approaches may be directly applicable to multipartite design networks. However, it is the current author's hypothesis that if the magnitude of all the arc weights in a loop are less than one, or even enough of them, then eventually the values of the path influence matrix will converge. The maximum exponent required will have little or no relation to n, but will likely have a strong correlation with the length of the loop and the weight of the arcs on it, and could also be an indicator for the length of time spent iterating. A final thought on Chapter V is another metric or influence implementation, a cross between Katz centrality and path influence. Something of the form $\sum \alpha^n \mathbf{A}^n$. The idea is essentially to discount the influence of longer paths. If applied it may speed the convergence rate for networks containing loops.

9.5 Continuation of Chapter VII

Simulation of design networks, especially large ones, is a very time consuming operation. The obvious extension of Chapter VII is simulating larger networks under a wider variety of scenarios, to further explore and validate static metrics. However, the area of particular interest is the strong apparent link between the geodesic length from input to output (i.e. function level), hierarchy and the first moment of information lag. The fact that there is a link is intuitive, but it should be investigated under a wider variety of network structures. The Sen Bulker Organization network is almost perfectly hierarchical, the major exceptions occur at the nodes where the link between function level and information lag break down.

9.6 New Directions

Almost completely unexplored in the present research are concepts for partitioning networks, either in clusters, components, communities, etc. Classifying nodes as a type is necessary to build multipartite networks. There are algorithms such as that for social agony that automate this process (Gupte et al., 2011). The Sen Bulker and Watson & Gilfillan networks contain no social agony, and naturally fell into a multipartite structure. Though the structure is intuitive, larger networks will likely require some automation. Social agony is a concept that has been explored by affiliated researchers looking at physical systems, but it is a promising area for application to multipartite design networks.

Social agony may also be a network equivalent of the clustering and tearing algorithms used in DSMs. If this is the case, then the computational efficiency of one method over another should be investigated. One research field may be missing out on the improvements of another.

APPENDICES

APPENDIX A

Watson & Gilfillan

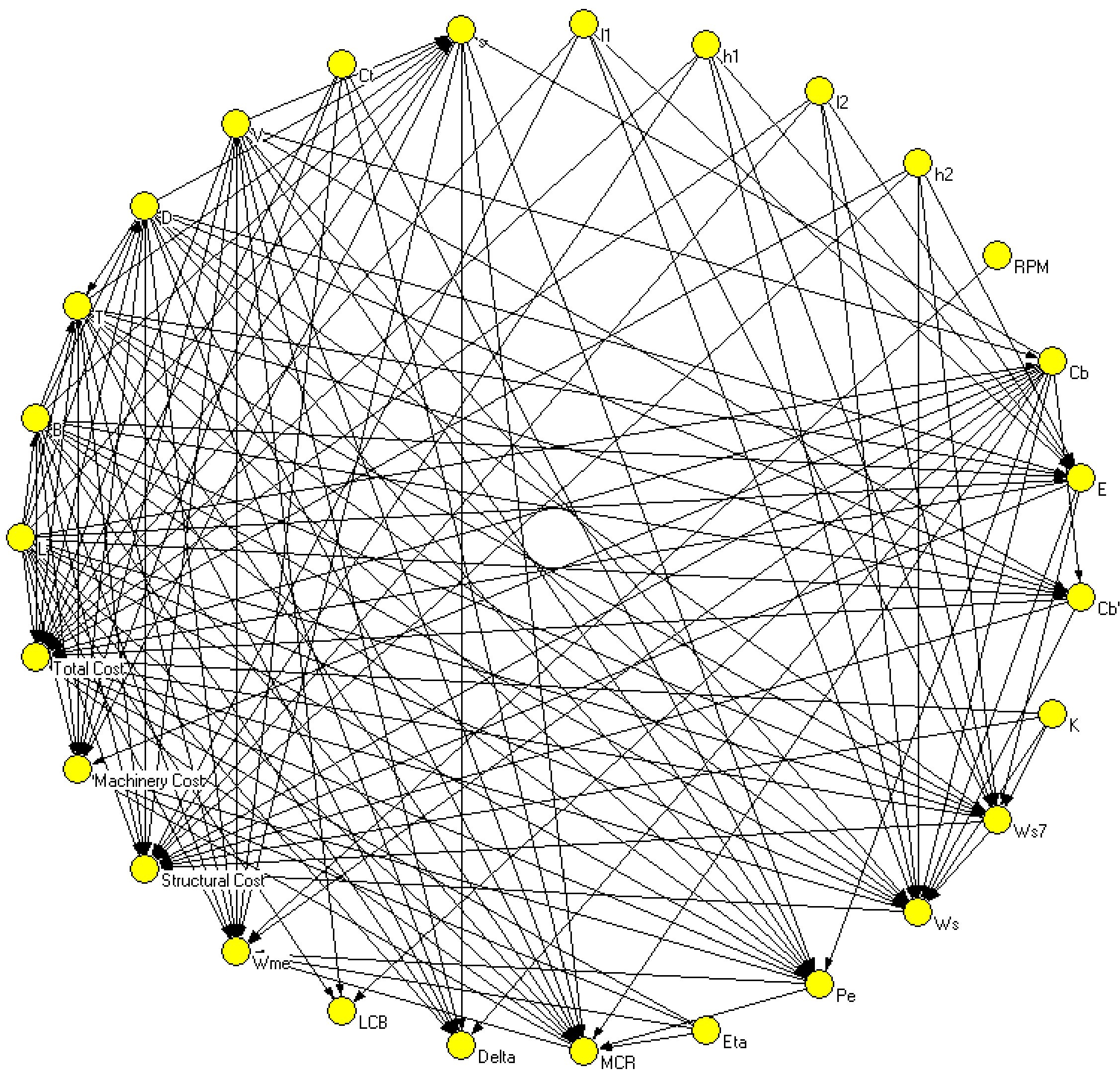


Figure A.1: Watson & Gilfillan One Mode Projection of Variables

APPENDIX B

Sen Bulker Problem

B.1 Model Definition from Sen and Yang (1998)

Table B.1: Sen Bulker Variables

Variable	Units	Symbol
Length	m	L
Draft	m	T
Depth	m	D
Block Coefficient	non-dimensional	C_b
Beam	m	B
Speed	kts	V

Table B.2: Sen Bulker Parameters

Parameter	Value	Parameter	Value
η_1	-10847.2	ζ_1	4977.06
η_2	12817	ζ_2	-8105.61
η_3	-6960.32	ζ_3	4456.51

Table B.3: Sen Bulker Constants

Parameter	Value	Units
Round Trip Miles	5000	nm
Fuel Cost	100	$£/tonne$
Cargo Handling Rate	8000	$tonnes/day$
g	9.8065	m/s^2

First Level Functions

$$F_n = \frac{V}{\sqrt{gL}} \tag{B.1}$$

$$\Delta = 1.025 \times L \times B \times T \times C_b \tag{B.2}$$

$$\text{Steel Mass} = 0.034 \times L^{1.7} \times B^{0.7} \times D^{0.4} \times C_b^{0.5} \tag{B.3}$$

$$\text{Outfit Mass} = 1.0 \times L^{0.8} \times B^{0.6} \times D^{0.3} \times C_b^{0.1} \tag{B.4}$$

$$a(C_b) = \eta_1 C_b^2 + \eta_2 C_b + \eta_3 \tag{B.5}$$

$$b(C_b) = \zeta_1 C_b^2 + \zeta_2 C_b + \zeta_3 \tag{B.6}$$

$$\text{Sea Days} = \frac{\text{Round Trip Miles}}{24 \times V} \tag{B.7}$$

$$BM = \frac{(0.85 \times C_b - 0.002) \times B^2}{T \times C_b} \tag{B.8}$$

$$KG = 1.0 + 0.52 \times D \tag{B.9}$$

$$KB = 0.53 \times T \tag{B.10}$$

Second Level Functions

$$P = \frac{\Delta^{2/3} \times V^3}{b(C_b) \times F_n + a(C_b)} \tag{B.11}$$

$$GM = KB + BM - KG \tag{B.12}$$

Third Level Functions

$$\begin{aligned}\text{Ship Cost} =& 1.3 \times (2000 \times \text{Steel Mass}^{0.85} + 3500 \\ & \times \text{Outfit Mass} + 2400 \times P^{0.8})\end{aligned} \tag{B.13}$$

$$\text{Machinery Mass} = 0.17 \times P^{0.9} \tag{B.14}$$

$$\text{Daily Consumption} = P \times 0.19 \times \frac{24}{1000} + 0.2 \tag{B.15}$$

Fourth Level Functions

$$\text{Fuel Cost} = 1.05 \times \text{Daily Consumption} \times \text{Sea Days} \times \text{Fuel Price} \quad \text{(B.16)}$$

$$\text{Light Ship Mass} = \text{Steel Mass} + \text{Machinery Mass} + \text{Outfit Mass} \quad \text{(B.17)}$$

$$\text{Capital Charges} = 0.2 \times \text{Ship Cost} \quad \text{(B.18)}$$

$$\text{Fuel Carried} = \text{Daily Consumption} \times (\text{Sea Days} + 5) \quad \text{(B.19)}$$

