USING PARAMETERIZED EFFICIENT SETS TO MODEL ALTERNATIVES FOR SYSTEMS DESIGN DECISIONS

A Dissertation
Presented to
The Academic Faculty

by

Richard J. Malak Jr.

In Partial Fulfillment
of the Degree
Doctor of Philosophy in the George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology

December 2008

USING PARAMETERIZED EFFICIENT SETS TO MODEL ALTERNATIVES FOR SYSTEMS DESIGN DECISIONS

Approved by:

Dr. Christiaan J.J. Paredis
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Leon McGinnis
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Bert Bras
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Ruchi Choudhary
School of Architecture
Georgia Institute of Technology

Dr. David Rosen
School of Mechanical Engineering
Georgia Institute of Technology

ACKNOWLEDGEMENT

Above all others, I must thank my family for their years of support and encouragement. My mother has been particularly important to me. Looking back on my childhood, her devotion to my education is clear: Before I could read, she would read to me. Once I could read on my own, she would drive me regularly to the nearest public library despite it being nearly an hour from our home. Throughout my college years, she has been a steady source of encouragement and has consistently filled the financial gaps left by my scholarships and loans But perhaps most importantly, she instilled in me at an early age that I could do anything I wanted with my life—that I am not constrained by the paths taken by my ancestors and predecessors, but by the limits of my imagination and determination.

I also must acknowledge my younger siblings, my brother Daniel and my sister Amanda. Their support has been invaluable to me. They keep me grounded and focused on what is important. I value greatly my regular conversations with them.

To Donna, my wife and best friend, I offer the deepest thanks. It is hard to articulate the role she plays in my life or the impact that she has had on this work. Always there for me, she is the difference maker in my life.

Academically speaking, I am most deeply indebted to my advisor, Dr. Chris Paredis. He is simultaneously open-minded and critical—a combination that is as valuable as it is rare. His insights and criticisms have helped to shape both the ideas in this thesis and my broader understanding of engineering.

I thank my committee members, Drs. Paredis, Rosen, Bras, McGinnis and Choudhary for their questions and insights. Their input certainly has strengthened my research and will be valuable to me in the future.

I owe gratitude to all the members of the Systems Realization Laboratory here at Georgia Tech, where I have found both professional advice and personal friendship. I have learned much from other members of the SRL, which is one of the greatest compliments I can give. It is impossible for me to name everyone who has had a positive impact on me in my time here. Those in the "266 crowd" standout in my mind due to our relatively close interactions. Over the past few years I have enjoyed sharing MaRC 266 office space and engaging in discussions (on all topics) with Jason Aughenbaugh, Stephanie Thompson, Scott Duncan, Steve Rekuc, Alek Kerzhner, Tommy Johnson, Jonathan Jobe, Roxanne Moore, Morgan Bruns, Jay Ling and Tarun Rathnam.

Two undergraduate research assistants were instrumental in completing this research. Lina Jensen played a large role in gathering the product information for the design problem in Chapter 6. Michael Lennard is responsible for much of the development and debugging work associated with the dynamical models for the hybrid vehicle architectures in Chapter 7.

Finally, I gratefully acknowledge the support of the National Science Foundation via grant DMI-0522116 and the ERC for Compact and Efficient Fluid Power under grant EEC-0540834.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

$z \in \mathbb{R}^N$, $\mathbf{z} = [z_1, z_2 \ldots, z_N]$	Respectively, a system-level attribute and a vector of such attributes.
$y \in \mathbb{R}^M$, $\mathbf{y} \in [y_1, y_2, \ldots, y_M]$	Respectively, a component-level attribute and a vector of such attributes.
$z_i = S_i(\cdot)$	System composition model; used to compute a system-level attribute based on one or more component-level attributes.
$\mathbf{S}(\cdot) = [S_1(\cdot), S_2(\cdot), \ldots, S_N(\cdot)]$	Vector-valued system composition model. Note that $\mathbf{z} = \mathbf{S}(\mathbf{y})$.
$v(\cdot)$, $u(\cdot)$	Respectively, a value or utility function, indicating preference for a particular attribute or making tradeoffs among attributes. In this work, a value function, $v(\cdot)$, is indicative of a decision under certainty and a utility function, $u(\cdot)$, is indicative of a decision under uncertainty.
$E[\cdot]$	Expectation operator.
$T(\cdot)$	Tradoeff model.
$\mathbf{T}(\cdot) = [T_1(\cdot), T_2(\cdot), \ldots, T_K(\cdot)]$	Vector-valued tradeoff model.
$\tilde{\mathbf{y}}$	Vector of attributes used as inputs to a tradeoff model.
$\tilde{\mathbf{Y}}$	Set of valid $\tilde{\mathbf{y}}$ vectors.
x, $\mathbf{x}$	Respectively, a design variable and a vector of design variables.
$\mathbf{z} = \mathbf{F}(\mathbf{x})$	An engineering analysis model that computes system-level attributes given design variable values.
a	A decision alternative
A	A set of decision alternatives

U_i, where $i \in \{0,1,2\}$	A set of utility functions
$\mathbf{x}_i$	A data vector input into the SVDD algorithm or a support vector identified using the method.
$\mathbf{a}$	Center of the SVDD bounding hypersphere.
R	Radius of the SVDD bounding hypersphere.
C	SVDD outlier detection sensitivity parameter.
β_i	Weight corresponding to the i^{th} support vector.
$K(\cdot,\cdot)$	Kernel function.
$\Phi : \mathbb{R}^N \to \mathscr{F}$	Nonlinear mapping from data space to feature space.
q	Gaussian kernel function width parameter.
$\mathbf{A}$, A_{ij}	Respectively, adjacency matrix identified during SVC and the ij^{th} element of the matrix.

SUMMARY

The broad aim of this research is to contribute knowledge that enables improvements in how designers model decision alternatives at the systems level—i.e., how they model different system configurations and concepts. There are three principal complications:

- Design concepts and systems configurations are partially-defined solutions to a problem that correspond to a large set of possible design implementations,
- Each concept or configuration may operate on different physical principles, and
- Decisions typically involve tradeoffs between multiple competing objectives that can include "non-engineering" considerations such as production costs and profits.

This research is an investigation of a data-driven approach to modeling partially-defined system alternatives that addresses these issues. The approach is based on compositional strategy in which designers model a system alternative using abstract models of its components. The component models are representations of the rational tradeoffs available to designers when implementing the components. Using these models, designers can predict key properties of the final implementation of each system alternative.

A new construct, called a parameterized efficient set, is introduced as the decision-theoretic basis for generating the component-level tradeoff models. Appropriate efficiency criteria are defined for the cases of deterministic and uncertain data. It is shown that the model composition procedure is mathematically sound under reasonable assumptions for the case of deterministic data. This research also introduces an approach

for describing the valid domain of a data-driven model based on the use of support-vector machines. Engineering examples include performing requirements allocation for a hydraulic log splitter and architecture selection for a hybrid vehicle.

CHAPTER 1:

INTRODUCTION

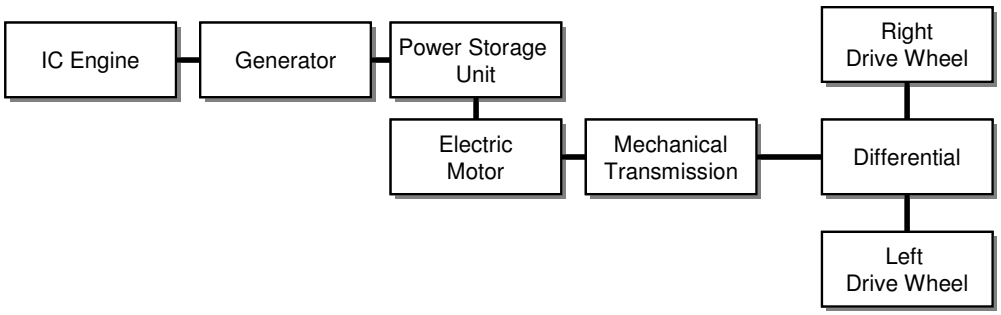
The focus of this chapter is on establishing the context for and scope of the investigation. Section 1.1 is an overview of the research topic, including why it is important and what are the main challenges. Section 1.2 is a summary of the main desirable characteristics for an approach to modeling alternatives for system-level decisions. Section 1.3 is an overview of the particular modeling approach studied in this research. Section 1.4 is an explanation of the main research questions and the corresponding hypotheses. Finally Section 1.5 is a roadmap to the remainder of this document.

1.1 Research Overview

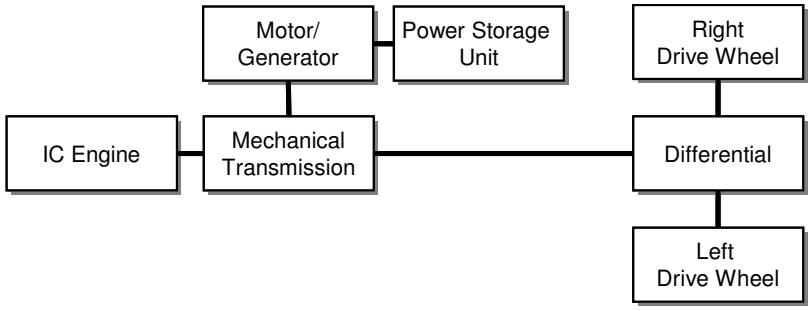
1.1.1 What is this Research About?

This research is about identifying new ways in which systems designers can think about making decisions. This can lead to new ways of formulating and solving decision problems as well as an improved understanding of decision methods in current use. Some examples of the types of decisions targeted in this investigation include:

- Automotive designers are choosing between different architectural configurations for a hybrid car. Options include both serial and parallel power train arrangements (e.g., Figure 1.1). They care about profitability of the new design, which they believe to be a function of fuel efficiency, reliability, performance characteristics (acceleration, top speed, etc.), environmental impact, lifecycle considerations and production costs.



(a)



(b)

Figure 1.1: Possible power train configurations for a (a) serial and (b) parallel hybrid vehicle.

- The same designers are considering different technologies to implement their system: gas-electric and gas-hydraulic hybrids. Gas-electric hybrid vehicles have been on the market for a few years and the designers are confident that they understand the tradeoffs they will encounter if they choose that approach. In contrast, gas-hydraulic systems have received less study. The designers believe gas-hydraulic technology potentially has greater upside—particularly for heavier vehicles such as sport-utility vehicles and service vehicles (garbage trucks, delivery trucks, etc.)—but are uncertain about whether it is superior to gas-electric technology for their particular problem.

- Suppose these automotive designers choose to develop a gas-hydraulic system using a parallel power train configuration. The next step is to assign domain and/or subsystem specialists to design its components. The system-level designers must identify implementation-neutral specifications for the components such that the specifications are technically feasible and, if met, lead to good system performance.

Other examples exist spanning numerous application domains. Collectively, these types of decisions are referred to as *system-level decisions* in this research because they require system-wide information and have a system-wide impact.

System-level decision making is a critical part of a systems design process, which tends to follow a top-down, hierarchical progression (Royce 1970, Boehm 1988, Forsberg and Mooz 1992, Wertz and Larson 1999, Buede 2000, Sage and Armstrong Jr. 2000). Two main types of system-level decisions are considered in this research:

- System Selection Decisions. Decisions between different *types* of systems, such as competing system architectures, design concepts or implementation technologies.
- Requirements Allocation Decisions. Also called requirements flowdown or component sizing, this type of decision entails determining appropriate design specifications for the components of a particular system architecture.

In the preceding examples, the first two are systems selection problems and the last is a requirements allocation problem.

Some authors use the terms compromise decisions (e.g., (Shupe, et al. 1987, Karandikar, et al. 1989, Mistree, et al. 1993b)) or parametric design to refer to

requirements allocation decisions. The latter term is adopted in this research because it better better fits with the systems engineering literature and it better reflects the specialized, system-level nature of the decisions of interest. Moreover, a requirements allocation decision is a *type of* compromise decision or parametric design problem. Like all such problems, designers determine the most preferred settings for certain parameters associated with a particular system architecture (in this case, the specifications for its components). However, unlike many compromise decisions, requirements allocation is just one step in a sequence of allocation and selection decisions—after requirements allocation, designers still must determine how to achieve the preferred specifications.

Practicing designers do make system-level decisions successfully. They design automobiles, aircraft, ships, trains, computers, robots and all manner of complex systems. However, this does not mean their decision methodology is ideal—or even good. Designers do the best they can within the limits of their decision-making resources. The goal of this research is to contribute knowledge about how designers can get more return on their limited resources. Ideally, this would lead to designers being able to make sound and quantitative decisions earlier in a design project than would be practical otherwise. Another desirable, and related, outcome would be for designers to be able to evaluate a greater number of heterogeneous system alternatives than they typically are able to do.

1.1.2 Why is it Important?

One perspective on the importance of system-level decision making relates to its lasting impact on a design process. Numerous authors assert that system-level design decisions are particularly important to the success of a design project on the basis that these early-phase decisions constrain subsequent decisions. For example, in his survey of

conceptual design research Hsu and Woon (1998) state that “a poorly conceived design concept can never be compensated for by a good detailed design”. Pahl and Beitz (1996, pp. 68) make a similar assertion, stating that “in the subsequent embodiment and detail design phases it is extremely difficult or impossible to correct fundamental shortcomings of the solution principle.”.

From a more quantitative perspective, Boehm (1981). and Davis (1993). both conclude based on data about software system design projects that the cost of correcting design mistakes increases the longer the mistakes go unrecognized. The implication is that value exists in researching approaches for improved decision making in design. Although their results are based on data about software systems, the general trend applies to systems in general. Authors in the systems engineering community commonly cite these studies to emphasize the importance of good decision-making practices early in a project and systematic decision making in general.

Overall, quantitative data regarding the value and impact of any decision making practices is sparse. However, there is some evidence to suggest that good practices can be a source of competitive advantage. Ward and coauthors (1995) argue that the success of Toyota Motor Company is due in part to the way in which they approach such decisions. Sobek and coauthors (1999) build upon this work and characterize the decision-making process at Toyota in terms of what they call set-based concurrent engineering. Liker and coauthors (1996) demonstrate that set-based concurrent engineering is in fact more prevalent among Toyota Motor Company and its suppliers, as compared to American auto manufacturers and their suppliers.

Despite the success of Toyota Motor Company and the insights gained by studying its practices, it is unclear whether or to what extent any particular company would gain by adopting these practices. Toyota relies heavily on highly experienced personnel to execute their approach, which emerged over many years rather than being imposed suddenly. In order to achieve improvements in system-level decision making on a widespread basis, it is necessary to achieve a more substantial and fundamental understanding of such decisions and approaches for solving them.

1.1.3 What is the Challenge?

The principles of decision making are well-established in the literature. One can think of a decision process as a sequence of four main steps (Clemen 1996):

1. Formulate the decision problem in terms of solution-independent objectives and willingness to make tradeoffs between competing objectives.
2. Identify alternative courses of action.
3. Evaluate each decision alternative relative to the stated objectives and identify the most-preferred alternative.
4. Execute the most preferred course of action.

The challenge addressed in this research is how to achieve the third step for system-level decision problems. In particular, *the fundamental problem is how to model the decision alternatives*. Designers have limited resources, which means it is not necessarily in their interest to create the most accurate or highest-fidelity model.

This research addresses both the theory and associated practical considerations for how designers can model partially-defined systems at a high level of abstraction. An assumption in this research is that systems designers generally can model the

relationships between a system's components, but that it is more difficult for them to understand the tradeoffs involved with implementing the components. Although it is true that most systems engineers have a tacit understanding of the components they would used in a system, this knowledge is difficult formalize. Ideally, designers would use formalized computer-interpretable models to enable fast and efficient exploration of many different alternatives rather than fixating on only a handful of choices. Furthermore, a model-based approach may be the only viable option for novel systems and technologies about which systems designers lack a preexisting intuition.

Section 1.3 is an overview of the modeling approach investigated in this research. It is based on component-level models that designers can use to predict the tradeoffs that they or other designers would make when implementing the components of a system under particular decision objectives. The models are data-driven, which enables system-level designers to consider the far-reaching consequences of a course of action without having to model every downstream consideration (detail design, manufacturing, etc.) explicitly. As discussed in Section 2.1, this type of information is important when one considers the relationships between different decisions in a systems realization process.

1.2 Desirable Characteristics for a Modeling Approach

It is instructive to consider which characteristics are desirable for an approach to modeling system-level decision alternatives. To some degree, such characteristics are a matter of perspective and different people may identify different considerations or assign emphasis differently. The following are the main characteristics considered important in this research.

Soundness from a Decision-Making Perspective

At a minimum, a decision process should be internally consistent in that it does not result in selections contrary to stated preferences or available information. The way in which designers model their decision alternatives has a large impact on whether they can make decisions soundly. Essentially, any model that ignores information—e.g., through simplifying assumptions—potentially can lead to unsound decisions. In practice, no model is free from assumptions and it is desirable to develop an understanding of the potential impact of any major assumptions from this decision-making perspective. Ideally, it would be provable that, at least under some mild assumptions, a particular modeling strategy supports sound decision making.

Based on Quantitative, Computer-Interpretable Models

The principles of decision making place no requirements on how one models decision alternatives except to say that models should reflect the available information and be consistent with the beliefs of the decision maker. However, in the context of system-level decision making it is desirable for models also to be quantitative and computer-interpretable. One reason is that at this stage of a design process it typically is valuable for designers to explore the space of alternatives widely. This exploration is difficult without the use of quantitative, computer-interpretable models. Another reason is that qualitative models (e.g., mental models) can lack the fidelity required to discriminate between alternatives. They can be useful for eliminating inferior design concepts with minimal effort or for identifying an obviously-superior concept, but are less effective for discriminating between similarly-performing decision alternatives. Furthermore, lacking a quantitative basis, they offer little support for allocation decisions. Although a quantitative model may be sufficient for identifying whether a gear drive is superior to a

belt drive for a particular application, it is poorly suited for deciding what specific gear ratio the drive should have.

Comprehensive in Scope

A common pitfall for designers is that sometimes they ignore important system attributes while making decisions. This usually is because they lack a good model for computing the attributes. For example, a designer might consider only the "engineering attributes" of a car, such as its top speed, acceleration and breaking figures of merit, but ignore other equally important attributes, such as cost. Although this practice always is undesirable, it is particularly deleterious for system-level decision making. Failure to consider major system attributes and adopt a comprehensive view of the available tradeoffs can lead designers down a path toward an undesirable outcome.

Based on Models that are Fast to Compute

If designers are to use models for system-level alternatives inside of an optimization loop or in the context of design space exploration, then computational complexity becomes an important concern. These computational procedures can require tens- or hundreds-of-thousands of model evaluations. Therefore, it is desirable that the models be fast to compute. The time to create the model also is important, but much less so than the time required to evaluate it.

1.3 From Component-Level Data to System-Level Models

This research examines an approach to modeling system-level alternatives that relies on predictive modeling and model composition. The vision is for designers to generate reusable predictive models of common engineering components that they can compose quickly and confidently to model a system-level alternatives of interest. Unlike other applications of predictive modeling in design, the approach investigated here is

based on models formulated strictly in terms of a component's top-level attributes. This allows designers to abstract away lower-level implementation details and focus on system-level issues. The following is an overview of several key concepts associated with the approach.

1.3.1 Component-Level Attributes

In this research, a top-level attribute for a component—or, a *component-level attribute*—is an implementation-independent characteristic that applies to all components of equivalent functionality. Designers commonly use these attributes when making decisions about how to implement a component of that type. Figure 1.2 is a simplified illustration of this idea for hydraulic pumps with efficiency and cost as attributes. Designers can visualize physically heterogeneous pump implementations in terms of these attributes. Other pump attributes potentially of interest include maximum operating speed, mass, external dimensions and maximum flow rate.

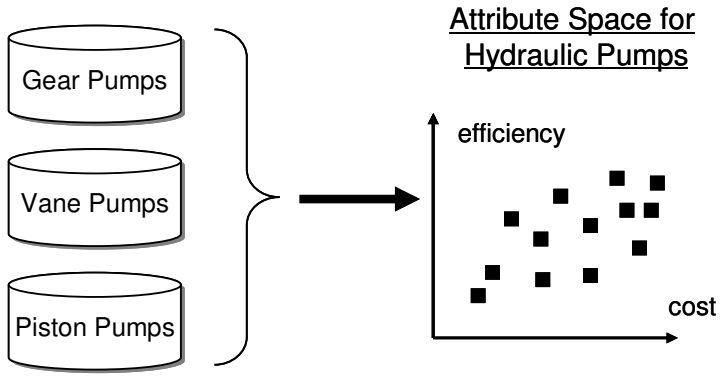


Figure 1.2: Physically heterogeneous implementations of a functional component share the same attribute space.

For most types of components, the top-level attributes are characteristics that would appear in a parts catalog or that companies would publish in their product literature. Lower-level implementation details—information that would indicate precisely how designers achieve the top-level attribute values—are not included. For example, one would not include gear geometry, orifice geometry or materials specifications as part of the top-level attributes for a gear pump.

Another aspect of component-level attributes is that designers typically cannot determine their values independently. For example, increasing the flow rate of a pump typically leads to an increase in its size since it would have to accommodate more fluid. Similarly, increasing the efficiency typically leads to an increase in cost due to the tighter tolerances required.

1.3.2 Preferences and Modeling Component-Level Attributes

In order to model components accurately in terms of their top-level attributes, it is important for one to consider the effect of preferences and how designers would make decisions when implementing that type of component. For example, suppose a designer of a hydraulic system cares only about the efficiency of a pump and what it cost to buy it (such that efficiency should be maximized and cost should be minimized). Several data points depicted in Figure 1.2 would be considered inferior by such a designer because there exist other pumps that are better in one or both of the attributes of interest.

The reason for this relates to the decision-theoretic notion of dominance. Figure 1.3 is a depiction of the data points from Figure 1.2 classified as either dominated (ones no designer would choose) and non-dominated (also called efficient or Pareto optimal). Informally, one alternative dominates another if it is at least as good in all attributes and

strictly better in at least one (see Chapter 3 for a formal definition). The significance of this from a modeling perspective is that some of the data is misrepresentative of what designers can achieve by implementing a component. Thus, in the approach studied here, one constructs predictive models based only on efficient set data (i.e., the set of non-dominated points). Figure 1.4 is an illustration of the difference between a model fit to all data about a component and a model fit only to the efficient set. The model fit to all data is a poor predictor of the relationship between cost[a] and efficiency for pump implementations that designers would actually choose. This is discussed further in Chapter 3.

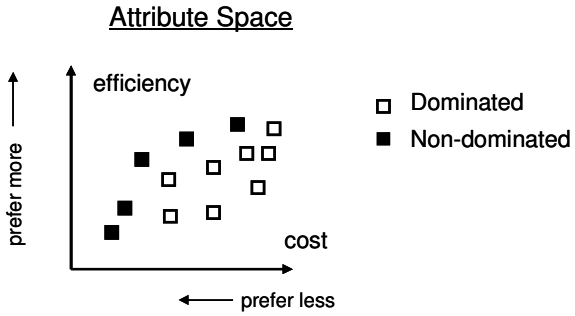


Figure 1.3: Illustrative visualization of dominance.

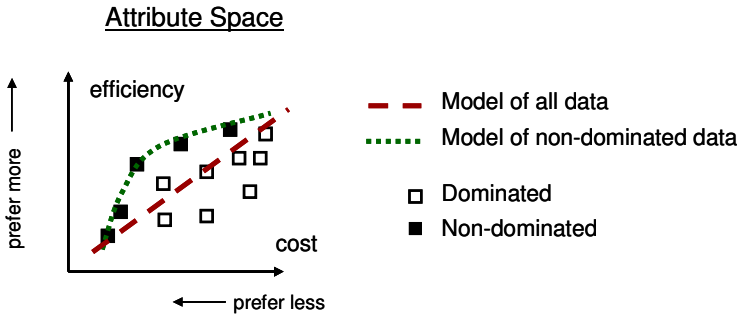


Figure 1.4: Illustration of distinction between model fit to all data and model fit to only the non-dominated data.

[a] Note that in this context, cost refers to the purchase price of the pump because this is an expense incurred by the system-level designer. The term cost is used in this way throughout this research. Clarification is stated whenever the meaning is ambiguous.

1.3.3 Parameterized Efficient Sets

Some types of components have attributes for which preference orderings are not universal. For example, although designers generally will seek to minimize cost or maximize efficiency, there is no problem-independent preference ordering for the bore of a hydraulic cylinder or the gear ratio of a transmission. Such attributes are incompatible with the classical notion of dominance.

To solve this problem, a novel extension of the classical Pareto dominance criterion is introduced in Chapter 3. Under the new dominance rule, one classifies attributes as dominator attributes or parameter attributes (respectively, those with and without problem-independent preference orderings). One can determine whether one component implementation dominates another only if their parameter attribute values are equivalent. If this is the case, one uses the classical dominance rule applied to the dominator subset of the attributes to draw conclusions about domination.

One can interpret the resulting non-dominated data—called the *parameterized efficient set*—as a family of efficient sets. Designers can recover a specific efficient set by specifying values for the parameter attributes. This construct is the basis for component-level predictive modeling in this research.

1.3.4 Predictive Tradeoff Models

In this research, a *predictive tradeoff model* is a mathematical model fit to parameterized efficient set data about a type of component. Typically, this is an algebraic relationship designers can use to compute the value of one or more component-level attributes as a function of the others. The intent of this model is to capture dependencies

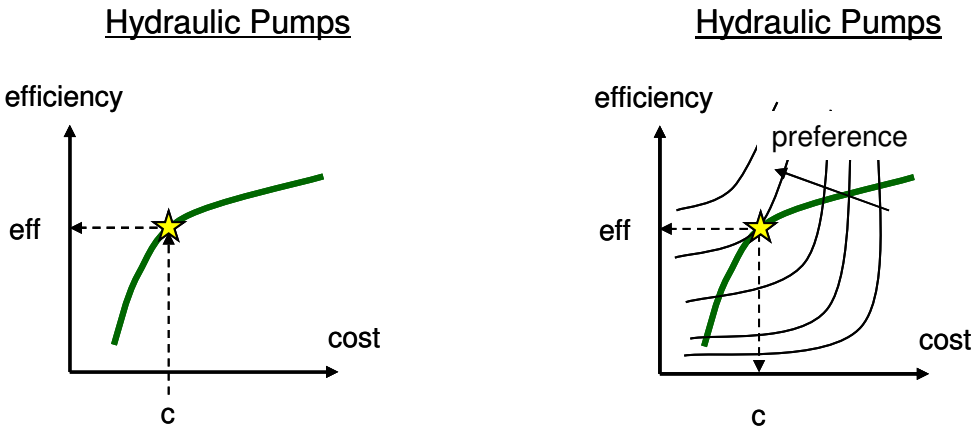
among the attribute values along the technology frontier—i.e., on the surface that delineates between implementations that are feasible and those that are not.

Component implementations at different points along the model represent different tradeoffs that designers can make. For example, one designer may place more emphasis on cost while another emphasizes efficiency. Both solutions—if they are rational in a decision theoretic sense—are in the parameterized efficient set and lie on the tradeoff model. This is why they are called "tradeoff" models.

The "predictive" part of the term requires some additional explanation. Strictly speaking, designers use the models to compute certain component-level attributes as a function of the others. However, this is not the prediction of interest in this research. The question of interest is:

> *What would be the resulting attribute vector if a designer implemented the component in question with particular preferences for making tradeoffs among the component-level attributes?*

This prediction problem speaks to the actual use of the mathematical relationship. Figure 1.5 is an illustration of the two different interpretations. In Figure 1.5(a), one computes pump efficiency as a function of its cost. In Figure 1.5(b), one searches along the modeled relationship for the combination of cost and efficiency that maximizes decision maker preferences. The second interpretation is adopted in this research.



(a) (b)

Figure 1.5: Illustration of different interpretations of predictive tradeoff models: (a) one of the attributes is predicted given the other and (b) the complete attribute vector is predicted given the relationship between the attributes and the decision preferences.

1.3.5 Composing System-Level Models from Component-Level Models

Designers can compose a model for a system-level decision alternative by combining component-level predictive tradeoff models with models of the interactions among the components. They can use the resulting model to compute system-level attributes as a function of component-level attributes. The advantage of taking a compositional approach is that data is more likely to exist for the components of a system than for the system itself. In principle, designers can model a novel system for which they have no prior data provided it consists of well-understood components that interact in a well-understood way.

The mathematical form of the interaction model depends on the system in question and the system-level attributes of interest. The example problems in research involve both algebraic interaction models (the log splitter system of Chapter 6) and dynamical interaction models (the hydraulic hybrid vehicle example of Chapter 7).

Designers formulate system-level decision problems as searches over the component-level attributes that are used as independent variables in their respective tradeoff models. One can conduct this search using standard optimization techniques. This problem formulation is described in Chapter 5.

1.4 Research Questions and Validation Strategy

One can summarize the motivation for this investigation in terms of a research question:

> *How can designers model system-level decision alternatives quantitatively in order to support sound and effective system-level decision making?*

This question is too broad to be answered satisfactorily in a single study. Instead, it serves as a launching pad to several specific research questions that revolve around the concepts introduced in Section 1.3. The broad aim is to discover new and general knowledge about system-level decision making. There are four main research questions:

RQ1. *How can designers conclude that one implementation of a component dominates another when they lack specific knowledge of the system in which the component will be used?*

RQ2. *How can designers describe the set of valid inputs to a tradeoff model mathematically?*

RQ3. *Under what conditions can designers compose component-level tradeoff models in order to model a system-level decision alternative soundly?*

RQ4. *How should designers identify and visualize the (parameterized) efficient set of tradeoffs when the attribute data is uncertain?*

1.4.1 Component-Level Dominance Analysis

Designers typically apply dominance analysis—usually in the form of the classical Pareto dominance criterion—at the system level and in the context of a specific decision problem. Although useful in many situations, classical Pareto dominance is inadequate at the component level or when problem-independence is desirable. The following hypothesis is studied in this research:

H1. Designers can use the parameterized Pareto dominance rule to eliminate attribute data about dominated implementations of a component.

This novel dominance rule is defined and validated in Chapter 3. The evidence in support of this hypothesis is a mathematical proof that the rule is sound under reasonable assumptions. In this context, soundness means that if the rule indicates that one implementation dominates another then the utility of a system that includes the dominated implementation is guaranteed to be lower than that of a system that includes the other implementation.

1.4.2 Tradeoff Model Domain Description

In the context of predictive modeling, a domain description is a mathematical representation of the set of valid inputs to a model. In many applications, one can describe the input domain using upper and lower bounds on the individual input variables. However, the inputs to a tradeoff model seldom are independent and the valid domain usually is not hypercube-shaped. When examining attribute data for a component, one typically finds irregularly-shaped, non-convex regions of data. Since the approach to generating tradeoff models is data-driven, designers ideally would have a

data-driven approach to describing its domain. Thus, the hypothesis corresponding to RQ2:

H2. Designers can use a domain description procedure based on kernel-based support vector domain description and clustering methods.

The domain description methodology is presented in Chapter 4. it is an extension of methods found in the machine learning literature. Chapter 4 includes basic examples to illustrate the approach and to provide evidence that it is effective. Further validation of the methodology stems from the example problems of Chapter 6 (requirements allocation for a hydraulic log splitter) and Chapter 7 (architecture selection for a hydraulic hybrid vehicle). Domain descriptions of the component-level tradeoff models are essential to the success of both examples.

1.4.3 Composing Tradeoff Models

Tradeoff model composition is an essential part of the modeling approach investigated in this research. However, it is prudent to question the validity of this procedure. Designers fit a tradeoff model to parameterized efficient set data, which is a subset of the original implementation data. Consequently, the composition procedure could be invalid if any of the excluded implementations could be part of the optimal system configuration.

The hypothesis corresponding to this question is as follows:

H3. One can compose predictive tradeoff models soundly if the tradeoff models are based on parameterized Pareto sets and all induced preferences for any component-level dominator attribute are monotonic in the same direction.

An induced preference is the preference ranking on the values of a component-level attribute implied indirectly by preferences for another attribute. This situation arises in systems problems where one has preferences directly for system-level attributes (e.g., to minimize cost or maximize system performance measures). The induced preference relates to how a component-level attribute affects the system-level attributes. If an induced preference is monotonically increasing, then increasing the value of that component-level attribute (whilst holding all others equal) leads to an increase in overall utility. The soundness conditions are explained in detail in Chapter 5.

This hypothesis is subjected to both mathematical and empirical validation efforts. The mathematical analysis, reported in Chapter 5, deals with the composition of parameterized Pareto sets. It is proved that provided one treats dominator attributes appropriately, any component implementation eliminated by the parameterized Pareto dominance rule cannot be part of the optimal system. This result has significance beyond tradeoff modeling, which is discussed in Chapter 5.

Designers forfeit any mathematical guarantees once they begin approximating the data using a continuous model. This necessitates an empirical investigation. The example problems of Chapter 6 and Chapter 7 serve this purpose. In both examples, the results one obtains using the composed model compares favorably with an exhaustive search of the components database.

1.4.4 Tradeoffs under Uncertainty

Two complications emerge when attribute data is uncertain. First, one cannot conduct the comparisons required for the Pareto dominance test because the attributes do

not take on precise values. Second, one now has additional flexibility in terms of tradeoffs since, informally speaking, one can trade performance for reductions in risk.

In the deterministic case, one can visualize a parameterized Pareto set as a surface in the space of attributes. A similar representation is desirable for tradeoffs under uncertainty. In the interest of problem scope, this investigation is limited to the special case of attributes modeled as being normally distributed and statistically independent. Under this assumption, the hypothesis corresponding to RQ4 is:

H4. Designers can identify the (parameterized) efficient set of tradeoffs under uncertainty using (parameterized) stochastic dominance criteria and can visualize this set as a surface in mean-variance space.

Stochastic dominance criteria are analogous to Pareto dominance, but account for uncertainty in the attributes and a designers attitude toward risk. Chapter 8 contains an explanation of stochastic dominance and the generalized interpretation of tradeoffs when uncertainty is present. One can extend the stochastic dominance criteria to operate in a manner similar to parameterized Pareto dominance.

Validation efforts associated with H4 include a logical development of the result from the appropriate decision theoretic principles and an empirical investigation. In the example problem, decision results obtained using a tradeoff model fit to the parameterized efficient set data correspond well to a decision obtained using a trusted, but impractical, solution method. These efforts are reported in Chapter 8.

1.5 Investigation Roadmap

This document is organized into several major theme groups, each of which has one or more associated chapters. Figure 1.6 is a high-level summary of the material.

The first theme area is introductory and contextual material. This group consists of the current and succeeding chapters and focuses on establishing the conceptual foundations for ensuing chapters. These are summarized as follows:

- Chapter 1 is a high-level introduction to the problem of modeling system-level decision alternatives and the aims of this research. It contains a vision for how designers can model system-level decision alternatives quantitatively as well as a description of the key research questions addressed in this investigation.
- Chapter 2 is an exploration of the problem background in greater detail. The conceptual foundations for system-level decision making are identified and it is argued that the approach being studied is reasonable in light of these foundations. The chapter also contains a survey of the related literature, which has a focus on the limitations of existing approaches in light of the characteristics of the problem and the desired solution characteristics (identified in Section 1.2).

The focus next shifts to the problem of generating tradeoff models. This group consists of two chapters that deal with foundational and practical issues relating to how to generate a tradeoff model of a component from data about the component.

- Chapter 3 includes the basic definition for parameterized Pareto dominance and tradeoff modeling. A general methodology for generating tradeoff models is described and the formulation of basic decisions (involving a single tradeoff model) is demonstrated on a gearbox design problem. The potential consequences

of not performing dominance analysis prior to model fitting are discussed and it is concluded that dominance analysis is worthwhile.

- Chapter 4 focuses on the practical problem of describing the valid domain of a tradeoff model. Existing theory for domain modeling and clustering are reviewed and a methodology for creating domain described tradeoff models is presented.

Having tackled some basic issues about tradeoff model generation, the next group of chapters focus on validating the use of tradeoff models to compose system-level models and make system-level decisions. This is done using both theory and demonstration.

- Chapter 5 is a theoretical analysis of the composition problem. It contains the general formulation of system-level decisions using composed tradeoff models. It also contains several original mathematical statements regarding the conditions under which it is theoretically valid to compose tradeoff models.
- Chapter 6 is an engineering example of performing requirements allocation for a hydraulic log splitter system. Tradeoff models are fit to data about commercially-available components. These are composed to model the system and requirement allocation is performed by searching the component-level attribute spaces. The results compare favorably to an exhaustive search of the components database.
- Chapter 7 is an engineering example of performing system architecture selection for a hydraulic hybrid vehicle system. The problem involves multiple system architectures, each of which is modeled by composing tradeoff models of the underlying components.

The final two chapters are not part of a larger theme group. The first covers the problem of how to deal with uncertain data:

- Chapter 8 is a discussion of the foundations of generating tradeoff models under data uncertainty. The conclusion is that doing so is difficult in general but is straightforward in certain special cases, one of which is presented. A critical insight is that one no longer can use Pareto-based dominance criteria to identify an efficient set and instead must use a stochastic dominance criterion.

The final chapter brings closure to the research:

- Chapter 9 is a description of the contributions and limitations of this research. It also includes a discussion of open questions raised during the course of this research.

There also is an appendix worth noting:

- Appendix A contains proofs of the mathematical statements made elsewhere in this document. The proofs are collected here so that they do not disrupt the flow of the other chapters.