Fifth Level Functions

$$\Delta_{DW} = \Delta - \text{Light Ship Mass} \quad \text{(B.20)}$$

Sixth Level Functions

$$\text{Running Costs} = 40000 \times {\Delta_{DW}}^{0.3} \quad \text{(B.21)}$$

$$\text{Port Costs} = 6.3 \times {\Delta_{DW}}^{0.8} \quad \text{(B.22)}$$

$$\text{Stores\&Water} = 2.0 \times {\Delta_{DW}}^{0.5} \quad \text{(B.23)}$$

Seventh Level Functions

$$\text{Voyage Costs} = \text{Fuel Costs} + \text{Port Costs} \quad \text{(B.24)}$$

$$\Delta_{Cargo} = \Delta_{DW} - \text{Fuel Carried} - \text{Stores\&Water} \quad \text{(B.25)}$$

Eighth Level Functions

$$\text{Port Days} = 2 \times \left(\frac{\Delta_{Cargo}}{\text{Cargo Handling Rate}} + 0.5 \right) \tag{B.26}$$

Ninth Level Functions

$$RTPA = \frac{350}{\text{Sea Days} + \text{Port Days}} \tag{B.27}$$

Tenth Level Functions

$$\text{Annual Cargo} = \Delta_{Cargo} \times RTPA \tag{B.28}$$

$$\text{Annual Costs} = \text{Capital Charges} + \text{Running Costs} + \text{Voyage Costs} \times RTPA \tag{B.29}$$

Eleventh Level Functions

$$\text{Transportation Cost} = \frac{\text{Annual Costs}}{\text{Annual Cargo}} \tag{B.30}$$

Constraints

$$g_1 = 6 - L/B \tag{B.31}$$

$$g_2 = \frac{L}{D} - 15 \tag{B.32}$$

$$g_3 = \frac{L}{T} - 19 \tag{B.33}$$

$$g_4 = T - (0.45 \times \Delta_{DW}^{0.31}) \tag{B.34}$$

$$g_5 = T - (0.7 \times D + 0.7) \tag{B.35}$$

$$g_6 = 3000 - \Delta_{DW} \tag{B.36}$$

$$g_7 = \Delta_{DW} - 500000 \tag{B.37}$$

$$g_8 = 0.63 - C_b \tag{B.38}$$

$$g_9 = C_b - 0.75 \tag{B.39}$$

$$g_{10} = 14 - V \tag{B.40}$$

$$g_{11} = V - 18 \tag{B.41}$$

$$g_{12} = F_n - 0.32 \tag{B.42}$$

$$g_{13} = 0.07 \times B - GM \tag{B.43}$$

B.2 Sen Bulker Network Arc List

Table B.4: Sen Bulker Multipartite Network Arc List Part One

#	Node	Incoming Arcs
1	L	
2	T	
3	D	
4	C_b	
5	B	
6	V	
7	F_n	1, 6
8	Steel Mass	1, 5, 3, 4
9	Outfit Mass	1, 5, 3, 4
10	a	4, 41, 42, 43
11	b	4, 38, 39, 40
12	Δ	1, 5, 2, 4
13	Sea Days	6, 44
14	P	12, 6, 10, 11, 7
15	A_{co}	10, 11, 7
16	Ship Costs	8, 9, 14
17	Machinery Mass	14
18	Daily Consumption	14
19	Fuel Cost	18, 13, 45
20	Light Ship Mass	8, 9, 17
21	Fuel Carried	18, 13
22	Capital Charges	16
23	Δ_{DW}	12, 20
24	Running Costs	23
25	Port Costs	23
26	Stores&Water	23
27	Voyage Costs	19, 25
28	Δ_{Cargo}	23, 21, 26
29	Port Days	28, 46
30	RTPA	13, 29

Table B.5: Sen Bulker Multipartite Network Arc List Part Two

#	Node	Incoming Arcs
31	Annual Cargo	28, 30
32	Annual Costs	24, 27, 30, 22
33	Transportation Costs	32, 31
34	BM	4, 5, 2
35	KG	3
36	KB	2
37	GM	34, 35, 36
38	ζ_1	
39	ζ_2	
40	ζ_3	
41	η_1	
42	η_2	
43	η_3	
44	Round Trip Miles	
45	Fuel Price	
46	Cargo Handling Rate	
47	g_1	1, 5
48	g_2	1, 3
49	g_3	1, 2
50	g_4	2, 23
51	g_5	2, 3
52	g_6	23
53	g_7	23
54	g_8	4
55	g_9	4
56	g_{10}	6
57	g_{11}	6
58	g_{12}	7
59	g_{13}	5, 37

B.3 Sen Bulker Centrality Results

Table B.6: Sen Bulker Centrality Results Part One

#	Node	In-degree	Out-degree	Park	Betweenness
1	L	0	7	98.04	0.00
2	T	0	6	44.92	0.00
3	D	0	5	40.07	0.00
4	C_b	0	8	120.59	0.00
5	B	0	6	76.89	0.00
6	V	0	5	60.85	0.00
7	F_n	2	3	23.82	0.03
8	Steel Mass	4	2	15.95	0.06
9	Outfit Mass	4	2	15.95	0.06
10	a	4	2	20.82	0.31
11	b	4	2	20.82	0.31
12	Δ	4	2	38.65	0.30
13	Sea Days	2	3	11.50	0.08
14	P	5	3	10.41	1.00
15	A_{co}	3	0	-11.40	0.00
16	Ship Costs	3	1	-21.25	0.21
17	Machinery Mass	1	1	1.74	0.40
18	Daily Consumption	1	2	-5.67	0.35
19	Fuel Cost	3	1	-14.80	0.08
20	Light Ship Mass	3	1	-3.56	0.56
21	Fuel Carried	2	1	-10.07	0.22
22	Capital Charges	1	1	-19.15	0.14
23	Δ_{DW}	2	7	-2.94	0.82
24	Running Costs	1	1	-19.46	0.04
25	Port Costs	1	1	-18.75	0.05
26	Stores & Water	1	1	-15.03	0.00
27	Voyage Costs	2	1	-32.62	0.03
28	Δ_{Cargo}	3	2	-48.64	0.36
29	Port Days	2	1	-44.04	0.14
30	RTPA	2	2	-40.43	0.08

Table B.7: Sen Bulker Centrality Results Part Two

#	Node	In-degree	Out-degree	Park	Betweenness
31	Annual Cargo	2	1	-84.18	0.04
32	Annual Costs	4	1	-104.52	0.12
33	Transportation Costs	2	0	-162.19	0.00
34	BM	3	1	-1.16	0.02
35	KG	1	1	0.84	0.01
36	KB	1	1	0.84	0.00
37	GM	3	1	-6.20	0.03
38	ζ_1	0	1	21.85	0.00
39	ζ_2	0	1	21.85	0.00
40	ζ_3	0	1	21.85	0.00
41	η_1	0	1	21.85	0.00
42	η_2	0	1	21.85	0.00
43	η_3	0	1	21.85	0.00
44	Round Trip Miles	0	1	12.34	0.00
45	Fuel Price	0	1	3.14	0.00
46	Cargo Handling Rate	0	1	4.44	0.00
47	g_1	2	0	-2.00	0.00
48	g_2	2	0	-2.00	0.00
49	g_3	2	0	-2.00	0.00
50	g_4	2	0	-22.30	0.00
51	g_5	2	0	-2.00	0.00
52	g_6	1	0	-21.30	0.00
53	g_7	1	0	-21.30	0.00
54	g_8	1	0	-1.00	0.00
55	g_9	1	0	-1.00	0.00
56	g_{10}	1	0	-1.00	0.00
57	g_{11}	1	0	-1.00	0.00
58	g_{12}	1	0	-2.68	0.00
59	g_{13}	2	0	-8.05	0.00