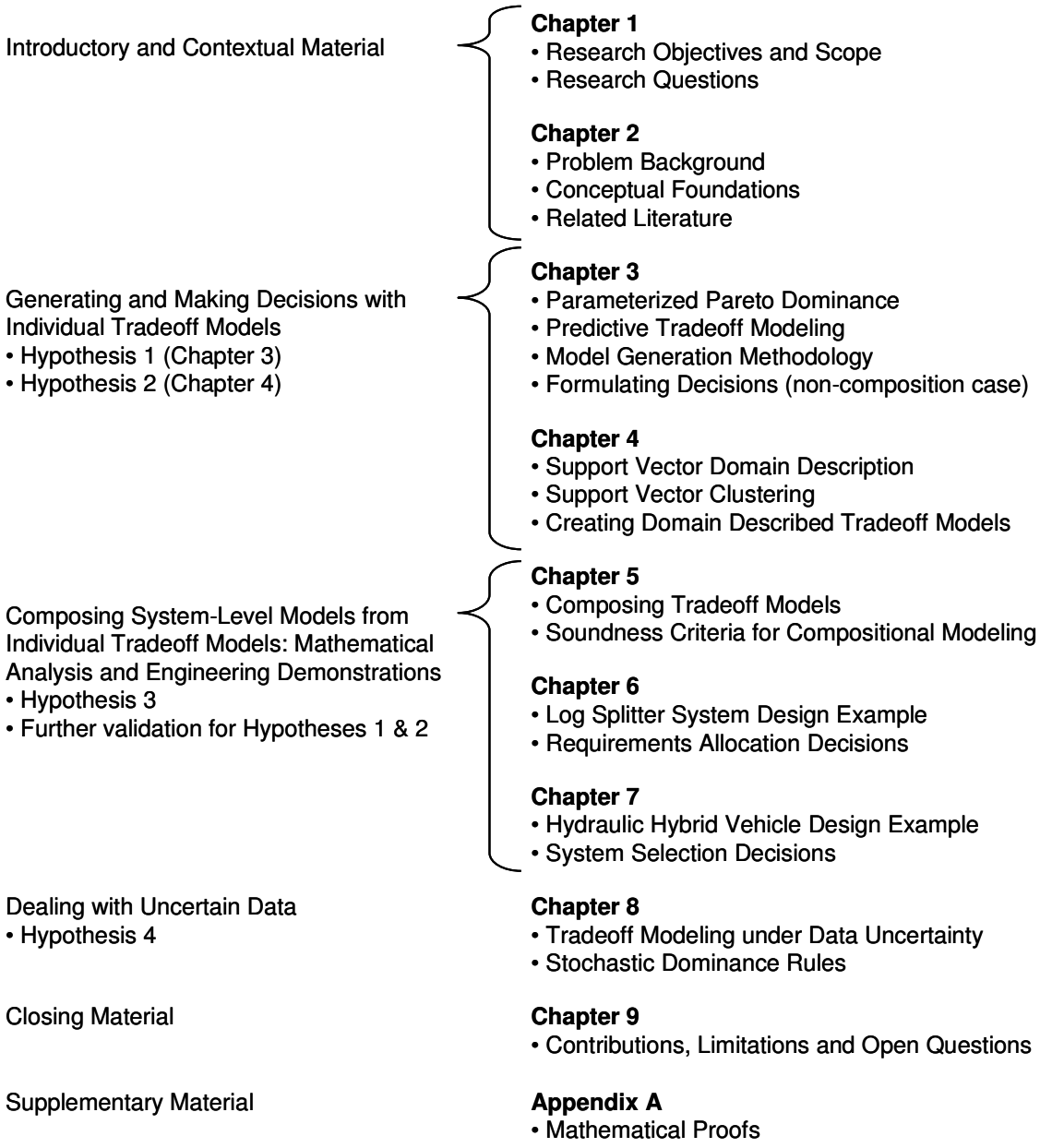


Figure 1.6: Overview of content with major theme groupings indicated.

CHAPTER 2:

PROBLEM BACKGROUND

A basic premise of this research is that one cannot understand system-level decisions properly without considering how they relate to other decisions in a design process. This chapter is a review and synthesis of the prior thinking on this relationship and an examination of the limitations of current approaches to decision making in light of this assertion.

Section 2.1 is a review of the literature on interrelated decisions in a systems realization process. Several authors conceive of a systems realization process as a chain of related decisions and describe various related problems, such as conflict resolution and decision sequencing. However, few consider the problem of modeling system-level decision alternatives in a decision chain context and the approaches that are described in the literature have limitations from a practical perspective. Section 2.2 is a survey of other approaches to decision making that designers sometimes apply to system-level decisions. Most of these methods are useful within a particular context, but are not good general approaches to system-level decision making.

2.1 Systems Realization Processes as Decision Chains

Many in the design research community recognize decision making as a central aspect of engineering design (Thurston 1991, Mistree, et al. 1993b, Hazelrigg 1998, Lewis, et al. 2006). The principles of how to model and solve an individual, isolated decision are well understood. However, most design problems are too complex for designers to solve directly as a single decision problem. Instead, designers simplify the

search for a satisfactory design by decomposing the problem into a series of related decisions. Several authors recognize and discuss the implications of this decomposition and the consensus is that a design process—or, more broadly, a systems realization process—consists of many interrelated decisions that occur in time. However, there is no consensus on how to model this chain of decisions mathematically.

2.1.1 Relationships among Decisions in a Decision Chain

Mistree and coauthors (1990) describe a design process as consisting of many interrelated decisions. They characterize the inception of a design process as the point at which designers begin establishing a hierarchicy of "decision entities" from the naturally heterarchical description of a design problem. They also describe an approach for identifying and formulating such decision entities. However, they do not address the problem of modeling how decisions that designers will make later in a process affect those that come earlier. Herrmann and Schmidt (2006) also view a systems realization process as consisting of numerous interrelated decisions and attribute this to a need to decompose a large problem into manageable sub-problems. They explore the question of whether such a process can be rational (in the sense of profit maximization; they argue in the affirmative) and offer insights into the complexities of information flow in a product development organization. However, they do not address the issue of how to model chains of decisions mathematically. Donndelinger (2006) identifies that the common "series of tasks" and "series of decisions" views of design are both valid and consistent with one another. He discusses how a series of decisions leads to iteration and distinguishes between two types of iteration. However, he offers no specific

recommendations about how to formulate the decisions or model the outcome of a particular chain of decisions.

Figure 2.1 is a simple example of how a decision maker might decompose a larger decision into a chain of smaller ones. Figure 2.1(a) is a depiction of a "one-shot" decision in which a decision maker considers all possible alternatives at once. Although tractable for very small problems, this is unrealistic for systems realization problems. Figure 2.1(b) is a decision tree depiction of one possible decomposition of the "one-shot" decision into a chain of related decisions. The first decision is about which power train to use and the second is between power storage technologies.

This example is simple for conceptual reasons. The need for decision chains is more apparent when one considers that a systems realization process can involve design, manufacturing, supply chains and other concerns. Imagine modeling a decision problem that encapsulates the design of a car, its manufacturing system and the associated supply chain. In this broader context, it is difficult for one to evaluate the consequences of a decision alternative without considering the chain of decisions that will follow from it. For example, a particular choice of material may appear favorable from a an engineering standpoint (e.g., it is strong and lightweight) but this choice is poor when one considers the implications on manufacturing, supply chains or other enterprise concerns (e.g., it may be difficult to machine, expensive or difficult to acquire in sufficient quantities, or unfavorable in light of end-of-life considerations).

This same consideration underlies the "Design for X" and concurrent engineering perspectives (e.g., see (Huang 1996, Prasad 1996)). These perspectives have led to many

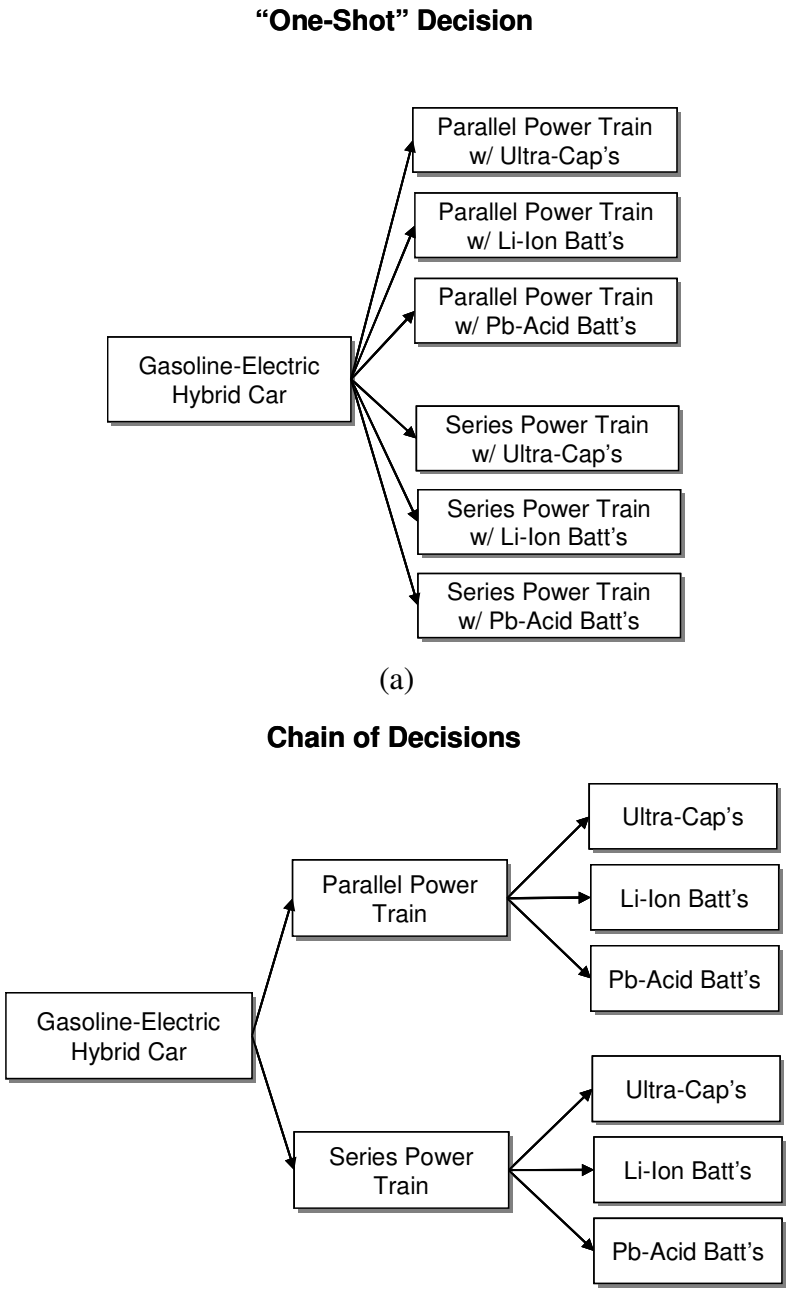


Figure 2.1: The same decision modeled as: (a) a "one-shot" decision, (b) a chain of decisions.

contributions to the theory and practice of systems realization. However, they have not resulted in a comprehensive understanding of how to model individual decisions in a decision chain context. This literature tends to have an emphasis on managing dependencies between decisions and tasks as opposed to how to make an individual decision or perform an individual task.

Several authors have investigated the problem of modeling information flow among a series of related tasks and/or decisions. The Design Structure Matrix (DSM), originally described by Steward (1981), is a common representation for this type of information. Although several extensions of the DSM representation exist, the most basic form involves a symmetric binary matrix that indicates whether two tasks share information. Researchers have applied the DSM and its extensions to problems ranging from task scheduling to system dependency analyses (see (Browning 2001) for a survey). Although this perspective acknowledges that a systems realization process consists of a chain of related decisions, it deals with these relationships at a very high level of abstraction. Moreover, one can use a DSM to sequence a group of decisions, but it provides no support for making individual decisions.

2.1.2 Modeling Decision Alternatives in a Decision Chain

Relatively little literature exists in which the authors consider how to model a decision alternative in the context of a decision chain. There exist four different views on the subject:

- Designers should model downstream decisions explicitly and use simulation to evaluate the outcomes associated with their decision alternatives.

- Designers should model decision alternatives in a set-based fashion and delay making a decision until progressing further downstream, focusing on eliminating demonstrably inferior alternatives from the set in the meantime.
- Designers should model the decision alternatives as they would in an isolated decision and then treat downstream decisions as a source of uncertainty.
- Designers should model a decision alternative by abstracting from data about similar systems.

Explicit Modeling of Downstream Decisions

Barton and coauthors take a direct approach by modeling and simulating an entire business enterprise (Barton and Love 2000, Barton, et al. 2001). They refer to this as a "whole business simulator," and account for all relevant design, manufacturing and business decisions occurring within the scope of the decision chain. Using this approach, designers can evaluate consequences of any decision alternative at any point in a decision chain.

Although this approach is logically correct, it is questionable from a practical perspective. To construct and validate a model of an entire business enterprise would require significant resources and its upkeep would be burdensome. Also, the simulation, unless using very abstract models, is likely to be computationally intense. It is undesirable to include such a simulation into design space exploration and optimization routines.

Delaying Decisions in Set-based Design

Set-based design is an approach in which one focuses on eliminating inferior implementations from a set of alternatives rather than selecting the one that is most preferred. Ward (1989) originally introduced the idea as a means for dealing with

imprecisely defined design specifications and environmental conditions. Under his approach, designers can rule out types of components or large intervals of component attribute values while not having to commit to a final specification until later. An advantage of this is that designers can make some progress in a design process while awaiting information from decisions that normally would come later in a design process. These ideas form the conceptual basis for describing the design practices of Toyota Motor Company (c.f. Section 1.1.2).

Set-based design has a clear relationship to the decision chain perspective. Consider again the decision chain in Figure 2.1(b). Under a set-based approach, one would consider the design concept "gas-electric hybrid car" to be a set of potential implementations, each of which is represented as a unique path from the root to a leaf. Thus, one implementation would be "parallel power train + ultra capacitors."[b] Designers reason about a set-based design concept by eliminating specific implementations from the set.

Ward focuses on elimination methods based on interval-based constraint propagation. Finch and Ward (1997) extends this to the use of more generally defined sets. Other researchers report computer-based tools that support set-based design (e.g., (Parunak, et al. 1997, Nahm and Ishikawa 2004)). Research also exists on extending the set-based design perspective to include eliminations based on domination (i.e., preference-based) criteria (Rekuc, et al. 2007, Malak, et al. 2008).

The main limitation of set-based design is that it is not a comprehensive approach to decision making—it focuses on the elimination of inferior alternatives, but not the

[b] Strictly speaking, this is a simplification for the sake of brevity. A fully-defined implementation would include more information, such as the storage capacity and other characteristics of the capacitors.

selection of the most preferred alternative. The fundamental problem is that an explicit set-based approach to modeling decision alternatives leads designers to rate a decision alternative with an interval of utility. Designers cannot distinguish between alternatives with overlapping utility intervals without appealing to heuristics. This is known as the problem of indeterminacy (Kyburg and Pittarelli 1996, Rekuc, et al. 2007).

Downstream Decisions as a Source of Uncertainty

In the context of conceptual design, Chen and coauthors adopt the perspective that one cannot model the downstream portion of a decision chain adequately and therefore one should make decisions that are insensitive to the uncertainties involved (Chen, et al. 1996, Chen, et al. 1997). They describe a method in which a designer models uncertainty about the final implementation of a concept explicitly using uniform probability distributions applied to the design parameters and makes decisions using a modified robust design formulation. Thus, designers can make a tradeoff between performance and risk.

The main limitation of this approach is that the uncertainty distribution does not account for the impacts of designer rationality in a systems realization process. The uniform distribution ascribes equal odds to all outcomes in its domain, when in reality there are some combinations of values that no designer would choose. This could lead one to conclude an alternative unduly risky even though it may be the best choice.

Predictive Modeling

Predictive modeling is the practice of formalizing relationships among the various factors in a study (Hand, et al. 2001). A predictive model typically does not have explanatory power or imply causation (Geisser 1982, 1993, Rygielski, et al. 2002). In general, one can fit a predictive model to observational data (as is typical in a data mining

problem) or experimental data (where one has control over the sample locations). Construed broadly, predictive modeling can include response surface modeling based on computer experiments, a practice sometimes called meta-modeling (Simpson, et al. 2001b, Wang and Shan 2007).

In the context of conceptual design, Wood and coauthors utilize observational data about existing systems in a decision-based approach to design exploration (Wood and Agogino 2005, Wood and Dong 2006). They model a concept by generating a "design PDF" from data about existing implementations to serve as beliefs in a utility-theoretic decision framework. They also discuss how designers can use this approach to model concepts at different levels of abstraction (e.g., one could model all motors and then refine the model to represent only AC motors). This approach has some similarity to that of Chen and coauthors in that a designer models a concept in terms of a distribution function. However, Wood and coauthors construct their distribution function directly from data and use a more general decision formulation (utility theory in lieu of robust design).

Although the approach Wood and coauthors describe has some advantages, it is questionable in two key respects. First, they provide no concrete interpretation for the "design PDF" they construct. They suggest a functional form for this distribution with little justification and never define clearly the meaning of "the probability of a point in the design space." One might assume this means the odds that a newly-designed component of that type will be at that point in the design space. However, intuitively, such odds should depend on designer objectives and their willingness to make tradeoffs

among competing objectives. Yet, their "design PDF" is independent of designer preferences.

A second limitation of their approach is that it does not provide for composing systems from component-level models. This is a practical concern. For all but the most common systems, designers are unlikely to have sufficient data for generating probability distributions. Although one could attempt to compose "design PDF" models together, it is unclear whether such an operation would be sound.

Several other authors apply predictive modeling techniques to design decisions, but their motivation is to support traditional engineering design analysis. Rather than modeling a system-level decision alternative abstractly based on data about similar systems, they use the data to establish a relationship between the physical form of the system and one of the system-level attributes. Cost estimation is perhaps the most common application (e.g., see (Dean 1976, Daschbach and Apgar 1988, Farineau, et al. 2001, Seo, et al. 2002, Shabani and Yekta 2006)). Others include power transmissions (Krus 2005) and environmentally benign design (Dewulf 2003). This type of model is useful in the context of traditional engineering analysis and optimization, particularly in cases where it is difficult to develop a model analytically. However, they are formulated in terms of lower-level design details that are inconvenient or inappropriate for system-level decision making.

2.1.3 A Foundation for Modeling System-Level Decision Alternatives

Although the methods reviewed in the preceding section have limitations, they each have roots in a meritorious point of view:

- Barton and coauthors are correct to assert that to make a particular decision in a decision chain, one should consider all of the decisions that will follow it. Without doing so, one cannot understand comprehensively the consequences of choosing a particular decision alternative.
- Ward's notion of set-based design is correct in that system-level decision alternatives relate to a set of specific design implementations. His premise of eliminating demonstrably inferior solutions from consideration also is sensible.
- Chen and coauthors are correct in that downstream decisions are difficult to model explicitly and that designers should consider uncertainty and their risk attitude when engaging in such decisions.
- Wood and coauthors are correct to use observational data about design implementations and predictive modeling techniques in order to model system-level decision alternatives. Using this approach, one can consider the impact of decisions that will occur downstream in a decision chain without having to model the decision processes explicitly.

These perspectives are complementary, rather than contradictory. The modeling approach investigated in this research is based on a global perspective that blends these individual ideas:

- Designers consider the chain of future decisions implicitly using predictive models fit to data about prior implementations of similar systems or obtained from a validated model of the system (that accounts for the relevant enterprise considerations).

- Designers eliminate demonstrably inferior solutions using dominance analysis. This results in a set-based characterization called the efficient set.
- Designers deal with uncertainty in the data using stochastic dominance criteria and an appropriate tradeoff space representation. They consider their risk attitude by formulating their decisions using utility theory.

This is not to suggest that the approach is simply a hybridization of the prior methods. It is formulated with the aim of improving upon the practical and fundamental limitations of prior approaches. This requires significant extensions beyond current methods (c.f. the approach overview in Section 1.3).

2.2 Related Literature on Modeling and Decision Making

The following is a review of other literature related to system-level modeling and decision making. The approaches described here are useful in a particular context, but have limitations in light of the decision chain perspective or lack other desirable characteristics (c.f. Section 1.2). However, this is not to say that all of the following ideas described are fundamentally incompatible with this perspective. For example, one can extend many of the decision-based design and optimization approaches (Section 2.2.3) to incorporate the modeling approach investigated in this research.

2.2.1 Component Sizing Procedures

The engineering product literature contains several examples of what one can characterize as "component sizing procedures." A sizing procedure is a sequence of computational steps through which a designer can identify the appropriate component model number in a particular company's product line. Essentially, these are simplified requirements allocation procedures. The term "sizing" is common because the primary

distinction between products in a single product family often is the magnitude of one or more attributes (e.g., for a family of hydraulic pumps, larger "sizes" would correspond to larger flow rates, masses and outer dimensions).

One can find procedures such as these in the literature associated with many types of engineering components. For example, Sauer-Danfoss publishes an applications manual on how to select and size driveline components (primarily hydraulic pumps and motors) for mobile applications (Sauer-Sundstrand Co. 1997). Given assumptions about the engine, loading characteristics, gearing and design requirements (e.g., lifetime), they define a procedure for determining which of their pumps and/or motors are suitable for the given application. Eaton publishes a similar document for their hydraulic pumps and motors (Eaton Corp. 1998).

The main limitation of component sizing procedures is that they restrict the decisions that designers can make. They require designers to assume values for several key parameters and then have them work backwards to identify which component in a particular company's product line is suited for the task. However, this leaves designers with minimal discretion in making tradeoffs between the performance characteristics and other factors such as cost or service life.

Another limitation is that sizing procedures are effective only in the context of one product line or, at best, one company's products. This is because they generally assume "size" and price (which would be a cost to a systems designer) relate directly and thus there is no tradeoff to be had between the two (i.e., ranking components by price and ranking by "size" yields the same ordering). This assumption commonly holds among the product line of a single company, but may not hold when including products from

different manufacturers. Designers require a general means to deal with the tradeoffs that occur in such situations.

2.2.2 Decision Approaches Involving Qualitative Models

A primary concern in this research is how to model a system-level decision alternative quantitatively. However, this is not the only approach one can take. Decision theory places no restrictions on the form of the model a decision maker uses. It requires only that a decision maker use a model that reflects his or her beliefs about the outcomes that will occur upon choosing an alternative. Consequently, if designers feel confident in their ability to assess the outcomes of their decision alternatives, they can formulate a system-level decision problem using an appropriate variant of utility theory—e.g., the single- and multi-attribute formulations of von Neumann-Morgenstern utility theory or the analogous deterministic formulation (Luce and Raiffa 1957, Fishburn 1965, von Neumann and Morgenstern 1980, Keeney and Raiffa 1993). Note that such an approach would result in quantitative evaluations of the decision alternatives (i.e., numerical utilities), even though the models are the tacit understanding of the designers.

The design literature contains descriptions of several other selection methods that rely on designer expertise to evaluate decision alternatives but that are not utility-based. Pugh selection (Pugh 1991), Quality Function Deployment (Akao 2004), various rating matrix approaches and the analytic hierarchy process (Saaty 1990) are common examples. A debate exists about the suitability of these methods, but this has nothing to do with the use of designer expertise to model the alternatives. The primary concerns revolve around whether the methods are mathematically sound and could lead to self-contradictory decisions (Saari 2000, Hazelrigg 2003, Mullur, et al. 2003). For example,

one can construct simple examples in which Pugh selection leads one to choose the worst alternative rather than the best.

Regardless the debate about certain methods, all decision approaches founded on using mental models to evaluate design alternatives have limitations. Such models are not useful in the context of broad design space exploration or automated search routines, which require many thousands of model evaluations. They also can be inadequate for discriminating between decision alternatives and provide little support for allocation decisions. Although a utility-based decision approach based on designer expertise sometimes is the best one can do, there is ample motivation to research quantitative modeling approaches.

2.2.3 Decision-based Design and Optimization

Much research exists on decision-based design and optimization methods in engineering design. Decision making and optimization methods are closely linked in the design literature because optimization methods are a typical means for solving decision problems. A rich body of literature exists in both areas. Although this work has limitations in the context of system-level decision making, it is best viewed as synergistic with the current research rather than in conflict with it.

Decision-Based Design and Decision Support Problems

Decision-based design (DBD) is the perspective that decision-making is a central activity in a design process and that designers therefore should formulate and solve decision problems in a sound and rational manner (Shupe 1988, Hazelrigg 1998, Thurston 2001, Lewis, et al. 2006). The topic of much of the DBD literature is the tasks of formulating decisions and organizing the information relating to a decision. Muster

and Mistree (1988) describe the Decision Support Problem Technique, which is an approach to formulating design problems in terms of a construct called a Decision Support Problem (DSP). A DSP is a template for structuring various types of decision problems. The most basic types are the compromise DSP (Karandikar, et al. 1989, Mistree, et al. 1993a) and selection DSP (Kuppuraju, et al. 1985). These are summarized in Figure 2.2 and Figure 2.3, respectively. Other types include hierarchical formulations (Shupe, et al. 1987, Bascaran, et al. 1989), decisions under uncertainty (Vadde, et al. 1994), robust design (Bras and Mistree 1993) and utility-based selection (Fernandez, et al. 2005).

Given	A candidate alternative.
Find	• The values of *system variables,* which describe the physical characteristics of an artifact. • The values of *deviation variables*, which indicate the extent to which goals are achieved.
Satisfy	• *System Constraints*: Define what constitutes a feasible combination of system variables. • *System Goals*: Target values for system attributes of interest. • *Bounds*: Lower and upper bounds on system variables.
Minimize	An objective function that quantifies the deviation of system performance from that implied by the goals and their associated priority levels or relative weights.

Figure 2.2: Compromise Decision Support Problem.

Given	A set of feasible alternatives
Identify	The principal attributes influencing selection
Rate	The alternatives with respect to each attribute.
Rank	The feasible alternatives in order of preference based on the attributes and their relative importance.

Figure 2.3: Selection Decision Support Problem.

Although the literature relating to the DSP and DSP Technique offers a rich language with which designers can describe their decision problems, it contributes relatively little to how designers should model the decision alternatives themselves. For example, Step 3 of the utility-based selection DSP is to "specify levels and/or probability distributions for each attribute for each alternative" and the authors presume designers are able to do this easily (Fernandez, et al. 2005). However, a basic premise of the current research is that evaluating the attributes of a partially-defined system-level alternative is a challenging task and that designers require a better approach for doing it. In this sense, the current research is synergistic with the prior work on formulating selection decisions.

A similar gap exists in the context of requirements allocation decisions, which are a subset of compromise decisions. An underlying assumption of the compromise DSP is that designers can formulate their decision problem in terms of system variables that describe the physical characteristics of the system. The typical interpretation of this is that these are physical dimensions, material specifications and other relatively low-level details. However, such details are either unavailable or impractical to work with when making system-level decisions. The approach summarized in Section 1.3 is intended to provide designers a means by which they can make decisions at a higher level while still accounting for the impact of lower-level details through predictive tradeoff models.

The same gap exists in other DBD frameworks. For example, Hazelrigg (1998) advocates a utility-based framework in which one treats a design problem essentially as a large optimization problem. In this framework, designers perform a parametric optimization for each alternative system configuration and then select from them the most

preferred. The parametric optimization is, like the compromise DSP, based on low-level system variables that relate to the physical construction of a system.

The Use of Surrogate Models

The introduction of and advances in surrogate modeling within the DBD community have reduced concerns about dealing with lower levels of abstraction. However, this has not eliminated concern entirely. A surrogate model (sometimes called a meta-model or a reduced-order model) is a computationally simple abstraction of a more complex model. One achieves this by sampling the more complex model and fitting the simpler model to the input-output data. Several examples of surrogate modeling exist in the DBD community. Pacheco and coauthors (2001) demonstrate the use of Bayesian techniques to produce surrogate models for a heat transfer problem. Simpson and coauthors (2001a) investigate the use of Kriging models as surrogates in the context of multidisciplinary design optimization and Martin and Simpson (2006) apply Kriging models of subsystems to estimate system-level uncertainties. There exist several surveys of surrogate modeling techniques in a DBD context (Simpson, et al. 2001b, Jin, et al. 2003, Wang and Shan 2007).

The advantage of surrogate models is that designers can explore the space of system variables more quickly using the simpler model. However, a surrogate model retains an input-output structure similar to that of the more complex model—i.e., inputs are physical descriptions of the artifact and the output consists of one or more attributes of interest from a decision-making perspective. Consequently, designers still must formulate their decisions in terms of the lower-level system variables. This can be burdensome when designers want to consider multiple heterogeneous implementations of different components.

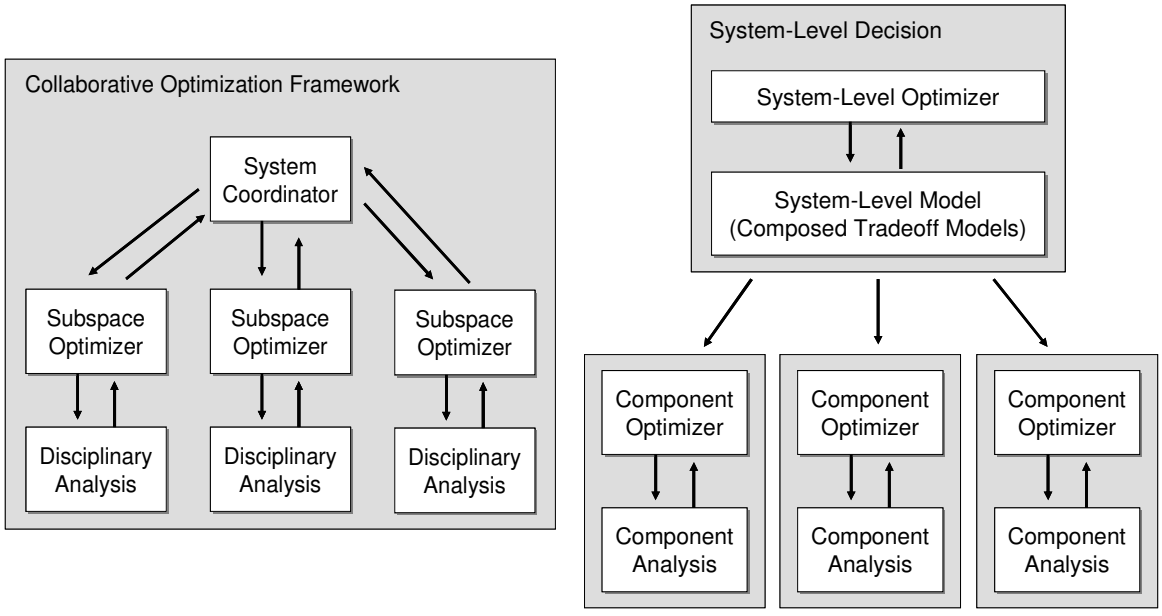
For example, Chapter 3 includes a design problem involving different concepts for a fixed-ratio gearbox. Although these all have the same interface to the rest of the system and the decision problem, they have very different physical descriptions. Using the modeling approach summarized in Section 1.3, designers can abstract several heterogeneous implementations into a single predictive tradeoff model and solve the problem using a single optimization run. However, if using surrogate models for each concept, they would require an optimization run for each concept. The situation becomes worse when designers consider different concepts for multiple components due to the combinatorial effect.

Parametric Optimization in Systems Design

An extensive body of literature exists on applying optimization in the context of design. However, of particular interest in this review are the approaches that are intended explicitly for system-wide optimization. These approaches enable designers to apply optimization methods to a well-defined system efficiently. They typically operate by introducing auxiliary variables that enable designers to partially decompose the search problem. This yields a number of smaller, coordinated search problems. Different formulations of this general strategy are useful depending upon the desired decomposition. Collaborative optimization (CO) methods are based on decomposing the system according to various analyses (i.e., different computational models) (Alexandrov and Lewis 1999, Kroo and Manning 2000, Gu, et al. 2002). Multidisciplinary design optimization (MDO) methods are based on decomposing the system analysis according to various analysis disciplines (e.g., statics, dynamics, thermal) (Cramer, et al. 1994, Sobieszczanski-Sobieski and Haftka 1997). Analytical target cascading (ATC) involves a

system-subsystem hierarchical decomposition (Kim, et al. 2003, Michelena, et al. 2003, Kokkolaras, et al. 2004).

Although these methods clearly are system-wide in scope, they actually operate at a relatively low level of abstraction compared to the decisions considered in this research. Figure 2.4 is an illustration of how a typical collaborative optimization framework (Figure 2.4(a), which is adapted from (Kroo and Manning 2000)) compares to the decision framework investigated in the current research (Figure 2.4(b)). The CO framework involves several coordinated optimization sub-problems, and designers formulate each of these sub-problems in terms of low-level design details—i.e., what are called system variables in the DSP literature. Due to system-level interactions and decision objectives, designers may have to re-execute the lower-level optimization



(a) (b)

Figure 2.4: Distinction between (a) a CO framework and (b) system-level decision making.

problems many times. In this sense, a CO framework is a closed-loop structure. An ATC framework operates on a similar principle—it is a closed-loop decomposition of a larger optimization problem—but differs in the structure of the decomposition.

In contrast with collaborative optimization and related approaches, the system-level decision making procedure investigated in this research is open loop. Rather than coordinate several lower-level problems interactively, designers use a predictive procedure to identify specifications that are communicated to lower-level problem solvers. The idea is that designers can use a system-level model composed from component-level tradeoff models to predict the most preferred component-level attributes accurately. Designers at the component level can use these predictions as design-to requirements when developing the individual components (this part of the problem is beyond the scope of the current research). Assuming these predictions are reasonably accurate, the need additional coordination among the sub-problems is minimized. Furthermore, abstracting the system in this way allows designer to make system-level decisions in the absence of the disciplinary analysis models required in the CO framework.

Ultimately, one should not view the system-level decision making perspective strictly as competing with the CO, MDO and ATC literature. Opportunities exist for combining both approaches. For example, one can apply any of those approaches at the lower levels of the procedure illustrated in Figure 2.4(b) (e.g., the “component” may be a large subsystem that would benefit from a multi-level approach to optimization).

2.2.4 Tradeoff Visualization in Design

Several authors investigate the use of efficient set information in the context of design decision making. These approaches typically require designers to model a system using a an engineering analysis model which they sample at different settings of the design variables. The model outputs are system-level attribute vectors, which the designer filters using the classical Pareto dominance criterion. The resulting efficient set is the basis for visualization and tradeoff analysis.

Representing Concepts with Efficient Sets

Mattson and Messac (2003) investigate a method for conceptual design in which a designer represents each concept using a different efficient set and makes decisions by comparing the sets. Ulrich (2005) applies a similar method to evaluate technology options in the context of mobility scooters. Both Gurnani and coauthors (2006) and Ferguson and coauthors (2005) use an efficient set to abstract more complex engineering models in order to improve preliminary design efforts. They use a model fit to efficient set data in order to speed computation during design exploration and to ensure the solution will be technically feasible. Thus, their approach has similarities with the one investigated in the current research. However, they rely on a different dominance criterion and do not perform model composition.

Although these approaches are useful in specific situations, they are poor general solutions to the problem of making system-level decisions. They require that designers already have a validated model of the entire system in question or have access to ample observational data about implementations of similar systems. This may be the case for systems that are simple or common, but these assumptions typically are not met for

systems that are complex or novel. Model composition is not a viable solution under these approaches due to their reliance on classical Pareto dominance (see Chapter 3).

Visual Decision Methods

Some authors use efficient sets in concert with visualization tools to support decision making and design exploration. Balling (1999) advocates making design decisions by visualizing the efficient set and choosing a solution based on the visual information (as opposed for formalizing a utility function). He refers to this as "design by shopping" since the selection method is analogous to how one might shop at a store. One also can find similar arguments in favor of making decisions based on visualizations of the efficient set from outside the design community (Lotov, et al. 2004). Other work on tradeoff visualization includes methods to steer optimization (Winer and Bloebaum 2002) and design space exploration (Yukish, et al. 2007).

One objection to visual decision making is that it should not matter. That is, a decision maker should have preferences that exist independent of the available alternatives. The proponents of visual decision methods would counter that human decision makers do not behave in such an idealized manner and that this information really is relevant.

Philosophical disagreements aside, there are practical limitations to visual decision methods. First, they suffer the same drawbacks as the other tradeoff modeling approaches—they require preexisting models of the system or ample data about prior implementations of similar systems. Secondly, and perhaps more problematically, the visual decision methods do not scale beyond a handful of decision attributes. Problems involving two or three attributes are straightforward and advanced visualization techniques exist that push the practical limit up to perhaps five or six (e.g., see the tool

described in (Stump, et al. 2004)). However, visualization-based decision methods are impractical beyond this point.

Dealing with Uncertainty

Another limitation of the preceding modeling approaches is that they include no means by which designers can consider uncertainty in their decisions. These methods rely on the classical Pareto dominance criterion, which is an inherently deterministic construct.

Mattson and Messac (2005) extend their concept modeling approach to support decisions involving uncertainty. However, their approach remains rooted in the Pareto domination construct. In their formulation, one constructs a Pareto-like frontier in the space of the attribute means; variation tends to "shift" the location of this frontier. Moreover, they treat risk essentially by incorporating a safety factor that is related to variance. They include no means by which designers can consider tradeoffs between, say, average performance and risk.

2.3 Conclusions and Chapter Summary

Based on this review of the literature, one can conclude that a gap exists between the accepted fundamental understanding of system-level decision making and the practice of modeling system-level decision alternatives. The consensus is that a systems realization process consists of a chain or interrelated decisions and that designers should consider the consequences of a decision alternative in light of the decisions that will follow it. However, prior approaches to modeling system-level decision alternatives have significant limitations. Thus, there is motivation to study new approaches and techniques.

One could argue that the system-level decision making situation is not all that dire. After all, designers clearly make such decisions in practice and quite often realize a

system that operates as desired. Designers routinely apply sizing procedures, methods based on qualitative models, system-level parametric optimization and tradeoff visualization to design problems and it would be foolish to dismiss these approaches outright. However, none of them are good general solutions for system-level decision making as it is defined in this research. They lack a quantitative basis or require a well-defined system. Therefore, they are inappropriate for design exploration over a space of partially-defined decision alternatives.

CHAPTER 3:

PARAMETERIZED PARETO DOMINANCE AND PREDICTIVE TRADEOFF MODELING

The preceding chapters are about the motivation and vision for this research. This chapter is the first step toward formalizing and studying the modeling approach introduced in Chapter 1. This approach is based on the idea of composing a system-level model from predictive tradeoff models of its components. The tradeoff models enable designers to predict the component-level attributes they would achieve should they implement the system according to their preferences. The topic for this chapter is how designers can generate these component-level predictive tradeoff models.

A the core of this chapter is a new decision-theoretic construct called *parameterized Pareto dominance*. This is an extension of the classical Pareto dominance criterion and is instrumental in the generation of reusable component-level tradeoff models. Recall Research Question 1 from Chapter 1:

RQ1. How can designers conclude that one implementation of a component dominates another when they lack specific knowledge of the system in which the component will be used?