B.4 Sen Bulker Organization Simulation Definition

```
********************************************************************************
*                                                                              *
*                          Formatted Listing of Model:                         *
* C:\Users\mcparker\Documents\Research\PhD\Network Visualizations\ProModel\Simulation Cases\SenBulkerS
*                                                                              *
********************************************************************************

  Time Units:                        Minutes
  Distance Units:                    Meters

********************************************************************************
*                                  Locations                                   *
********************************************************************************

  Name       Cap Units Stats       Rules      Cost
  ---------- --- ----- ----------- ---------- ------------
  Node_1     INF 1     Time Series Oldest, ,
  Node_2     INF 1     Time Series Oldest, ,
  Node_3     INF 1     Time Series Oldest, ,
  Node_4     INF 1     Time Series Oldest, ,
  Node_5     INF 1     Time Series Oldest, ,
  Node_6     INF 1     Time Series Oldest, ,
  Node_7     INF 1     Time Series Oldest, ,
  Node_8     INF 1     Time Series Oldest, ,
  Node_9     INF 1     Time Series Oldest, ,
  Node_10    INF 1     Time Series Oldest, ,
  Node_11    INF 1     Time Series Oldest, ,
  Node_12    INF 1     Time Series Oldest, ,
  Node_13    INF 1     Time Series Oldest, ,
  Node_14    INF 1     Time Series Oldest, ,
  Node_16    INF 1     Time Series Oldest, ,
  Node_17    INF 1     Time Series Oldest, ,
  Node_18    INF 1     Time Series Oldest, ,
  Node_19    INF 1     Time Series Oldest, ,
  Node_20    INF 1     Time Series Oldest, ,
  Node_21    INF 1     Time Series Oldest, ,
  Node_22    INF 1     Time Series Oldest, ,
  Node_23    INF 1     Time Series Oldest, ,
  Node_24    INF 1     Time Series Oldest, ,
  Node_25    INF 1     Time Series Oldest, ,
  Node_26    INF 1     Time Series Oldest, ,
  Node_27    INF 1     Time Series Oldest, ,
  Node_28    INF 1     Time Series Oldest, ,
  Node_29    INF 1     Time Series Oldest, ,
  Node_30    INF 1     Time Series Oldest, ,
  Node_31    INF 1     Time Series Oldest, ,
  Node_32    INF 1     Time Series Oldest, ,
  Node_33    INF 1     Time Series Oldest, ,
  Node_38    INF 1     Time Series Oldest, ,
  Node_39    INF 1     Time Series Oldest, ,
  Node_40    INF 1     Time Series Oldest, ,
  Node_41    INF 1     Time Series Oldest, ,
  Node_42    INF 1     Time Series Oldest, ,
  Node_43    INF 1     Time Series Oldest, ,
  Node_44    INF 1     Time Series Oldest, ,
  Node_45    INF 1     Time Series Oldest, ,
  Node_46    INF 1     Time Series Oldest, ,
  Arc_0107   INF 1     Time Series Oldest, ,
  Arc_0108   INF 1     Time Series Oldest, ,
  Arc_0109   INF 1     Time Series Oldest, ,
  Arc_0112   INF 1     Time Series Oldest, ,
  Arc_0212   INF 1     Time Series Oldest, ,
  Arc_0308   INF 1     Time Series Oldest, ,
  Arc_0309   INF 1     Time Series Oldest, ,
  Arc_0408   INF 1     Time Series Oldest, ,
  Arc_0409   INF 1     Time Series Oldest, ,
  Arc_0410   INF 1     Time Series Oldest, ,
  Arc_0411   INF 1     Time Series Oldest, ,
  Arc_0412   INF 1     Time Series Oldest, ,
  Arc_0508   INF 1     Time Series Oldest, ,
  Arc_0509   INF 1     Time Series Oldest, ,
  Arc_0512   INf 1     Time Series Oldest, ,
  Arc_0607   INF 1     Time Series Oldest, ,
  Arc_0613   INF 1     Time Series Oldest, ,
  Arc_0614   INF 1     Time Series Oldest, ,
  Arc_0714   INF 1     Time Series Oldest, ,
  Arc_0816   INF 1     Time Series Oldest, ,
```

```
  Arc_0820   INF 1     Time Series Oldest, ,
  Arc_0916   INF 1     Time Series Oldest, ,
  Arc_0920   INF 1     Time Series Oldest, ,
  Arc_1014   INF 1     Time Series Oldest, ,
  Arc_1114   INF 1     Time Series Oldest, ,
  Arc_1214   INF 1     Time Series Oldest, ,
  Arc_1223   INF 1     Time Series Oldest, ,
  Arc_1319   INF 1     Time Series Oldest, ,
  Arc_1321   INF 1     Time Series Oldest, ,
  Arc_1330   INF 1     Time Series Oldest, ,
  Arc_1416   INF 1     Time Series Oldest, ,
  Arc_1417   INF 1     Time Series Oldest, ,
  Arc_1418   INF 1     Time Series Oldest, ,
  Arc_1622   INF 1     Time Series Oldest, ,
  Arc_1720   INF 1     Time Series Oldest, ,
  Arc_1819   INF 1     Time Series Oldest, ,
  Arc_1821   INF 1     Time Series Oldest, ,
  Arc_1927   INF 1     Time Series Oldest, ,
  Arc_2023   INF 1     Time Series Oldest, ,
  Arc_2128   INF 1     Time Series Oldest, ,
  Arc_2232   INF 1     Time Series Oldest, ,
  Arc_2324   INF 1     Time Series Oldest, ,
  Arc_2325   INF 1     Time Series Oldest, ,
  Arc_2326   INF 1     Time Series Oldest, ,
  Arc_2328   INF 1     Time Series Oldest, ,
  Arc_2432   INF 1     Time Series Oldest, ,
  Arc_2527   INF 1     Time Series Oldest, ,
  Arc_2628   INF 1     Time Series Oldest, ,
  Arc_2732   INF 1     Time Series Oldest, ,
  Arc_2829   INF 1     Time Series Oldest, ,
  Arc_2831   INF 1     Time Series Oldest, ,
  Arc_2930   INF 1     Time Series Oldest, ,
  Arc_3031   INF 1     Time Series Oldest, ,
  Arc_3032   INF 1     Time Series Oldest, ,
  Arc_3133   INF 1     Time Series Oldest, ,
  Arc_3233   INF 1     Time Series Oldest, ,
  Arc_3811   INF 1     Time Series Oldest, ,
  Arc_3911   INF 1     Time Series Oldest, ,
  Arc_4011   INF 1     Time Series Oldest, ,
  Arc_4110   INF 1     Time Series Oldest, ,
  Arc_4210   INF 1     Time Series Oldest, ,
  Arc_4310   INF 1     Time Series Oldest, ,
  Arc_4413   INF 1     Time Series Oldest, ,
  Arc_4519   INF 1     Time Series Oldest, ,
  Arc_4629   INF 1     Time Series Oldest, ,


********************************************************************************
*                                    Entities                                  *
********************************************************************************

  Name       Speed (mpm)  Stats       Cost
  ---------- ------------ ----------- -----------
  Length     50           Time Series
  Draft      50           Time Series
  Depth      50           Time Series
  Cb         50           Time Series
  Beam       50           Time Series
  V          50           Time Series
  Zeta_1     50           Time Series
  Zeta_2     50           Time Series
  Zeta_3     50           Time Series
  Eta_1      50           Time Series
  Eta_2      50           Time Series
  Eta_3      50           Time Series
  RTM        50           Time Series
  FP         50           Time Series
  CHR        50           Time Series
  F          50           Time Series
  F7         50           Time Series
  F8         50           Time Series
  F9         50           Time Series
  F10        50           Time Series
  F11        50           Time Series
  F12        50           Time Series
  F13        50           Time Series
  F14        50           Time Series
  F16        50           Time Series
```

```
 F17        50          Time Series
 F18        50          Time Series
 F19        50          Time Series
 F20        50          Time Series
 F21        50          Time Series
 F22        50          Time Series
 F23        50          Time Series
 F24        50          Time Series
 F25        50          Time Series
 F26        50          Time Series
 F27        50          Time Series
 F28        50          Time Series
 F29        50          Time Series
 F30        50          Time Series
 F31        50          Time Series
 F32        50          Time Series
 F33        50          Time Series


********************************************************************************
*                                  Processing                                  *
********************************************************************************

                             Process                    Routing

 Entity   Location Operation            Blk  Output   Destination Rule     Move Logic
 -------- -------- -----------------    ---- -------- ----------- -------  ------------
 Length   Node_1   node_1_ic=ic
                   ic=ic*NodeWeight[1]*M1
                   WAIT Vwait


                   node_1_mt=N(mt[1,1],mt[1,2])
                                        1    Length   Arc_0107    FIRST 1
                                        2*   Length   Arc_0108    FIRST 1
                                        3*   Length   Arc_0109    FIRST 1
                                        4*   Length   Arc_0112    FIRST 1
 Draft    Node_2   node_2_ic=ic
                   ic=ic*NodeWeight[2]*M2
                   WAIT Vwait           1    Draft    Arc_0212    FIRST 1
 Depth    Node_3   node_3_ic=ic
                   ic=ic*NodeWeight[3]*M3
                   WAIT Vwait           1    Depth    Arc_0308    FIRST 1
                                        2*   Depth    Arc_0309    FIRST 1
 Cb       Node_4   node_4_ic=ic
                   ic=ic*NodeWeight[4]*M4
                   WAIT Vwait           1    Cb       Arc_0408    FIRST 1
                                        2*   Cb       Arc_0409    FIRST 1
                                        3*   Cb       Arc_0410    FIRST 1
                                        4*   Cb       Arc_0411    FIRST 1
                                        5*   Cb       Arc_0412    FIRST 1
 Beam     Node_5   node_5_ic=ic
                   ic=ic*NodeWeight[5]*M5
                   WAIT Vwait           1    Beam     Arc_0508    FIRST 1
                                        2*   Beam     Arc_0509    FIRST 1
                                        3*   Beam     Arc_0512    FIRST 1
 V        Node_6   node_6_ic=ic
                   ic=ic*NodeWeight[6]*M6
                   WAIT Vwait           1    V        Arc_0607    FIRST 1
                                        2*   V        Arc_0613    FIRST 1
                                        3*   V        Arc_0614    FIRST 1
 Zeta_1   Node_38  node_38_ic=ic
                   Wait Vwait
                   ic=ic*NodeWeight[38]*M38
                                        1    Zeta_1   Arc_3811    FIRST 1
 Zeta_2   Node_39  node_39_ic=ic
                   Wait Vwait
                   ic=ic*NodeWeight[39]*M39
                                        1    Zeta_2   Arc_3911    FIRST 1
 Zeta_3   Node_40  node_40_ic=ic
                   Wait Vwait
                   ic=ic*NodeWeight[40]*M40
                                        1    Zeta_3   Arc_4011    FIRST 1
 Eta_1    Node_41  node_41_ic=ic
                   Wait Vwait
                   ic=ic*NodeWeight[41]*M41
                                        1    Eta_1    Arc_4110    FIRST 1
 Eta_2    Node_42  node_42_ic=ic
```

```
                  Wait Vwait
                  ic=ic*NodeWeight[42]*M42
                                          1    Eta_2     Arc_4210    FIRST 1
Eta_3    Node_43  node_43_ic=ic
                  Wait Vwait
                  ic=ic*NodeWeight[43]*M43
                                          1    Eta_3     Arc_4310    FIRST 1
RTM      Node_44  node_44_ic=ic
                  Wait Vwait
                  ic=ic*NodeWeight[44]*M44
                                          1    RTM       Arc_4413    FIRST 1
FP       Node_45  node_45_ic=ic
                  Wait Vwait
                  ic=ic*NodeWeight[45]*M45
                                          1    FP        Arc_4519    FIRST 1
CHR      Node_46  node_46_ic=ic
                  Wait Vwait
                  ic=ic*NodeWeight[46]*M46
                                          1    CHR       Arc_4629    FIRST 1
Length   Arc_0107 ic=ic*A[1,7]            1    Length    Node_7      FIRST 1  INC node_7_ic, ic
Length   Arc_0108 ic=ic*A[1,8]            1    Length    Node_8      FIRST 1  INC node_8_ic, ic
Length   Arc_0109 ic=ic*A[1,9]            1    Length    Node_9      FIRST 1  INC node_9_ic, ic
Length   Arc_0112 ic=ic*A[1,12]           1    Length    Node_12     FIRST 1  INC node_12_ic, ic
Draft    Arc_0212 ic=ic*A[2,12]           1    Draft     Node_12     FIRST 1  INC node_12_ic, ic
Depth    Arc_0308 ic=ic*A[3,8]            1    Depth     Node_8      FIRST 1  INC node_8_ic, ic
Depth    Arc_0309 ic=ic*A[3,9]            1    Depth     Node_9      FIRST 1  INC node_9_ic, ic
Cb       Arc_0408 ic=ic*A[4,8]            1    Cb        Node_8      FIRST 1  INC node_8_ic, ic
Cb       Arc_0409 ic=ic*A[4,9]            1    Cb        Node_9      FIRST 1  INC node_9_ic, ic
Cb       Arc_0410 ic=ic*A[4,10]           1    Cb        Node_10     FIRST 1  INC node_10_ic, ic
Cb       Arc_0411 ic=ic*A[4,11]           1    Cb        Node_11     FIRST 1  INC node_11_ic, ic
Cb       Arc_0412 ic=ic*A[4,12]           1    Cb        Node_12     FIRST 1  INC node_12_ic, ic
Beam     Arc_0508 ic=ic*A[5,8]            1    Beam      Node_8      FIRST 1  INC node_8_ic, ic
Beam     Arc_0509 ic=ic*A[5,9]            1    Beam      Node_9      FIRST 1  INC node_9_ic, ic
Beam     Arc_0512 ic=ic*A[5,12]           1    Beam      Node_12     FIRST 1  INC node_12_ic, ic
V        Arc_0607 ic=ic*A[6,7]            1    V         Node_7      FIRST 1  INC node_7_ic, ic
V        Arc_0613 ic=ic*A[6,13]           1    V         Node_13     FIRST 1  INC node_13_ic, ic
V        Arc_0614 ic=ic*A[6,14]           1    V         Node_14     FIRST 1  INC node_14_ic, ic
Zeta_1   Arc_3811 ic=ic*A[38,11]          1    Zeta_1    Node_11     FIRST 1  INC node_11_ic, ic
Zeta_2   Arc_3911 ic=ic*A[39,11]          1    Zeta_2    Node_11     FIRST 1  INC node_11_ic, ic
Zeta_3   Arc_4011 ic=ic*A[40,11]          1    Zeta_3    Node_11     FIRST 1  INC node_11_ic, ic
Eta_1    Arc_4110 ic=ic*A[41,10]          1    Eta_1     Node_10     FIRST 1  INC node_10_ic, ic
Eta_2    Arc_4210 ic=ic*A[42,10]          1    Eta_2     Node_10     FIRST 1  INC node_10_ic, ic
Eta_3    Arc_4310 ic=ic*A[43,10]          1    Eta_3     Node_10     FIRST 1  INC node_10_ic, ic
RTM      Arc_4413 ic=ic*A[44,13]          1    RTM       Node_13     FIRST 1  INC node_13_ic, ic
FP       Arc_4519 ic=ic*A[45,19]          1    FP        Node_19     FIRST 1  INC node_19_ic, ic
CHR      Arc_4629 ic=ic*A[46,29]          1    CHR       Node_29     FIRST 1  INC node_29_ic, ic
F7       Arc_0714 ic=ic*A[7,14]           1    F7        Node_14     FIRST 1  INC node_14_ic, ic
F8       Arc_0816 ic=ic*A[8,16]           1    F8        Node_16     FIRST 1  INC node_16_ic, ic
F8       Arc_0820 ic=ic*A[8,20]           1    F8        Node_20     FIRST 1  INC node_20_ic, ic
F9       Arc_0916 ic=ic*A[9,16]
                                          1    F9        Node_16     FIRST 1  INC node_16_ic, ic
F9       Arc_0920 ic=ic*A[9,20]           1    F9        Node_20     FIRST 1  INC node_20_ic, ic
F10      Arc_1014 ic=ic*A[10,14]          1    F10       Node_14     FIRST 1  INC node_14_ic, ic
F11      Arc_1114 ic=ic*A[11,14]          1    F11       Node_14     FIRST 1  INC node_14_ic, ic
F12      Arc_1214 ic=ic*A[12,14]          1    F12       Node_14     FIRST 1  INC node_14_ic, ic
F12      Arc_1223 ic=ic*A[12,23]
                                          1    F12       Node_23     FIRST 1  INC node_23_ic, ic
F13      Arc_1319 ic=ic*A[13,19]          1    F13       Node_19     FIRST 1  INC node_19_ic, ic
F13      Arc_1321 ic=ic*A[13,21]          1    F13       Node_21     FIRST 1  INC node_21_ic, ic
F13      Arc_1330 ic=ic*A[13,30]          1    F13       Node_30     FIRST 1  INC node_30_ic, ic
F14      Arc_1416 ic=ic*A[14,16]          1              Node_16     FIRST 1  INC node_16_ic, ic
F14      Arc_1417 ic=ic*A[14,17]          1              Node_17     FIRST 1  INC node_17_ic, ic
F14      Arc_1418 ic=ic*A[14,18]          1              Node_18     FIRST 1  INC node_18_ic, ic
F16      Arc_1622 ic=ic*A[16,22]          1              Node_22     FIRST 1  INC node_22_ic, ic
F17      Arc_1720 ic=ic*A[17,20]          1              Node_20     FIRST 1  INC node_20_ic, ic
F18      Arc_1819 ic=ic*A[18,19]          1              Node_19     FIRST 1  INC node_19_ic, ic
F18      Arc_1821 ic=ic*A[18,21]          1              Node_21     FIRST 1  INC node_21_ic, ic
F19      Arc_1927 ic=ic*A[19,27]          1              Node_27     FIRST 1  INC node_27_ic, ic
F20      Arc_2023 ic=ic*A[20,23]          1              Node_23     FIRST 1  INC node_23_ic, ic
F21      Arc_2128 ic=ic*A[21,28]          1              Node_28     FIRST 1  INC node_28_ic,ic
F22      Arc_2232 ic=ic*A[22,32]          1              Node_32     FIRST 1  INC node_32_ic, ic
F23      Arc_2324 ic=ic*A[23,24]          1              Node_24     FIRST 1  INC node_24_ic, ic
F23      Arc_2325 ic=ic*A[23,25]          1              Node_25     FIRST 1  INC node_25_ic, ic
F23      Arc_2326 ic=ic*A[23,26]          1              Node_26     FIRST 1  INC node_26_ic, ic
F23      Arc_2328 ic=ic*A[23,28]          1              Node_28     FIRST 1  INC node_28_ic, ic
F24      Arc_2432 ic=ic*A[24,32]          1              Node_32     FIRST 1  INC node_32_ic, ic
F25      Arc_2527 ic=ic*A[25,27]          1              Node_27     FIRST 1  INC node_27_ic, ic
F26      Arc_2628 ic=ic*A[26,28]          1              Node_28     FIRST 1  INC node_28_ic, ic
```

```
F27      Arc_2732 ic=ic*A[27,32]     1          Node_32     FIRST 1  INC node_32_ic, ic
F28      Arc_2829 ic=ic*A[28,29]     1          Node_29     FIRST 1  INC node_29_ic, ic
F28      Arc_2831 ic=ic*A[28,31]     1          Node_31     FIRST 1  INC node_31_ic, ic
F29      Arc_2930 ic=ic*A[29,30]     1          Node_30     FIRST 1  INC node_30_ic, ic
F30      Arc_3031 ic=ic*A[30,31]     1          Node_31     FIRST 1  INC node_31_ic, ic
F30      Arc_3032 ic=ic*A[30,32]     1          Node_32     FIRST 1  INC node_32_ic, ic
F31      Arc_3133 ic=ic*A[31,33]     1          Node_33     FIRST 1  INC node_33_ic, ic
F32      Arc_3233 ic=ic*A[32,33]     1          Node_33     FIRST 1  INC node_33_ic, ic
Length   Node_7   WAIT FLw[1]

                  ic=ic*NodeWeight[7]*M7
                                     1    F7    Arc_0714    FIRST 1
V        Node_7   WAIT FLw[1]



                  ic=ic*NodeWeight[7]*M7
                                     1    F7    Arc_0714    FIRST 1
Length   Node_8   WAIT FLw[1]



                  ic=ic*NodeWeight[8]*M8
                                     1    F8    Arc_0816    FIRST 1
                                     2*   F8    Arc_0820    FIRST 1
Depth    Node_8   WAIT FLw[1]


                  ic=ic*NodeWeight[8]*M8
                                     1    F8    Arc_0816    FIRST 1
                                     2*   F8    Arc_0820    FIRST 1
Cb       Node_8   WAIT FLw[1]



                  ic=ic*NodeWeight[8]*M8
                                     1    F8    Arc_0816    FIRST 1
                                     2*   F8    Arc_0820    FIRST 1
Beam     Node_8   WAIT FLw[1]



                  ic=ic*NodeWeight[8]*M8
                                     1    F8    Arc_0816    FIRST 1
                                     2*   F8    Arc_0820    FIRST 1
Length   Node_9   WAIT FLw[1]

                  ic=ic*NodeWeight[9]*M9


                                     1    F9    Arc_0916    FIRST 1
                                     2*   F9    Arc_0920    FIRST 1
Depth    Node_9   WAIT FLw[1]


                  ic=ic*NodeWeight[9]*M9


                                     1    F9    Arc_0916    FIRST 1
                                     2*   F9    Arc_0920    FIRST 1
Cb       Node_9   WAIT FLw[1]



                  ic=ic*NodeWeight[9]*M9


                                     1    F9    Arc_0916    FIRST 1
                                     2*   F9    Arc_0920    FIRST 1
Beam     Node_9   WAIT FLw[1]

                  ic=ic*NodeWeight[9]*M9
```

```
                                          1    F9        Arc_0916    FIRST 1
                                          2*   F9        Arc_0920    FIRST 1
Cb        Node_10  WAIT FLw[1]


                   ic=ic*NodeWeight[10]*M10
                                          1    F10       Arc_1014    FIRST 1
Eta_1     Node_10  WAIT FLw[1]


                   ic=ic*NodeWeight[10]*M10
                                          1    F10       Arc_1014    FIRST 1
Eta_2     Node_10  WAIT FLw[1]

                   ic=ic*NodeWeight[10]*M10
                                          1    F10       Arc_1014    FIRST 1
Eta_3     Node_10  WAIT FLw[1]


                   ic=ic*NodeWeight[10]*M10
                                          1    F10       Arc_1014    FIRST 1
Cb        Node_11  WAIT FLw[1]


                   ic=ic*NodeWeight[11]*M11
                                          1    F11       Arc_1114    FIRST 1
Zeta_1    Node_11  WAIT FLw[1]


                   ic=ic*NodeWeight[11]*M11
                                          1    F11       Arc_1114    FIRST 1
Zeta_2    Node_11  WAIT FLw[1]

                   ic=ic*NodeWeight[11]*M11
                                          1    F11       Arc_1114    FIRST 1
Zeta_3    Node_11  WAIT FLw[1]

                   ic=ic*NodeWeight[11]*M11
                                          1    F11       Arc_1114    FIRST 1
Length    Node_12  WAIT FLw[1]

                   ic=ic*NodeWeight[12]*M12
                                          1    F12       Arc_1214    FIRST 1
                                          2*   F12       Arc_1223    FIRST 1
Draft     Node_12  WAIT FLw[1]

                   ic=ic*NodeWeight[12]*M12
                                          1    F12       Arc_1214    FIRST 1
                                          2*   F12       Arc_1223    FIRST 1
Cb        Node_12  WAIT FLw[1]

                   ic=ic*NodeWeight[12]*M12
                                          1    F12       Arc_1214    FIRST 1
                                          2*   F12       Arc_1223    FIRST 1
Beam      Node_12  WAIT FLw[1]


                   ic=ic*NodeWeight[12]*M12
                                          1    F12       Arc_1214    FIRST 1
                                          2*   F12       Arc_1223    FIRST 1
V         Node_13  WAIT FLw[1]
                   ic=ic*NodeWeight[13]*M13
                                          1    F13       Arc_1319    FIRST 1
                                          2*   F13       Arc_1321    FIRST 1
                                          3*   F13       Arc_1330    FIRST 1
RTM       Node_13  WAIT FLw[1]

                   ic=ic*NodeWeight[13]*M13
```

```