Without an answer to this question, designers have no basis for generating reusable component-level tradeoff models. Parameterized Pareto dominance is a critical part of the hypothesized answer:

H1. Designers can use the parameterized Pareto dominance rule to eliminate attribute data about dominated implementations of a component.

This chapter contains an explanation of the limitations of classical Pareto dominance, a formal definition of the new dominance rule, proof that it is mathematically sound and a demonstration that is an appropriate basis for tradeoff modeling. The demonstration is a gearbox design problem that involves both concept selection and requirements allocation decisions. Tradeoff models are generated from parameterized efficient set data and used during decision making. The decision results compare favorably with results obtained using classical optimization methods.

Chapter organization is as follows. Section 3.1 includes an explanation of dominance analysis, formal definitions and properties of the classical and parameterized Pareto rules, and an explanation of the shortcomings of the classical rule. Section 3.2 is a description of a tradeoff modeling methodology based on the use of parameterized Pareto dominance. Section 3.3 is an explanation of how to formulate decision problems using a tradeoff model. Section 3.4 contains the gearbox design problem demonstration. Section 3.5 addresses the question of whether it is necessary, as a practical matter, to perform dominance analysis. This question is answered in the affirmative.

3.1 Dominance Analysis

A dominance criterion is a mathematical test that indicates whether one alternative is assured of being more preferred to another based on limited information about a decision maker's preferences on a multi-attribute decision problem. The significance is that it allows a decision maker to draw conclusions in situations when formalizing preferences precisely. Dominance analysis is the practice of applying a dominance criterion.

The classical Pareto criterion is by far the most widely used of such tests, but others exist and each requires different assumptions about the structure or characteristics of decision maker preferences. For example, Yu (1974) describes a cone-based extension of Pareto dominance that Hunt and coauthors (2007) apply to problems in which a decision maker knows something about the relative importance of different decision attributes. Other dominance criteria are appropriate for problems in which the data is uncertain. These are reviewed in Chapter 8, which is an expansion of the ideas in this chapter to deal with uncertainty.

3.1.1 Multi-Attribute Decisions

Before defining any dominance criteria, it is necessary to establish some notation regarding multi-attribute decisions. For decisions under certainty, one can state a multi-attribute decision problem as (Keeney and Raiffa 1993, Marler and Arora 2004):

$$\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathbb{R}^P} \left(V\left(\mathbf{F}\left(\mathbf{x}\right)\right)\right)$$

subject to

$$\mathbf{g}\left(\mathbf{x}\right) \le \mathbf{0}$$
$$\mathbf{h}\left(\mathbf{x}\right) = \mathbf{0}$$

where $\mathbf{x}\in\mathbb{R}^P$ is a vector of design variables; $\mathbf{F}\left(\mathbf{x}\right) = \left[F_1\left(\mathbf{x}\right), F_2\left(\mathbf{x}\right), \ldots, F_N\left(\mathbf{x}\right)\right]$, such that $F_i\left(\cdot\right): \mathbb{R}^P \to \mathbb{R}$ for $i = 1\ldots N$, is a vector-valued design model that relates design variables to decision attributes (also called criteria and denoted $z_i = F_i\left(\mathbf{x}\right)$ for $i = 1\ldots N$); $\mathbf{g}\left(\cdot\right)$ and $\mathbf{h}\left(\cdot\right)$ are vector-valued inequality and equality constraints, respectively; $V: \mathbb{R}^N \to \mathbb{R}$ is a formalization of a decision maker's preferences for making tradeoffs among the competing objectives; and $\mathbf{x}^*$ is the most preferred design vector. For

notational brevity, let $\mathbf{X}$ denote the set of feasible design vectors as indicated by the equality and inequality constraints. Thus, one can restate the above as:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathbf{X}} V\left(\mathbf{F}\left(\mathbf{x}\right)\right). \tag{3.1}$$

The function $V(\cdot)$ is known variously as an objective function, aggregation function, value function or utility function. There are many approaches for formalizing $V(\cdot)$. According to multi-attribute value theory, a decision maker elicits $V(\cdot)$ to be sound with respect to stated decision preferences (i.e., such that if a decision maker prefers the consequences of $\mathbf{x}'$ to $\mathbf{x}''$, then $V\left(\mathbf{F}\left(\mathbf{x}'\right)\right) > V\left(\mathbf{F}\left(\mathbf{x}''\right)\right)$; see (Fishburn 1965, Keeney and Raiffa 1993) for elicitation procedures). Researchers also describe various means for defining value functions that are not based on the elicitation of decision-maker preferences (e.g., see the surveys of (Otto and Antonsson 1991, Marler and Arora 2004)). The results of this research hold regardless of how one determines $V(\cdot)$.

3.1.2 Classical Pareto Dominance

The classical Pareto dominance criterion is based on the notion of market efficiency described in the early 20th century by the economist Vilfredo Pareto (1971). In that context, one market situation dominates another if it makes at least one person better off without making anyone worse off. Transformed into engineering terms, one decision alternative dominates another if it is better in at least one attribute and no worse in any other.

Suppose a decision maker defines his or her value function, $V(\cdot)$, such that it is monotonically increasing in each decision attribute, z_i for $i = 1 \ldots N$. Since larger values

of $V(\cdot)$ are more preferred (implied by the maximization operator in Equation (3.1)), "better" in this case means larger attribute values. Let $\mathbf{z} = [z_1, z_2, \ldots, z_N]$ denote a vector of attributes and $\mathbf{Z}$ denote the set of achievable attribute vectors. This leads to the following definition:

Definition 3.1 (Classical Pareto Dominance): An alternative with attribute vector $\mathbf{z}'' \in \mathbf{Z}$ *is said to be Pareto dominated by one with attribute vector* $\mathbf{z}' \in \mathbf{Z}$ *if and only if* $z_i' \geq z_i'' \quad \forall i = 1 \ldots N$ *and* $z_i' > z_i'' \quad \exists i = 1 \ldots N$.

In many practical situations, a decision maker can formulate problems such that the monotonicity condition on $V(\cdot)$ holds by defining the top-level attributes (i.e., the z_i) appropriately. Thus, the significance of classical Pareto dominance in this context is that a dominated alternative cannot be the most preferred solution. This is summarized in the following theorem. Let $\mathbf{z}' \,\mathrm{DOM}\, \mathbf{z}''$ denote that $\mathbf{z}'$ dominates $\mathbf{z}''$ according to Definition 1. Then one has (see Appendix A for the proof):

Theorem 3.1: If a value function, $V(\mathbf{z} = \mathbf{F}(\mathbf{x}))$, *is monotonically increasing in each attribute and* $\mathbf{z}' \,\mathrm{DOM}\, \mathbf{z}''$ *for two alternatives with attribute vectors* $\mathbf{z}', \mathbf{z}'' \in \mathbf{Z}$, *then* $V(\mathbf{z}') > V(\mathbf{z}'')$.

Some authors develop classical Pareto dominance from a slightly different perspective, but the preceding is equally valid. For example, Marler and Arora (2004) describe the Pareto set as the set of possible solutions to the following[c]:

[c] Many authors, including the work cited, express this using minimization. However, we use maximization without loss of generality in order to maintain consistency with Equation (3.1).

$$\max_{\mathbf{x}\in\mathbf{X}} \left(\mathbf{F}(\mathbf{x}) = \left[F_1(\mathbf{x}), F_2(\mathbf{x}), \ldots, F_N(\mathbf{x}) \right] \right), \tag{3.2}$$

where the terms are defined as above and $z_i = F_i(\mathbf{x})$ for $i = 1 \ldots N$. One interprets this as having a set of solutions—i.e., the efficient set—from which a decision maker will make a final selection using some indeterminate method. This is analogous to assuming that a decision maker will choose using a monotonic value function, but has not yet elicited it precisely.

3.1.3 Limitations of Classical Pareto Dominance

In the current context—of using efficient sets or tradeoff models as an abstract representation for a design concept or system architecture—the crucial limitation of classical Pareto dominance lies in what a decision maker must do to ensure that their value function is monotonic in every attribute. As noted in the previous section, it often is possible for a decision maker to reformulate the attributes of a problem such that preference increases monotonically with each. However, this reformulation almost always is problem-specific, which creates challenges when one wishes to use efficient sets as a basis for a general and reusable representation.

In order to promote reusability of these models, designers must formulate them in terms of relatively generic attributes that pertain more to the system being modeled than to any particular decision about it. This is analogous to how design catalogs list parts using general characteristics of the part; designers must relate these to their specific decision problem. For example, one might describe hydraulic pumps in terms of their displacements, maximum operating pressures, mechanical and volumetric efficiencies and masses. However, a designer would formulate a decision involving hydraulic pumps

in terms of design objectives. For an excavator, these might include maximizing the load it can lift, its operating speed or fuel efficiency.

The problem with classical Pareto dominance is that it often is unclear how to apply it independently of a particular decision problem. Suppose designers wish to abstract gearbox concepts using tradeoff models. Attributes relevant to gearboxes might include mass, mechanical efficiency, reliability, expected lifetime, manufacturing cost and, most importantly, gear ratio. Most of these correspond to monotonic preferences—rarely would a designer not prefer less mass, more reliability, etc.—but gear ratio poses a special problem in that the most preferred ratio is problem-dependent. Because designers cannot establish a problem-independent preference ordering of this attribute, they cannot apply classical Pareto dominance in order to generate a reusable tradeoff model.

This problem can exist even when creating tradeoff models at high levels of abstraction. Suppose designers create a tradeoff model for various implementations of an excavator for use in high-level decision making (defining product families, selecting a system architecture, allocating requirements, etc.). Hazelrigg (1998) argues that designers should formulate decisions as a profit-maximization problem. Taking this approach requires designers to relate their multi-attribute excavator model to profit through one or more models (for manufacturing, supply chain, customer demand, etc.). If they are to apply classical Pareto dominance when modeling the excavator, they must be sure that profit is monotonically increasing in each excavator attribute. Whether this is possible depends on the structure of the profit-computing models. However, it seems unlikely to be the case given the potential complexity of the models involved.

3.1.4 Parameterized Pareto Dominance

Parameterized Pareto dominance is an extension intended to address the limitation of classical Pareto dominance in the context of tradeoff modeling. The name follows from the interpretation of the corresponding efficient set: a parameterized Pareto set consists of multiple classical Pareto sets, any of which one can recover by specifying a vector of parameters.

The new rule requires one to distinguish between two types of attributes: those for which any associated preference ordering is monotonic and those for which it is not. The former category, which are referred to as *dominator attributes*, includes things such as mass, cost, efficiency and reliability—all are characteristics of a system that any designer would prefer strictly more (or less) of, all other factors being equal, and for which the classical Pareto assumptions hold. The latter category, called the *parameter attributes*, include attributes that associate commonly with preferences that are non-monotonic or likely to be influenced in opposing directions by competing objectives. One cannot determine a preference ordering for such attributes without problem-specific information.

To define the parameterized Pareto dominance, one requires notation to account for the two categories of attributes. For a system with attributes indexed 1 through N, let D denote the non-empty set of indices for the dominator attributes and P denote the set of indices corresponding to parameter attributes. Note that $D \cup P = \{1 \ldots N\}$ and $D \cap P = \varnothing$.

Definition 3.2 (Parameterized Pareto Dominance): An alternative having attributes $\mathbf{z}'' \in \mathbf{Z}$ *is parametrically Pareto dominated by one with attributes* $\mathbf{z}' \in \mathbf{Z}$ *if* $z_i' = z_i'' \ \forall i \in P$, $z_i' \geq z_i'' \ \forall i \in D$ *and* $z_i' > z_i'' \ \exists i \in D$.

Notice that this rule essentially applies the classical rule to the dominator attributes provided the parameter attributes are equal; one solution cannot dominate another by this rule if the parameter attributes are not equal. Also notice that the parameterized and classical Pareto dominance rules are equivalent if and only if there are no parameter attributes (i.e., if $P = \varnothing$).

Like the classical rule, the significance of parameterized Pareto dominance in the current context is that decision makers can use it to identify and eliminate alternatives that cannot possibly be their most preferred. Let $\mathbf{z}' \text{PDOM}\, \mathbf{z}''$ denote that $\mathbf{z}'$ parametrically Pareto dominates $\mathbf{z}''$.

Theorem 3.2: If a value function, $V\left(\mathbf{z} = \mathbf{F}\left(\mathbf{x}\right)\right)$*, is monotonically increasing in every dominator attribute and* $\mathbf{z}' \text{PDOM}\, \mathbf{z}''$ *for two alternatives with attribute vectors* $\mathbf{z}', \mathbf{z}'' \in \mathbf{Z}$*, then* $V\left(\mathbf{z}'\right) > V\left(\mathbf{z}''\right)$.

A proof of this statement is provided in Appendix A.

The task of determining which attributes one should model as parameters and which to model as dominators requires some domain expertise. When generating a tradeoff model, designers should consider the types of problems in which they are likely to use it. A survey of these problems should reveal most of the parameter attributes that are relevant. This procedure is demonstrated in the gearbox design problem in Section 3.4 as well as the example problems of Chapter 6 and Chapter 7. However, before tackling design problems it is necessary to discuss how to generate tradeoff models from efficient set data and how to formulate decisions in terms of tradeoff models.

3.2 A Methodology for Generating Tradeoff Models

Designers can create tradeoff models in two ways: from data sources and from other tradeoff models.

3.2.1 Generating Tradeoff Models from Data

Figure 3.1 is a summary of a data-driven methodology for generating tradeoff models. With the exception of the dominance analysis step, this phase is similar to other predictive modeling procedures (e.g., see (Hand, et al. 2001, Kutner, et al. 2005, Witten and Frank 2005)). The main concerns are defining what data to collect, how to validate the data prior to generalization and how best to generalize it into a valid continuous model. What follows is a description of the major steps in this methodology. As with any predictive modeling process, some iteration of the steps may be necessary.

Step 1: Model Planning

In this step, one's objective is to determine precisely what to model. This includes deciding which component to model and which attributes to use when modeling it. The attributes should be properties of the component about which designers typically have preferences (e.g., reliability, technical specifications) or use to compute other attributes for which they have preferences (e.g., cost, used to compute profit). There is no firm rule for what qualifies, but a reasonable heuristic is to include those properties that would be valid descriptors of any possible implementation of the component and avoid ones that are implementation-specific. Also, attributes appearing in spec sheets commonly are good choices since the reason they are listed is because designers tend care about them.

Designers also must classify each attribute as a dominator or a parameter. For the dominators, they also must identify a preference ordering for it (prefer larger values or

Step	Description
1. Model planning	Decide what to model and how to model it. Identify a type of component to model and the attributes designers typically associate with it in a decision-making context. Classify attributes as dominators or parameters.
2. Data collection	Gather data about components that fall within tradeoff model scope. Possible data sources: published datasheets and catalogs, manufacturers and vendors, experimental test data, and mathematical models of a component.
3. Data validation	Verify that data fits tradeoff model scope. Data should appear plausible upon inspection by an expert. Examine for outliers. If necessary, use clustering analysis to re-scope into multiple tradeoff models. Many texts cover the required data analysis methods (e.g., (Hand, et al. 2001, Kutner, et al. 2005, Witten and Frank 2005)).
4. Dominance analysis	Eliminate data points that are dominated by the parameterized Pareto dominance criterion.
5. Domain characterization	Identify valid domain for model inputs to prevent automated search routines from extrapolating too far beyond the data. Often more complex than upper and lower bounds. See Chapter 4.
6. Model fitting	Fit a tradeoff model to the non-dominated data using function approximation (e.g., regression, artificial neural network) or interpolation (e.g., Kriging). Model computes one or more attributes as a function of the others. Choice of inputs and outputs is arbitrary.
7. Model validation	Validate the model fit. For regression models, standard statistical analyses are reasonable. For other function approximation methods (e.g., artificial neural networks) and interpolation methods (e.g., Kriging), the hold-out or cross-validation approaches are more appropriate.

Figure 3.1: Summary of the procedure for generating predictive tradeoff models.

prefer smaller values). Note that these preference orderings assume all other attributes are equal (i.e., for a fixed cost and fuel efficiency, would you prefer more or less power?). One considers tradeoffs between attributes when making a particular decision.

Step 2: Data Collection

Identify potential data sources and gather data. If a validated model of the component exists, one can obtain data by sampling the model. Research exists on improving the sampling and representation of efficient set data (e.g., (Wilson, et al. 2001, Messac and Mattson 2004, Berezkin, et al. 2006, Harada, et al. 2007)). These approaches are intended for use with classical Pareto dominance, but one can extend them to work with parameterized Pareto dominance.

For model generation from observational data, suitable data sources depend on the component of interest. Parts catalogs, product data sheets, product literature and vendor-supplied information all can be useful. One of the main challenges in this setting is to ensure that all data sources are consistent with the attribute definitions. For example, some component pricing data is a function of the purchase quantity. Similarly, two manufacturers may measure or calculate particular attributes differently (e.g. lifetime estimates using different operating assumptions). Designers must do their best to ensure semantic consistency.

Step 3: Data Validation

This step is easily overlooked, but highly important. Designers must validate the correctness and completeness of their data. They should remove from the data set samples for which some attributes are missing. They also should remove duplicates and any obvious outliers. Numerous data mining techniques exist for identifying possible outliers (see (Han and Kamber 2001, Hand, et al. 2001, Kutner, et al. 2005)). Another

opportunity for eliminating outliers exists during the domain description step. However, removing obvious outliers early on simplifies subsequent steps of the process.

Step 4: Dominance Analysis

During dominance analysis, one's objective is to eliminate data about any designs that no rational designer would choose as an implementation for the concept of interest. This is achieved using the parameterized Pareto dominance rule (Definition 3.2). The preceding section is a description of the motivation for and mathematical foundations of this rule.

Step 5: Domain Description

One's objective in this step is to prevent search routines from extrapolating too far beyond the data during decision making phase. A fitted model will return predictions for any inputs values, regardless of whether these predictions are meaningful. Sometimes, one can describe the valid domain of a model using upper and lower bounds on the model inputs. Although these prove sufficient for the gearbox design problem of Section 3.4, this approach often is unreliable. A new, general approach is described in Chapter 4.

An analysis of the domain may reveal multiple clusters of data between which few points exist. In such cases, it may be advisable to fit a different model to each cluster. The potential advantage is improved fitting accuracy compared to one model fit to both clusters. In this case, one applies steps 6 and 7 to each cluster individually. Note that even if fitting a single model to all clusters, the domain may be divided into disjoint regions.

Step 6: Model Fitting

Model fitting involves two critical decisions: (1) choosing inputs and outputs for the model and (2) choosing a mathematical structure for the model. Both are influenced strongly by convenience and effectiveness.

In principle, the only restriction on inputs and outputs is that parameter attributes must be inputs. This is necessary to ensure a functional relationship exists (i.e., avoiding a many-to-one relationship) and to reflect the meaning of a parameterized efficient set (i.e., for a parameterized model, the parameters should be inputs). Beyond this, the choice is a matter of what is convenient for the designer and what yields good predictive accuracy.

The mathematical structure of the generalizing function is a more critical choice. Several alternatives are available, including regression models, artificial neural networks and interpolators. No general rule exists for selecting a model structure, though some approaches may be impractical depending on the data. For example, interpolators may be cumbersome if there are too many data points. However, unless carefully constructed, regression models tend to "average out" key features of a tradeoff relationship. This behavior is desirable for many statistical estimation problems, but not for tradeoff modeling where "elbows" and other irregular relationships often are the critical determinants in the location of the most preferred tradeoff.

A third consideration is whether or not there is an advantage to fit different models to different subsets of the data. This can be the case when multiple clusters of data exist. This relates to the domain description problem. See Section 4.4 for a discussion.

Step 7: Model Validation

Model validation relates closely with model fitting and one may prefer to think of them as one larger step in the process. For regression models, standard statistical analyses are reasonable. See (Kutner, et al. 2005). For other function approximation methods (e.g., artificial neural networks) and interpolation methods (e.g., Kriging), the hold-out or

cross-validation approaches are more appropriate (Han and Kamber 2001, Hand, et al. 2001).

3.2.2 Generating a Model from Other Tradeoff Models

It can be advantageous from a computational perspective for designers to abstract a single tradeoff model from several existing tradeoff models. This is possible when the tradeoff models represent different implementation strategies for a component or system that achieves the same function. For example, designers could combine tradeoff models for four-cylinder, V6 and V8 engines to obtain a generic engine tradeoff model. This also assumes that all the tradeoff models deal with the same attributes, which is somewhat likely since they represent the same functional component.

The computational advantage comes during decision making. Rather than search each tradeoff model independently, designers can search the abstract model. This helps to reduce the combinatorial explosion that can occur in system-level decision making. As explained in Section 3.3, designers are able to identify which of the implementation strategies is most preferred based on the search results. For example, designers could identify that a V6 engine is the most preferred using an abstract engine tradeoff model.

Figure 3.2 is a summary of the procedure for generating an abstract tradeoff model from other tradeoff models. It is similar to the procedure from Figure 3.1. The main difference is the data source prior to dominance analysis. Designers should sample all tradeoff models at the same sites—i.e., same input values. This makes it easier to compare the models during dominance analysis. Design of experiments or other efficient sampling methods are advisable in order to achieve a good fit at a minimum of sample points.

Step	Description
1. Gather Tradeoff Models	Gather tradeoff models for different implementation concepts of the same functional unit. Verify that they all use the same attributes.
2. Sample Tradeoff Models	Sample the tradeoff models at the same sample sites. Design of experiment techniques can be used.
3. Dominance analysis	Eliminate data points that are dominated by the parameterized Pareto dominance criterion.
4. Domain characterization	Identify valid domain for model inputs. This will be union of domains of constituent tradeoff models.
5. Model fitting	Same as Step 6 of Figure 3.1.
6. Model validation	Same as Step 7 of Figure 3.1.

Figure 3.2: Summary of the procedure for generating an abstract tradeoff models from other tradeoff models.

3.3 Formulating Decisions using Tradeoff Models

This section is an explanation of how designers can formulate a design decision in terms of a tradeoff model. This chapter covers requirements allocation and selection decisions, but is limited to the case in which designers represent only one component using a tradeoff model. The formulation for multiple composed tradeoff models is given in Chapter 5.

The discussion that follows is organized into three parts. First is a rudimentary decision formulation in which a decision maker has preferences directly for the component-level attributes (i.e., the component-level attributes and the system-level attributes are equivalent). Next is a formulation in which one accounts for transformations of the component-level attributes into system-level attributes (e.g., computing profit using component costs). The third formulation is a special case in which

a decision maker abstracts several tradeoff models (each representing a different implementation concept) into a single tradeoff model. Doing so has a computational advantage and is possible when the model that transforms component-level attributes to system-level attributes is the same for all of the implementations.

3.3.1 Rudimentary Formulation

Requirements Allocation Decisions

A requirements allocation decision is a straightforward specialization of the general decision problem stated in Equation (3.1). Rather than search over the space of design variables, one searches over the space of component-level attributes as constrained by the tradeoff model.

Let $\mathbf{T}(\cdot)$ denote the tradeoff model associated with a component of interest. Furthermore, let $\tilde{\mathbf{z}}$ denote a vector of inputs to the tradeoff model and $\hat{\mathbf{z}}$ denote the attribute vector output by the tradeoff model[d]. Finally, let $\mathbf{z}$ denote the complete vector of component-level attributes such that $\mathbf{z} = \left[\tilde{\mathbf{z}}, \hat{\mathbf{z}} = \mathbf{T}(\tilde{\mathbf{z}})\right]$. If $\tilde{\mathbf{Z}}$ denotes the set of valid tradeoff model inputs (i.e., it is the set indicated by the domain description), then one can formulate a requirements allocation decision as:

$$\mathbf{z}^* = \left[\tilde{\mathbf{z}}^*, \mathbf{T}(\tilde{\mathbf{z}}^*)\right],$$

where

$$\tilde{\mathbf{z}}^* = \arg\max_{\tilde{\mathbf{z}} \in \tilde{\mathbf{Z}}} V\left(\left[\tilde{\mathbf{z}}, \mathbf{T}(\tilde{\mathbf{z}})\right]\right).$$

[d] Note that a tradeoff model often will output a scalar. However, the decisions formulations are given here in their most general form.

System Selection Decisions

A system selection decision is a generalization of a requirements allocation decision. The approach is to choose the alternative that has the most preferred requirements allocation. In essence, it is a nested search problem.

Suppose there are L distinct alternatives from which designers must choose. Let $l = 1 \ldots L$ be an index denoting each of these alternatives. Thus, $\mathbf{T}_l(\cdot)$ is the tradeoff model corresponding to the l^{th} discrete alternative and $\tilde{\mathbf{Z}}_l$ is the corresponding set of valid model inputs. The most preferred requirements allocation for the l^{th} alternative is

$$\mathbf{z}_l^* = \left[\tilde{\mathbf{z}}_l^*, \mathbf{T}_l\left(\tilde{\mathbf{z}}_l^*\right) \right],$$

where

$$\tilde{\mathbf{z}}_l^* = \arg\max_{\tilde{\mathbf{z}}_l \in \tilde{\mathbf{Z}}_l} V\left(\left[\tilde{\mathbf{z}}_l, \mathbf{T}_l\left(\tilde{\mathbf{z}}_l\right) \right] \right).$$

Thus, the most preferred system alternative is

$$l^* = \arg\max_{l=1\ldots L} V\left(\mathbf{z}_l^*\right).$$

3.3.2 General Formulation

In general, a decision maker may formulate system-level decisions in terms of attributes that are transformations of component-level attributes. One example is a preference to maximize profit, where profit is computed from the component-level attribute of cost. Another example occurs in the gearbox example of the next section. Designers have a preference for maximizing winnings in a race, which they compute as a function of gearbox attributes (gear ratio, cost and reliability). Reformulating the preceding to address this situation is straightforward, but requires some additional notation.

Let $\mathbf{S}_l(\cdot)$ denote a system model for the l^{th} system alternative that computes system-level attributes in terms of component-level attributes. Let $\mathbf{z}$ denote a vector of system-level attributes and $\mathbf{y}_l$ represent a vector of component-level attributes for the l^{th} system alternative. Thus, $\mathbf{z} = \mathbf{S}_l(\mathbf{y}_l)$.

As in the simpler formulation, designers use tradeoff models to search the space of component-level attributes. Let $\tilde{\mathbf{Y}}_l$ denote the set of valid inputs to tradeoff model $\mathbf{T}_l(\cdot)$. Thus, $\mathbf{y}_l = \left[\tilde{\mathbf{y}}_l, \mathbf{T}(\tilde{\mathbf{y}}_l)\right]$ and $\mathbf{z} = \mathbf{S}_l\left(\left[\tilde{\mathbf{y}}_l, \mathbf{T}(\tilde{\mathbf{y}}_l)\right]\right)$. For a particular system alternative, one can predict the most preferred requirements allocation as

$$\mathbf{y}_l^* = \left[\tilde{\mathbf{y}}_l^*, \mathbf{T}_l\left(\tilde{\mathbf{y}}_l^*\right)\right], \tag{3.3}$$

where

$$\tilde{\mathbf{y}}_l^* = \arg\max_{\tilde{\mathbf{y}}_l \in \tilde{\mathbf{Y}}_l} V\left(\mathbf{S}_l\left(\left[\tilde{\mathbf{y}}_l, \mathbf{T}_l(\tilde{\mathbf{y}}_l)\right]\right)\right). \tag{3.4}$$

Given this, the most preferred system alternative is

$$l^* = \arg\max_{l=1\ldots L} V\left(\mathbf{S}_l\left(\mathbf{y}_l^*\right)\right). \tag{3.5}$$

It is worth noting that the system model, $\mathbf{S}_l(\cdot)$, plays a significant role in determining whether it is best to treat a component-level attribute as a dominator or as a parameter. Chapter 5 contains a more thorough discussion of this topic and includes a mathematical analysis that indicates when an attribute is a dominator and it is a parameter.

3.3.3 Formulation using an Abstract Tradeoff Model

Suppose designers of a hydraulic system wish to choose a type of pump and allocate requirements to the system components. Several types of pumps exist—gear

pumps, vane pumps and piston pumps are a few examples—and designers might have a tradeoff model for each type of pump. Provided the system model, $\mathbf{S}(\cdot)$, is the same regardless of the type of pump, the solution procedure from the previous section (Equations (3.3) through (3.5)) is inefficient. It requires that designers solve an optimization problem for each type of pump. They can do better by abstracting the different tradeoff models, the $\mathbf{T}_l(\cdot)$ for $l = 1 \ldots L$, into a single abstract model, $\mathbf{T}_0(\cdot)$. Figure 3.3 is an illustration of the distinction between the two approaches.

The requirements allocation problem is similar to before, with the main difference being that there is only one system model, $\mathbf{S}(\cdot)$, and one tradeoff model, $\mathbf{T}_0(\cdot)$, to consider. Formally, one can state the most preferred allocation as

$$\mathbf{y}^* = \left[\tilde{\mathbf{y}}^*, \mathbf{T}_0\left(\tilde{\mathbf{y}}^*\right)\right], \tag{3.6}$$

where

$$\tilde{\mathbf{y}}^* = \arg\max_{\tilde{\mathbf{y}} \in \tilde{\mathbf{Y}}_0} V\left(\mathbf{S}\left(\left[\tilde{\mathbf{y}}, \mathbf{T}_0\left(\tilde{\mathbf{y}}\right)\right]\right)\right), \tag{3.7}$$

And $\tilde{\mathbf{Y}}_0$ is the domain description for the abstract tradeoff model, $\mathbf{T}_0(\cdot)$. Note that this entails a single search problem over the abstract tradeoff model.

To make a selection decision, designers must determine to which of the source tradeoff models the solution to Equation (3.7) corresponds. Designers can accomplish this by computing each of the source tradeoff models at the solution point. The appropriate decision is to choose the source tradeoff model that most closely matches the solution obtained by solving the abstract model. One can formalize this as

$$l^* = \arg\min_{l \in 1 \ldots L} \left| \mathbf{T}_0\left(\tilde{\mathbf{y}}^*\right) - \mathbf{T}_l\left(\tilde{\mathbf{y}}^*\right) \right|. \tag{3.8}$$

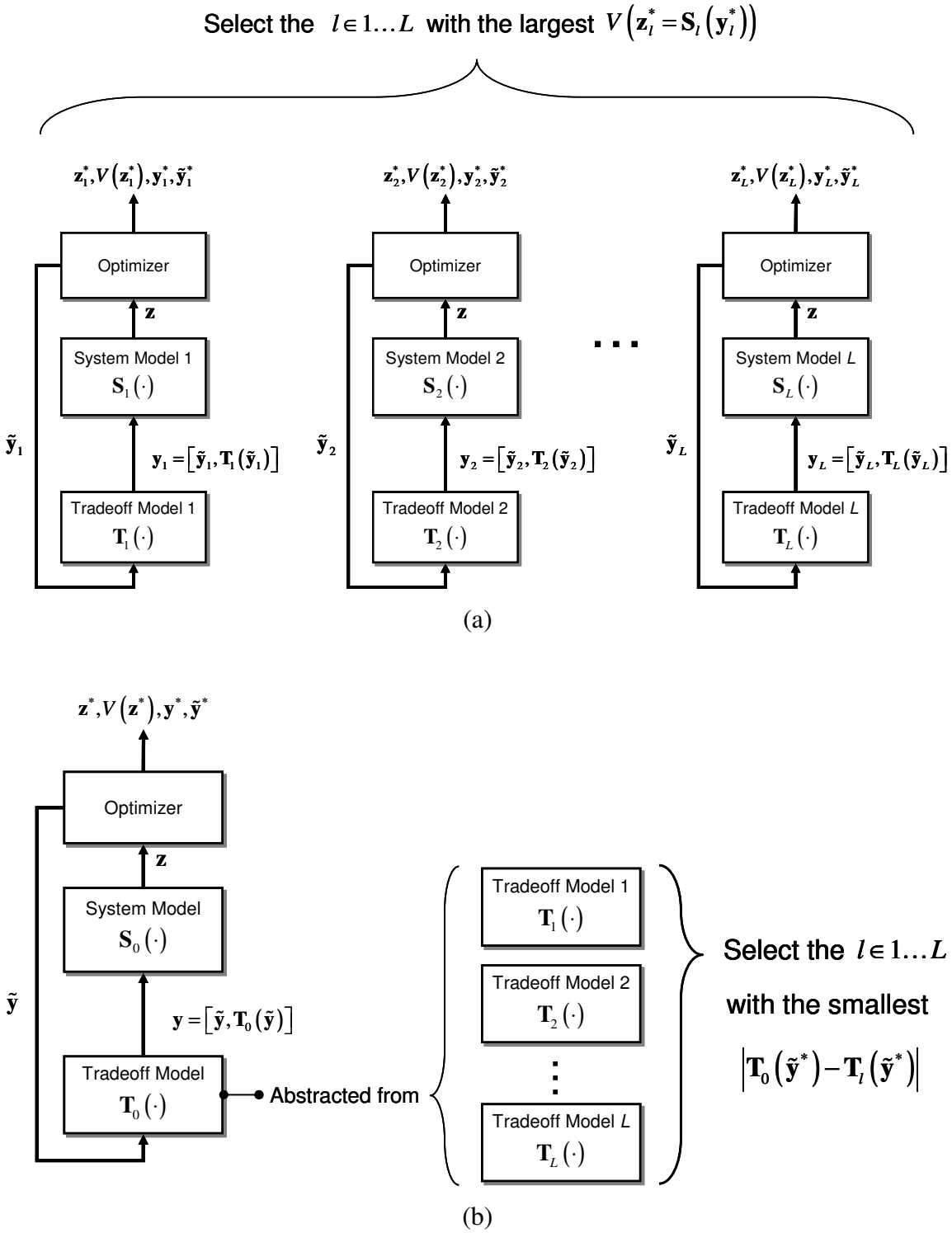


Figure 3.3: Illustration of two procedures for making selection decisions using tradeoff models. Approach (a) involves an independent optimization problem for each tradeoff model; approach (b) requires only one optimization problem, but assumes the system model is the same for all alternatives.

The rationale for Equation (3.8) is that the abstract tradeoff model, $\mathbf{T}_0(\cdot)$, essentially is a piece-wise aggregation of the source tradeoff models. Moreover, the abstract model is equivalent to the source model that is non-dominated at a particular point in the tradeoff space. Consequently, designers can determine the most preferred solution by checking to see which of the source models yield nearly the same prediction as the abstract model.

Note that this formulation assumes all of the source tradeoff models have the same interface—i.e., they have matching inputs and outputs. This should be reasonable for functionally equivalent components. If designers can legitimately replace one type of component with another in a system, then both should be described by the same set of attributes. It is possible that two tradeoff models involve the same attributes, but do not use the same inputs and outputs. In this case, designers can reformulate one of the models to match the other by sampling it and refitting a new relationship to the sample data. This is reasonable since tradeoff models do not imply causation among the inputs and outputs.

3.4 Gearbox Design Problem

This section is a demonstration of applying tradeoff modeling to a design problem. There are three main objectives:

- To illustrate the tradeoff modeling approach, including model generation and decision making.
- To provide evidence that a model fit to parameterized efficient set data can be an effective representation of the capabilities and limitations of a particular design component.

- To demonstrate the computational advantage of abstracting heterogeneous implementations of a functional component into a single tradeoff model.

The topic for this example is a gearbox design problem situated in the context of a small off-road racing vehicle similar to an SAE Mini Baja car[e]. It is an adaptation of a problem originally formulated by Bruns (2006), which involves retrofitting an existing vehicle to include an additional fixed-ratio gearbox. Designer objectives are to maximize profits, which depend on race winnings and fabrication costs. The design problem is to identify the best concept for implementing a gearbox in the vehicle transmission and determining the most preferred specifications for the winning concept. The problem entails both concept selection and requirements allocation decisions using tradeoff models to represent each of three physically heterogeneous gearbox concepts.

To show that the tradeoff models are good representations of the gearbox concepts, the problem also is solved using standard engineering optimization methods. This is tractable due to the simple nature of the problem, but often this option is unavailable to systems designers.

3.4.1 Generating Tradeoff Models for Different Gearbox Configurations

Gearbox Concepts

Preexisting vehicle components constrain the gearbox to have co-axial input and output shafts that rotate in the same direction. Three gearbox concepts are considered. Figure 3.4 is an illustration of the different configurations.

- Planetary Gearbox (PGB): Basic planetary gear system, with input on sun, output on arm and fixed ring. Depicted in Figure 3.4(a).

[e] http://students.sae.org/competitions/bajasae/

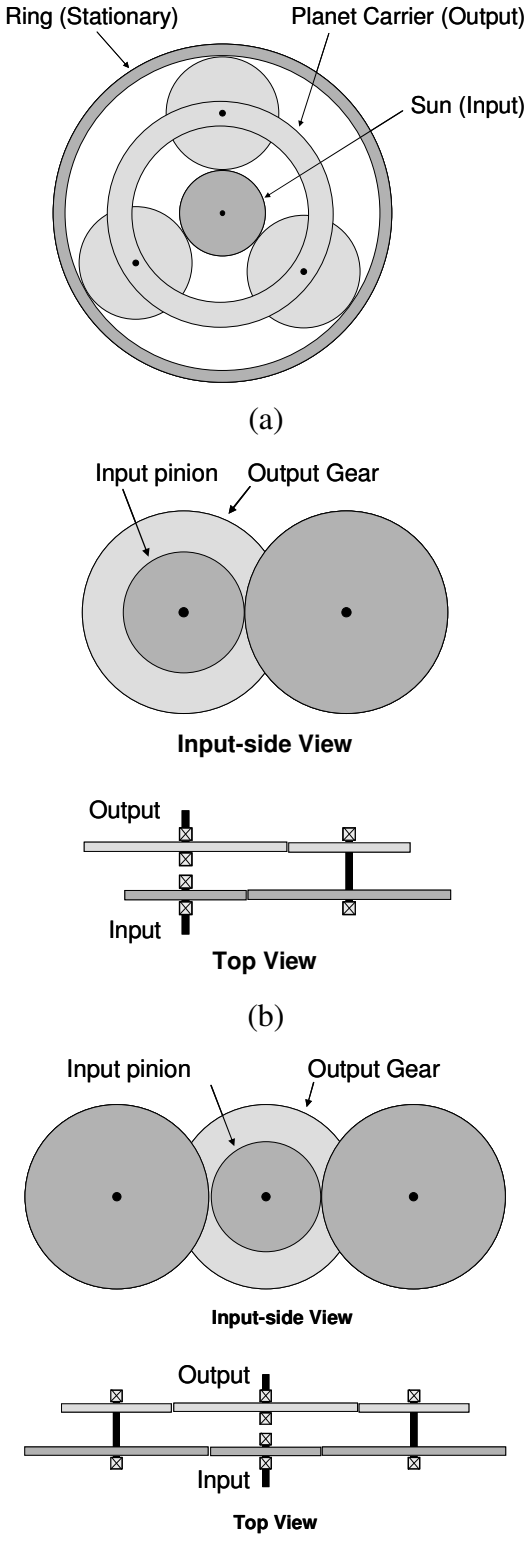


Figure 3.4: Layout of the three gearbox concepts: (a) planetary gearbox; (b) single-sided reverted gear train; (c) double-sided reverted gear train.

- Single-Sided Fully-Reverted Gearbox (SGB): Four-gear system with two identical pinions and two identical gears. Depicted in Figure 3.4(b).
- Double-Sided Fully-Reverted Gearbox (DGB): Similar to single-sided concept, but includes two paths for torque flow. Depicted in Figure 3.4(c).

Each concept is an abstraction of many possible implementations that conform to a particular structure. Within each concept, designs have a common parametric structure. These control the number of teeth on each gear, the gear face widths and the gear module. Other parameters, such as gear material, quality factor, etc., are assumed the same for all concepts; it is possible to vary these, but doing so would add little to the demonstration.

All three concepts are defined over a wide domain in their respective design spaces. The number of teeth on any gear is allowed to vary from 15 to about 50. The face width, constant for all gears in the same gearbox, is permitted to vary from 6.35 mm to 8.75 cm. Gear module can take on any of the 25 standard Series 1 values, which range from 0.1 mm to 5 cm.[f]

Gearbox Attributes and Preference Classifications

The gearbox tradeoff models account for three attributes.

- Cost[g]: The expense of purchasing the gearbox, computed as a function of the material and parts involved. Classified as a dominator attribute (less is better).

[f] Source: http://www.qtcgears.com/Q410/QTC/Q410P337.htm

[g] Price and cost are closely-related terms. In this research, the cost for a component refers to what the system designer would have to pay to purchase the component in question. The term "cost" is used because this is an expense incurred by the system designer. However, one should not confuse this with the cost of manufacturing the component. From a manufacturer's perspective, what a system designer calls "cost" is better described as the "price" for the component (i.e., cost plus markup).

- Reliability: The probability that the gearbox operates without failure, considering both static and dynamic loading phenomena. Classified as a dominator attribute (more is better).
- Gear ratio: The ratio of transformation from input to output. Classified as a parameter attribute. In the current context (power transmission), decision objectives such as maximizing power and torque throughput result in opposing preferences for gear ratio.

In general, other attributes may be important for considering gearboxes. For example, outside dimensions and mass may be important in some problems. However, the three attributes listed above are sufficient for this example.

Data Generation and Dominance Analysis

A model-based data gathering approach is used for this example (for examples involving observational data, see Chapter 6 and Chapter 7). For each concept, one can compute gear ratio and reliability from the design variables using standard engineering models (see e.g., (Norton 2000)). Each gearbox concept has a different design space representation. Cost is an empirical relationship based on gear dimensions fit from catalog data (i.e., it is a traditional application of predictive modeling techniques).

Implementations of each concept are generated by sampling the design space systematically. Although it is possible to use design-of-experiments (DOE) techniques to reduce data requirements, the engineering models are computationally simple in this case and efficiency is not a major concern. Each sample point is verified for technological feasibility—e.g., vetted against basic geometric constraints—and the ones deemed feasible are analyzed according to the engineering models to compute the attributes of

interest. The sample contains many implementations that are feasible but poor from a decision-making perspective. These are eliminated during dominance analysis.

Dominance analysis is conducted using parameterized Pareto dominance (Definition 3.2) using reliability and cost as dominator attributes and gear ratio as a parameter attribute. Mathematically, the negative of cost is used so that it is consistent with Definition 3.2. However, graphs and tables display cost in its natural sense.

Model Fitting and Validation for Concept-Specific Tradeoff Models

For each concept, gear ratio and reliability are used as inputs to the tradeoff model. Gear ratio must be an input because it is a parameter attribute. The decision to use cost as an output is arbitrary (initial experiments indicated no significant difference in fitting accuracy for using cost or reliability as the output). Strictly speaking, gear ratio is not a continuous variable since it is determined by the numbers of gear teeth in mesh. However, it is approximated as continuous for modeling purposes.

To generalize the parameterized efficient set data, Kriging interpolation methods and the DACE Matlab Kriging Toolbox are used (Lophaven, et al. 2002). Fifty non-dominated implementations of each concept are reserved for estimating prediction error (i.e., they are not used during model fitting). This approach is known as hold-out validation (Han and Kamber 2001, Hand, et al. 2001). Tradeoff models are fit to the rest of the parameterized efficient set data for each concept.

According to the holdout validation procedure, the estimated root mean square prediction errors for the tradeoff models are:

- PGB Model: $2.17
- SGB Model: $6.14
- DGB Model: $4.81

The minimum cost of any gearbox being considered is about $165, so this represents an error of less than 5%.

Domain characterization is straightforward in this example. The upper and lower bounds on the model inputs are found to be adequate representations of the valid input domain. Table 3.1 is a summary of the tradeoff model domain descriptions.

Figure 3.5 is a visualization of the fitted models for gear ratios up to 5 (the models are valid for ratios up to about 9). It is impractical to report the closed-form equations for these models since the Kriging models are interpolators (i.e., the equation is a function of every individual point in the data set).

Table 3.1: Domain descriptions for the gearbox tradeoff models.

	Input Variable	*Lower Bound*	*Upper Bound*
SGB and DGB Concepts	Reliability, R	0.85	1
	Gear Ratio, N_g	1.13	9.0
PGB Concept	Reliability, R	0.85	1
	Gear Ratio, N_g	2.54	9.2

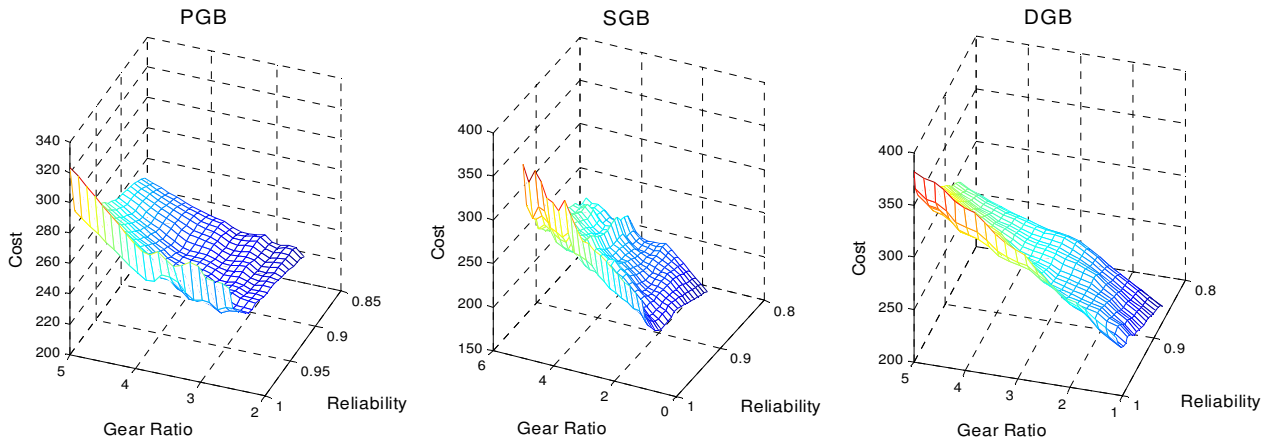


Figure 3.5: Fitted tradeoff models for each of the design concepts for gear ratios up to 5.

Generating an Abstract Tradeoff Model

Because the tradeoff models for the gearbox concepts share the same interface—they all compute cost as a function of gear ratio and reliability—it is possible to abstract them into a single tradeoff model. Figure 3.6 is a graph with all three gearbox tradeoff models plotted on the same axes. One can observe that the PGB and SGB concepts are dominant in particular regions of the tradeoff space.

To generate an abstract tradeoff model, the tradeoff models for each concept are sampled at the same locations. A simple grid sampling scheme is used, consisting of 225 points in the model inputs space. These samples are fast to generate because the tradeoff models are algebraic equations. Parameterized Pareto dominance is applied to the resulting data set. This reduces the 675 sample points (225 samples each of three models) to 192 non-dominated points.

Like with the concept-specific tradeoff models, a Kriging model is used to represent the abstract surface and hold-out validation is performed using 50 data points. This results in a root-mean-squared error of $5.32. Figure 3.7 is a graph of the resulting model over its entire domain.

3.4.2 Design Problem Scenario

System and Environment

The system under consideration is a small, single-person off-road vehicle. The components relevant to this problem are its engine, continuously-variable transmission (CVT), a fixed-ratio gearbox and a rear differential with a fixed gear ratio, arranged according to Figure 3.8. All the components are preexisting on the vehicle except the gearbox that is the focus of this design problem. Table 3.2 is a summary of the system and environmental parameters that affect vehicle performance.

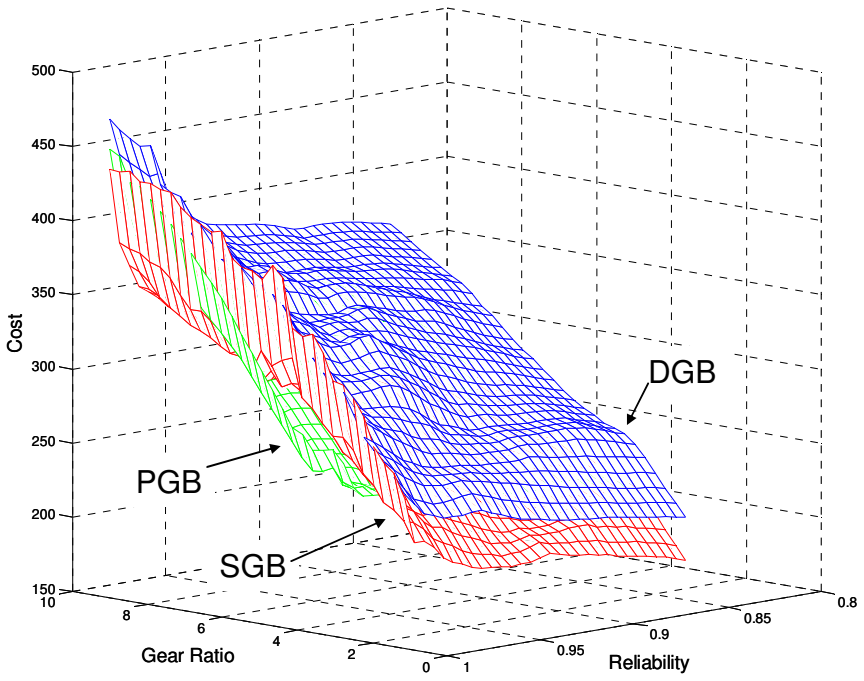


Figure 3.6: Tradeoff models for gearbox concepts plotted in same graph.

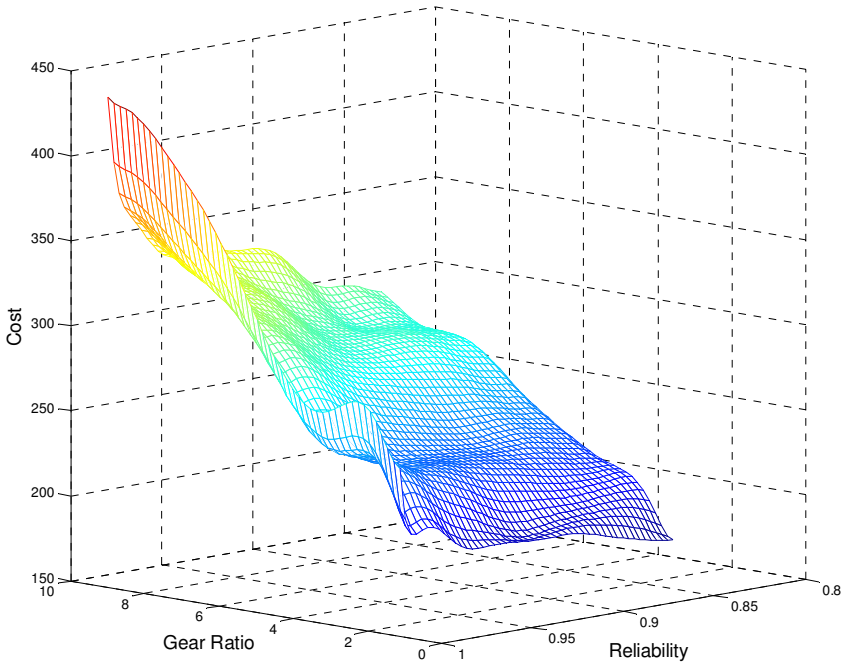


Figure 3.7: Visualization of the abstracted gearbox model.

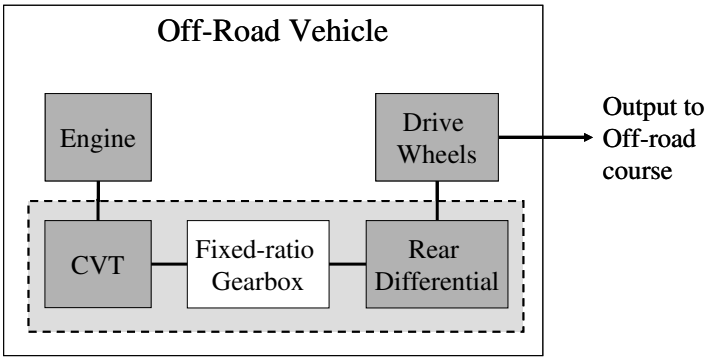


Figure 3.8: Configuration of off-road vehicle components. Grayed components are already designed; the fixed-ratio gearbox is of interest in this demonstration.

Table 3.2:Values for parameters used in the example problem.

Parameter	Value
System and Environment	
Total Vehicle Mass	300 [kg]
External Drag Coefficient	0.45 [N/(m/s^2)]
Internal Drag Coefficient	0.02 [N/rpm]
Gearing	
Application Factor	1.7
Gear Quality Factor	8
Bending Strength Geometry Factor	0.24
Gear Material, Bending Fatigue Strength	200e6 [Pa]

Decision Problem

The problem is to design a gearbox for the vehicle that competes in a race. Designer preferences are to maximize profit, which in this case yields a value function of

$$V(R,W,C) = RW - C,$$

where R is the reliability of the gearbox, W is the anticipated winnings assuming perfect reliability and C is the cost of building the gearbox. At the decision level, reliability and cost are equivalent to the corresponding gearbox attributes. The anticipated winnings decision attribute is a non-monotonic function of gear ratio, N_g. Figure 3.9 is a summary

of how winnings relates to gear ratio. Vehicle dynamics account for aerodynamic drag, rolling resistance and engine characteristics. Maximum velocity and maximum acceleration are computed at the appropriate engine operating points. Race finish time is approximated using an algebraic relationship developed originally by Bruns (2006). The first term in this expression is the time it would take to complete the entire race if traveling at top speed, and the second term accounts for the time to accelerate to top speed. The K_c parameter accounts for the relative amount of accelerating for a given course. Finish time is used to compute the anticipated winnings for the race. The finish time model is non-monotonic in maximum velocity and the vehicle dynamics are non-monotonic in N_g.

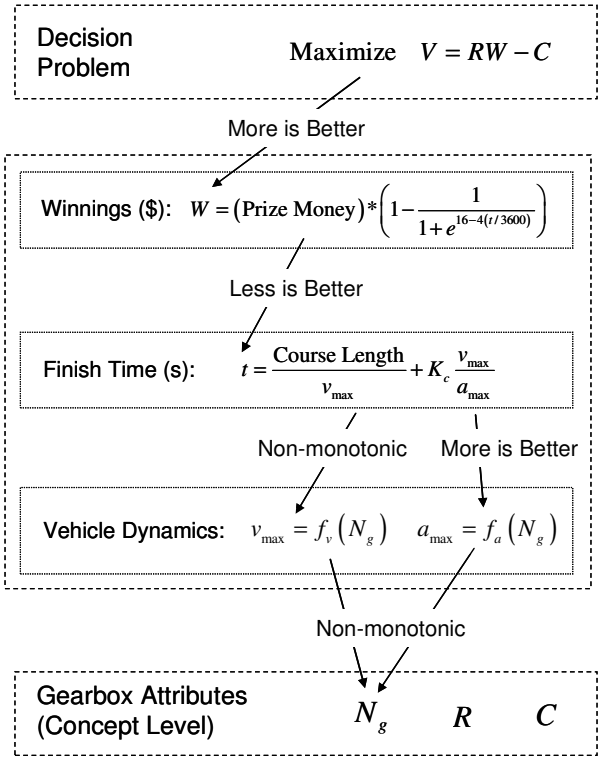


Figure 3.9: Schematic view of how top-level preferences propagate down to the concept-level attribute gear ratio. Vehicle dynamics are non-monotonic functions of gear ratio, and account for aerodynamic drag, rolling resistance and engine characteristics.

Solution Formulation using Concept-Specific Tradeoff Models

The decision problem is solved using both concept-specific tradeoff models and a tradeoff model abstracted from them in order to compare the approaches. The concept-specific approach follows the formulation from 3.3.2. Concept selection begins with performing requirements allocation for each concept. The first step is to specialize Equation (3.4) to the problem at hand. This yields

$$\left[R, N_g\right]_l^* = \underset{\left[R,N_g\right]\in\tilde{\mathbf{Y}}_l}{\arg\max}\, V\left(\mathbf{S}\left(R, N_g, T_l\left(\left[R, N_g\right]\right)\right)\right), \tag{3.9}$$

where N_g is the gear ratio, $T_l(\cdot)$ is the tradeoff model corresponding to the l^{th} gearbox concept, $\mathbf{S}(\cdot)$ represents the system model that calculates system-level attributes from the concept-level attributes and $\tilde{\mathbf{Y}}_l$ is search domain for the reliability and gear ratio attributes for the l^{th} concept. The domain descriptions that define $\tilde{\mathbf{Y}}_l$ are specified in Table 3.1. Note that the system model in this example is the same for all concepts and consists of all the models listed in Figure 3.9.

One can solve Equation (3.9) using standard optimization methods. A pattern search method is used in this example. After Equation (3.3), one can construct the allocation decision for concept l as

$$\mathbf{y}_l^* = \left[\left[R, N_g\right]_l^*, T_l\left(\left[R, N_g\right]_l^*\right)\right].$$

Given this result, one can use Equation (3.5) to find the most preferred concept. This yields the expression

$$l^* = \underset{l\in\{PGB,SGB,DGB\}}{\arg\max}\; V\left(\mathbf{S}\left(\mathbf{y}_l^*\right)\right),$$

where PGB, SGB and DGB denote the planetary gearbox, single-sided gearbox and double-sided gearbox, respectively.

Solution Formulation using Abstract Tradeoff Model

To predict the most preferred requirements allocation using the abstracted model, denoted $T_0(\cdot)$, one must solve the following search problem:

$$\left[R, N_g\right]_0^* = \underset{\left[R,N_g\right]\in\tilde{\mathbf{Y}}_0}{\arg\max}\, V\left(\mathbf{S}\left(R, N_g, T_0\left(\left[R, N_g\right]\right)\right)\right), \tag{3.10}$$

where the $\tilde{\mathbf{Y}}_0$ denotes the search domain for the abstract model and the other terms are as defined above. The most preferred gearbox concept is the one that satisfies the following:

$$l^* = \underset{l\in\{PGB,SGB,DGB\}}{\arg\min} \left| T_0\left(\left[R, N_g\right]_0^*\right) - T_l\left(\left[R, N_g\right]_0^*\right)\right|.$$

3.4.3 Results

Approach based on Concept-Specific Tradeoff Models

Table 3.3 contains results from the gearbox concept selection problem. The table contains the attributes and value for the most preferred implementation of each design concept as predicted using the fitted tradeoff models. The planetary concept has the largest value of all three design concepts, and therefore is the most preferred. In practice, designers would continue by designing a planetary gearbox using the indicated attributes as design targets.

The table also contains reference solutions for each gearbox configuration. These solutions are computed using optimization methods to search the design parameter space for each concept. This relies on using the appropriate engineering models for each concept. These are the same models that were sampled when generating the tradeoff

models. Thus, if the solutions differ significantly, one could question the use of tradeoff models based on parameterized efficient set data.

A comparison of the results yields two main observations. First, the selection decision obtained using the tradeoff models appears to be correct. For both the tradeoff modeling solutions and the reference solutions, the preference order is $PGB \succ SGB \succ DGB$ (where $\succ$ means "is preferred to"). Second, the allocation predictions are accurate for each configuration, differing by no more than a few percent. This indicates that no important information was lost by applying parameterized Pareto dominance criterion to the sampled data. These tradeoff models yield the correct decision for the correct reason.

Table 3.3: Results from the gearbox concept selection problem (using the tradeoff models) and the reference solution for each configuration. The most preferred design in each case is the PGB concept.

		Using Tradeoff Models	*Reference Solution*	*Percent Difference*
PGB	***Maximum Value***	682.65	681.72	0.14
	Gear Ratio	4.14	4.13	0.12
	Reliability	0.994	0.994	0.00
	Cost ($)	262.42	262.91	0.19
SGB	***Maximum Value***	651.20	670.87	2.93
	Gear Ratio	4.12	4.27	3.51
	Reliability	0.968	0.988	2.02
	Cost ($)	268.20	266.55	0.62
DGB	***Maximum Value***	613.40	606.34	1.16
	Gear Ratio	4.16	4.27	2.58
	Reliability	0.980	0.984	0.41
	Cost ($)	319.00	327.47	2.59

Approach based on an Abstract Tradeoff Model

Table 3.4 contains the results of solving Equation (3.10) using the abstract tradeoff model. The allocation decision compares favorably with the reference solution obtained using engineering optimization methods.

Table 3.5 contains the results from comparing the three concept-specific tradeoff models to the solution obtained using the abstract model. Based on these results, a designer would choose the PGB concept. Compared to the other concepts, it agrees very closely with the cost computed using the abstract model. This selection is consistent with the engineering optimization results reported in Table 3.3.

Table 3.6 contains a comparison of the computational times involved in using the three different approaches. Times represent actual process execution time on a personal computer (i.e., not wall clock time). Since computer processor power varies, one should focus on the relative computational times rather than the absolute numbers. All times are low because the system model involves simple algebraic equations.

Both tradeoff modeling approaches are two orders of magnitude faster than the engineering optimization approach. This may be an extreme example since the engineering optimization solution is computed using a very thorough search to avoid local optima (to yield a quality reference solution; this involved multiple optimizer runs starting from different initial points). However, this does not explain the difference completely and one can conclude that it is faster to make the decision using tradeoff models.

It is worth noting that the time data does not include the time one requires to generate the tradeoff models. Adding this to the time required to solve the decision

Table 3.4: Comparison of allocation solution using abstract tradeoff model to reference solution for best concept.

	Using Tradeoff Model	*Reference Solution*	*Percent Difference*
Maximum Value	677.52	681.72	0.62
Gear Ratio	4.13	4.13	0.0
Reliability	0.994	0.994	0.0
Cost ($)	266.85	262.91	1.51

Table 3.5: Comparison of predictions from concept-specific tradeoff models to prediction from abstract model.

	Predicted Cost ($)	*Cost from Abstract Model ($)*	*Absolute Difference ($)*
PGB	262.33	266.85	4.52
SGB	344.17		77.32
DGB	304.95		38.10

Table 3.6: Comparison of computational times for three approaches to the gearbox design problem.

Model		*Partial Times (sec)*	*Total Time (sec)*
Abstract Tradeoff Model		n/a	0.139
Concept-Specific Tradeoff Models	PGB	0.131	0.451
	SGB	0.175	
	DGB	0.145	
Engineering Optimization Approach	PGB	19.46	46.48
	SGB	13.57	
	DGB	13.46	

problem would be misleading since one can reuse a tradeoff model across many problems. Moreover, one designer might incur the time but another would not because the model already exists.

The approach relying on the abstract tradeoff model is about three times faster than the approach that uses concept-specific tradeoff models. This is approximately what one would expect since there are three concepts. These results confirm the computational advantage of the approach based on abstract tradeoff models.

3.5 The Practical Value of Performing Dominance Analysis

The example from the preceding section addresses the practical viability of applying parameterized Pareto dominance to the gearbox data, but not the practical value of doing so. The practical value of dominance analysis is due to two main factors:

- Parameterized Pareto dominance is not expensive to apply.
- Designers expose themselves to risk by not performing this step.

The cost argument itself has two considerations, both of which are favorable to the value proposition. First, dominance analysis is something designers do during model generation, but not during decision making. This means they incur the computational expense exactly once—it is not something that is repeated inside an optimization loop or in the context of design space exploration. Second, the algorithm is relatively fast to apply for the sizes of data sets designers are likely to encounter. Because designers can draw conclusions about domination only at equivalent parameter settings, it is unnecessary for them to compare most pairs of attribute vectors. This reduces computational costs compared to that of applying classical Pareto dominance to a similarly-sized data set. For all of the design problems considered in this research, the parameterized Pareto dominance step takes on the order of seconds complete.

The risk argument is based on the observation of how including dominated implementations in the training set will bias the predictive model away from the tradeoff relationship it is intended to represent. This idea is illustrated graphically in Figure 1.4 (page 12). Although the bias may be small at times, designers have no means by which to ascertain whether this is the case without conducting dominance analysis in the first place. Furthermore, the importance of the bias depends on the particular decision at hand.

Given the low cost of performing dominance analysis and the risks associated with forgoing it, designers are well advised to not skip this step.

Another perspective is to consider the relative proportion of implementations that one would eliminate using parameterized Pareto dominance. This provides no evidence about the amount of bias that the dominated points might introduce. However, larger numbers of dominated points would tend to introduce larger biases into a tradeoff model, all other factors remaining equal.

Dominance analysis is essential whenever one wishes to fit a tradeoff model to data generated by sampling an analysis model (i.e., as opposed to using observational data). Such data necessarily includes implementations that are technically feasible but grossly inferior to the best in the sample set. Figure 3.10 is an illustration of this for data used in the gearbox design example. This graph is for one setting of the parameter attribute for the planetary gearbox, but is representative of other parameter levels and the

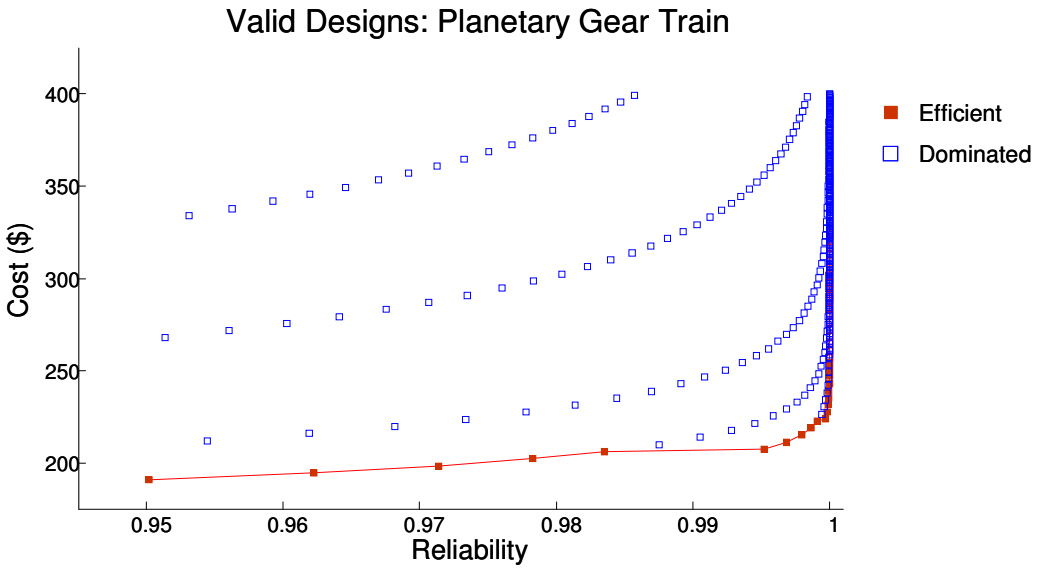


Figure 3.10: Efficient and dominated implementations of planetary gear train for a fixed value of gear ratio.

other gearbox concepts. Clearly, fitting a model to all of this data would misrepresent the tradeoff relationship between reliability and cost.

Rather than sampling of the design space, one might incorporate optimization criteria to arrive at solutions that more closely represent how designers might implement the component in question. For some problems, this could be more efficient that sampling (randomly or systematically). However, this simply is a form of dominance analysis since optimization entails the use of an objective function.

The situation is more varied for the case of using observational data, but it is clear that many dominated implementations exist in such data. Table 3.7 is a summary of dominance analysis results for observational data about the components used in the example problem of Chapter 6. Some implementations of each component are removed from the database initially based on outlier analysis. This includes both "common sense" eliminations and points excluded using the domain description and clustering procedure described in Chapter 4. The stated percentage of each data set removed via parameterized Pareto dominance is relative to the post-outlier quantities. The proportion of implementations removed is significant.

Table 3.7: Percentage of implementations eliminated by parameterized Pareto dominance for components used in log splitter design problem (Chapter 6).

Component	*Engine*	*Pump*	*Cylinder*	*Control Valve*
Total # in DB	59	61	188	36
# after Outlier Analysis	49	43	158	32
# after Dominance Analysis	19	24	137	8
% removed by PDOM	61%	44%	13%	75%

Several factors may explain why so much of the observed data is dominated. One factor is that designers have bounded rationality and design components and systems in this context. As such, it is reasonable to expect that some proportion of the available implementations of a component are dominated by other implementations.

Another factor is that implementations that *seem* to be dominated when one examines them with a particular set of attributes may be non-dominated when one considers additional attributes. For example, a hermetically sealed motor may be considerably more expensive than a comparable standard motor with similar engineering characteristics. If one does not account for an attribute relating to sealing, the hermetically sealed implementation likely will appear to be dominated (since odds are that it would cost more). To eliminate such implementations actually is desirable so long as the sealing or other attributes are unimportant for the current context. Thus, dominance analysis is a useful data cleansing step.

3.6 Conclusions and Chapter Summary

Based on the results of this chapter, one can conclude that parameterized efficient sets are an appropriate basis for generating reusable predictive tradeoff models of engineered components. Several observations support this conclusion.

The first observation is that parameterized Pareto dominance is mathematically sound and appropriate for the problem of eliminating data about dominated implementations of a component without problem-specific information. This is established in Section 3.1. It is an extension of classical Pareto dominance, which is inappropriate when problem-specific information is unavailable. In order for tradeoff models to be reusable—i.e., applicable across many design problems—it is necessary for

the underlying dominance criterion to be largely problem independent. According to Theorem 3.2, any component implementation eliminated by parameterized Pareto dominance cannot be the most preferred. When one considers this in light of the decision chain arguments of Barton and coauthors (Barton and Love 2000, Barton, et al. 2001) and the set-based design principles described by Ward, Sobek and other authors (Ward 1989, Ward, et al. 1995, Sobek II 1996) (see Chapter 2), one can conclude reasonably that the parameterized efficient set of a component is what designers *should* use as a basis for predictive tradeoff models.

Another observation is that a reasonable procedure exists by which designers can generate a predictive tradeoff model for a particular type of component. This approach is the topic of Section 3.2. It is based on typical data-driven modeling procedures, but is specialized for dealing with the problem of tradeoff model generation. Extensions to the typical procedures follow logically from the needs of the problem.

The design example of Section 3.4 also is evidence that parameterized efficient sets are an effective basis for tradeoff modeling. It is possible to generalize a continuous model effectively from parameterized efficient set data and use it to make decisions about partially-defined design solutions through design concept selection and requirements allocation. The results obtained using the tradeoff modeling approach match well with results obtained via a traditional engineering optimization approach. Although this design problem is relatively simple, the results are encouraging.

A final observation is that there is a practical motivation for applying parameterized Pareto dominance. The cost of eliminating data about dominated implementations from a data set is small and designers need perform the operation only

once during model generation (as opposed to repeatedly inside of an optimization loop). Furthermore, the risk of not performing dominance analysis is difficult to assess on a case-by-case basis and possibly is significant. In observational data sets about four common engineering components, only one data set had fewer than 40% of the implementations eliminated via the parameterized Pareto dominance rule. Thus, there is practical value in performing the dominance analysis using parameterized Pareto dominance.

Although one can conclude that parameterized efficient sets are a reasonable basis for tradeoff modeling, many open questions remain. The example problem in this chapter is of modest complexity, does not involve observational (i.e., mined) data and includes tradeoff models of only one functional component. Subsequent chapters and example problems expand the range of evidence supporting the overall approach through deeper mathematical analysis and more complex example problems. However, the immediate next step is to resolve a practical challenge associated with tradeoff model generation. This relates to the domain description step of the procedure summarized in Figure 3.1. It is possible to deal with domain description in a naïve way in this chapter, but this approach will not suffice in general. Chapter 4 is a description of a general approach for describing the valid input domain of a tradeoff model.

CHAPTER 4:

USING SUPPORT VECTOR DOMAIN DESCRIPTION TO IMPROVE PREDICTIVE TRADEOFF MODELING

Based on the analysis of the previous chapter, predictive tradeoff models are promising as an abstract representation for sets of designs in the context of system-level decision making. Because tradeoff models are attribute-space representations of a subsystem, they are abstract of implementation details. However, creating predictive models at this high level of abstraction poses a special challenge: certain attribute combinations for a component may be unobtainable. These restrictions are due to fundamental physical constraints that would be evident when modeling a component at a lower level of abstraction, but can be difficult to infer when abstracting from attribute data.

From a modeling perspective, the challenge is to identify the domain of the tradeoff model input space over which predictions can be valid. Despite the fact that solutions beyond this domain are physically infeasible, one still can compute predictions using a tradeoff model. These are meaningless and to be avoided. To constrain automated search routines (optimization and design exploration codes), a formalized domain description is necessary. Thus, this chapter addresses the second research question:

RQ2. How can designers describe the set of valid inputs to a tradeoff model mathematically?

The hypothesized answer to this question is that designers can apply an approach based on specific machine learning techniques:

H2. Designers can use a domain description procedure based on kernel-based support vector domain description and clustering methods.

Using the approach, designers can create a mathematical model for the valid input domain of a tradeoff model. They also can identify clusters (disjoint sub-domains) and outlying data points, both of which are useful in constructing accurate tradeoff models.

The focus of this chapter is on defining the tradeoff model domain description procedure and the methods upon which it is based. The chapter organization is as follows. Section 4.1 contains a definition of the domain description problem and a discussion of the requirements for its solution. Sections 4.2 and 4.3 are reviews of the theory underlying the approach for tradeoff model domain description. The former is focused on Support Vector Domain Description (SVDD) and the latter on Support Vector Clustering (SVC). Section 4.4 is a summary of the domain description approach for tradeoff modeling.

4.1 The Problem of Domain Description for Tradeoff Models

The problem of domain description is to define in a mathematical way what constitutes a valid input to a tradeoff model. This problem is analogous to the one-dimensional problem of preventing extrapolation. Figure 4.1 is a depiction of this simpler case with a model $y = f(x)$ fit to (x, y) pairs that fall within the input domain $[x_{lb}, x_{ub}]$. A rule of thumb is that one should avoid using the model $f(\cdot)$ to make predictions beyond the domain of the observed data—i.e., for $x < x_{lb}$ or $x > x_{ub}$. The rationale for this is that one cannot be confident that the local trend observed on $[x_{lb}, x_{ub}]$ will hold for

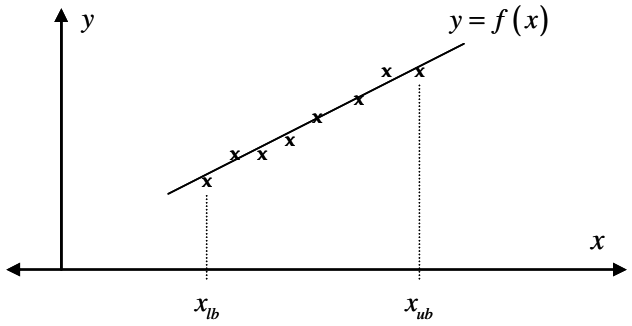


Figure 4.1: A simple domain description for the one-dimensional case.

any other region of x[h]. For example, it is common for a nonlinear relationship to appear linear over a small neighborhood. In the one-dimensional case, the simple solution is to restrict evaluations of the model to between the upper and lower bounds on the model input. However, this approach does not generalize to higher-dimensional problems.

Figure 4.2 is a graphical depiction of the domain description problem for a model $z = f(x, y)$. In this example, an input domain defined by the upper and lower bounds on each of the input variables yields an overly conservative domain description. This domain includes all the observed data, but also includes many regions in which no observations exist. Using such a domain definition can lead to unwarranted extrapolation beyond the observations.

The need for a proper domain description is particularly acute when generating a tradeoff model. Unlike many other predictive modeling scenarios, the inputs to a tradeoff

[h] In principle, one cannot even know what will happen between samples in the observed region. This is the Problem of Induction described in the 18th century by David Hume (1965, Orig. 1739-40). The question of how to validate a continuous model based on a finite data sample has received considerable attention in the literature (see e.g., (Balci 1997, Kleindorfer, et al. 1998, Law and McComas 2001, Sargent 2001, Oberkampf and Trucano 2002, Malak and Paredis 2007)) and several professional and governmental organizations publish model validation guidelines (including the AIAA (AIAA 1998), ASME (ASME 2006), and the U.S. Department of Defense (US DoD 2003)). However, this issue is beyond the scope of this research.