                                   1    F13     Arc_1319    FIRST 1
                                   2*   F13     Arc_1321    FIRST 1
                                   3*   F13     Arc_1330    FIRST 1
V       Node_14  WAIT FLw[2]


         ic=ic*NodeWeight[14]*M14
                                   1    F14     Arc_1416    FIRST 1
                                   2*   F14     Arc_1417    FIRST 1
                                   3*   F14     Arc_1418    FIRST 1
F7      Node_14  WAIT FLw[2]


         ic=ic*NodeWeight[14]*M14
                                   1    F14     Arc_1416    FIRST 1
                                   2*   F14     Arc_1417    FIRST 1
                                   3*   F14     Arc_1418    FIRST 1
F10     Node_14  WAIT FLw[2]

         ic=ic*NodeWeight[14]*M14
                                   1    F14     Arc_1416    FIRST 1
                                   2*   F14     Arc_1417    FIRST 1
                                   3*   F14     Arc_1418    FIRST 1
F11     Node_14  WAIT FLw[2]

         ic=ic*NodeWeight[14]*M14
                                   1    F14     Arc_1416    FIRST 1
                                   2*   F14     Arc_1417    FIRST 1
                                   3*   F14     Arc_1418    FIRST 1
F12     Node_14  WAIT FLw[2]


         ic=ic*NodeWeight[14]*M14
                                   1    F14     Arc_1416    FIRST 1
                                   2*   F14     Arc_1417    FIRST 1
                                   3*   F14     Arc_1418    FIRST 1
F8      Node_16  WAIT FLw[3]


         ic=ic*NodeWeight[16]*M16
                                   1    F16     Arc_1622    FIRST 1
F9      Node_16  WAIT FLw[3]


         ic=ic*NodeWeight[16]*M16
                                   1    F16     Arc_1622    FIRST 1
F14     Node_16  WAIT FLw[3]


         ic=ic*NodeWeight[16]*M16
                                   1    F16     Arc_1622    FIRST 1
F14     Node_17  WAIT FLw[3]

         ic=ic*NodeWeight[17]*M17
                                   1    F17     Arc_1720    FIRST 1
F14     Node_18  WAIT FLw[3]


         ic=ic*NodeWeight[18]*M18
                                   1    F18     Arc_1819    FIRST 1
                                   2*   F18     Arc_1821    FIRST 1
FP      Node_19  WAIT FLw[4]


         ic=ic*NodeWeight[19]*M19
                                   1    F19     Arc_1927    FIRST 1
F13     Node_19  WAIT FLw[4]

         ic=ic*NodeWeight[19]*M19
                                   1    F19     Arc_1927    FIRST 1
F18     Node_19  WAIT FLw[4]
```

```
                    ic=ic*NodeWeight[19]*M19
                                             1    F19      Arc_1927    FIRST 1
F8        Node_20   WAIT FLw[4]


                    ic=ic*NodeWeight[20]*M20
                                             1    F20      Arc_2023    FIRST 1
F9        Node_20   WAIT FLw[4]


                    ic=ic*NodeWeight[20]*M20
                                             1    F20      Arc_2023    FIRST 1
F17       Node_20   WAIT FLw[4]


                    ic=ic*NodeWeight[20]*M20
                                             1    F20      Arc_2023    FIRST 1
F13       Node_21   WAIT FLw[4]


                    ic=ic*NodeWeight[21]*M21
                                             1    F21      Arc_2128    FIRST 1
F18       Node_21   WAIT FLw[4]

                    ic=ic*NodeWeight[21]*M21
                                             1    F21      Arc_2128    FIRST 1
F16       Node_22   WAIT FLw[4]

                    ic=ic*NodeWeight[22]*M22
                                             1    F22      Arc_2232    FIRST 1
F12       Node_23   WAIT FLw[5]

                    ic=ic*NodeWeight[23]*M23
                                             1    F23      Arc_2324    FIRST 1
                                             2*   F23      Arc_2325    FIRST 1
                                             3*   F23      Arc_2326    FIRST 1
                                             4*   F23      Arc_2328    FIRST 1
F20       Node_23   WAIT FLw[5]

                    ic=ic*NodeWeight[23]*M23
                                             1    F23      Arc_2324    FIRST 1
                                             2*   F23      Arc_2325    FIRST 1
                                             3*   F23      Arc_2326    FIRST 1
                                             4*   F23      Arc_2328    FIRST 1
F23       Node_24   WAIT FLw[6]


                    ic=ic*NodeWeight[24]*M24
                                             1    F24      Arc_2432    FIRST 1
F23       Node_25   WAIT FLw[6]


                    ic=ic*NodeWeight[25]*M25
                                             1    F25      Arc_2527    FIRST 1
F23       Node_26   WAIT FLw[6]


                    ic=ic*NodeWeight[26]*M26
                                             1    F26      Arc_2628    FIRST 1
F19       Node_27   WAIT FLw[7]


                    ic=ic*NodeWeight[27]*M27
                                             1    F27      Arc_2732    FIRST 1
F25       Node_27   WAIT FLw[7]
```

```
                  ic=ic*NodeWeight[27]*M27
                                          1    F27      Arc_2732    FIRST 1
F21       Node_28  WAIT FLw[7]


                  ic=ic*NodeWeight[28]*M28
                                          1    F28      Arc_2829    FIRST 1
                                          2*   F28      Arc_2831    FIRST 1
F23       Node_28  WAIT FLw[7]

                  ic=ic*NodeWeight[28]*M28
                                          1    F28      Arc_2829    FIRST 1
                                          2*   F28      Arc_2831    FIRST 1
F26       Node_28  WAIT FLw[7]


                  ic=ic*NodeWeight[28]*M28
                                          1    F28      Arc_2829    FIRST 1
                                          2*   F28      Arc_2831    FIRST 1
F28       Node_29  WAIT FLw[8]


                  ic=ic*NodeWeight[29]*M29
                                          1    F29      Arc_2930    FIRST 1
CHR       Node_29  WAIT FLw[8]


                  ic=ic*NodeWeight[29]*M29
                                          1    F29      Arc_2930    FIRST 1
F13       Node_30  WAIT FLw[9]


                  ic=ic*NodeWeight[30]*M30
                                          1    F30      Arc_3031    FIRST 1
                                          2*   F30      Arc_3032    FIRST 1
F29       Node_30  WAIT FLw[9]


                  ic=ic*NodeWeight[30]*M30
                                          1    F30      Arc_3031    FIRST 1
                                          2*   F30      Arc_3032    FIRST 1
F28       Node_31  WAIT FLw[10]


                  ic=ic*NodeWeight[31]*M31
                                          1    F31      Arc_3133    FIRST 1
F30       Node_31  WAIT FLw[10]

                  ic=ic*NodeWeight[31]*M31
                                          1    F31      Arc_3133    FIRST 1
F22       Node_32  WAIT FLw[10]

                  ic=ic*NodeWeight[32]*M32
                                          1    F32      Arc_3233    FIRST 1
F24       Node_32  WAIT FLw[10]

                  ic=ic*NodeWeight[32]*M32
                                          1    F32      Arc_3233    FIRST 1
F27       Node_32  WAIT FLw[10]

                  ic=ic*NodeWeight[32]*M32
                                          1    F32      Arc_3233    FIRST 1
F30       Node_32  WAIT FLw[10]
```

```
                   ic=ic*NodeWeight[32]*M32
                                     1    F32      Arc_3233    FIRST 1
 F31      Node_33  WAIT FLw[11]


                   ic=ic*NodeWeight[33]*M33
                                     1    F33      EXIT        FIRST 1
 F32      Node_33  WAIT FLw[11]



                   ic=ic*NodeWeight[33]*M33
                                     1    F33      EXIT        FIRST 1


********************************************************************************
*                                    Arrivals                                  *
********************************************************************************

  Entity   Location Qty Each   First Time Occurrences Frequency  Logic
  -------- -------- ---------- ---------- ----------- ---------- ------------
  Length   Node_1   1                     1                      ic=1
  Draft    Node_2   01                    1                      ic=1
  Depth    Node_3   1                     1                      ic=1
  Cb       Node_4   01                    1                      ic=1
  Beam     Node_5   01                    1                      ic=1
  V        Node_6   01                    1                      ic=1
  Zeta_1   Node_38  1          0          1                      ic=1
  Zeta_2   Node_39  01         0          1                      ic=1
  Zeta_3   Node_40  1          0          1                      ic=1
  Eta_1    Node_41  01         0          1                      ic=1
  Eta_2    Node_42  01         0          1                      ic=1
  Eta_3    Node_43  01         0          1                      ic=1
  RTM      Node_44  01         0          1                      ic=1
  FP       Node_45  01         0          1                      ic=1
  CHR      Node_46  01         0          1                      ic=1


********************************************************************************
*                                   Attributes                                 *
********************************************************************************

  ID         Type         Classification
  ---------- ------------ --------------
  ic         Real         Entity


********************************************************************************
*                              Variables (global)                              *
********************************************************************************

  ID         Type         Initial value Stats
  ---------- ------------ ------------- -----------
  node_1_ic  Real         0             Time Series
  node_1_mt  Real         0             Time Series
  node_2_ic  Real         0             Time Series
  node_3_ic  Real         0             Time Series
  node_4_ic  Real         0             Time Series
  node_5_ic  Real         0             Time Series
  node_6_ic  Real         0             Time Series
  node_7_ic  Real         0             Time Series
  node_8_ic  Real         0             Time Series
  node_9_ic  Real         0             Time Series
  node_10_ic Real         0             Time Series
  node_11_ic Real         0             Time Series
  node_12_ic Real         0             Time Series
  node_13_ic Real         0             Time Series
  node_14_ic Real         0             Time Series
  node_16_ic Real         0             Time Series
  node_17_ic Real         0             Time Series
  node_18_ic Real         0             Time Series
  node_19_ic Real         0             Time Series
  node_20_ic Real         0             Time Series
  node_21_ic Real         0             Time Series
  node_22_ic Real         0             Time Series
  node_23_ic Real         0             Time Series
  node_24_ic Real         0             Time Series
```