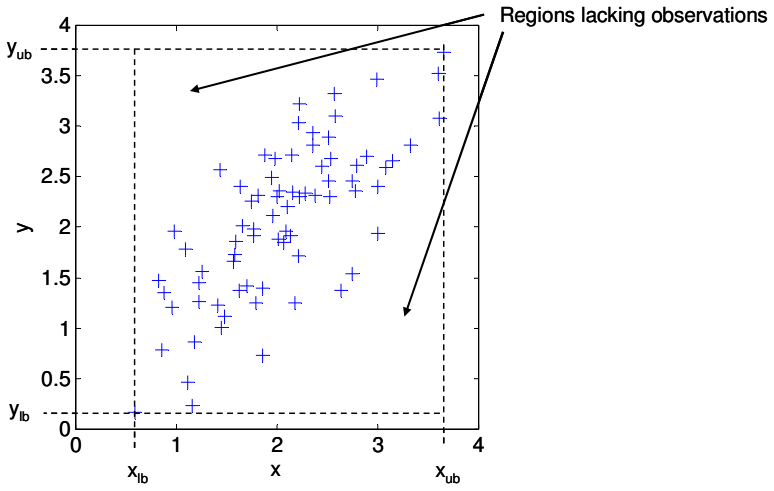
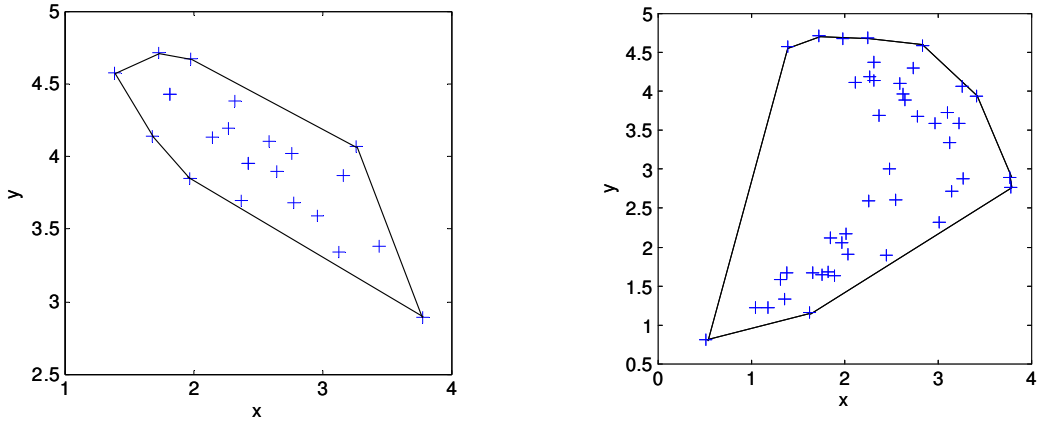


Figure 4.2: An illustration of two independent variables with a valid domain that occupies less than the rectangular region defined by their upper and lower bounds.

model may not be independent of one another. Consider the reliability and cost of an engine. These attributes seldom are independent—they tend to have a positive correlation—and one cannot, for example, produce an engine with 100% reliability for zero cost. Such relationships exist because the attributes of interest during tradeoff modeling depend on lower-level design variables. When multiple attributes relate to the same design variables, correlations occur and certain combinations of attribute values can be impossible to achieve. The practical consequence is that predictions from these unobtainable regions in the attribute space are meaningless. This is somewhat worse than the ordinary problem of extrapolation (at least the extrapolated model *might* be correct) and designers must take care to describe a tight domain for their tradeoff models.

Although there are many approaches by which a designer can describe the domain associated with a tradeoff model, most of these are appropriate only under certain circumstances or have other limitations. For example, one approach is to estimate a

probability density function associated with the data and to define the domain according to a probability threshold (Tarassenko, et al. 1995). However, to obtain a good probability model can require a great deal of data. The data sets used to generate tradeoff models in this research often contain only dozens of points (see the example problems of Chapter 6 and Chapter 7), which is insufficient. Another approach is to use a convex hulling algorithm such as Quickhull (Barber, et al. 1996) to define a geometric boundary for the data. However, many of the data sets encountered in this research are sufficiently non-convex for their convex hull to be a poor domain description. Figure 4.3 is an illustration of how a convex hull can be inappropriate as a domain description.



(a) (b)

Figure 4.3: Two convex hull domain descriptions. In (a), the convex hull envelops the data somewhat tightly leaving little extra space in the domain. In (b), a large region without any data still falls within the domain description.

At times it may be possible for designers to transform their data nonlinearly such that they can more easily describe the domain of the remapped data. For example, it may be possible for one to transform the x-axis in Figure 4.3(b) such that the point cloud is convex in the transformed space. This strategy is akin to the use of nonlinear

transformations in regression analysis to achieve a better fit. However, as in regression, the appropriate transformation is highly problem-dependent and not automated easily. It is preferable for designers to have a more general approach to domain description.

A final complication to domain description is that the attribute data—particularly when it is observational data of existing components—may contain gaps or occur in disjoint groupings. Figure 4.4 is an illustration of this problem. The data contains void regions internal to the main data cloud as well as a secondary cluster that is disjoint from the outer ring of data. Although this is a synthetic data set, gaps and disjoint clusters occur in real engineering data sets. Voids likely are due to a physical constraint that prevents designers from achieving a particular combination of attributes. Clusters in observational data may indicate popular niches in the market. Designers require a general approach for dealing with such situations.

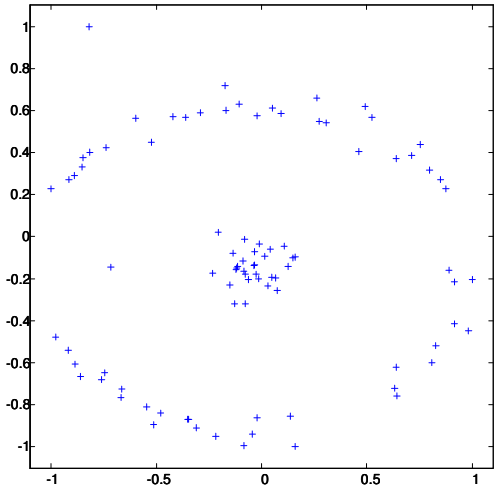


Figure 4.4: A synthetic data set with internal voids and distinct clusters.

The capability to handle outliers is another desirably quality for a domain description approach. In Figure 4.4, there is an isolated data point at approximately (-0.6,-0.2) and another at approximately (-0.75, 1). These are far from all other data points and one might be justified in removing it from the analysis. Ideally, one could identify such situations more easily using a domain description method since visual approaches do not scale well beyond a few dimensions.

In addition to the preceding considerations, designers require an approach for tradeoff model domain description that yields fast evaluations of whether or not a given point is in the domain. This requirement is particularly important because designers often will use tradeoff models in concert with automated optimization and design exploration algorithms. Such algorithms can require upwards of tend of thousands of tradeoff model evaluations. A slow domain evaluation routine will slow such searches noticeably.

To summarize the requirements, an approach for tradeoff model domain description should be:

- capable of handling non-convex domains,
- capable of handling gaps in a domain,
- capable of handling disjoint sub-domains,
- capable of identifying potential outliers, and
- fast at determining whether a given point is in the domain.

The approach described in this chapter addresses each of these concerns using two key methods: support vector domain description and support vector clustering.

4.2 Support Vector Domain Description

The approach to creating a domain description for tradeoff models proposed in this research is based on an existing method called support vector domain description (SVDD) described originally by Tax and Duin (1999b). Their method is inspired by Vapnik's support vector machines (SVMs) (Vapnik 1995). A SVM is a classifier that distinguishes between classes geometrically using a hyperplane. One determines this hyperplane during the learning phase based on labeled examples of each class. Tax and Duin developed the SVDD method using mathematics similar to that of a SVM. However, instead of identifying classes using a hyperplane, they determine whether a point is in the data domain using a hypersphere that envelops the data. This formulation requires only data about what is in the domain (i.e., one requires no examples of what is not in the domain).

The principal motivation to apply SVDD in the current context is its capability to establish a domain classification boundary without examples of points that are outside the domain. This sets it apart from most other machine learning algorithms. Another advantage is that one can apply kernel techniques to model domains for which a hyperspherical domain is inappropriate (including non-convex domains). It also accommodates gaps in the data and disjoint sub-domains, and one can use it to identify outliers. Finally, one can extend it to perform clustering analysis (see Section 4.3). Thus, the approach satisfies all of the requirements identified in the preceding section.

4.2.1 Basic SVDD

For the basic (non kernel-based) SVDD method, one tries to find the minimum-radius hypersphere that contains a set of N data points, $\{\mathbf{x}_i, i = 1 \ldots N\}$. Thus, the domain

description consists of a sphere center, $\mathbf{a}$, and radius R. The domain description problem is to determine the center and radius given the data. In the most rudimentary formulation, one has the constraint

$$\|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2 \quad \forall i,$$

Where $\mathbf{a}$ is the sphere center, R is its radius and $\|\cdot\|$ is the Euclidean norm. However, this representation is sensitive to outliers in the data. It is advantageous to formulate the problem in a way that allows one to exclude the most extreme outliers. In the original formulation of SVDD, Tax and Duin achieve this by introducing slack variables, $\xi_i \geq 0$, in a manner analogous to the classical SVM formulation. Thus, the domain description problem becomes

$$\min_{R,\mathbf{a},\xi_i} \left(F(R,\mathbf{a},\xi_i) = R^2 + C\sum_i \xi_i \right), \tag{4.1}$$

where C is a constant scalar and $C\sum_i \xi_i$ is a penalty term. The minimization is subject to the modified constraints

$$\|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2 + \xi_i \quad \forall i. \tag{4.2}$$

One can combine Equations (4.1) and (4.2) in order to construct the Lagrangian

$$L(R,\mathbf{a},\beta_i,\xi_i) = R^2 + C\sum_i \xi_i - \sum_i \beta_i \left(R^2 + \xi_i - \|\mathbf{x}_i - \mathbf{a}\|^2 \right) - \sum_i \mu_i \xi_i,$$

which has Lagrange multipliers $\beta_i \geq 0$ and $\mu_i \geq 0$[i]. Optimality conditions require the partial derivatives to equal zero. This yields new constraints:

[i] This notation follows that of Ben-Hur and coauthors (Ben-Hur, et al. 2001). In their original paper on SVDD, Tax and Duin (Tax and Duin 1999b) use the symbols α_i and γ_i rather than β_i and μ_i, respectively.

$$\sum_i \beta_i = 1, \tag{4.3}$$

$$\mathbf{a} = \frac{\sum_i \beta_i \mathbf{x}_i}{\sum_i \beta_i} = \sum_i \beta_i \mathbf{x}_i \text{, and} \tag{4.4}$$

$$C - \beta_i - \mu_i = 0 \quad \forall i . \tag{4.5}$$

Since $\beta_i \geq 0$ and $\mu_i \geq 0$, it is valid to remove the variables μ_i from Equation (4.5) and to use instead the constraint $0 \leq \beta_i \leq C \ \forall i$.

By rewriting the Lagrangian problem by substituting in the preceding constraints, one can obtain the Wolfe dual form problem

$$\max_{\beta_i} W = \sum_i \beta_i \left(\mathbf{x}_i \cdot \mathbf{x}_i \right) - \sum_{i,j} \beta_i \beta_j \left(\mathbf{x}_i \cdot \mathbf{x}_j \right), \tag{4.6}$$

subject to constraints $0 \leq \beta_i \leq C \ \forall i$ and $\sum_i \beta_i = 1$.

Note that Equation (4.4) defines the location of the center of the hypersphere as a linear combination of the data points. One finds the weights for this linear combination, the β_i, by solving Equation (4.6). For each data point, $\mathbf{x}_i$ for $i = 1 \ldots N$, there are three possible classifications:

- It is inside the hypersphere, which is indicated by $\beta_i = 0$.
- It is on the boundary of the hypersphere, which is indicated by $0 < \beta_i < C$.
- It is an outlier outside of the hypersphere, which is indicated by $\beta_i = C$.

Data falling inside the hypersphere has no effect on the computation of the centroid and is not part of the representation of the domain description. This is an advantage in cases involving many data points. Typically only a small percentage of the data falls at the hypersphere boundary.

Data on the boundary of the hypersphere are called *support vectors* and are essential to the domain description representation. For any given point in the data space, $\mathbf{z}$, the squared distance to the hypersphere centroid is

$$R^2(\mathbf{z}) = \|\mathbf{z} - \mathbf{a}\|^2 .$$

Substituting in the definition of the center, Equation (4.4), one obtains

$$R^2(\mathbf{z}) = \mathbf{z} \cdot \mathbf{z} - 2\sum_i \beta_i \mathbf{z} \cdot \mathbf{x}_i + \sum_{i,j} \beta_i \beta_j \mathbf{x}_i \cdot \mathbf{x}_j . \tag{4.7}$$

Since support vectors are on the hypersphere boundary, their distance to the centroid defines the hypersphere radius. One can compute this radius as

$$R = R(\mathbf{x}_i), \quad \text{s.t. } \mathbf{x}_i \text{ is a support vector},$$

where $R(\cdot)$ is the square root of $R^2(\cdot)$.

Note that one can represent the two critical characteristics of the bounding hypersphere—its radius and the location of its centroid—using only the support vectors and their corresponding β_i. This is a significant result, and one of the main advantages of SVDD. For most data sets, only a small proportion of the data will lie on the hypersphere boundary. Thus, the resulting domain representation is compact relative to the data set.

To determine whether a test point is in the domain defined by the support vectors, one need only compute its distance from the hypersphere centroid. Test point $\mathbf{z}$ is in the domain if

$$R^2(\mathbf{z}) \le R^2 ,$$

where $R^2(\cdot)$ is from Equation (4.7) and R^2 is the squared hypersphere radius.

Outliers are data points for which $\beta_i = C$ and some authors refer to these as *bounded support vectors*. They are not part of the domain description. Whether one

detects any outliers depends on one's choice of C. Due to the constraint in Equation (4.3), choosing $C \geq 1$ yields no outliers. There must be at least one support vector (i.e., $0 < \beta_i < C$), which means $\beta_i = C$ cannot occur for $C \geq 1$. Also note that no solution is possible for $C < 1/N$ due also to the constraint in Equation (4.3). Empirical evidence presented by Tax and Duin in their original investigation of SVDD suggests that the value of C is not highly critical on the domain description (large ranges of C values yielded identical domain descriptions). However, the specific effect is problem-dependent.

4.2.2 Mercer Kernels

The basic formulation of SVDD is valuable when a hypersphere is a good model for the domain. However, this will not be the case in most situations. Tax and Duin address this problem in their original work on SVDD. Their solution is to use kernel functions to remap the data nonlinearly into a higher-dimensional feature space in which a hypersphere is a good model for the domain. Kernel functions allow one to perform this nonlinear transformation without explicit representations of the transformation or the higher-dimensional space. In fact, the feature space could be of infinite dimension (Scholkopf and Smola 2002).

Many machine learning algorithms perform better in after being "kernelized." In their survey of clustering methods, Filippone and coauthors (2007) attribute the first use of kernel methods to Aizerman and coauthors (1964) and the recent popularity of such methods to the success of the kernel-based version of Vapnik's SVM (Vapnik 1995). Kernel methods are particularly useful for support vector approaches because the classification scheme in such approaches is a simple geometric construct (a hyperplane or

hypersphere). Data that is not separable by such a surface in the original space may be separable after a non-linear remapping to a feature space.

Mercer kernels form the foundation of kernel-based learning methods. One can define a Mercer kernel as follows (Aronszajn 1950):

Definition 4.1 (Mercer Kernel): Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ *be a non-empty set of data such that* $\mathbf{x}_i \in \mathbb{R}^d \ \forall i$. *A function* $K : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ *is called a positive definite kernel or Mercer kernel if and only if* K *is symmetric (i.e.* $K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)$*) and*

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \quad \forall N \geq 2,$$

where $c_k \in \mathbb{R} \ \forall k = 1 \ldots N$.

Strictly speaking, this definition also holds for data defined in the complex plane. However, this research involves only real-valued data.

The significance of Mercer kernels in the current context is that one can express any Mercer kernel as the dot product of a nonlinear mapping of the data space (Scholkopf and Smola 2002, Shawe-Taylor and Cristianini 2004). That is,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j), \tag{4.8}$$

where $\Phi : \mathbf{X} \to \mathcal{F}$ is a nonlinear mapping from the data space to a high-dimensional *feature space,* $\mathcal{F}$. This representation of the kernel is powerful. One can use it to reformulate machine learning problems that involve dot products, such as the SVDD problem of Equation (4.6). Essentially, it allows one to compute the dot product of two

feature-space vectors *without knowing explicitly the nonlinear mapping* $\Phi(\cdot)$. This is an important result that several machine learning researchers have exploited. This procedure is at the heart of the so-called distance kernel trick, which is valuable for several clustering and pattern recognition problems (Scholkopf and Smola 2002, Shawe-Taylor and Cristianini 2004, Filippone, et al. 2007).

There exists a multitude of valid kernel functions. Ones common in machine learning include (Scholkopf and Smola 2002):

- Linear: $K_L(\mathbf{x}_i, x_j) = \mathbf{x}_i \cdot \mathbf{x}_j$.
- Polynomial (degree p): $K_p(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^p, \quad p \in \mathbb{N}$.
- Gaussian: $K_G(\mathbf{x}_i, \mathbf{x}_j) = e^{-q\|\mathbf{x}_i - \mathbf{x}_j\|^2}$.

The prior literature on SVDD favors the Gaussian kernel because it leads to closed domains in the data space (Tax and Duin 1999a).

4.2.3 Kernel-based SVDD

One can recast the SVDD problem as a kernel-based method. The strategy is to map the data into a higher-dimensional feature space and apply the representation from Equation (4.8) to express the result in terms of a Mercer kernel rather than the explicit transformation function. The domain description remains a hypersphere, but now the sphere is described in the feature space.

One develops kernel-based SVDD by revising the constraint that defines the enclosing hypersphere. For non-kernel SVDD, this is given in Equation (4.2), which involves a distance measure in the data space ($\mathbb{R}^d$). To reformulate this in the feature

space, one can replace the data-space distance with the corresponding feature-space distance. Recall Equation (4.2):

$$\left\|\mathbf{x}_i - \mathbf{a}\right\| \leq R^2 + \xi_i \quad \forall i\,.$$

This now becomes

$$\left\|\Phi\left(\mathbf{x}_i\right) - \mathbf{b}\right\|^2 \leq R^2 + \xi_i \quad \forall i\,, \tag{4.9}$$

where $\mathbf{b}$ is the centroid of the feature-space hypersphere and $\Phi(\cdot)$ is a nonlinear mapping from the data space to the feature space. From this point, one can develop the corresponding Wolfe dual problem in a way that directly parallels Section 4.2.1 to obtain

$$W = \sum_i \beta_i \Phi\left(\mathbf{x}_i\right) \cdot \Phi\left(\mathbf{x}_i\right) - \sum_{i,j} \beta_i \beta_j \Phi\left(\mathbf{x}_i\right) \cdot \Phi\left(\mathbf{x}_j\right),$$

where $0 \leq \beta_i \leq C,\ \forall i$. This formulation is in terms of dot products of the nonlinear transformation, $\Phi(\cdot)$. Recalling Equation (4.8), one can rewrite the above in terms of a Mercer kernel, $K(\cdot,\cdot)$:

$$W = \sum_i \beta_i K\left(\mathbf{x}_i, \mathbf{x}_i\right) - \sum_{i,j} \beta_i \beta_j K\left(\mathbf{x}_i, \mathbf{x}_j\right). \tag{4.10}$$

Given a set of data, one maximizes Equation (4.10) over the β_i subject to $0 \leq \beta_i \leq C,\ \forall i$ and $1/N < C \leq 1$.

One can compute the center of the hypersphere as

$$\mathbf{b} = \sum_i \beta_i \Phi\left(\mathbf{x}_i\right),$$

where the $\mathbf{x}_i$ are the support vectors (i.e., $0 < \beta_i < C$). Using this relationship, the squared distance of the feature-space image of a point, $\mathbf{z}$, to the center of the hypersphere is

$$R^2(\mathbf{z}) = \left\| \Phi(\mathbf{z}) - \mathbf{b} \right\|^2 = K(\mathbf{z},\mathbf{z}) - 2\sum_i \beta_i K(\mathbf{x}_i,\mathbf{z}) + \sum_{i,j} \beta_i \beta_j K(\mathbf{x}_i,\mathbf{x}_j). \quad (4.11)$$

This expression depends on the support vectors, their weights and the kernel, but not on the nonlinear transformation. The test to evaluate whether a point, $\mathbf{z}$, is in the domain is directly analogous to the non-kernel case: one compares $R^2(\mathbf{z})$ with the radius of the hypersphere (i.e., feature space distance from the centroid to a support vector).

4.2.4 Kernel-Based SVDD Example

Figure 4.5 is an illustration of kernel-based SVDD. The data depicted in Figure 4.4 is input to the algorithm, which uses a Gaussian kernel function. From these graphs, one can observe the general effect of varying the kernel width parameter, q. Smaller values yield fewer support vectors and a broader domain description. As the value increases, so does the number of support vectors. The domain description also follows the data more closely and breaks into multiple disjoint regions. However, readers should understand that the precise effect is problem-dependent.

4.3 Support Vector Clustering

Support vector clustering (SVC) is an extension to kernel-based SVDD described originally by Ben-Hur and coauthors (2001). It is a numerical approach for identifying when a domain description contains multiple disjoint regions in the original data space. The method takes a domain description and the original data set as inputs and outputs the appropriate cluster labels for the original data set. One can use this information to label new data points.

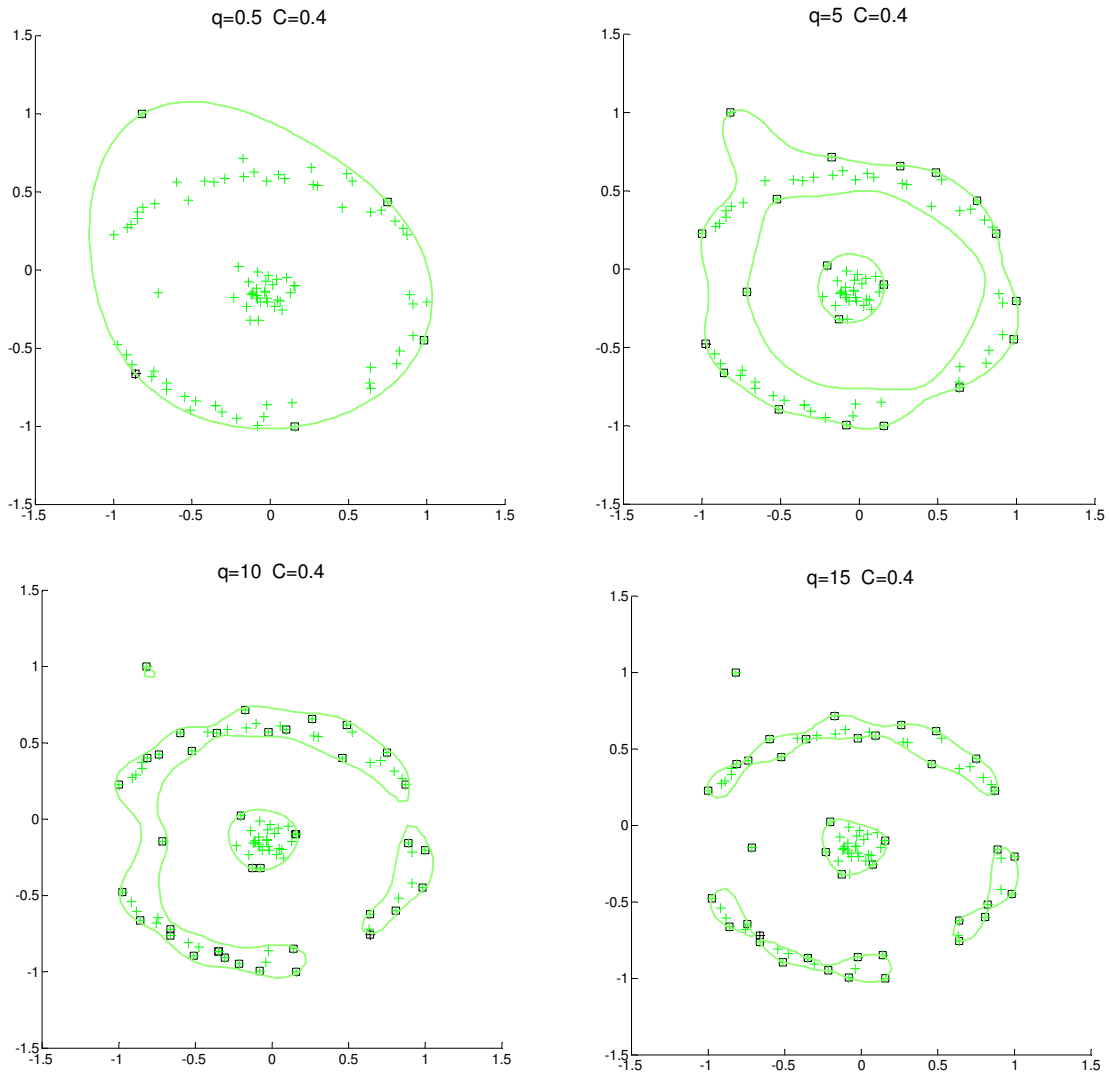


Figure 4.5: SVDD results for different settings of the Gaussian kernel width parameter, q. The regularization constant is held at $C = 0.4$. Support vectors are indicated by boxes.

This can be useful in the context of tradeoff modeling for two reasons. First, one sometimes can benefit in terms of model accuracy by fitting a different model to each cluster. In this case, it is necessary to identify these clusters and to associate each model with the appropriate cluster. Second, even when a single model describes the relationship in each cluster accurately, one must have a means of ensuring that automated search

routines (optimization or design space exploration codes) explore all valid regions of the domain. One can achieve this by modeling each cluster independently.

The current research focuses on the original formulation of SVC by Ben-Hur and coauthors. Other researchers report extensions to the basic SVC method, any of which one can apply to the problem of generating a tradeoff model. Lee and Daniels (2005) report a numerical method for determining the Gaussian kernel width parameter in the context of SVC. Typically, one must rely on intuition or experience to determine an appropriate value for this parameter. Similarly, Jun and Oh (2006) describe an evolutionary method for determining the Gaussian kernel width parameter and the regularization constant, C. Several researchers have investigated advanced cluster labeling methods, since this is a critical bottleneck in the original SVC method. These are reviewed in Section 4.3.2.

Other approaches to clustering exist that are not based on SVDD. These include partitioning-based methods (e.g., the K-means algorithm), hierarchical methods, probabilistic methods (approaches based on mixture models and use of the expectation maximization algorithm) and grid-based methods—see Han and Kamber (2001, Chapter 8), Hand and coauthors (2001, Chapter 9), Xu and Wunsch (2005) and Filippone and coauthors (2007) for surveys of clustering methods. Although it is clear from the literature that many effective clustering methods exist, the focus of this research is on the SVC approach because it is closely allied with the SVDD method.

4.3.1 Cluster Identification Theory

Figure 4.6 is a domain description obtained using kernel-based SVDD for a two-dimensional data set. The areas enclosed by the contours represent the two-dimensional

projection of the feature-space hypersphere. Any point enclosed by one of the contours is internal to the hypersphere; all other points are outside of it. Ben-Hur and coauthors recognized that one can determine whether two points are in the same cluster by checking to see whether it is possible to connect them via line segments between the data points without crossing the domain boundary contours.

The simplest scenario for this is illustrated in Figure 4.6. Points A and B are part of the same cluster because segment AB remains wholly within the domain boundary. In contrast, points C and D potentially are in different clusters because segment CD crosses the domain boundary. That is, the feature space representation of the segment leaves and then re-enters the enclosing hypersphere.

Two points can be part of the same cluster even when their connecting segment crosses a domain boundary. This is because cluster membership is associative: if A is in the same cluster as B and B is in the same cluster as C, then A must be in the same cluster as C. This logic is important when the domain region is a non-convex shape. Figure 4.6 is an illustration of this situation.

4.3.2 Cluster Labeling Methods

Due to the associative nature of cluster labeling, one requires an algorithm capable of identifying indirect relationships among the data points. Most approaches to labeling the clusters operate using adjacency information based on the theory presented in the preceding section. Two data points are considered adjacent if the segment connecting them in the data space does not cross the domain boundary.

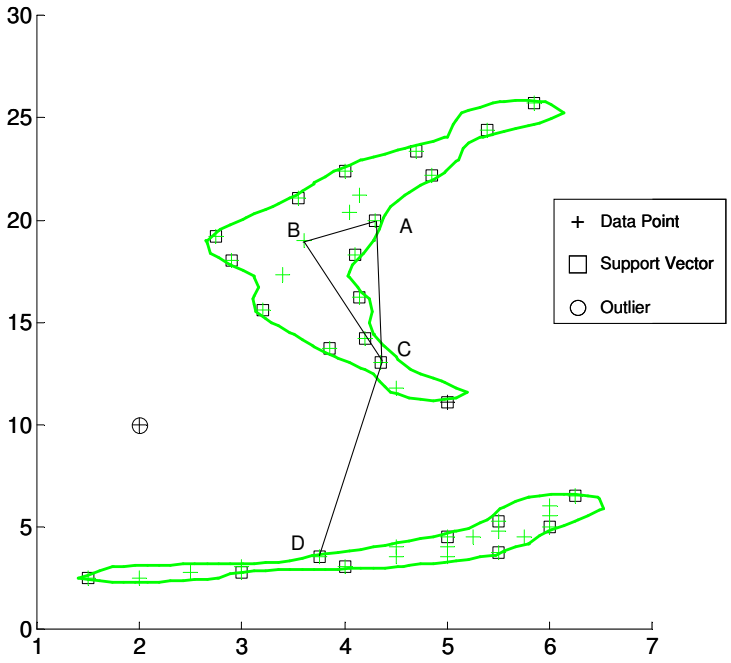


Figure 4.6: Points A and B are in the same cluster because segment AB stays within the domain boundary. Points B and C are in the same cluster for the same reason. Points A and C are in the same cluster via association through point B, even though segment AC leaves the boundary. Point D is in a different cluster than the other points.

The most straightforward approach is to build a complete adjacency matrix for all combinations of (non-outlier) points. Let $\mathbf{A}$ denote this matrix and A_{ij} denote whether points $\mathbf{x}_i$ and $\mathbf{x}_j$ are adjacent. Thus, A_{ij} is a binary variable such that

$$A_{ij} = \begin{cases} 1 & \text{if } R\left(\mathbf{x}_i + \lambda\left(\mathbf{x}_j - \mathbf{x}_i\right)\right) \le R, \quad \forall \lambda \in [0,1]; \\ 0 & \text{otherwise.} \end{cases} \tag{4.12}$$

Given the matrix $\mathbf{A}$, one labels clusters as connected components of the graph implied by $\mathbf{A}$. To evaluate A_{ij}, one can sample values of $\lambda \in [0,1]$. Several researchers report 10 to 20 samples being a reasonable level of resolution.

The preceding approach is called the Complete Graph (CG) method in the literature and is known to be fairly inefficient. Ben-Hur and coauthors (2001) suggest an

approximate method of evaluating only the A_{ij} such that $\mathbf{x}_i$ or $\mathbf{x}_j$ is a support vector. This approach, sometimes called the Support Vector Graph (SVG) method, is slightly faster than the CG method, but it sometimes can fail to detect that two clusters are in fact a single larger cluster. Yang and coauthors (2002) describe a labeling approach based on proximity graphs intended to be fast while solving this problem. Their approach is actually a family of methods because one can use different proximity graph (PG) models, such as k-nearest neighbors, Delaunay diagrams and minimum spanning trees. Their approach is among the faster of the adjacency-based approaches, but it can be difficult to implement and still can lead to classification errors. Lee and Lee (2005) describe a method for cluster labeling based on a gradient descent (GD) algorithm and the notion of stable equilibrium points. An advantage of their approach over the PG approach is that it tends to have a low error rate. However, because one must compute gradients the GD method is slow on high dimensional data sets. Lee and Daniels (2006) describe a method that does not rely on sampling a segment between pairs of points. Their method, called Cone Cluster Labeling (CCL), is approximate but is considerably faster than other methods.

For tradeoff modeling, one can apply any of the aforementioned methods. Although computational cost is not unimportant, the likely workflow for tradeoff modeling would require one to perform SVC only once when the model is created. That is, it is not something one must perform during design space exploration or optimization and, as such, it is not essential that one use the fastest algorithm possible. The CG algorithm is used in the current investigation.

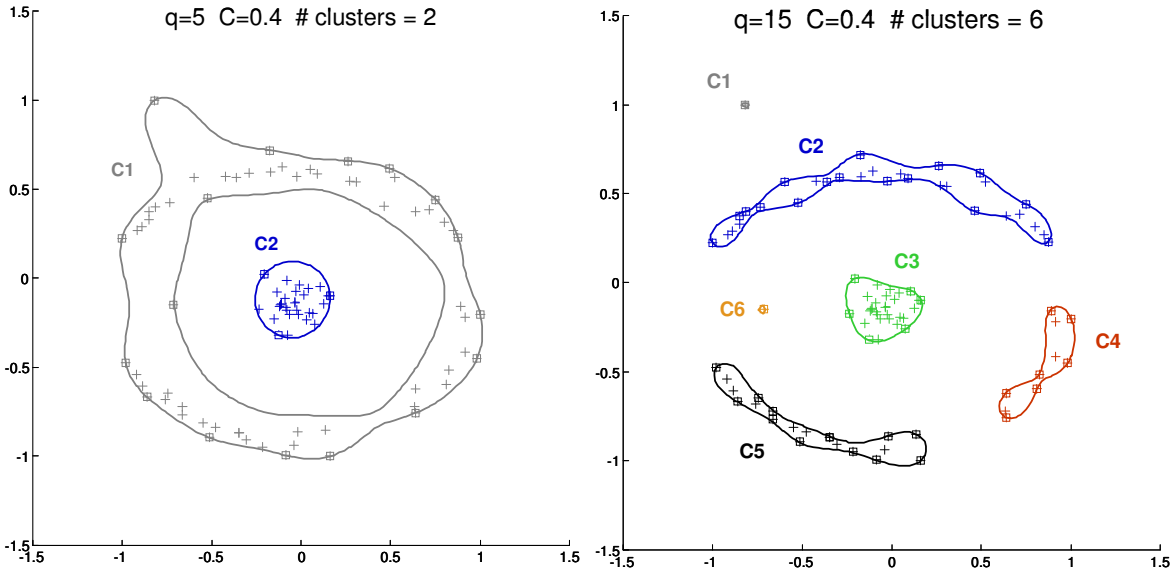
Table 4.1, adopted from (Lee and Daniels 2006), is a summary of the theoretical space and time bounds for each of the methods surveyed. In this notation, N is the total number of data points, N_{sv} is the number of support vectors, m is the number of samples made along segments between points (used to evaluate Equation (4.12)), N_{sep} is the number of stable equilibrium points (used only for the GD method) and k is the number of iterations for GD convergence. In all cases, $N \geq N_{sv}$ and $N \geq N_{sep}$. Usually, one chooses $10 \leq m \leq 20$.

Table 4.1: Worst-case asymptotic running times and memory requirements for several cluster labeling algorithms as reported by (Lee and Daniels 2006).

Method	***CG***	***SVG***	***PG***	***GD***	***CCL***
Adjacency Matrix Size	$O(N^2)$	$O(NN_{sv})$	$O(N^2)$	$O(N_{sep}^2)$	$O(N_{sv}^2)$
Total Asymptotic Time	$O(mN^2N_{sv})$	$O(mNN_{sv}^2)$	$O(N^2 + mNN_{sv})$	$O(mN^2(k + N_{sv}))$	$O(NN_{sv})$

4.3.3 Example

Figure 4.7 is a demonstration of SVC on the data set depicted in Figure 4.4. It contains results for different settings of the kernel width parameter. Figure 4.7(a) contains two large clusters. The larger value for the width parameter used to generate Figure 4.7(b) leads to a larger number of smaller clusters. There are six in all, two of which contain only one point and enclose a very small surrounding domain (C1 and C6). Clusters of such small size are candidates for being discarded as outliers. This issue is discussed in Section 4.4.



(a) (b)

Figure 4.7: SVC on the dataset from Figure 4.4 for two different settings of the kernel width parameter.

Figure 4.8 is a visualization of the adjacency information corresponding to the graph of Figure 4.7(a). From this, one can observe how the algorithm associates points into clusters. One typically would not visualize this information during model building. It is useful primarily for pedagogical and debugging purposes.

4.4 Creating Domain-Described Tradeoff Models

Figure 4.9 is a summary of the process for creating a domain-described predictive tradeoff models. This is an expansion of Step 5 of Figure 3.1 (domain description) into four sub-steps and a clarification of Step 6 (model fitting) in light of the possibility that the data contains multiple distinct clusters.

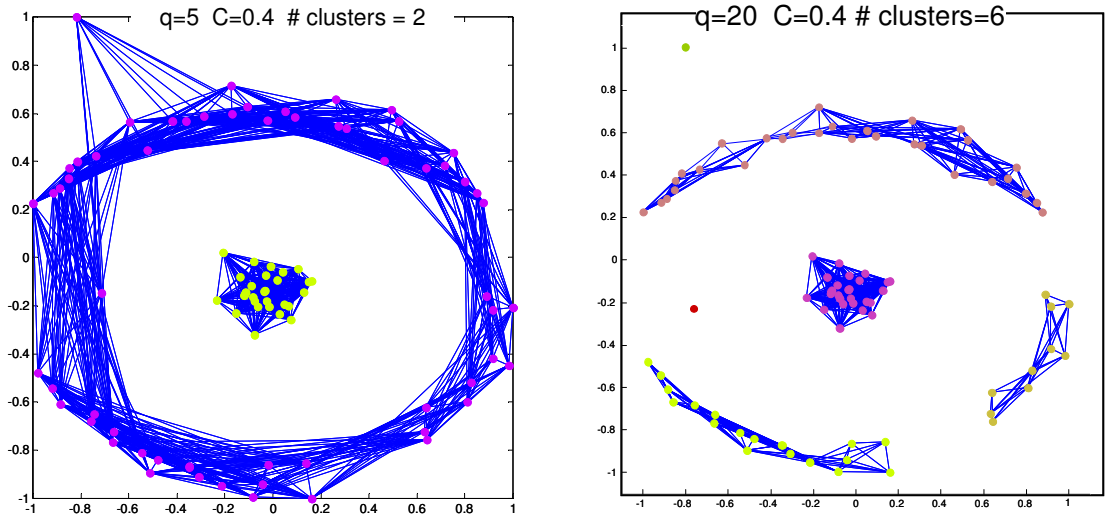


Figure 4.8: A visualization of the adjacency information corresponding to the graphs in Figure 4.7. Two distinct clusters are evident in (a). In (b), six distinct clusters are evident, including two that contain only one point each.

Step 5.1: Preprocessing. Scale raw data. Usually normalizing (map to [0,1] scale) or centralizing (map to [-1,1] scale) is effective.
Step 5.2: SVDD. Apply the SVDD method using a Gaussian kernel function. This step may require iteration if domain description is too loose or too tight (as indicated through visualization or other diagnostics). Can use adaptive methods to tune the Gaussian kernel width parameter, q.
Step 5.3: SVC. Apply SVC method (with cluster labeling method of choice). If too many clusters, repeat Step 2 with a smaller value for the kernel width parameter, q. Adaptation of q to achieve desired number of clusters can be automated.
Step 5.4: Eliminate degenerate clusters. Clusters with too few points can be considered as outliers. Either reduce the kernel width parameter, q, until those points become part of a larger cluster or drop them from the analysis as outliers.
Step 6: Model fitting. ▪ *If fitting a different model to each cluster*: Apply SVDD to each cluster individually to model sub-domains. Adjust width parameter, q, as necessary to obtain a single major cluster for each sub-domain (a few points may be lost as outliers). Fit models to each cluster of data. ▪ *If fitting one model to all data*: Fit models to all non-outlier data.

Figure 4.9: Refinement of process for generating tradeoff models (Figure 3.1) to include a generalized domain description procedure.

Step 5.1 involves scaling the data. Although one could skip this step, the algorithms involve in the process are more reliable when the data is scaled and relocated. Anecdotally, centralizing the data—scaling everything to the range [-1,1] and centering on the origin—appeared most effective during the course of this research. Normalization is another common scaling scheme.

Step 5.2 involves applying kernel-based SVDD using a Gaussian kernel, as described in Section 4.2.3. One may wish at this point to adapt the kernel width parameter, q, until an acceptable domain description is found. For example, one may want to reduce q if there is a larger proportion of support vectors relative to the number of data points (e.g. more than half the data are support vectors) because this may be an indication of overfitting. In the current context, it can be easier to iterate over both Steps 5.2 and 5.3 using the number of clusters as a driver for width parameter adaptation. The basic rule is to reduce q if the number of actual clusters is larger than the number desired.

Step 5.3 involves applying the SVC method in concert with the cluster labeling method of choice as described in Section 4.3. As already noted, the number of clusters can be a driver for adapting the kernel width parameter (either manually or through an automated numerical method). Whenever possible, visualization of the clusters and their domain boundaries is recommended as a final validation step.

Step 5.4 is necessary because it is common for SVC to yield clusters with very few points and a very small domain region. This occurs for example in Figure 4.7(b), where clusters C1 and C6 each contain only one data point. In such cases, one has two choices: (1) decrease parameter q until the points are absorbed into a nearby cluster or

(2) eliminate the data points as outliers. The first choice is appropriate when the degenerate cluster is close to another larger one. Otherwise, it is common practice to eliminate the data.

After constructing a domain description and identifying any disjoint clusters, one must fit one or more continuous models to the data. This step is discussed originally in Section 3.2. Here, this step is expanded to account for the additional consideration of how to model the data in each cluster. The appropriate action is one that balances fitting accuracy with convenience. One incurs additional organizational overhead by fitting several individual models, but often achieves a superior fit to each cluster of data. Regardless of how many individual models one fits, it is necessary to ensure that any optimization or design exploration activities consider all the relevant sub-domains. For a gradient-based search, it may be best to initiate multiple optimization runs, with at least one originating in each sub-domain. Typically, such solvers search only within a single contiguous domain and are unable to explore disjoint sub-domains.

If electing to fit a different tradeoff model to each sub-domain, one must obtain an independent description of the sub-domains. A practical approach to this is to repeat SVDD independently for each cluster. It also is advisable to repeat SVC and to adapt the width parameter such that a single cluster is obtained (i.e., a single domain region). It is possible to establish a hierarchical clustering (such that the original clusters are subdivided into additional clusters), but doing so would have value only if it leads to a fitted model with considerably greater accuracy or if the sub-domain would have been better as multiple clusters in the first place. Note that the application of SVDD and SVC

to the sub-domain data will proceed faster than the initial application because there are fewer data points involved.

Figure 4.10 is a 3D visualization of the centralized domain description identified for data about small four-stroke engines. The actual data domain is five-dimensional. In addition to mass, maximum power and maximum torque, the remaining dimensions are efficiency and maximum speed. The data is for commercially-available engines and is obtained from publicly-available data sources (e.g., spec sheets and catalogs) as well as data supplied by vendors and companies supporting this research. Engine retail price (the cost to the system designer) is predicted by a tradeoff model fit to the underlying data, and so is not part of the domain description (only inputs to the tradeoff model require a domain description).

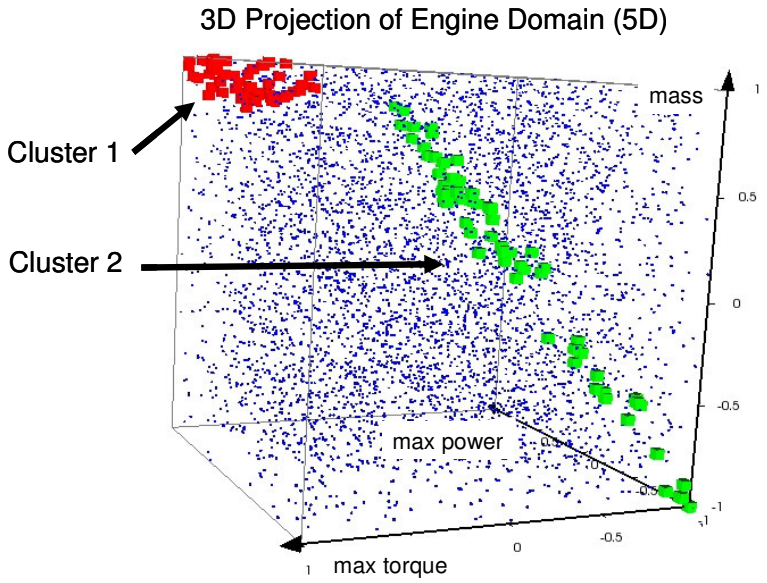


Figure 4.10: A 3D visualization of the 5D domain description for small four-stroke engines. This figure is generated by randomly sampling the 5D unit hypercube and classifying each point. Larger points belong to one of the clusters; smaller points are not part of the engine tradeoff model domain.

The figure is generated by randomly sampling the five-dimensional unit hypercube. Points that fall within one of the two sub-domains (clusters in the original data) are labeled appropriately. Points that do not fall within one of those regions are rejected and appear as small dots in the figure. Although the two sub-domains appear discontinuous due to the sampling, each one is in fact continuous.

4.5 Conclusions and Chapter Summary

The main conclusion of this chapter is that the kernel-based SVDD and SVC methods are an appropriate and logical basis for solving the tradeoff model domain description problem. The approach, summarized in Section 4.4, is based on a well-founded mathematical theory for domain description and clustering. This theory is presented in Sections 4.2 and 4.3. The domain description procedure also fits directly within the overall model generation procedure from Section 3.2.

The kernel-based SVDD and SVC methods have several advantageous characteristics. By virtue of being kernel-based, they are capable of representing complex data domains. This includes domains that are non-convex or contain disjoint sub-regions. They also provide designers with the means to subdivide a data set into multiple independent tradeoff models should doing so improve the model fit. Several data sets encountered during the course of this research exhibit such complex input domains (e.g., the engine data of Figure 4.10). Consequently, these characteristics are important in practice.

Perhaps the most important characteristic is that it is fast to test whether a candidate point is in the domain. This is important because typically this must occur repeatedly within an optimization loop. The computation entails evaluating an algebraic

equation involving several weighted sums and kernel functions (Equation (4.11)). Although this computation would become slow for an extremely large numbers of support vectors, this situation has not been observed in this research and it seems highly unlikely to occur.

Algorithms for constructing the domain description and clusters do not scale well in the number of data points and number of attributes, but designers need only apply them during model generation. Moreover, the slower algorithms never need be part of a design exploration or optimization routine.

This chapter contains several synthetic examples to illustrate the concepts and provide a basic level of support for its effectiveness. Although these are successful, they are inadequate for one to judge fully the practical usefulness of the approach. Further empirical evidence comes in later chapters. Neither the log splitter design problem of Chapter 6 nor the hybrid vehicle problem of Chapter 7 would be solvable using tradeoff models without effective domain descriptions. Both problems involve tradeoff models fit to data with non-convex domains and the log splitter example involves a domain comprised of two disjoint clusters.

CHAPTER 5:

A MATHEMATICAL ANALYSIS

OF COMPOSING PARAMETERIZED PARETO SETS

The focus of the preceding chapters has been on individual tradeoff models: how to construct them, formulate decisions in terms of them and describe the domain over which they are valid. This chapter, along with the two that follow, is an investigation of system-level decision making using multiple tradeoff models. The vision is that designers can model a system using tradeoff models for its components along with a model for how they interact to yield system-level attributes.

This chapter addresses the third research question:

RQ3. Under what conditions can designers compose component-level tradeoff models in order to model a system-level decision alternative soundly?

The focus is on parameterized Pareto dominance, which designers use to eliminate ostensibly irrelevant data prior to fitting a tradeoff model. Whether these eliminations are appropriate is critical, since dominated data are left out of the final model. Designers must be confident that applying parameterized Pareto dominance will not lead them to eliminate data about solutions that could be the most preferred.

It is not immediately obvious whether parameterized Pareto dominance is sound in this sense. One applies parameterized Pareto dominance to each component individually when creating its tradeoff model. However, the attributes of a system depend on the interactions between its components. For example, one combination of gearbox

and motor may be effective while the same motor performs poorly when connected to a different gearbox.

This chapter contains a mathematical analysis of parameterized Pareto dominance in the context of composed system models. Using set theory and decision theory, it is shown that composing parameterized Pareto sets is sound under mild assumptions about how the components of a system interact. In particular, the following hypothesis is evaluated:

H3. One can compose predictive tradeoff models soundly if the tradeoff models are based on parameterized Pareto sets and all induced preferences for any component-level dominator attribute are monotonic in the same direction.

In this context, soundness refers to the informational completeness of the parameterized efficient set. The argument is that it contains all the component-level information one requires to search for the most preferred system-level solution. Similar soundness criteria are derived for classical Pareto dominance, and these are shown to be a subset of those for parameterized Pareto dominance. The main conclusion of this chapter is that parameterized Pareto dominance is a sound basis for generating tradeoff models.

The chapter organization is as follows. Section 5.1 is a review of the motivations for composing tradeoff models. Section 5.2 contains a mathematical definition for the compositional modeling framework. Section 5.3 is a theoretical analysis of the soundness of composing classical and parameterized Pareto sets in this framework.

5.1 Motivations for Composing Tradeoff Models

Tradeoff model composition is an appealing capability. One challenge of system-level decision making is the question of how to incorporate lower-level considerations

into a decision without having an explosion in problem complexity. The approach of composing tradeoff models addresses this directly. Component designers can formalize the capabilities and limitations of the various implementations of their components into a tradeoff model for use by system-level designers. This conveys to system designers the implications of the lower-level details in a quantitative way without encumbering them with the actual details.

In the context of data-driven predictive modeling, composition is essential to enable designers to model novel systems at a systems level. It is impossible for designers to construct a model of such a system from observational data because no prior implementations of it exist. It may be possible for designers to rely on data sampled from an existing model of the system, but this approach is questionable for similar reasons—if the system is sufficiently novel, would designers already have a detailed model during system-level decision making? Even if designers sometimes would have such a model, more often they would not and must rely on a compositional approach.

Model reuse is another major benefit of tradeoff model composition. The more often designers expect to use a particular model, the more easily they can justify devoting resources to developing, validating and maintaining it. It also is not far-fetched to imagine components manufacturers developing tradeoff models of their own components for use by potential clients. In this sense, a tradeoff model would be a computable version of a components catalog.

5.2 Mathematical Framework for Composition

Figure 5.1 is an illustration of the compositional modeling framework under consideration. At the lowest level are tradeoff models, $T_k(\cdot)$ for $k = 1 \ldots K$, that abstract

the characteristics of the non-dominated implementations of the system components. Let $\mathbf{y}_k$ denote the attribute vector associated with the k^{th} component and $\tilde{\mathbf{y}}_k$ denote a vector consisting of the attributes used as inputs to the tradeoff model. Thus, $\mathbf{y}_k = \left[\tilde{\mathbf{y}}_k, T_k\left(\tilde{\mathbf{y}}_k\right)\right]$ for $k = 1 \ldots K$. Also, let $\mathbf{y} = \left[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K\right]$ denote a vector comprised of all component-level attributes and let M denote the dimensionality of this vector. A vector-valued function, $\mathbf{S}(\cdot) = [S_1(\cdot), S_2(\cdot), \ldots, S_N(\cdot)]$, relates the component-level attributes to the system-level attributes, denoted z_i for $i = 1 \ldots N$. Thus, $z_i = S_i(\mathbf{y})$ and $\mathbf{z} = \mathbf{S}(\mathbf{y})$. The value function, $V(\cdot)$, is a formalization of designer preferences for the system-level attributes, $\mathbf{z} = \left[z_1, z_2, \ldots, z_N\right]$.

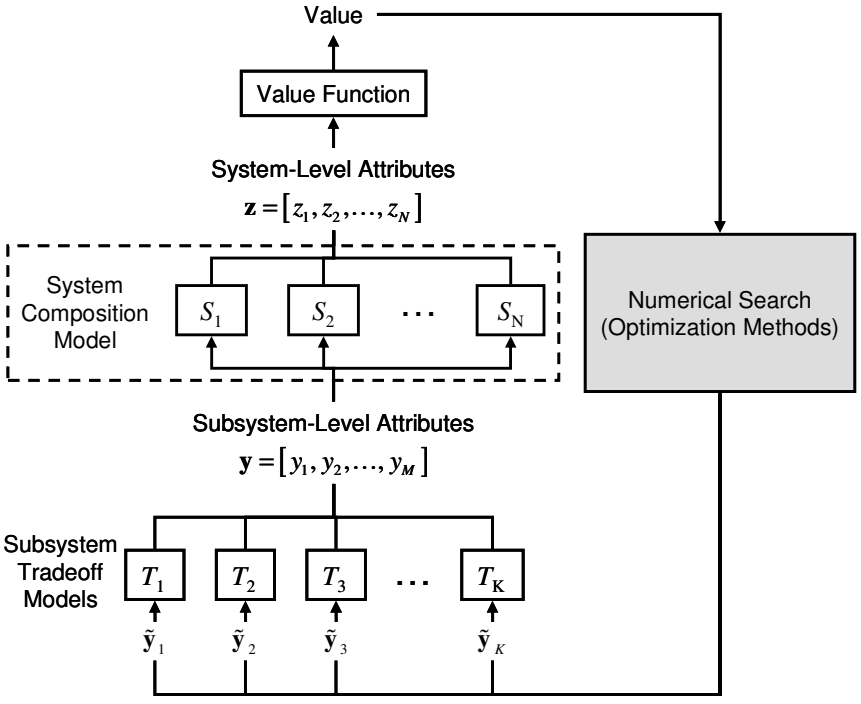


Figure 5.1: Tradeoff model composition framework.

Given this notation, one can express the value of the l^{th} system alternative as:

$$v_l = V\left(\mathbf{S}_l\left(\left[\tilde{\mathbf{y}}_l, T_{l1}\left(\tilde{\mathbf{y}}_{l1}\right), T_{l2}\left(\tilde{\mathbf{y}}_{l2}\right), \ldots, T_{lK}\left(\tilde{\mathbf{y}}_{lK}\right)\right]\right)\right), \tag{5.1}$$

where $\mathbf{S}_l$ denotes the composition models for the l^{th} system-level decision alternative, $\tilde{\mathbf{y}}_l = \left[\tilde{\mathbf{y}}_{l1}, \tilde{\mathbf{y}}_{l2}, \ldots, \tilde{\mathbf{y}}_{lK}\right] \in \tilde{\mathbf{Y}}_l$ is a vector consisting of the component-level attributes used as inputs to their respective tradeoff models and $\tilde{\mathbf{Y}}_l = \tilde{\mathbf{Y}}_{l1} \times \tilde{\mathbf{Y}}_{l2} \times \cdots \times \tilde{\mathbf{Y}}_{lK}$ is the set of feasible tradeoff model input vectors for alternative l. Note that one determines the $\tilde{\mathbf{Y}}_{lk}$ using the domain description procedure from Chapter 4.

Using Equation (5.1), one can formulate requirements allocation and system selection decision problems. These are direct extensions of the corresponding formulations from Section 3.3.2. For the l^{th} system-level alternative, one can predict the most preferred requirements allocation as:

$$\mathbf{y}_l^* = \left[\tilde{\mathbf{y}}_l^*, T_{l1}\left(\tilde{\mathbf{y}}_{l1}^*\right), T_{l2}\left(\tilde{\mathbf{y}}_{l2}^*\right), \ldots, T_{lK}\left(\tilde{\mathbf{y}}_{lK}^*\right)\right], \tag{5.2}$$

where

$$\tilde{\mathbf{y}}_l^* = \arg\max_{\tilde{\mathbf{y}}_l \in \tilde{\mathbf{Y}}_l} V\left(\mathbf{S}_l\left(\left[\tilde{\mathbf{y}}_l, T_{l1}\left(\tilde{\mathbf{y}}_{l1}\right), T_{l2}\left(\tilde{\mathbf{y}}_{l2}\right), \ldots, T_{lK}\left(\tilde{\mathbf{y}}_{lK}\right)\right]\right)\right). \tag{5.3}$$

Given this, the most preferred system alternative is:

$$l^* = \arg\max_{l=1\ldots L} V\left(\mathbf{S}_l\left(\mathbf{y}_l^*\right)\right). \tag{5.4}$$

One limitation of this notation is that different system configurations can have different numbers of components. Thus, the parameter K in the above notation really should be indexed according to the system number—i.e., we should use K_l for $l = 1 \ldots L$.

However, this detail is omitted to make the notation less cumbersome. It is inconsequential with respect to the analysis in the next section.

5.3 Soundness Conditions for Composing Non-Dominated Sets

The procedure defined in Equations (5.2) through (5.4) incorporates an assumption that warrants scrutiny. If designers fit tradeoff models to data identified using a component-level optimality criteria—a dominance analysis performed on data about component implementations—how can they be sure that some combination of tradeoffs that appear dominated at the level of their respective components will not yield the most preferred solution from a system-level perspective?

If it is valid to compose tradeoff models, then the answer to this question must be that it can never happen—that if a component tradeoff is eliminated by a dominance analysis, then it also would be eliminated when considered at higher levels in the system hierarchy. Whether this is the case depends on how component-level attributes relate to system-level attributes through the system composition model, $\mathbf{S}(\cdot)$. The aim in this section is to identify mathematical conditions that guarantee validity in this sense for both classical and parameterized Pareto dominance.

The analysis is limited to the idealized case of dealing directly with a non-dominated set, as opposed to a tradeoff model fit to it. However, the results are an important justification for trying to generalize a non-dominated set in the first place. If it could be that the most preferred tradeoff is not in this set, then the use of tradeoff models surely is unwarranted.

5.3.1 Soundness Condition for Composing Classical Pareto Sets

In the following, it is shown that it is sound to model a component using a classical Pareto set *provided the system composition model is strictly monotonic in all of the component's attributes*. Building on this result, one can reason that designers can compose classical Pareto sets *provided the system composition model is strictly monotonic in all attributes from all components*. This is a restrictive assumption and motivates the use of parameterized Pareto dominance, further considered in Section 5.3.2.

Since a system composition model, $\mathbf{S}(\cdot)$, is a vector-valued function, it is necessary to clarify the meaning of monotonicity in this context. It is useful to begin with the definition of monotonicity for a scalar function. Note that what follows adopts the convention used in (Keeney and Raiffa 1993) of denoting an attribute in the abstract using an uppercase letter and denoting a specific value for that attribute using a lowercase letter (e.g., if Y is the attribute "gear ratio", y would be 10.72 or some other numeric value).

Definition 5.1 (Strictly Increasing in Y_k, scalar function): A function $S:\mathbb{R}^M \to \mathbb{R}$ is strictly monotonically increasing in Y_k if for $\mathbf{y}',\mathbf{y}'' \in \mathbb{R}^M$, $y'_j = y''_j \quad \forall j = 1\ldots M, j \neq k$ and $y'_k > y''_k$ implies $S(\mathbf{y}') > S(\mathbf{y}'')$.

This discussion is restricted to increasing functions without loss of generality; if $S(y)$ is decreasing in Y, it is increasing in the negation of Y. One can extend the preceding definition to vector-valued functions as follows:

Definition 5.2 (Strictly Increasing in Y_k, vector function): A function $\mathbf{S}:\mathbb{R}^M \to \mathbb{R}^N$ such that $\mathbf{S}=\left[S_1(\cdot),S_2(\cdot),\ldots,S_N(\cdot)\right]$ is strictly increasing in Y_k if every $S_i(\cdot)$ that is a function of Y_k is strictly increasing in S_k.

Figure 5.2 is an illustration of different monotonicity scenarios for a system composition model, $\mathbf{S}=\left[S_1,S_2,S_3\right]$. The model is strictly increasing in Y_1 because S_1 and S_2 are strictly increasing in Y_1 and S_3 is not a function of Y_1. However, $\mathbf{S}$ is not strictly increasing in either Y_2 or Y_3. For Y_2, this is evident in that S_2 non-monotonic in Y_2. The situation for Y_3 is more complicated, but essentially the same. The function S_2 is monotonically decreasing in Y_3, but the function S_3 is monotonically *increasing* in Y_3. Consequently, the overall effect is non-monotonic. No simple reformulation of the problem can avoid this and, as shall be proved in the following, one cannot compose classical Pareto sets soundly under these circumstances.

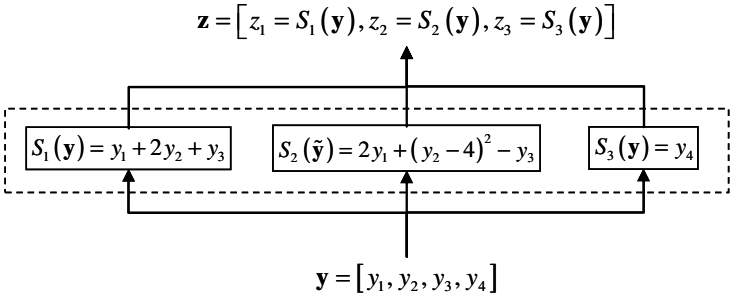


Figure 5.2: Different monotonicity possibilities for a system composition model.

To formalize conditions for when composing classical Pareto sets is mathematically sound, some additional notation is needed. Let $\mathbf{Y}_k$ and $\mathbf{Y}_k^*$ denote,

respectively, the feasible and non-dominated sets corresponding to component $k = 1 \ldots K$ and $\mathbf{Y} = \mathbf{Y}_1 \times \mathbf{Y}_2 \times \cdots \times \mathbf{Y}_K$ denote the set of all combinations of feasible component-level points. Also, let $\mathbf{Y}_{-k} = \mathbf{Y}_1 \times \cdots \times \mathbf{Y}_{k-1} \times \mathbf{Y}_{k+1} \times \cdots \times \mathbf{Y}_K$ denote the set of all combinations of feasible component-level points *except* those from the k^{th} component and $\mathbf{Y}^*_{-k}$ denote the analogously-defined non-dominated set. Finally, we use the notation $\mathbf{y} = [\mathbf{y}_k, \mathbf{y}_{-k}]$ such that $\mathbf{y}_k \in \mathbf{Y}_k$ and $\mathbf{y}_{-k} \in \mathbf{Y}_{-k}$ to mean that $\mathbf{y} \in \mathbf{Y}$.

Given this notation, one can make three related statements about the validity of using classical Pareto sets in a compositional modeling framework. As before, let $\mathbf{y}' \text{DOM} \, \mathbf{y}''$ denote that $\mathbf{y}'$ dominates $\mathbf{y}''$ by the classical Pareto criterion.

Theorem 5.1: If a system composition model, $\mathbf{S}: \mathbb{R}^M \to \mathbb{R}^N$, *is strictly increasing in all inputs originating from the* k^{th} *component and* $\mathbf{y}'_k \, \text{DOM} \, \mathbf{y}''_k$ *for* $\mathbf{y}'_k, \mathbf{y}''_k \in \mathbf{Y}_k$, *then there exist* $\mathbf{y}'_{-k}, \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$ *such that* $\mathbf{S}(\mathbf{y}') \text{DOM} \, \mathbf{S}(\mathbf{y}'')$, *where* $\mathbf{y}' = [\mathbf{y}'_k, \mathbf{y}'_{-k}] \in \mathbf{Y}$ *and* $\mathbf{y}'' = [\mathbf{y}''_k, \mathbf{y}''_{-k}] \in \mathbf{Y}$.

This means that under the assumption of a strictly increasing system composition model, it is guaranteed that a solution dominated at the component level also is dominated at the system level. In concrete terms, designers can, in principle, eliminate an alternative for a motor without considering the load it must drive. A proof of Theorem 5.1 is given in Appendix A. Two related corollaries follow (also proved in the appendix).

Corollary 5.1: Theorem 5.1 holds for $\mathbf{y}'_{-k}, \mathbf{y}''_{-k} \in \mathbf{Y}_{-k}$.

Whereas Theorem 5.1 is expressed in terms of classical Pareto sets—i.e., $\mathbf{y}'_{-k}, \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$—Corollary 5.1 is in terms of the corresponding feasible set. This means that applying classical Pareto dominance at a component level can be valid independent of whether one applies it to the other components in a system model. From a practical perspective, this indicates that one can combine classical Pareto sets or tradeoff models based on them with more traditional engineering models (i.e., ones that compute attributes from design variables).

Another useful corollary of Theorem 5.1 is that one can extend it to the system-wide scenario.

> *Corollary 5.2: If a system composition model,* $\mathbf{S}: \mathbb{R}^M \to \mathbb{R}^N$, *is strictly increasing in every component-level attribute, then Theorem 3 holds for all components,* $k = 1 \ldots K$.

This is a generalization of Theorem 5.1 to establish that one can compose classical Pareto sets whenever the system composition model is strictly increasing in all inputs. Although this indicates that it is possible for one to compose classical Pareto sets, the monotonicity requirement is restrictive in practice. Many system composition models resemble those of Figure 5.2. For example, in the gearbox design problem of Chapter 3, gear ratio is an attribute that relates to the system-level decision attributes through a non-monotonic composition model of engine and vehicle dynamics (i.e., it is analogous to Y_2 from Figure 5.2). The log splitter example (Chapter 6) also violates the strict monotonicity requirement. Larger cylinder bore is preferred to improve in a force-maximization objective, but smaller bore is preferred to improve in a speed-maximization objective

(i.e., it is analogous to Y_3 in Figure 5.2). These are relatively simple engineering problems and there is no reason to expect such circumstances would not exist for more complicated ones. Designers need an approach that is generally sound under realistic conditions.

5.3.2 Soundness Condition for Composing Parameterized Pareto Sets

Parameterized Pareto dominance does not have the limitations of the classical rule, provided one classifies the attributes correctly as parameters or dominators. Appendix A contains proofs for the following statements.

Theorem 5.2: If a system composition model, $\mathbf{S}:\mathbb{R}^M \to \mathbb{R}^N$, *is strictly increasing in inputs that are dominator attributes of the* k^{th} *component and* $\mathbf{y}'_k \,\text{PDOM}\, \mathbf{y}''_k$ *for* $\mathbf{y}'_k, \mathbf{y}''_k \in \mathbf{Y}_k$, *then there exist* $\mathbf{y}'_{-k}, \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$ *such that* $\mathbf{S}(\mathbf{y}')\,\text{DOM}\,\mathbf{S}(\mathbf{y}'')$, *where* $\mathbf{y}' = [\mathbf{y}'_k, \mathbf{y}'_{-k}] \in \mathbf{Y}$ *and* $\mathbf{y}'' = [\mathbf{y}''_k, \mathbf{y}''_{-k}] \in \mathbf{Y}$.

This has an interpretation similar to that of Theorem 5.1, with the main distinction being the application of PDOM at the component level. In terms of the example of Figure 5.2, Y_1 would be a dominator attribute while Y_2 and Y_3 are parameters. Like Theorem 5.1, the practical consequence of Theorem 5.2 is that designers can eliminate component-level alternatives soundly without considering how the properties of other components in the system. However, the monotonicity requirements for doing this are much less strict in this case. In fact, the circumstances in which using parameterized Pareto dominance is sound subsume those for the classical rule. This is evident in that Theorem 5.2 reduces to Theorem 5.1 in the event that all attributes from the k^{th} component are dominators.

One can extend Theorem 5.2 in a way that parallels the corollaries associated with Theorem 5.1:

Corollary 5.3: Theorem 5.2 holds for $\mathbf{y}'_{-k}, \mathbf{y}''_{-k} \in \mathbf{Y}_{-k}$.

Corollary 5.4: If a system composition model, $\mathbf{S}: \mathbb{R}^M \to \mathbb{R}^N$, *is strictly increasing in all component-level dominator attributes, then Theorem 5.2 holds for all components,* $k = 1 \ldots K$.

According to Corollary 5.3, it is sound to use parameterized Pareto dominance to model a particular component regardless of how one models other components provided the stated monotonicity property holds. Corollary 5.4 is an extension of Theorem 5.2 to all components in a system.

5.3.3 Practical Implications of the Mathematical Results

Designers cannot perform dominance analysis at the component level without some understanding of the system composition model. This is evident in the monotonicity assumptions required to make the preceding propositions work. Essentially, one cannot draw any conclusions about the soundness of composing non-dominated sets without saying something about how they are composed.

Ultimately, this requirement is not oppressive. Monotonicity is a relatively simple property that designers often can verify by inspecting their models. Designers who wish to develop reusable tradeoff models—which necessarily must be developed without a specific composition model in mind—can consider the odds of a particular type of attribute being used in a particular type of model. For example, cost, mass and reliability will, for most systems, map monotonically from the subsystem to the system level and

therefore designers can use those attributes as dominator attributes in the parameterized Pareto dominance rule. In cases where an attribute is likely to undergo a non-monotonic transformation (e.g., gear ratio, cylinder bore) or no clear usage is evident, designers can treat an attribute as a parameter. The risk of designers misusing a tradeoff model is low because they can validate any monotonicity assumptions incorporated into the model at the time of use.

The main practical consequence of the preceding theoretical analysis is that classical Pareto dominance is very limiting in a compositional modeling context. Only in cases where one can treat every one of a component's attributes as dominator attributes is this dominance rule helpful. This assumption appears unlikely to hold in most cases.

It is important to stress that the classical and parameterized dominance rules are not competing modeling formalisms. Parameterized Pareto dominance is a generalization of the classical rule, completely subsuming the situations in which the classical rule is valid. Thus, classical Pareto dominance *is* parameterized Pareto dominance for the case of no parameter attributes (i.e., when $P = \varnothing$). Although one could use the classical rule outside its soundness conditions, it is unclear what one would gain from this. To do so would require one to make aggressive assumptions about preferences for some of the attributes and could lead to unsound decisions. One can apply parameterized Pareto dominance without these assumptions and with the guarantee of a sound decision process.

5.4 Conclusions and Chapter Summary

The study reported in this chapter provides a firm theoretical basis for modeling system-level design alternatives by composing tradeoff models based on parameterized

efficient sets. Under reasonable assumptions, the parameterized efficient set for a component is guaranteed to contain the most preferred solution according to system-level preferences. The formal statements in this chapter are novel contributions to the study of dominance analysis and tradeoff modeling. Based on these, one can draw three important conclusions:

First, designers can develop component tradeoff models *independently of one another*. This is evident from Theorem 5.2, which includes no assumptions about subsystems other than the one being considered. At a glance, one may be inclined to believe this should not be so on the basis that the solution that is most preferred in one component depends on what tradeoffs are available in the other components as well as the way in which the components interact (via the system composition models). However, the parameterized Pareto set construct enables designers to identify the set of all tradeoffs for a component that *could be* the most preferred rather than identifying one individual solution.

Second, designers can develop component tradeoff models *without specific knowledge about the system-composition model* with which it will be used. Theorem 5.2 (and Corollary 5.4) requires only that a general property for the system composition model holds for certain attributes. Although this requires designers constructing tradeoff models to have some general insight into a particular problem domain, it does not require them to know exactly how a given system will be modeled. For example, it is sufficient for designers to know that increasing the mass of a subsystem leads to an increase in system mass—they need not know whether increasing the subsystem mass also requires enlarging a structural component elsewhere, thus adding to the increase at the system

level. It is reasonable to believe designers have this sort of insight, as it is similar to the type of knowledge upon which experts rely when applying traditional design methods.

Third, designers can develop component tradeoff models *independently of specific system-level preferences*. As with system composition models, designers require some level of insight into a system or application domain but problem-specific knowledge is unnecessary. Put simply, a designer constructing a tradeoff model need not know the degree to which a $90 pump is preferred to a $100 pump (all other factors being equal)—only that it is safe to assume that less cost is preferred to more.

Strictly speaking, the mathematical results of this chapter apply only to the case of composing parameterized efficient sets. However, one can infer that composing predictive tradeoff models, which are fit to parameterized efficient set data, can be effective. The next two chapters contain example problems that provide empirical evidence to this effect.

CHAPTER 6:

REQUIREMENTS ALLOCATION FOR A HYDRAULIC LOG SPLITTER

This is the first of a pair of example problems aimed at providing evidence that the tradeoff modeling approach is useful in engineering practice. The main conclusion of the previous chapter is that reasonable conditions exist under which it is mathematically sound to compose parameterized efficient sets. However, the main idea of tradeoff modeling is to generalize from the finite data sample using predictive models. These models preclude any mathematical guarantees and, consequently, empirical study is required.

This chapter addresses requirements allocation decisions using a hydraulic log splitter design example. A database of hydraulics and related components is assembled and tradeoff models are generated according to the procedure outlined in Figure 3.1. System composition models that describe the interactions among the various components are identified and used to relate component-level attribute to system-level attributes. Decision results obtained using the tradeoff modeling approach are compared to an exhaustive search of the components database. Although one would not expect both approaches to yield precisely the same decisions, comparing the two is instructive.

Chapter organization is as follows. Section 6.1 is an overview of the tradeoff model generation procedure for the hydraulics components. Section 6.2 is a description of designer preferences for the requirements allocation problem. Section 6.3 is a description of the system composition model for the log splitter system. Section 6.4 is a comparison

of the decision obtained using tradeoff models to that obtained via an exhaustive search of the components database.

6.1 Generating Tradeoff Models for Hydraulics Components

To demonstrate the tradeoff modeling methodology, a library of tradeoff models for hydraulics components is generated. The library consists of tradeoff models for gear pumps, directional control valves, cylinders and engines. Tradeoff models are generated using data primarily from publicly-available data sheets and catalogs, with the remainder obtained from corporate partner companies or their venders. All of the pricing data is for similar purchase quantities and, whenever necessary, the data has been "anonymized" to protect proprietary interests. Table 6.1 (next page) is a summary of the scope of the components in the database and Table 6.2 (page 140) is a summary of the results from data analysis and model fitting.

A number of points are removed prior to fitting according to standard data cleansing and analysis practices. A vast majority of the data had nearly the same maximum operating pressure, which rendered that attribute uninformative from a prediction standpoint. Accordingly, any components with a maximum pressure below this level—about 172 bar (2500 psi)—were eliminated and that attribute was removed from the analysis. A similar observation applies to the engine speed data: the speed at maximum power was the same for most engines in our database (3600 rpm), and the same was true for speed at maximum torque (2500 rpm). Consequently, any engine data deviating from these marks by more than 100 rpm was eliminated and the tradeoff

Table 6.1: Summary of hydraulic component database.

Component	*Description*	*Attribute*	*Symbol*	*Min*	*Max*	*Units*
Pump	Single-stage gear pump with relief valve	Cost	c_{pump}	213	859	$
		Mass	m_{pump}	2.26 (4.98)	20.5 (45.2)	kg (lb)
		Displacement	V_g	1.18 (0.072)	48 (2.93)	cm^3/rev (in^3/rev)
		Max. op. pressure	$\Delta p_{\text{max,pump}}$	120 (1740)	250 (3625)	bar (psi)
		Max. op. speed	$n_{\text{max,pump}}$	3000	4000	rpm
		Efficiency (total)	η	0.44	0.92	-
Cylinder	Dual-acting medium- or heavy-duty.	Cost	c_{cyl}	57	404	$
		Mass	m_{cyl}	25.3 (11.47)	390 (177)	kg (lb)
		Stroke length	L_{cyl}	0.2 (8)	1.52 (60)	m (in)
		Bore diameter	b_{cyl}	0.038 (1.5)	0.127 (5)	m (in)
		Max. op. pressure	$\Delta p_{\text{max,cyl}}$	172 (2500)	207 (3000)	bar (psi)
Directional Control Valve (DCV)	Manual, spool-type, three-way closed center or four-way closed center (w/ open position) w/ load-side relief valve or detent	Cost	c_{dcv}	70	168	$
		Mass	m_{dcv}	7 (15.4)	16 (35.3)	kg (lb)
		Max. op. flow rate	Q	60.6 (16)	113.6 (30)	l/min (gal/min)
		Max. op. pressure	$\Delta p_{\text{max,dcv}}$	138 (2000)	310 (4500)	bar (psi)
Engine	Internal combustion engine (gasoline-fueled)	Cost	c_{eng}	180	1907	$
		Mass	m_{eng}	3.4 (7.5)	58.5 (129)	kg (lb)
		Max. power output	$P_{\text{max,eng}}$	0.75 (1.0)	18.6 (25.0)	kW (hp)
		Speed at max. power output	$n_{\text{eng,maxP}}$	3600	7500	rpm
		Max. torque output	$\tau_{\text{max,eng}}$	1.08 (0.8)	55 (40.6)	Nm (lb-ft)
		Speed at max. torque output	$n_{\text{eng,maxT}}$	2200	5500	rpm

Table 6.2: Summary of data eliminations and tradeoff model generation.