node_25_ic	Real	0	Time Series
node_26_ic	Real	0	Time Series
node_27_ic	Real	0	Time Series
node_28_ic	Real	0	Time Series
node_29_ic	Real	0	Time Series
node_30_ic	Real	0	Time Series
node_31_ic	Real	0	Time Series
node_32_ic	Real	0	Time Series
node_33_ic	Real	0	Time Series
node_38_ic	Real	0	Time Series
node_39_ic	Real	0	Time Series
node_40_ic	Real	0	Time Series
node_41_ic	Real	0	Time Series
node_42_ic	Real	0	Time Series
node_43_ic	Real	0	Time Series
node_44_ic	Real	0	Time Series
node_45_ic	Real	0	Time Series
node_46_ic	Real	0	Time Series
Vwait	Real	1	Time Series

```
*******************************************************************************
*                                    Arrays                                   *
*******************************************************************************
```

ID	Dimensions	Type	Import File	Export File	Disable	Persis
mt	11,2	Real	SenBulkerStdIni_Rev_01.xls		None	Yes
FLw	11	Real	SenBulkerStdIni_Rev_01.xls		None	Yes
NodeWeight	59	Real	SenBulkerStdNodeINI_Rev01.xls		None	Yes
A	46,46	Real	SenBulkerStd_A_Rev01.xls		None	Yes
vp	46	Real	SenBulkerStdIni_Rev_01.xls		None	No

```
*******************************************************************************
*                                    Macros                                   *
*******************************************************************************
```