Component	*Engine (A)*	*Engine (B)*	*Pump*	*Cylinder*	*DCV*
Total # in DB	59		61	188	36
# after outlier analysis	49		43	158	32
# after dominance analysis	14	5	24	137	8
Validation results: $\sqrt{MSE}$ (% of mean)	45.8 (15%)	33.2 (2%)	3.63 (1%)	14 (9%)	14.2 (13%)

Notes:
- Kriging interpolation used for all tradeoff models
- All tradeoff models predict cost as a function of the other attributes
- Engine split into two models after clustering analysis

models did not incorporate this attribute. A small percentage of components were eliminated on the basis of being outliers or appearing suspect in some way (e.g., unusually high or low price for the stated performance attributes).

Parameterized Pareto dominance is applied after making eliminations based on the above grounds. This accounts for the largest number of eliminations (see Table 6.2). Applying the tradeoff modeling domain description approach (Figure 4.9) does not result in further eliminations, but SVC results indicate that two disjoint clusters exist in the engine data. It was determined that considerably better tradeoff model accuracy could be had by fitting independent models to each sub-domain. This was deemed worthwhile, despite the expense of requiring two independent optimization searches to allocate requirements for the system (one using each engine tradeoff model).

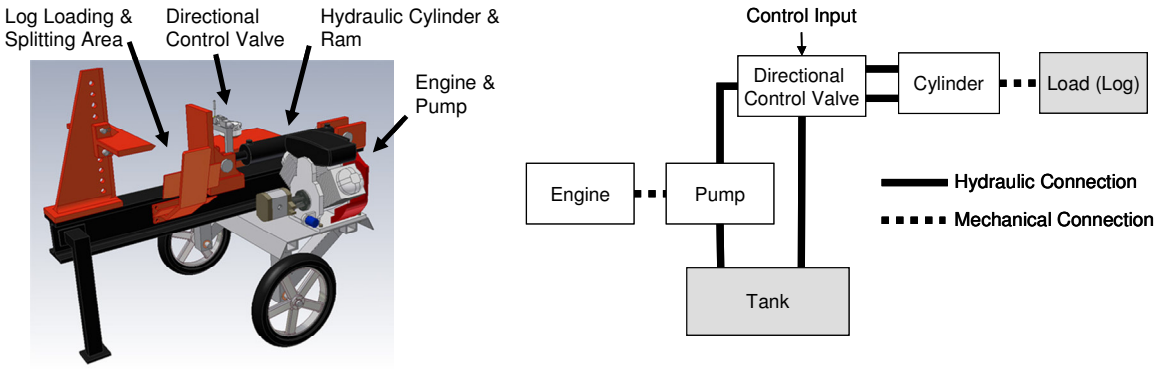
The tradeoff model for each component is formulated to predict cost as a function of its other attributes. Kriging methods and the DACE Matlab Kriging Toolbox (Lophaven, et al. 2002) are used to fit the tradeoff models. Validation is conducted using leave-one-out cross validation (Witten and Frank 2005).

6.2 Hydraulic Log Splitter System and Design Preferences

A log splitter is a system that divides a roughly cylindrical log into two or more pieces, typically in association with the harvesting of firewood. Several physical configurations are possible, but the current example is limited to a horizontal-acting type (Figure 6.1). An operator loads a log into the system and then operates a control to drive a wedge into the log. The wedge action is aligned with the grain of the wood, so minimal effort is required after initiating the split. Critical requirements include portability (typically light weight, has wheels for transport, etc.), cost and splitting capabilities (maximum size of log it can handle, maximum force it can apply at wedge, etc.).

As a first step in formulating the decision problem, it is useful to identify an objectives hierarchy. This is given in Figure 6.2. Each leaf of the tree associates with an attribute the one must compute:

- Cost: Sum of the purchase prices of the hydraulic components and the engine (which are costs to a system designer). Assembly or other cost factors are not considered in this example.
- Mass: Sum of the masses of the hydraulic components and the engine. Structure weight is not considered in this example.
- Ram Force: Maximum force the system can apply to the log.
- Log Length: The maximum length of log that will fit into the system.
- Cycle Time: An index for how long it takes to split a log. Defined as the time for the wedge to extend 0.15 meters (6 inches) at maximum engine torque (i.e., maximum ram force) plus the time to retract it with the engine running at maximum power (a conservative approximation of maximum ram speed).



(a) (b)

Figure 6.1: A hydraulic log splitter: (a) physical layout, (b) functional configuration, where white boxes correspond to components modeled in this example using tradeoff models.

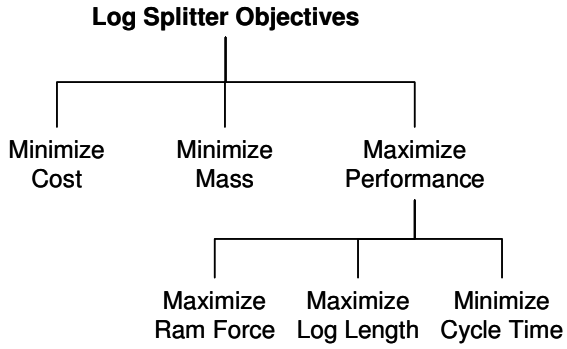


Figure 6.2: Objectives hierarchy for the log splitter problem.

Figure 6.3 contains graphs of the individual value functions that represent designer preferences corresponding to the individual system attributes[j]. Preferences for tradeoffs are elicited in a hierarchical fashion by first eliciting a value function for the three performance attributes and then combining this result with weight and cost for the top-level elicitation (for a discussion on eliciting preferences, see (Keeney and Raiffa 1993, Section 5.8, Clemen 1996, pp. 546-52)). The performance attribute is a tradeoff among its constituent objectives:

$$v_{\mathrm{P}} = 0.15v_{\mathrm{T}} + 0.17v_{\mathrm{F}} + 0.01v_{\mathrm{L}} + 0.3v_{\mathrm{T}}v_{\mathrm{F}} + 0.1v_{\mathrm{T}}v_{\mathrm{L}} + 0.1v_{\mathrm{F}}v_{\mathrm{L}} + 0.17v_{\mathrm{T}}v_{\mathrm{F}}v_{\mathrm{L}},$$

where $v_T = V_{\mathrm{T}}(z_{\mathrm{T}})$ is the value function result for the cycle time attribute at z_{T}, $v_F = V_{\mathrm{F}}(z_{\mathrm{F}})$ is for the ram force attribute and $v_L = V_{\mathrm{L}}(z_{\mathrm{L}})$ is for the log length attribute. The top-level

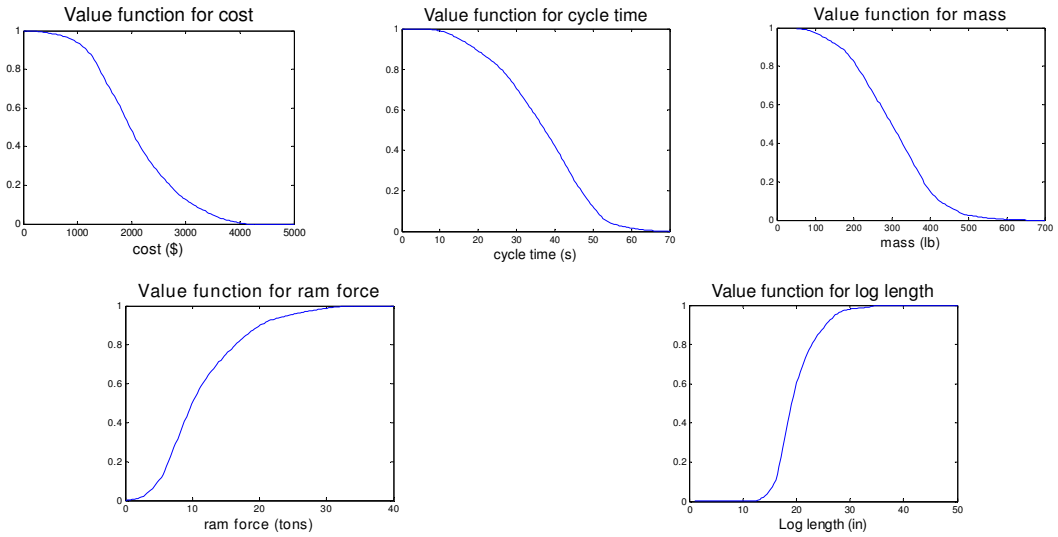


Figure 6.3: Graphs of the individual value functions for the five system-level attributes of the log splitter design problem.

[j] All preferences are those of the researcher.

value function is

$$V_{\text{sys}}(\mathbf{z}) = 0.45v_{\text{C}} + 0.12v_{\text{P}} + 0.02v_{\text{M}} + 0.16v_{\text{C}}v_{\text{P}} + 0.07v_{\text{C}}v_{\text{M}} + 0.07v_{\text{P}}v_{\text{M}} + 0.11v_{\text{C}}v_{\text{P}}v_{\text{M}},$$

where the $v_{\text{C}} = V_{\text{C}}(z_{\text{C}})$ is the value function for the cost evaluated at z_C, $v_{\text{M}} = V_{\text{M}}(z_{\text{M}})$ is for the mass attribute, v_{P} is the performance attribute defined above and $\mathbf{z}$ is the system-level attribute vector. Thus, the requirements allocation problem is to find the component-level attribute values that maximize $V_{sys}(\cdot)$.

6.3 System Composition and Requirements Allocation

Figure 6.4 is a summary of the system composition model for the log splitter example. The model consists of the algebraic relationships designers typically use to analyze hydraulic systems. The cost and mass models are sums of the costs and masses of

Cost	$z_{\text{C}} = S_{\text{C}}(\mathbf{y}) = c_{\text{pump}} + c_{\text{cyl}} + c_{\text{dcv}} + c_{\text{eng}}$
Mass	$z_{\text{M}} = S_{\text{M}}(\mathbf{y}) = m_{\text{pump}} + m_{\text{cyl}} + m_{\text{dcv}} + m_{\text{eng}}$
Ram Force	$z_{\text{F}} = S_{\text{F}}(\mathbf{y}) = \left(\Delta p_{\text{max,sys}}\right)\left(b_{\text{cyl}}^2 \frac{\pi}{4}\right)$
	where $\Delta p_{\text{max,sys}}$ is the maximum operating pressure of the system as dictated by the rating limitations of components or the pressure that can be generated by the engine-pump combination.
Log Length	$z_{\text{L}} = S_{\text{L}}(\mathbf{y}) = L_{\text{cyl}}$
Cycle Time	$z_{\text{T}} = S_{\text{T}}(\mathbf{y}) = 1.56\left(b_{\text{cyl}}^2 \frac{\pi}{4}\right)\left(\frac{1}{Q_\tau} + \frac{1}{Q_{\text{PWR}}}\right)$
	where Q_τ is the maximum flow rate the system can achieve at $\tau_{\text{max,eng}}$ and Q_{PWR} is the maximum flow rate (in gal/min) at $P_{\text{max,eng}}$. Both are non-decreasing functions of pump displacement, the engine speeds at the respective operating points and component rating limitations.

Figure 6.4: Summary of system composition model for log splitter design problem.

the individual components. Ram force is a function of the maximum system operating pressure and cylinder bore. Log length is equivalent to the cylinder stroke length. Cycle time is an index (i.e., not the actual cycle time, but a measure that correlates with it strongly) that depends on the maximum system flow rate at maximum engine torque, the maximum system flow rate at maximum engine power and cylinder bore.

To formalize the requirements allocation problem, let $\mathbf{S}$ represent the vector-valued system composition model (comprised of the $S_i(\cdot)$ from Figure 6.4), $\tilde{\mathbf{y}}$ denote the vector of component-level attributes controlled by the optimization routine, and $\mathbf{T}_i(\tilde{\mathbf{y}})$ denote the vector of all the component-level tradeoff model predictions using the i^{th} engine tradeoff model, where $i = 1, 2$. Thus, requirements allocation problem is to find

$$\mathbf{y} = \left[\tilde{\mathbf{y}}^*, \mathbf{T}\left(\tilde{\mathbf{y}}^*\right)\right],$$

such that

$$\tilde{\mathbf{y}}^* = \underset{\tilde{\mathbf{y}}_i^* \in \left\{\tilde{\mathbf{y}}_1^*, \tilde{\mathbf{y}}_2^*\right\}}{\arg\max} V_{sys}\left(\mathbf{S}\left(\left[\tilde{\mathbf{y}}_i^*, \mathbf{T}_i\left(\tilde{\mathbf{y}}_i^*\right)\right]\right)\right), \tag{7.1}$$

and

$$\tilde{\mathbf{y}}_i^* = \underset{\tilde{\mathbf{y}} \in \tilde{\mathbf{Y}}_i}{\arg\max} V_{sys}\left(\mathbf{S}\left(\left[\tilde{\mathbf{y}}, \mathbf{T}_i\left(\tilde{\mathbf{y}}\right)\right]\right)\right), \tag{7.2}$$

where $i = 1, 2$ indicates which engine tradeoff model is used and $\tilde{\mathbf{Y}}_i$ is the domain in which the tradeoff model predictions are valid (i.e., the domain description determined via SVDD). Note that although Equation (7.2) entails an optimization search over the input domain of the tradeoff models, Equation (7.1) simply is a discrete choice between two alternatives.

A couple of implementation notes are worth mentioning. First, in general it is important that designers use every attribute of a tradeoff model in the system model or assign them constant value. Passing unused variables to the optimization method can reduce solver efficiency significantly. In this example, the system integration model uses all available component-level attributes. Second, the optimization problem of Equation (7.2) is, in general, nonlinear and non-convex. Designers must choose their optimization algorithms appropriately. Gradient-based methods are used in this example, but with a number of random-restart runs to improve the odds of finding the global maximum.

6.4 Comparison to Exhaustive Search of Components Database

To demonstrate that the tradeoff modeling approach yields a reasonable requirements allocation solution for the log splitter, the results of the requirements allocation problem are compared to an exhaustive search of the components database. One typically would not do an exhaustive search in practice due to the large number of combinations that can exist. Even after removing outliers, the modestly-sized database yields nearly 13 million possible combinations for the log splitter system. Although one does not necessarily expect the two approaches to yield equivalent solutions—since the tradeoff models are able to generalize beyond the database contents—the exhaustive solution does provide a meaningful baseline for comparison. The tradeoff modeling approach should do no worse than the exhaustive search on the basis that the tradeoff models are representations of the database contents.

Table 6.3 contains results from the exhaustive search and the two tradeoff modeling optimization runs (one with each engine tradeoff model). The Engine 1 tradeoff

Table 6.3: Comparison of log splitter requirements allocation results from tradeoff modeling approach and exhaustive search.

Component	*Attribute*	*Composed Tradeoff Models (Engine 1)*	*Composed Tradeoff Models (Engine 2)*	*Exhaustive Search of DB*
Pump	Cost	$ 223	$ 221	$ 223
	Weight	11.7 kg (5.3 lb)	7.9 kg (3.6 lb)	11.7 kg (5.3 lb)
	Displacement	6.1 cc/rev (0.37 in^3/rev)	5.7 cc/rev (0.35 in^3/rev)	6.1 cc/rev (0.37 in^3/rev)
	Max. op. speed	4000 rpm	4000 rpm	40000 rpm
	Efficiency	0.88	0.63	0.88
Cylinder	Cost	$ 233	$ 213	$ 260
	Weight	181 kg (82.3 lb)	155 kg (70 lb)	254 kg (115 lb)
	Stroke length	0.68 m (27 in)	0.88 m (34.5 in)	0.71 m (28 in)
	Bore diameter	0.114 m (4.5 in)	0.102 m (4 in)	0.127 m (5 in)
Directional Control Valve	Cost	$ 83	$ 75	$ 90
	Weight	18.7 kg (8.5 lb)	26 kg (12 lb)	15.4 kg (7 lb)
	Max. op. flow rate	68.1 l/min (17 gal/min)	73.8 l/min (19.5 gal/min)	68 l/min (18 gal/min)
Engine	Cost	$ 330	$ 800	$ 300
	Weight	105 kg (47 lb)	192 kg (87 lb)	121 kg (55 lb)
	Maximum Power	6.7 kW (8.9 hp)	11.2 kW (15 hp)	6.7 kW (9 hp)
	Maximum Torque	18.8 N-m (13.8 ft-lb)	32.3 N-m (23.8 ft-lb)	19 N-m (14 ft-lb)
Value Components	Ram Force (v_F)	0.897	0.775	0.958
	Log Length (v_L)	0.944	0.996	0.963
	Cycle Time (v_T)	0.987	0.993	0.967
	Performance (v_P)	0.874	0.812	0.918
	Weight (v_W)	0.928	0.887	0.87
	Cost (v_C)	0.955	0.850	0.955
System Value (v)		**0.958**	**0.933**	**0.956**

model corresponds to the tradeoff model for lower-torque engines. According to the tradeoff modeling approach, a system that includes an engine from the Engine 1 domain is preferred to a system with an engine from the Engine 2 domain (preference value of 0.958 compared to 0.933). The exhaustive search corroborates this result, with its engine being virtually identical to the engine predicted using the Engine 1 tradeoff model.

Overall, the tradeoff modeling approach yields results similar to the exhaustive search solution. The tradeoff modeling approach identifies targets for the pump and engine that are virtually identical to those of the exhaustive search solution. However, the tradeoff modeling approach does generalize beyond the database contents for the cylinder and DCV requirements. Upon examining the system attribute valuations, one can see that the tradeoff modeling solution sacrifices small amounts in terms of the performance attributes in order to improve in the weight attribute. That this particular solution is not in the database underscores a strength of the tradeoff modeling approach to generalize beyond the discrete data.

6.5 Conclusions and Chapter Summary

The results of the log splitter design problem provide evidence that designers can make requirements allocation decisions in a practical setting by using component-level tradeoff models to compose a system-level model. The process of generating tradeoff models begins with observational data about existing implementations of each type of component. For each component, a tradeoff model is fit to the parameterized efficient set of the observed data. These tradeoff models are composed together using a system composition model (Figure 6.4) based on models designers typically use when designing fluid power systems.

Requirements allocation is an important type of decision during systems design, which means these results have practical significance. As noted in Section 2.2.1, the typical procedures that designers use for accomplishing this have significant limitations. They tend to require strong assumptions about system operation and preclude designers from considering all the tradeoffs that are important to them. The approach based on composing tradeoff models has no such limitations.

It is noteworthy that the requirements allocation problem would not have been solvable using tradeoff models if not for the application of the domain description approach from Chapter 3. Most obviously, the engine data contains two distinct clusters that one can model much more accurately using independent models. However, even though the other components feature only a single predominance cluster, they all have complex domain boundaries (i.e., they are irregular and non-convex).

The results of this chapter build upon the mathematical results of Chapter 5, providing empirical evidence that the theoretical conclusions hold in practice. However, further empirical study is required. The log splitter example is not highly complex. Only one system configuration is considered and the system composition model is algebraic in nature. The next chapter contains a more complex example that involves multiple system configurations and dynamical system models.

CHAPTER 7:

ARCHITECTURE SELECTION FOR A HYBRID HYDRAULIC VEHICLE

This is the last in a trio of chapters aimed at evaluating whether it is effective for designers to compose system-level models using component-level tradeoff models. Theoretical results from Chapter 5 indicate that composing parameterized efficient sets, which are the basis for predictive tradeoff models, is a sound operation in principle. Empirical evidence from Chapter 6 indicates that tradeoff modeling can be effective in practice. The current chapter contains additional empirical evidence through a more complex design example.

The log splitter problem from Chapter 6 involves a requirements allocation decision and algebraic system interaction models. This is a realistic design problem, but is not representative of all system-level decision problems. The design problem in the current chapter is the selection of a power train architecture for a hybrid hydraulic vehicle (HHV). This problem involves dynamical system models, which are considerably more complex than the algebraic models from the previous example.

Section 7.1 is a brief overview of the problem. Section 7.2 is a summary of the tradeoff model generation procedure and results associated with this problem. Section 7.3 is a description of the alternative HHV drive train architectures to be considered in the decision problem. Section 7.4 is an overview of the dynamical models used to evaluate the alternatives. Section 7.5 contains the results of the decision problem. Readers should note that the emphasis in this chapter is on the approach to tradeoff modeling and decision making rather than the design of a hybrid vehicle.

7.1 Hybrid Hydraulic Vehicles

A hybrid vehicle combines two different sources of power for propulsion: an internal combustion engine and a second source that is capable of storing regenerated energy during the driving cycle. Energy regeneration occurs when a driver slows a vehicle. Rather than dissipate the kinetic energy of the vehicle as heat through friction brakes, a hybrid vehicle converts the energy into a storable form. Two common technologies for storing and using the regenerated energy are batteries with electric motors/generators and accumulators with hydraulic motors/pumps.

Both types of systems have advantages and disadvantages for this application. The main advantage of hybrid electric vehicles is that electric power technology has a relatively high energy density (i.e., they can store a larger quantity of energy per unit mass). Conversely, hybrid hydraulic vehicles have an advantage in terms of the power density.

Presently, hybrid electric vehicles (HEVs) are more common, particularly for passenger vehicles. However, hybrid hydraulic vehicles (HHVs) have found application in service vehicles such as delivery and garbage trucks and have received increasing attention from the research community. The large power density of hydraulic power systems makes them attractive since this means they can recuperate a larger proportion of the energy typically dissipated during breaking.

Although vehicle power train components have grown increasingly sophisticated, the architecture for a conventional internal-combustion vehicle remains straightforward: there is a single flow of power flows from the engine to the drive wheels through a mechanical transmission. In contrast, several power transmission architectures are possible using hybrid technology and most of these can involve different energy flows

depending on the vehicle state. The search for effective and efficient hybrid vehicle architectures is a topic of ongoing research; one can find several studies and surveys in the recent literature—e.g., see (Bernhard 2004, Achten 2008, Van de Ven, et al. 2008), among others.

The complex nature and novelty of hybrid vehicle power trains makes them an interesting example problem in the context of this research. Hybrid vehicle technology—HHV technology in particular—continues to evolve. It is difficult for designers to identify whether one architecture is superior to another simply by inspection, which means they need some amount of quantitative modeling. The tradeoff modeling approach investigated in this research can benefit designers of HHVs by providing them with a means to evaluate their alternatives quickly and confidently.

Another interesting consideration is that comparing HHV power train architectures typically requires an analysis of vehicle dynamics. Although it is possible to make comparisons using algebraic relationships akin to those in the log splitter problem, these are limiting. System-level objectives for vehicles often include things such as maximizing fuel economy and acceleration. It is difficult for designers to evaluate these accurately without considering the dynamical characteristics of a vehicle.

7.2 Tradeoff Model Generation

For this example, tradeoff models are generated for three types of components: variable-displacement hydraulic pump/motors, hydraulic accumulators and gasoline-fueled internal combustion engines. Like the log splitter example, the tradeoff models are

Table 7.1: Summary of component database for HHV problem.

Component	*Description*	*Attribute*	*Min*	*Max*	*Units*
Pump	Axial piston type, variable displacement	Cost	980	2980	$
		Mass	4.99	41.73	kg
		Max. Displacement	10.49	140	cm^3/rev
		Max. Pressure	138	361	bar
		Max. Speed	1800	3200	rpm
Accumulator	Bladder type hydraulic accumulator	Cost	330	1500	$
		Mass	14.0	167.83	kg
		Volume	0.0036	0.0568	m^3
		Max Pressure	207	350	bar
Engine	Gasoline-fueled internal combustion engine	Cost	180	2845	$
		Mass	3.4	80.2	kg
		Max. Power	0.75	26.1	kW
		Speed at Max. Power	3600	7500	rpm

based on data that originate from published sources and information requests from vendors. Table 7.1 is an overview of the database of components collected for the HHV architecture selection problem.

Tradeoff model generation follows the procedure outlined in Figure 3.1 and the domain description procedure outlined in Figure 4.9. During data validation it was observed that each component had at least one uninformative attribute. For example, nearly all the engines had the maximum power at the same engine speed. Similar observations apply to maximum pressure and maximum speed for the pumps and maximum pressure for the accumulators. Each of these attributes is removed from the analysis and the few data points with different values for these attributes is removed from the database.

The parameterized Pareto dominance rule is applied to the remaining data. Table 7.2 includes a summary of how each attribute is treated during this step. Note that all

Table 7.2: Summary of domination and model structure for HHV tradeoff models.

Component	*Attribute*	*Attribute Type*	*Tradeoff Model Role*
Pump	Cost	Dominator (less is better)	Output
	Mass	Dominator (less is better)	Input
	Max. Displacement	Parameter	Input
Accumulator	Cost	Dominator (less is better)	Output
	Mass	Dominator (less is better)	Input
	Volume	Dominator (more is better)	Input
Engine	Cost	Dominator (less is better)	Output
	Mass	Dominator (less is better)	Input
	Max. Power	Parameter	Input

accumulator attributes are dominators, which means in that case the dominance rule is equivalent to classical Pareto dominance. Maximum pump displacement is modeled as a parameter attribute because although it is tends to influence performance-oriented objectives positively (i.e., increasing it increases power output) it tends to influence efficiency-oriented objective negatively. Maximum engine power is a parameter attribute for a similar reason: although higher power generally means better performance, it also means a higher fuel consumption rate.

The domain description procedure (Figure 4.9) is performed on the parameterized efficient set for each component and tradeoff models are fit to the data using Kriging methods (Lophaven, et al. 2002). Table 7.3 is a summary of those results. Validation is performed using leave-one-out cross validation (Witten and Frank 2005). The results reported in the table are the mean percentage prediction error as measured using that technique. The engine model error is somewhat higher than is desirable, but the other models yield good fits.

Table 7.3: Summary of data eliminations and tradeoff model generation for HHV problem.

Component	*Engine*	*Pump*	*Accumulator*
Total # in DB	44	31	31
# after outlier analysis	24	31	29
# after dominance analysis	14	29	19
Validation Results (mean % error)	28%	5.5%	4.5%

7.3 HHV Power Train Architectures and Designer Preferences

Two HHV power train architectures are considered in this example. Both are variants of a hydro-mechanical parallel configuration in which power may be delivered to the drive wheels from either or both of the internal combustion engine and the hydraulic motor (Van de Ven, et al. 2008).

7.3.1 Independent Torque Hybrid Hydro-Mechanical Drive Train

The first alternative is an independent torque hybrid hydro-mechanical (ITHHM) drive train. Figure 7.1 is an illustration of the first configuration considered in this example. It is capable of supplying torque to each drive wheel independently by virtue of the dual pump and planetary gearbox arrangement (PGB1 and PGB2). The internal combustion engine, when running, operates at a fixed speed and power output. This allows one to set the operating point to minimize fuel consumption.

Power flow through the system changes depending on the vehicle state. Different operating scenarios include the following:

- The engine is off and the vehicle is propelled by the pumps (acting as motors) using energy stored in the accumulator. The displacement of the pumps is varied to provide the appropriate amount of torque to the wheels.

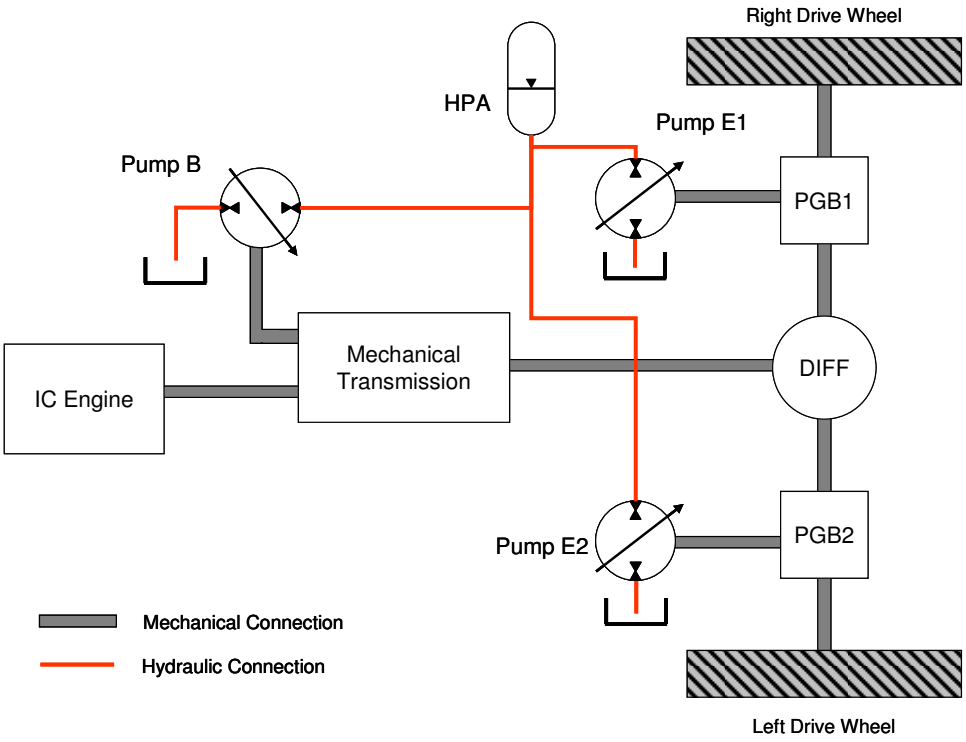


Figure 7.1: A hydro-mechanical drive train with independent wheel torque control.

- The accumulator has released most of its stored energy, but a higher vehicle speed is desired. In this case, the engine is turned on and drives the wheels directly. Any excess energy is used to restore the accumulator (with the pumps in pumping mode).
- Deceleration of the vehicle is desired. Kinetic energy of the vehicle is transformed into potential energy stored in the accumulator by running the pumps. The rate of energy transformation (and, consequently, deceleration) is controlled by the pump displacement setting. Friction brakes may be engaged to assist or to bring the vehicle to a complete halt.

Some control logic is required to ensure the power train is in the appropriate state. The engine is controlled by a pressure sensor on the high-pressure accumulator, HPA. The

engine starts when system pressure falls below a predetermined threshold and is shut off when the pressure becomes sufficient. Hysteresis is designed in to the controller to prevent the engine from cycling on and off rapidly. Speed and torque controllers set the appropriate pump displacements based on the current and desired state of the vehicle. The displacement of Pump B is varied to maintain a constant engine operating point. The displacements of Pump E1 and Pump E2 are set as a function of the desired speed of the vehicle.

7.3.2 Simplified Hybrid Hydro-Mechanical Drive Train

The second alternative is a simplified hybrid hydro-mechanical (SHHM) drive train. This is a variant of the ITHHM concept. Figure 7.2 is an illustration of this concept. Relative to that of Figure 7.1, this architecture has slightly less complex hydraulics (only one pump on the high-pressure side), but is incapable of independent control of wheel torque.

The simpler architecture potentially is advantageous because it requires fewer components. Potentially, this can result in a lighter and less expensive transmission with fewer hydraulic losses to detract from efficiency.

The mechanical transmission implementation is identical to the more complex configuration. The control logic also is largely the same.

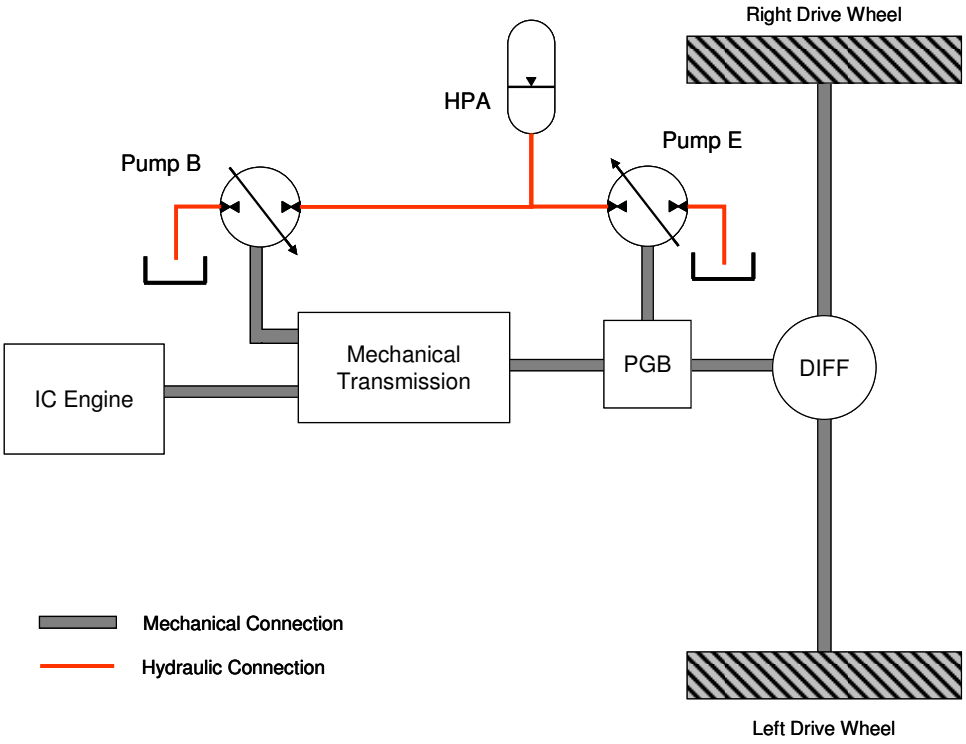


Figure 7.2: A hybrid hydro-mechanical drive train with high-low gear.

7.3.3 System-Level Decision Objectives

There are three system-level design objectives for this problem: maximize fuel economy, maximize vehicle acceleration and minimize costs. They are quantified as follows:

- Fuel economy: Measured in miles-per-gallon (liters-per-kilometer) achieved on the EPA Urban Dynamometer Drive Cycle (U.S. EPA 2008). Computed using a simulation of vehicle dynamics.
- Acceleration: Measured as time required to accelerate from full stop to 60 mph (96.56 kph). Computed using a simulation of vehicle dynamics.

- Costs: Measured as purchase prices of all major drive train components in US$ (which, from the perspective of a system designer, are costs). Assembly and transport costs not considered. Computed using tradeoff model predictions.

Figure 7.3 contains graphs of the value functions associated with each attribute. These represent the preferences of the researcher and are elicited using standard decision-

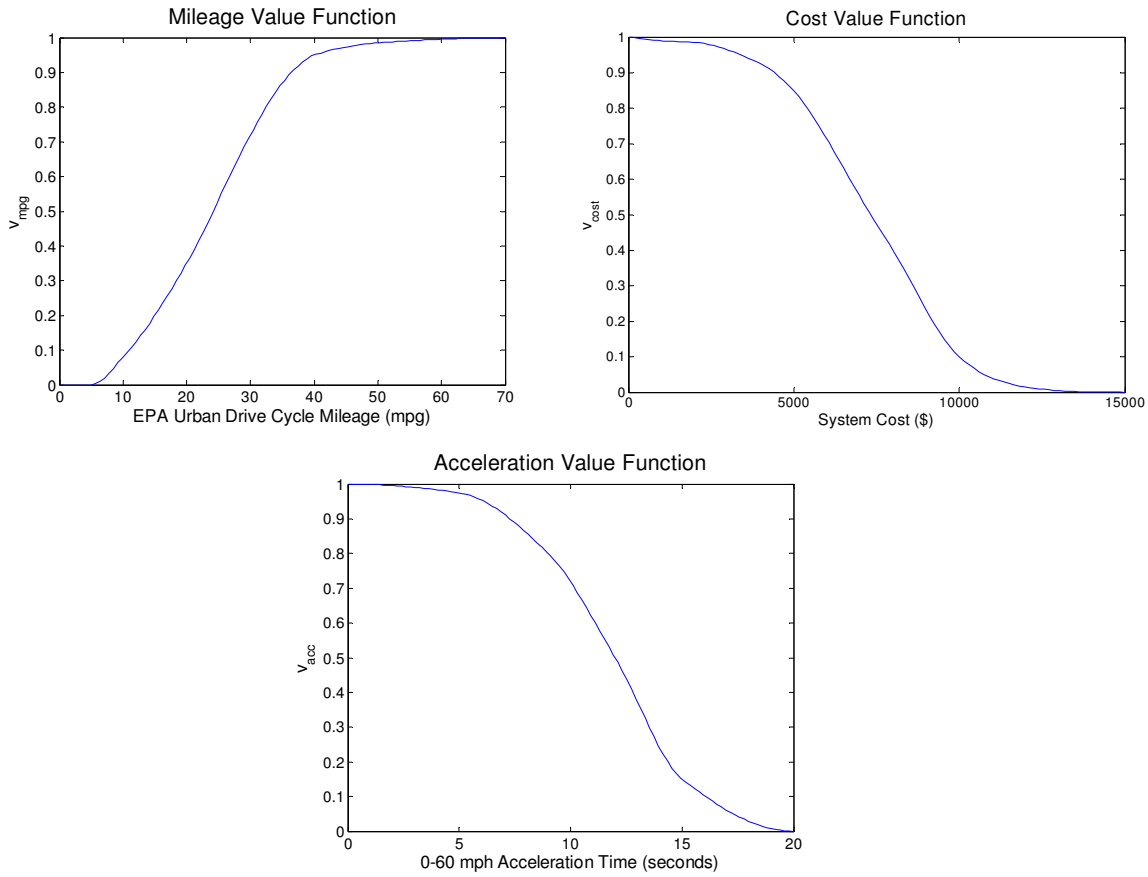


Figure 7.3: Graphs of the individual value functions for the HHV power train design problem.

analysis techniques (see (Keeney and Raiffa 1993, Section 5.8, Clemen 1996, pp. 546-52)). The individual value functions are combined using the following value function:

$$\begin{aligned} v_{sys} = {} & 0.35v_{mpg} + 0.22v_{acc} + 0.1v_{price} + \\ & 0.1v_{mpg}v_{acc} + 0.1v_{mpg}v_{price} + 0.06v_{acc}v_{price} + \\ & 0.11v_{mpg}v_{acc}v_{price} \end{aligned}$$

where v_{sys} is the overall system value and v_{mpg}, v_{acc} and v_{price} are the values obtained from the mileage, acceleration and cost value functions, respectively.

7.4 System Dynamics

7.4.1 System Model Overview

Both power train alternatives are modeled in the Modelica modeling language[k] using an open-source Fluid Power library (Paredis 2008). The models account for engine behavior, vehicle dynamics and energy losses. Figure 7.4 (page 162) is a top-level view of the Modelica model for the ITHHM drive train alternative. The key model components are:

- driveTrainMech. Encapsulates all of the mechanical transmission components. In addition to the "mechanical transmission" block from Figure 7.1, it also includes the differential and planetary drives located at the wheels.
- engine. The engine is modeled as having inertia and the model accounts for fuel consumption.
- vehicleBody. This model accounts for inertia as well as rolling and aerodynamic resistances.
- HPA. The high-pressure accumulator.