ID	Text
M1	1
M2	1
M3	1
M4	1
M5	1
M6	1
M7	1
M8	1
M9	1
M10	1
M11	1
M12	1
M13	1
M14	1
M15	1
M16	1
M17	1
M18	1
M19	1
M20	1
M21	1
M22	1
M23	1
M24	1
M25	1
M26	1
M27	1
M28	1
M29	1
M30	1
M31	1
M32	1
M33	1
M38	1
M39	1
M40	1
M41	1
M42	1

```
  M43              1
  M44              1
  M45              1
  M46              1


******************************************************************************
*                               External Files                               *
******************************************************************************

  ID         Type              File Name                     Prompt
  ---------- ----------------- ----------------------------- ----------
  (null)                       SenBulkerStd_Rev_02.xls
  (null)                       SenBulkerStdIni_Rev_01.xls
  (null)                       SenBulkerStdNodeINI_Rev01.xls
  (null)                       SenBulkerStd_A_Rev01.xls
```

BIBLIOGRAPHY

BIBLIOGRAPHY

Ahlswede, R., Cai, N., Li, S.-Y. R., and Yeung, R. W. (2000). Network Information Flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216.

Andrews, D. J. (1981). Creative Ship Design. *Transactions of the Royal Institution of Naval Architects*, 123:447–471.

Andrews, D. J. (2006). Simulation and the design building block approach in the design of ships and other complex systems. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 462(2075):3407–3433.

Andrews, D. J. (2012). Arts and science in the design of physically large and complex systems. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 468(2139):891–912.

Baldwin, A., Austin, S., Hassan, T., and Thorpe, A. (1999). Modelling information flow during the conceptual and schematic stages of building design. *Construction Management and Economics*, 17(2):155–167.

Banks, J., Carson, J. S., Nelson, B. L., and Nicol, D. M. (2010). *Discrete-Event Simulation*. Prentice Hall, Upper Saddle River, N.J., fifth edition.

Bartolomei, J. E. (2007). *Qualitative Knowledge Construction for Engineering Systems : Extending the Design Structure Matrix Methodology in Scope and Procedure*. PhD thesis, Massachusetts Institute of Technology.

Batallas, D. A. and Yassine, A. (2006). Information Leaders in Product Development Organizational Networks: Social Network Analysis of the Design Structure Matrix. *IEEE Transactions on Engineering Management*, 53(4):570–582.

Bazaraa, M. S., Sherali, H. D., and Shetty, C. (2006). *Nonlinear Programming*. John Wiley & Sons, Hoboken, New Jersey, third edition.

Bernstein, J. I. (1998). *Design methods in the aerospace industry: looking for evidence of set-based practices*. PhD thesis, Massachusetts Institute of Technology.

Bertsimas, D. and Tsitsiklis, J. N. (1997). *Introduction to Linear Optimization*. Athena Scientific and Dynamic Ideas LLC, Belmont, Massachusetts.

Biedermann, W., Browning, T. R., Eppinger, S. D., Kreimeyer, M., Maurer, M. S., and Lindemann, U. (2013). Different DSM Types.

Black, T. A., Fine, C. H., and Sachs, E. S. (1990). *A Method for Design Using Precedence Relationships: An Application to Automotive Brake Systems.* PhD thesis, Massachusetts Institute of Technology.

Braha, D. and Bar-Yam, Y. (2004a). Information flow structure in large-scale product development organizational networks. *Journal of Information Technology*, 19(4):244–253.

Braha, D. and Bar-Yam, Y. (2004b). Topology of large-scale engineering problem-solving networks. *Physical Review E*, 69(1):016113.

Brown, D. K. (1986). Defining a Warship. *Naval Engineers Journal*, 98(2).

Brown, D. K. (1993a). History as a Design Tool. *Transactions of the Royal Institution of Naval Architects*, 135:41–59.

Brown, D. K. (1993b). Naval Architecture. *Naval Engineers Journal*, 105(1).

Brown, D. K. (1997). *Warrior to Dreadnought: Warship Design and Development 1860-1905*. Seaforth Publishing, Barnsley, UK.

Browning, T. R. (2001). Applying the Design Structure Matrix to System Decomposition and Integration Problems : A Review and New Directions. *IEEE Transactions on Engineering Management,*, 48(3):292–306.

Clausen, H., Lutzen, M., Friis-Hansen, A., and Bjørneboe, N. (2001). Bayesian and neural networks for preliminary ship design. *Marine Technology*, 38(4):268–277.

Cooper, S., Allen, G., Smith, R., Billingsley, D., and Helgerson, D. (2011). Ship Design Process Modeling: Capturing a Highly Complex Process. In *13th International Dependency and Structure Modelling Conference (DSM 11)*, Cambridge, MA.

Daft, R. L. and Lengel, R. H. (1986). Organizational information requirements, media richness and structural design. *Management science*, 32(5):554–571.

Daly, E. M. and Haahr, M. (2009). Social Network Analysis for Information Flow in Disconnected Delay-Tolerant MANETs. *IEEE Transactions on Mobile Computing*, 8(5):606–621.

Danilovic, M. and Browning, T. R. (2007). Managing complex product development projects with design structure matrices and domain mapping matrices. *International Journal of Project Management*, 25(3):300–314.

de Nooy, W., Mrvar, A., and Batagelj, V. (2005). *Exploratory social network analysis with Pajek*. Cambridge University Press, New York.

Deb, K., Bandaru, S., Greiner, D., Gaspar-Cunha, A., and Tutum, C. C. (2014). An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering. *Applied Soft Computing*, 15:42–56.

Devendorf, E. D., Devendorf, M., and Lewis, K. E. (2010). Using Network Theory to Model Distributed Design Systems. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, Texas.

Devendorf, E. D. and Lewis, K. E. (2011). The Impact of Process Architecture on Equilibrium Safety in Distributed Design. *Journal of Mechanical Design*, 133(10).

Diestel, R. (2005). *Graph Theory*. Springer-Verlag, Berlin Heidelberg, third edition.

Diez, M., Chen, X., Campana, E. F., and Stern, F. (2013). Reliability-Based Robust Design Optimization for Ships in Real Ocean Environment. In *12th International Conference on Fast Sea Transportation (FAST 2013)*, Amsterdam.

Doerry, N. (2009). Using the Design Structure Matrix to Plan Complex Design Projects. In *ASNE Intelligent Ships Symposium*, Philadelphia, PA.

Eppinger, S. D. and Browning, T. R. (2012). *Design Structure Matrix Methods and Applications*. MIT Press, Cambridge, MA.

Eppinger, S. D. and Rowles, C. M. (2000). Understanding the Effects of Product Architecture on Technical Communication in Product Development Organizations.

Euler, L. (1741). Solutio Problematis ad Geometriam Situs Pertinentis. *Commentarii Academiae Scientiarum Imperialis Petropolitanae*, 8:128–140.

Euler, L. (1956). The Seven Bridges of Königsberg. In Newman, J. R., editor, *The World of Mathematics*, pages 573–580. Simon and Schuster, New York.

Ford, D. N. and Sobek, D. K. (2005). Adapting real options to new product development by modeling the second Toyota paradox. *IEEE Transactions on Engineering Management,*, 52(2):175–185.

GAO (2002). Capturing Design and Manufacturing Knowledge Early Improves Acquisition Outcomes. Technical report, United States General Accounting Office, Washington, D.C.

Gillespie, J. W. (2012). *A Network Science Approach to Understanding and Generating Ship Arrangements in Early-Stage Design*. PhD thesis, University of Michigan.

Gillespie, J. W., Daniels, A. S., and Singer, D. J. (2010). An Agent-based Framework for Simultaneously Arranging Spaces, Components, and Systems. In Sphaier, S. H., Estefan, S. F., and Pasqualino, I. P., editors, *Proceedings of the 11th International Symposium on Practical Design of Ships and Other Floating Structures (PRADS 2010)*, pages 863–869, Rio de Janeiro, Brazil.

Gillespie, J. W., Daniels, A. S., and Singer, D. J. (2011). Decomposing ship arrangements using signed networks. In *Proceedings of the 15th International Conference on Computer Applications in Shipbuilding (ICCAS 2011)*, Trieste, Italy. The Royal Institution of Naval Architects.

Gillespie, J. W. and Singer, D. J. (2011). Gaining insight into the structure of an early-stage ship design. In Minai, A., Braha, D., and Bar-Yam, Y., editors, *The Eighth International Conference on Complex Systems*, pages 916–917, Quincy, MA. Springer Berlin Heidelberg.

Gillespie, J. W. and Singer, D. J. (2013). Identifying drivers of general arrangements through the use of network measures of centrality and hierarchy. *Ocean Engineering*, 57(January):230–239.

Goldratt, E. M. and Cox, J. (2004). *The Goal*. The North River Press, Great Barrington, MA, third edition.

Greenberg, M. D. (1998). *Advanced Engineering Mathematics*. Prentice Hall, Upper Saddle River, N.J., second edition.

Guenov, M. D. and Barker, S. G. (2005). Application of Axiomatic Design and Design Structure Matrix to the Decomposition of Engineering Systems. *Systems Engineering*, 8(1):29–40.

Gupte, M., Shankar, P., Li, J., Muthukrishnan, S., and Iftode, L. (2011). Finding Hierarchy in Directed Online Social Networks. In *Proceedings of the 20th International Conference on World Wide Web (WWW 2011)*, pages 557–566, Hyderabad, India. ACM.

Holme, P. and Saramäki, J. (2012). Temporal Networks. *Physics Reports*, 519(3):97–125.

Holtrop, J. and Mennen, G. (1982). An Approximate Power Prediction Method. *International Shipbuilding Progress*, 29(335).

Johnson, R. S. (1980). The Changing Nature of the U.S. Navy Ship Design Process. *Naval Engineers Journal*, 92(2):88–113.

Keller, R., Eckert, C. M., and Clarkson, P. J. (2006). Matrices or node-link diagrams: which visual representation is better for visualising connectivity models? *Information Visualization*, 5(1):62–76.

Kerns, C. (2011). *Naval Ship Design and Synthesis Model Architecture Using a Model-Based Systems Engineering Approach*. Masters thesis, Virginia Polytechnic Institute and State University.

Knight, J. T. and Singer, D. J. (2014). Applying Real Options Analysis to Naval Ship Design. In *Proceedings of ASNE Day 2014*, Washington, D.C. American Society of Naval Engineers.

Kreimeyer, M., Braun, S., Gürtler, M., and Lindemann, U. (2008). Relating Two Domains Via a Third - An Approach to Overcome Ambiguous Attributions using Multiple Domain Matrices. In *Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering*

Conference (IDETC/CIE 2008), Brooklyn, NY. American Society of Mechanical Engineers.

Laverghetta, T. and Brown, A. (1999). Dynamics of Naval Ship Design: A Systems Approach. *Naval Engineers Journal*, 113(3).

Leopold, R. (1975). Should the Navy Design Its Own Ships? *United States Naval Institute Proceedings*, 101(5).

Leveson, N. G. (2002). *A New Approach To System Safety Engineering*.

Liker, J. K. (2004). *The Toyota Way*. McGraw-Hill, New York.

Lindemann, U., Maurer, M., and Braun, T. (2009). *Structural complexity management*. Springer-Verlag, Berlin Heidelberg.

MacCallum, K. (1982). Understanding Relationships in Marine Systems Design. In *First International Marine Systems Design Conference (IMSDC 82)*, pages 1–9, London. Royal Institution of Naval Architects.

Maurer, M. S. (2007). *Structural Awareness in Complex Product Design*. PhD thesis, Technical University of Munich.

McKenney, T. A. (2013). *An Early-Stage Set-Based Design Reduction Decision Support Framework Utilizing Design Space Mapping and a Graph Theoretic Markov Decision Process Formulation*. PhD thesis, University of Michigan.

McKenney, T. A., Gray, A. W., Madrid, C., and Singer, D. J. (2012). The Use of a Fuzzy Logic Set-Based Design Tool to Evaluate Varying Complexities of Late-Stage Design Changes. *Transactions of the Royal Institution of Naval Architects Part A: International Journal of Maritime Engineering*, 154(A4):179–189.

Mebane, W. L., Carlson, C. M., Dowd, C., Singer, D. J., and Buckley, M. E. (2011). Set-Based Design and the Ship to Shore Connector. *Naval Engineers Journal*, 123(3):79–92.

Michelena, N. F. and Papalambros, P. Y. (1995). A Network Reliability Approach to Optimal Decomposition of Design Problems. *Journal of Mechanical Design*, 117(3):433–440.

Newman, M. E. J. (2003). The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167.

Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford University Press, New York.

NSWCCD (2005). Advanced Ship and Submarine Evaluation Tool.

Park, J. and Newman, M. E. J. (2005). A network-based ranking system for US college football. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(October).

Parker, M. C. (2013). The Temporal and Static Structure of Naval Design : A Network Theoretic Framework.

Parker, M. C. and Singer, D. J. (2012). Flexibility and Modularity in Ship Design : An Analytical Approach. In *Proceedings of the 11th International Marine Design Conference (IMDC 2012)*, Glasgow, Scotland.

Parker, M. C. and Singer, D. J. (2013). The Impact of Design Tools : Looking for Insights with a Network Theoretic Approach. In Bertram, V., editor, *12th International Conference on Computer and Information Technology Applications in the Maritime Industries (COMPIT 2013)*, pages 96–109, Cortona, Italy. Technische Universität Hamburg-Harburg.

Parker, M. C. and Singer, D. J. (2014). Analyzing the Dynamic Behavior of Marine Design Tools using Network Theory. *Submitted to Ocean Engineering*.

Parsons, M. G., Chung, H., Nick, E. K., Daniels, A. S., Liu, S., and Patel, J. (2008). Intelligent Ship Arrangements: A New Approach to General Arrangement. *Naval Engineers Journal*, 120(3):51–65.

Pasqual, M. C. (2010). *Multilayer Network Modeling of Change Propogation for Engineering Change Management*. PhD thesis, Massachusetts Institute of Technology.

Pasqual, M. C. and de Weck, O. L. (2012). Multilayer network model for analysis and management of change propagation. *Research in Engineering Design*, 23(4):305–328.

Ponstein, J. (1966). Self-avoiding paths and the adjacency matrix of a graph. *SIAM Journal on Applied Mathematics*, 14(3):600–609.

Rigterink, D., Piks, R., and Singer, D. J. (2013). The use of network theory to model disparate ship design information. In *Proceedings of the 12th Symposium on Practical Design of Ships and Floating Structures (PRADS 2013)*, Changwon City, Korea.

Scovel, D. (1975). The Effect of Public Opinion Upon National Security Policy. *United States Naval Institute Proceedings*, 101(5):118–133.

Sen, P. and Yang, J.-B. (1998). Multi Objective Decision Making. In *Multiple Criteria Decision Support in Engineering Design*, chapter 4, pages 113–175. Springer-Verlag, London.

Shahan, D. and Seepersad, C. C. (2012). Bayesian Networks for Set-Based Collaborative Design. *Journal of Mechanical Design*, 134(7).

Shai, O. (2003). Transforming engineering problems through graph representations. *Advanced Engineering Informatics*, 17(2):77–93.

Singer, D. J., Doerry, N., and Buckley, M. E. (2009). What Is Set-Based Design? *Naval Engineers Journal*, 121(4):31–43.

Smith, R. P. and Eppinger, S. D. (1997a). Identifying Controlling Features of Engineering Design Iteration. *Management Review*, 43(3):276–293.

Smith, R. P. and Eppinger, S. D. (1997b). Identifying Controlling Features of Engineering Design Iteration. *Management Science*, 43(3):276–293.

Sobek, D. K., Ward, A. C., and Liker, J. K. (1999). Toyota's principles of set-based concurrent engineering. *Sloan Management Review*, 40(2):67–83.

Sosa, M. E., Eppinger, S. D., and Rowles, C. M. (2007a). A Network Approach to Define Modularity of Components in Complex Products. *Journal of Mechanical Design*, 129(11):1118.

Sosa, M. E., Eppinger, S. D., and Rowles, C. M. (2007b). Are your engineers talking to one another when they should? *Harvard Business Review*, 85(11):133–142.

Steward, D. V. (1981a). *Systems Analysis and Management: Structure, Strategy, and Design*. Petrocelli Books.

Steward, D. V. (1981b). The Design Structure System: A Method for Managing the Design of Complex Systems. *IEEE Transactions on Engineering Management*, 28(3):71–74.

Steward, D. V. (1991). Planning and Managing the Design of Systems. In *Technology Management: the New International Language*, pages 189–193. IEEE.

Steward, D. V. (1993). Re-engineering the Design Process. In *Second Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 94–98. IEEE.

Strogatz, S. H. (1994). *Nonlinear Dynamics and Chaos*. Perseus Books Publishing, Cambridge, MA.

Valente, T. W. (1995). *Network Models of the Diffusion of Innovations*. Hampton Press, Cresskill, NJ, first edition.

Watson, D. (1962). Estimating Preliminary Dimensions in Ship Design. *Transactions of the Institution of Engineers and Shipbuilders in Scotland*, 105:110–184.

Watson, D. (1998). *Practical Ship Design*. Elsevier, London.

Watson, D. and Gilfillan, A. (1977). Some Ship Design Methods. *Transactions of the Royal Institution of Naval Architects*, 119:279–324.

Whitfield, R. I., Duffy, A. H., Wu, Z., and Meehan, J. (2003). Intelligent Design Guidance. In *14th International Conference on Engineering Design (ICED 2003)*, Stockholm, Sweden.

Winter, D. C. (2014). Personal Correspondance.

Work, R. O. (2013). The Littoral Combat Ship: How We Got Here, and Why. Technical report, Naval War College Press, Newport, RI.

Wyatt, D. F., Wynn, D. C., Jarrett, J. P., and Clarkson, P. J. (2011). Supporting product architecture design using computational design synthesis with network structure constraints. *Research in Engineering Design*, 23(1):17–52.

Yang, J.-B. and Sen, P. (1996). Interactive trade-off analysis and preference modeling for preliminary multiobjective ship design. *Systems Analysis, Modelling and Simulation*, 26:25–55.

Yassine, A. and Braha, D. (2003). Complex Concurrent Engineering and the Design Structure Matrix Method. *Concurrent Engineering*, 11(3):165–176.

Yassine, A., Falkenburg, D., and Chelst, K. (1999). Engineering design management: An information structure approach. *International Journal of Production Research*, 37(13):2957–2975.

Yassine, A., Whitney, D. E., Daleiden, S., and Lavine, J. (2003a). Connectivity maps : Modeling and analysing relationships in product development processes. *Journal of Engineering Design*, 14(3):377–394.

Yassine, A. A., Joglekar, N., Braha, D., Eppinger, S., and Whitney, D. (2003b). Information hiding in product development: the design churn effect. *Research in Engineering Design*, 14(3):145–161.

Zhou, T., Ren, J., Medo, M., and Zhang, Y.-C. (2007). Bipartite network projection and personal recommendation. *Physical Review E*, 76(4):1–7.