[k] The Modelica Association: http://www.modelica.org/

- Pumps (PumpE1, PumpE2 and PumpB). Variable displacement axial piston pumps modeled with losses (see Section 7.4.2).
- engineControl. Controller for turning engine on or off based on hydraulic system pressure.
- TorqueControl. Changes the displacement of Pump B to ensure that the engine remains at a fixed operating point.
- speedControl. Implements the speed commands for the system. Determines displacements for the E Pumps to achieve desired vehicle speed. Inputs are current vehicle speed and hydraulic pressure.

Figure 7.5 (next page) is the corresponding view of the SMMH drive train alternative. The main distinction is that there is only one E pump and the driveTrainMech component implementation is different to reflect the lack of independent torque control of the drive wheels.

7.4.2 Losses and Fuel Consumption

Hydraulic Pump Losses

One of the larger disadvantages of HHV technology relative to HEV technology is that variable displacement hydraulic pumps/motors are inefficient relative to electric generator/motors. Consequently, to evaluate any HHV drive train architecture accurately, one must account for these losses.

The pump model in the open source fluid power library incorporates a loss model based on the work of McCandlish and Dorey (1984). Using this

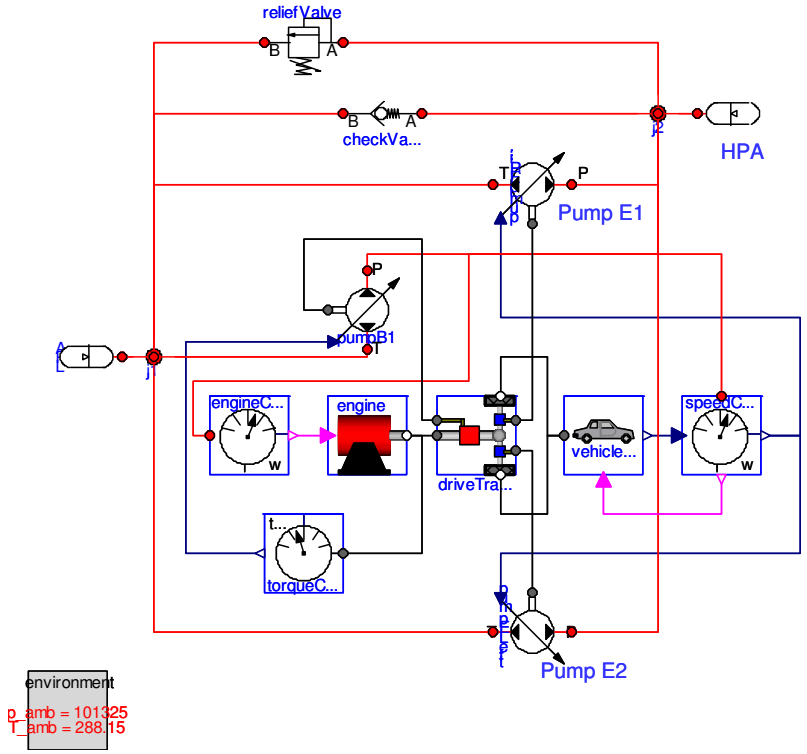


Figure 7.4: Top-level view of Modelica model for the ITHHM drive train.

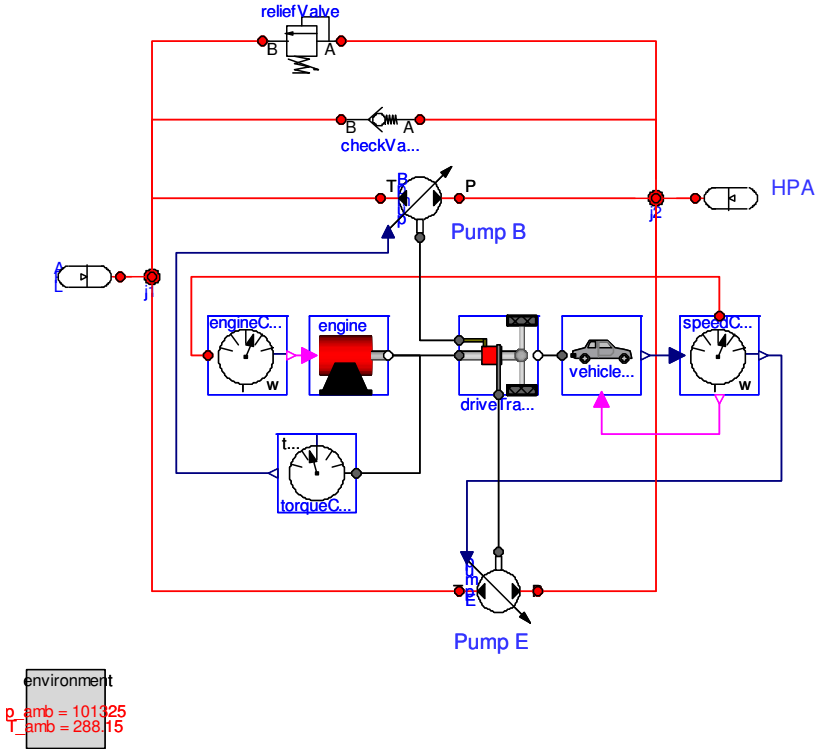


Figure 7.5: Top-level view of Modelica model for the SHHM drive train.

model, one can account for volumetric and mechanical losses with reasonable accuracy over a wide range of operating conditions. This model requires one to know four pump parameters:

- A volume ratio associated with flow loss
- Slip coefficient associated with flow loss
- Viscous drag coefficient associated with torque loss
- Coulomb friction coefficient associated with torque loss

One can determine these parameters for a given pump experimentally, but pump manufacturers tend not to publish these coefficients. For the current example problem, the same coefficients are used for all pumps. These coefficients are determined experimentally using a pump that is representative of those used to fit the tradeoff model. Ideally, one would incorporate these coefficients in a pump tradeoff model should they vary significantly over the set of pumps being modeled.

Fuel Consumption

The fuel consumption model is based on published brake-specific fuel consumption (BSFC) data from the engine product literature. Brake-specific fuel consumption is a measure of engine fuel consumption rate for a particular power output that one determines using a dynamometer. Minimum BSFC values for various engines of the same fuel type are remarkably constant (Aird 2000). For engines in the database used to generate the engine tradeoff model, the BSFC is about is 295 g/kW-hr. Since the engine in both architectures has a constant operating point, it is possible to compute a fuel consumption rate in volume per unit time. The calculation is

$$fcr = bsfc * \rho_{gasoline} * P_{eng} * \left(\frac{1}{3600} \right),$$

where fcr is the fuel consumption rate in gallons/second, $bsfc$ is the brake-specific fuel consumption rate in g/kW-hr, $\rho_{gasoline}$ is the density of gasoline expressed in gallons per gram, P_{eng} is the operating power of the engine in kW and the last term converts from hours to seconds. Because the engine operates at a fixed point when running, one can compute the total fuel consumed on a particular drive by multiplying the fcr by the total time the engine was running.

7.4.3 Computing the Decision Attributes

Estimating Fuel Economy

Fuel economy is estimated by simulating the EPA Urban Dynamometer Drive Schedule (UDDS) (U.S. EPA 2008). This consists of a velocity-time profile representative of a typical passenger vehicle in an urban setting. Figure 7.6 is a graph of the driving schedule in units of miles-per-hour versus time in seconds. The total driving duration is 1369 seconds, covering a distance of 7.45 miles (approximately 12 km) at an average speed of 19.59 mph (8.7575 m/s or 31.53 kph). To conduct the UDDS

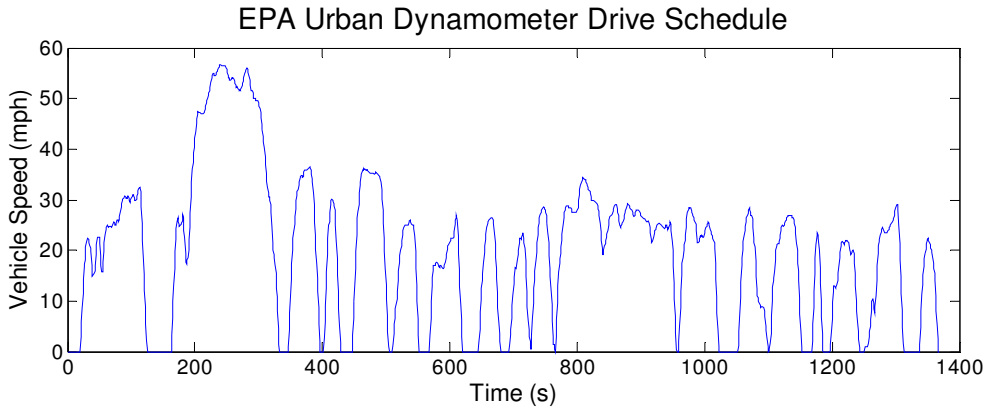


Figure 7.6: EPA Urban Dynamometer Drive Schedule.

simulation, the drive profile is input to the speed controller component in the Modelica model. This controller issues the signals required for the vehicle to follow the given schedule.

One challenge in estimating fuel economy is that some combinations of component-level attributes result in a system that is incapable of following the drive cycle. This usually happens when components are undersized. Figure 7.7 is an illustration of this effect. The problem is that the fuel economy computed for such a system may actually be quite good. In this sense, the system is "cheating" the test by deviating from it.

To avoid favoring systems that are incapable of following the EPA UDDS, a constraint is added to the decision problem based on a global trajectory error measure. The measure is computed as the integral of the squared error relative to the EPA drive schedule. Alternative solutions to this problem include using a maximum error criterion or creating an additional decision attribute to formalize a preference for solutions that follow the drive schedule closely.

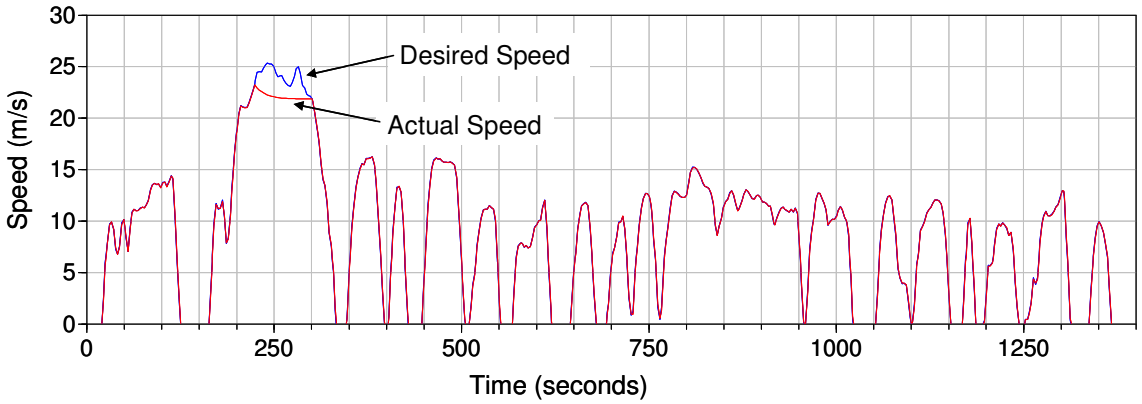


Figure 7.7: Illustration of a speed trajectory that does not follow the EPA UDDS.

Estimating Acceleration Time

The time required to accelerate from full stop to 60 mph (96.56 kph) is estimated using the same dynamical model as is used for the fuel economy simulations. However, rather than providing the speed controller with the urban driving schedule, the acceleration simulation is implemented by inputting a (large) step function into the speed controller. This prompts the simulated vehicle to its full acceleration in an attempt to track the step function. The simulation terminates when the desired speed is achieved.

If power train components are undersized, it is possible that a simulation does not obtain the desired top speed. In this case, the simulation proceeds for 60 seconds of simulated time and that solution is assigned a zero utility for the acceleration attribute (c.f. Figure 7.3). One need not enforce a constraint on the optimizer in this case because the solution receives such a poor rating.

7.5 Solving the Decision Problem

7.5.1 Component-Level Search Space

Because the aim of this example is illustrative, the mechanical transmission components are held fixed during this decision process. This is a significant simplification because the gear ratios have a significant impact on acceleration and fuel economy. Although this will not yield the best possible HHV drive train architecture, the problem retains adequate complexity to explore the role of tradeoff models in the decision solution process.

For each system alternative, tradeoffs for four components are explored: the engine, the accumulator, the B pump and the E pump(s). The two E-level pumps in the ITHHM drive train are assumed identical. Table 7.4 is a summary of the search space and tradeoff models for the problem.

Table 7.4: Summary of attribute search space for HHV drive train problem.

Component	*Search Attributes*	*Tradeoff Model*	*Predicted Attribute*
Pump B	Max displacement, mass	$T_{pump}(\cdot)$	Cost
Pump E	Max displacement, mass	$T_{pump}(\cdot)$	Cost
Engine	Max power, mass	$T_{eng}(\cdot)$	Cost
Accumulator	Max volume, mass	$T_{accum}(\cdot)$	Cost

7.5.2 Decision Formulation

The decision problem formulation follows the general formulation from Section 3.3.2. Requirements allocation is performed on each drive train architecture to predict the attributes designers would achieve if they continued to implement that alternative. The system selection decision is a discrete choice between the two predictions. To formalize the decision problem mathematically, let $a \in \{ITHHM, SHHM\}$ be an indicator to distinguish between the drive train architectures. Then let:

- $S_{\text{mpg},a}(\cdot)$ denote the simulation model for mileage estimation for architecture a,
- $S_{\text{acc},a}(\cdot)$ denote the simulation model for estimating 0-60 mph time for architecture a, and
- $S_{\text{cost},a}(\cdot)$ denote the algebraic model that computes system cost for architecture a given the prices of its components.

Also, let $\mathbf{S}_a(\cdot) = \left[S_{\text{mpg},a}(\cdot), S_{\text{acc},a}(\cdot), S_{\text{cost},a}(\cdot) \right]$ represent a vector-valued model composed of the three scalar models. As in previous chapters, let $\tilde{\mathbf{y}}$ denote the vector of search

attributes (second column of Table 7.4) and $\tilde{\mathbf{Y}}$ represent the valid search domain as identified by the support vector domain description procedure. Thus, the requirements allocation problem for each $a \in \{ITHHM, SHHM\}$ is to find

$$\mathbf{y}_a^* = \left[\tilde{\mathbf{y}}_a^*, \mathbf{T}\left(\tilde{\mathbf{y}}_a^*\right)\right]$$

such that

$$\tilde{\mathbf{y}}_a^* = \arg\max_{\tilde{\mathbf{y}} \in \tilde{\mathbf{Y}}} v_{sys}\left(\mathbf{S}_a\left(\left[\tilde{\mathbf{y}}, \mathbf{T}\left(\tilde{\mathbf{y}}\right)\right]\right)\right). \tag{8.1}$$

The final selection decision is formalized as

$$a^* = \arg\max_{a \in \{ITHHM, SHHM\}} v_{sys}\left(\mathbf{y}_a^*\right).$$

In addition to the usual domain description constraints, the optimization problem stated in Equation (8.1) is subject to an additional inequality constraint regarding the squared error integral of the vehicle tracking the speed profile for fuel economy estimation. This is implemented as an inequality constraint.

7.5.3 Results

For each architecture, a search of the component-level attribute space (Equation (8.1)) is conducted using a pattern search optimization method. Table 7.5 is a summary of the results. The ITHHM concept—the one with the more complex hydraulics and mechanical transmission—is the most preferred.

Part of the motivation for exploring the SHHM architecture was that it could be less expensive than the more complex architecture for similar levels of performance. However, according to these results this is not the case. In order to implement the SHHM architecture in a way that is capable of following the EPA UDDS successfully, one

Table 7.5: Summary of results from HHV drive train selection.

Component	*Variable*	*Architecture*		*Units*
		ITHHM	*SHHM*	
Engine	Max Power	12.6	15.6	kW
	Mass	36.2	40.3	kg
	Cost	591	1565	$
Accumulator	Volume	0.0322	0.0322	m^3
	Mass	69.4	69.3	kg
	Cost	1036	1036	$
Pump B	Max Displacement	11.78	16.96	cm^3/rev
	Mass	4.9	5.2	kg
	Cost	998	1076	$
Pump E	Max Displacement	12.42	33.79	cm^3/rev
	Mass	5.3	13.1	kg
	Cost	1020	1361	$
System Attributes	Mileage	49.2	36.6	mi/gal
	Acceleration	8.89	8.26	sec
	Cost	6167	6538	$
Value Function	System	0.8034	0.7588	
	Mileage	0.9834	0.9028	
	Acceleration	0.8062	0.8449	
	Cost	0.6867	0.6257	

requires a relatively large and expensive engine. Although the SHHM architecture is better than the ITHHM in terms of pumps (recall that there are two identical Pump E components in the ITHHM configuration), this difference is less than the added cost of the larger engine. This is an interesting result that would not have been apparent without the tradeoff model based optimization study.

It is interesting to note that both solutions have essentially identical solutions for the accumulator. Accumulator size impacts energy storage most directly; it has relatively little to do with the rate at which it can be stored or returned. Thus, the solution likely is the smallest accumulator that still allows for successful completion of the acceleration and fuel economy tests.

It is difficult to validate whether these predictions are valid. Unlike in the log splitter example, it is impractical to compare these results to an exhaustive search of the components database. Although the database is small, it still contains 21,576 combinations after removing gross outliers. The simulation models require on the order of two minutes to solve, which means this many combinations requires on the order of a month of computer time.

One can obtain some insight into the solutions by comparing them with the database entry that is the most similar. Table 7.6 (next page) is a summary of such a comparison. Nearest neighbors are selected using an equally-weighted normalized distance metric. The percent difference is relative to the entry from the database. Negative results indicate under-estimates.

Most of the solution points are within about 10% of a database entry, but the differences between others is much greater. One should not expect perfect correspondence between a solution found using the tradeoff model based attribute space search and the database. The tradeoff modeling approach is intended to generalize the database entries, potentially enabling designers to identify novel solutions that should be feasible.

In light of this, Pump B from the SHHM drive train is the only troubling result. Although the displacement and cost predictions match closely with a database entry, the mass is very different. Judging from the other pumps reported in the table, it seems the real mass of a pump having that displacement should be somewhere in between the two values. Notice that database entry listed for Pump E of the SHHM alternative has a higher

Table 7.6: Comparison of component sizing solutions to nearest database entry.

Component	*Variable*	*Predicted Solution*	*Nearest Neighbor in Database*	*Percent Difference*
Independent Torque Hybrid Hydro-Mechanical Drive Train				
Engine	Max Power	12.6	11.9	5.9
	Mass	36.2	36.3	-0.28
	Cost	591	500	18.2
Accumulator	Volume	0.0322	0.034	-5.29
	Mass	69.4	80	-13.25
	Cost	1036	1113	-6.92
Pump B	Max Displacement	11.78	10.48	12.4
	Mass	4.9	4.9	0
	Cost	998	980	1.84
Pump E	Max Displacement	12.42	13.76	-9.74
	Mass	5.3	5.9	-10.17
	Cost	1020	1000	2
Simplified Hybrid Hydro-Mechanical Drive Train				
Engine	Max Power	15.6	15.7	-0.64
	Mass	40.3	42.6	-5.4
	Cost	1565	1700	-7.94
Accumulator	Volume	0.0322	0.034	-5.29
	Mass	69.3	80	-13.37
	Cost	1036	1113	-6.92
Pump B	Max Displacement	16.96	16.39	3.48
	Mass	5.2	16.3	-68.1
	Cost	1076	1050	2.48
Pump E	Max Displacement	33.79	32.93	2.6
	Mass	13.1	13.6	-3.68
	Cost	1361	1470	-7.41

displacement but a lower mass than the one for Pump B. Typically, one would expect mass and displacement to correlate positively. It is likely that with more data gathering, one could identify a pump that would dominate this uncharacteristically heavy option.

7.6 Conclusions and Chapter Summary

The two HHV drive trains considered here serve as a good example for the importance of incorporating tradeoff modeling into architecture selection decisions. The

alternatives are closely matched, with neither being superior to the other. Certainly, this is not a decision that designers—no matter what their level of expertise—can make confidently without some amount of modeling and simulation.

Another factor is that the final result—that the more complex architecture is superior—is slightly counter-intuitive. The motivation for considering the simplified drive train is because it has fewer components and seems as though it could perform similarly for less cost. The tradeoff model based analysis yields a contrary conclusion.

One also can understand the part of value of applying tradeoff modeling to system selection decisions in the context of this example. Given that the alternatives have such similar levels of performance it is important that designers evaluate the solutions at realistic settings for the component-level attributes. Surely, it would be inappropriate to conclude that the SHHM alternative is superior based on a simulation that assumes a 25 kW engine cost only $500.

Perhaps more significantly is that tradeoff modeling combined with support vector domain description is a critical for enabling simulations at this level of abstraction. Although it is reasonable to suppose designers have an understanding of the upper and lower bounds on most of the component-level attributes, the odds are against their having a detailed understanding of the associations between them. Allowing an optimizer to search the entire hypercube defined by the upper and lower bounds certainly would yield invalid solutions. The alternative to tradeoff modeling is to model the constraints analytically, based on physical principles. Although this is reasonable to do at lower levels of abstraction, the aim is to avoid these lower levels when making system-level decisions.

CHAPTER 8:

TRADEOFF MODELING UNDER DATA UNCERTAINTY

An assumption underlying the research reported in the preceding chapters is that the attribute data designers use to generate a tradeoff model is known with certainty. Although this is reasonable in some cases, one cannot expect this assumption to hold in all situations. This chapter addresses the problem of dealing with uncertain data as expressed in the fourth research question:

RQ4. How should designers identify and visualize the (parameterized) efficient set of tradeoffs when the attribute data is uncertain?

The parameterized Pareto dominance rule is meaningless when attribute values are uncertain (this is true of *all* Pareto-based dominance criteria). Designers instead must use a dominance rule that accounts for the impact of uncertainty on their decision making. Uncertainty opens the door to designers making tradeoffs between their base objectives (e.g., to maximize reliability or to minimize cost) and their risk attitude toward uncertainty in achieving those objectives. In the simplest case, designers can trade between the mean value of an attribute and its variance. This problem in its full generality is too broad to tackle in this research. However, it is possible to handle an important special case and lay a foundation for future work.

This chapter addresses the case in which designers can model the uncertainty in attribute data using statistically independent normal distributions. A further assumption is that designers make decisions in a non-risk-taking manner (the precise meaning of which

is defined in Section 8.2). Under these conditions, designers can use a parameterized version of an appropriate stochastic dominance criteria as a basis for tradeoff modeling:

H4. Designers can identify the (parameterized) efficient set of tradeoffs under uncertainty using (parameterized) stochastic dominance criteria and can visualize this set as a surface in mean-variance space.

Stochastic dominance criteria are mathematically rigorous generalizations of the notion of dominance to the case of decisions under uncertainty. Under the current assumptions, the appropriate stochastic dominance rule yields a frontier in the mean-variance space of the distribution functions much in the way that classical Pareto dominance leads to a frontier in the attribute space. Like classical Pareto dominance, one can parameterize the stochastic dominance rule to enable model composition and reuse. The new dominance rule is demonstrated on a gearbox design problem similar to the one in Chapter 3.

Chapter organization is as follows. Section 8.1 is a review of an appropriate theory for decision making under uncertainty, called multi-attribute utility theory, and a description of how designers can reinterpret the notion of a tradeoff space in the case of uncertainty in the attribute data. Section 8.2 is a review of the literature on stochastic dominance. Section 8.3 is a description of a parameterized version of a stochastic dominance rule and how to apply it to tradeoff modeling. Section 8.4 contains the gearbox design example.

8.1 Making Tradeoffs under Uncertainty

8.1.1 Multi-Attribute Utility Theory

This research is based upon the decision theoretic framework of Multi-Attribute Utility Theory (MAUT) (Keeney and Raiffa 1993), which is an extension of the utility

theory by von Neumann and Morgenstern (von Neumann and Morgenstern 1980) to the case of multiple competing objectives. Using MAUT, designers can make decisions in a way that considers their preferences with regard to tradeoffs under uncertainty. One can formulate a decision in MAUT as:

$$a^* = \arg\max_{a \in \mathrm{A}} \mathrm{E}\left[u\left(\mathbf{z}_a\right)\right],$$

or

$$a^* = \arg\max_{a \in \mathrm{A}} \int u\left(\mathbf{z}_a\right) F_a\left(\mathbf{z}_a\right) d\mathbf{z}_a \,,$$

where A is the set of feasible decision alternatives, $a \in A$ is a specific alternative, a^* is the most preferred alternative, $\mathbf{z}_a \in \mathbb{R}^N$ is a random vector of attributes for alternative a having distribution function F_a, $u : \mathbb{R}^N \to \mathbb{R}$ is a suitably defined utility function, and $E[\cdot]$ is the expectation operator.

The function $u(\cdot)$ is analogous to $V(\cdot)$ from previous chapters. However, one must elicit $u(\cdot)$ in a manner that accounts for risk attitude. In contrast, $V(\cdot)$ accounts only for one's preferences for tradeoffs among the attributes. Several references describe procedures for eliciting utility functions (e.g., (Keeney and Raiffa, 1993; Clemen, 1996)).

8.1.2 Generalized Tradeoff Spaces

In preceding chapters, the space in which designers evaluate, visualize and model tradeoffs is the space of decision attributes. For example, the classical and parameterized Pareto dominance rules both involve comparisons of attribute values and one can visualize the corresponding efficient sets as surfaces in the space of attributes. This is

precisely the relationship designers capture in a tradeoff model. However, the attribute space is unsuitable for considering tradeoffs for decisions involving uncertainty.

One must extend the notion of a tradeoff space in order to deal with attributes under uncertainty. In this chapter, a *generalized tradeoff space* refers to *the space of probability distributions over the attribute space*. This follows from the axiomatic definition MAUT as decisions between lotteries (see (Keeney and Raiffa 1993) for a development of this theory) and is evidenced by the procedures for eliciting preferences in MAUT, which involves comparisons of lotteries over attributes (as opposed to comparisons of precise attribute values).

Strictly speaking, the notion of a "space" of probability distributions is ill-defined. However, one can specify several common distributions unambiguously by a small number of parameters. If those distributions—which includes normal, log-normal, uniform, Poisson, etc.—are good representations of the uncertainties, then designers can define a space of distributions in terms of such parameters.

The definition of a generalized tradeoff space makes intuitive sense when one considers the kinds of tradeoffs that occur when uncertainty is involved. For example, when product quality is a concern it is common for designers to trade mean performance to achieve reduced variability in that same attribute. One can visualize this tradeoff in the mean-variance space of that attribute. Figure 8.1 is an illustration of the relationship between the attribute space and the generalized tradeoff space for MAUT decision problems.

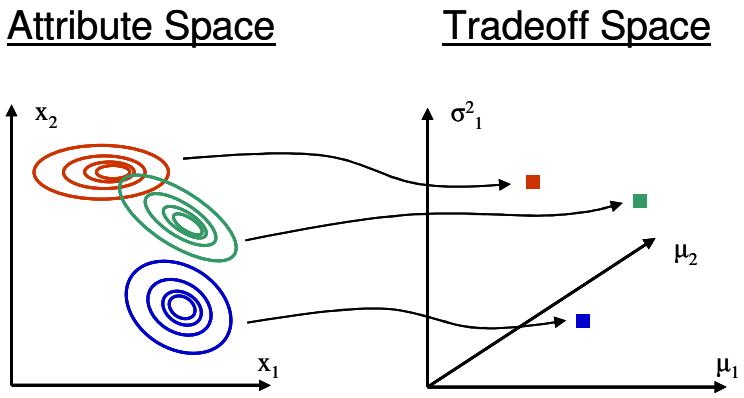


Figure 8.1: Illustration of distributions in an attribute space mapping to points in a tradeoff space defined in terms of parameters of the distributions.

8.2 Stochastic Dominance

Although the study of stochastic dominance dates back to the late 19060's (e.g., (Fishburn 1965, Hanoch and Levy 1969)), it has received little attention beyond the economics and operations research literature. A stochastic dominance test involves comparing distribution functions that are defined over an attribute space—i.e., it is an evaluation in the generalized tradeoff space. It also requires assumptions about the mathematical structure of the corresponding utility function. This is similar to the monotonicity assumption associated with the deterministic concept of Pareto dominance. However, the useful assumptions are more varied in the stochastic case owing to the greater complexity of preferences for decisions under uncertainty.

Researchers have identified a number of stochastic dominance conditions for different classes of utility functions and assumptions about the uncertainties involved. Examples include stochastic dominance in the case of statistically independent attributes (Huang, et al. 1978), for when attributes are mutually utility independent (Mosler 1984), for specialized uncertainty distributions (Levy 1973, 1990), for multi-attribute problems

(Levhari, et al. 1975, Huang, et al. 1978, Russell and Seo 1978, Scarsini 1988, Baccelli and Makowski 1989, O'Brien and Scarsini 1991), for non-transitive preferences (Fishburn 1978) and for the case in which stochastic dominance is equivalent to mean-variance analysis (Baron 1977). The following is an overview of the results most relevant to this research.

8.2.1 Stochastic Dominance for Single-Attribute Utility Theory

Researchers commonly distinguish between different classes of utility functions, each having different mathematical properties and to which different stochastic dominance rules apply. Table 8.1 is a summary of three common classes of single-attribute utility functions, their relationships to one another and the associated stochastic dominance criteria (see (Levy 1992) for a survey of these and other related criteria as well as mathematical proofs for the rules).

Class U_0 is the set of all utility functions. All other classes are a subset of this and no stochastic dominance rule applies to all of U_0. Class U_1 includes only the monotonic utility functions (i.e., increasing the value of the decision attribute cannot result in a decrease in utility). The stochastic dominance rule for this class is called *first-degree stochastic dominance* and involves a comparison of cumulative distribution functions as stated in Table 8.1. If one alternative dominates another by the first-degree stochastic dominance (FSD), then designers are assured its utility is higher than that of the dominated alternative for *any monotonic utility function*.

The FSD rule is the most directly analogous to the Pareto-based rules. However, it is relatively weak in the sense that it incorporates no assumptions about designer risk

Table 8.1: Summary of common classes of single-attribute utility functions and the relevant stochastic dominance criteria.

Class	*Defining Assumptions*	*Interpretation*	*Associated Dominance Criterion*
U_0	none	All utility functions.	n/a
U_1	$U_1 := \left\{ u \in U_0 \mid \frac{du(z)}{dz} \geq 0 \right\}$	Monotonic utility functions.	First-degree stochastic dominance (FSD): Alternative z_a with cumulative distribution function F_a dominates z_b with CDF F_b according to FSD if and only if $F_a(z) \leq F_b(z) \quad \forall z$.
U_2	$U_2 := \left\{ u \in U_1 \mid \frac{d^2u(z)}{dz^2} \leq 0 \right\}$	Monotonic and non-risk-taking (i.e., risk neutral or risk averse).	Second-degree stochastic dominance (SSD): Alternative z_a with CDF F_a dominates z_b with CDF F_b according to SSD if and only if: $\int_{-\infty}^{z} \left[F_b(t) - F_a(t) \right] dt \geq 0 \quad \forall z$.
U_3	$U_3 := \left\{ u \in U_2 \mid \frac{d^3u(z)}{dz^3} \geq 0 \right\}$	Monotonic, non-risk-taking and decreasing absolute risk aversion.	Third-degree stochastic dominance (TSD): Alternative z_a with CDF F_a dominates z_b with CDF F_b according to TSD if and only if: $\int_{-\infty}^{t} \int_{-\infty}^{v} \left[F_b(t) - F_a(t) \right] dt dv \geq 0 \quad \forall z$ and $\mathrm{E}[z_a] \geq \mathrm{E}[z_b]$.

attitude. If designers know something about their risk attitude, then they can eliminate a larger proportion of the alternatives by using a stronger stochastic dominance rule.

The class U_2 is a subset of U_1 and includes only those utility functions that are monotonic and that correspond to a non-risk-taking attitude. Under the typical interpretation, a utility function corresponding to a non-risk-taking attitude has a positive derivative with respect to the decision attribute. The corresponding stochastic dominance rule is called *second-degree stochastic dominance* (SSD). It also requires that one

compare cumulative distribution functions, but the comparison differs from that of first-degree stochastic dominance.

Note that any alternative that is dominated via FSD also is dominated via SSD. This is because the set U_1 subsumes U_2 (i.e., what is true of every element of U_1 must also hold for every element of U_2 because every element of U_2 is an element of U_1). However, the converse is not true in general: an alternative that is dominated assuming a risk-averse utility function may be non-dominated for a risk-taking attitude.

Another common class, U_3, is a subset of U_2 and includes only those utility functions that are monotonic, correspond to a non-risk-taking attitude and have decreasing absolute risk aversion. The corresponding stochastic dominance criterion is called *third-degree stochastic dominance* (TSD). Like the others, it involves a comparison of distribution functions.

8.2.2 Stochastic Dominance for Multi-Attribute Utility Theory

The stochastic dominance rules from the preceding section apply only to single-attribute utility functions. Researchers have generalized these criteria to the case of multi-attribute decisions, but the conditions are not always straightforward extensions of the single-attribute tests (Levhari, et al. 1975, Russell and Seo 1978).

Table 8.2 is an overview of several common classes of multi-attribute utility functions that researchers have investigated in the context of stochastic dominance. These are extensions of those from Table 8.1. One can define other classification schemes. For example, Scarscini (1988) identifies utility functions according to a non-classical characterization of risk, called multivariate risk aversion. However, only those listed in Table 8.2 are of interest in the current research.

Table 8.2: Common classes of multi-attribute utility functions.

Class	*Defining Assumptions*	*Interpretation*
U_0	None	All utility functions.
U_1	$U_1 := \left\{ u \in U_0 \mid \frac{\partial u(\mathbf{z})}{\partial z_i} \geq 0 \; \forall i = 1 \ldots N \right\}$	Monotonic utility functions.
U_2	$U_2 := \left\{ u \in U_1 \mid \frac{\partial^2 u(\mathbf{z})}{\partial z_i^2} \leq 0 \; \forall i = 1 \ldots N \right\}$	Monotonic and non-risk-taking (i.e., risk neutral or risk averse).
U_3	$U_3 := \left\{ u \in U_2 \mid \frac{\partial^3 u(\mathbf{z})}{\partial z_i^3} \geq 0 \; \forall i = 1 \ldots N \right\}$	Monotonic, non-risk-taking and decreasing absolute risk aversion.

In general, stochastic dominance tests for multi-attribute problems involve comparisons of multivariate distribution functions and are difficult to evaluate. However, researchers have identified that under certain circumstances one can simplify the multi-attribute rules considerably. One of the more powerful simplifying assumptions is statistical independence.

When attribute distributions are statistically independent, one can test any multi-attribute dominance condition by testing the corresponding single-attribute condition using the marginal distributions (the general proof is non-trivial; see Theorem 2 in (Huang, et al. 1978)). Single-attribute tests are more amenable to numerical computation. Although this assumption may not hold perfectly for many engineering problems, it is a reasonable approximation in a number of cases.

8.2.3 Stochastic Dominance for Specialized Distribution Functions

One can obtain a further reduction in the computational effort required to evaluate the stochastic dominance rules when the attribute distributions have a special form. Simply knowing the closed form expression for the distribution function can be helpful

because it may allow one to simplify the integrals or reformulate them in a way more conducive to numerical solutions. However, one can achieve considerably greater simplification for a few special distribution functions.

Researchers have shown that when the attributes are independent and normally distributed, the SSD test simplifies to a comparison of the means and variances (Tobin 1958, Hanoch and Levy 1969, Baron 1977). This significantly reduces computational complexity. To formalize this simplified rule, some notation is required. Let $\mathbf{z}_a, \mathbf{z}_b \in \mathbb{R}^N$ denote the attribute vectors for decision alternatives a and b. Also, let $F_a(\cdot)$ and $F_b(\cdot)$ denote the distribution functions for $\mathbf{z}_a$ and $\mathbf{z}_b$, respectively. Finally, let $\mu_{i,a}$ and $\sigma^2_{i,a}$ denote the mean and variance, respectively, of the i^{th} attribute of alternative a and $\mu_{i,b}$ and $\sigma^2_{i,b}$ the corresponding means and variances for alternative b. Thus, one can state the mean-variance dominance rule as:

> *Definition 8.1 (Multi-Attribute Mean-Variance SSD): If* $\mathbf{z}_a \sim F_a(\cdot)$ *and* $\mathbf{z}_b \sim F_b(\cdot)$ *are independent and normally distributed, then* $\mathbf{z}_a$ *dominates* $\mathbf{z}_b$ *by MV-SSD if an only if,* $\mu_{i,a} \geq \mu_{i,b} \quad \forall i = 1 \ldots N$, $\sigma^2_{i,a} \leq \sigma^2_{i,b} \quad \forall i = 1 \ldots N$, *and at least one of the inequalities is strict.*

Strictly speaking, one should compare means and variances using statistical tests. However, when sufficient data exists comparisons of the point estimates are a reasonable approximation. The important implication of the mean-variance SSD rule is formalized as follows. Let $\mathbf{z}_a$ MV-SSD $\mathbf{z}_b$ denote that $\mathbf{z}_a$ dominates $\mathbf{z}_b$ via the MV-SSD rule. Thus:

Theorem 8.1: For an alternative a with uncertain attributes $\mathbf{z}_a \sim F_a(\cdot)$ and an alternative b with uncertain attributes $\mathbf{z}_b \sim F_b(\cdot)$, $\mathbf{z}_a$ MV-SSD $\mathbf{z}_b$ if and only if $u(\mathbf{z}_a) > u(\mathbf{z}_b)\ \forall u \in U_2$.

Several authors prove that the mean-variance assumptions hold for the single-attribute case (e.g., (Tobin 1958, Hanoch and Levy 1969, Baron 1977)). Combining this result with Theorem 2 in (Huang, et al. 1978) yields a complete proof of the multi-attribute case of Theorem 8.1. A similar result exists for log-normal distributions (Levy 1973, 1990).

Figure 8.2 is an illustration in mean-variance space of this dominance rule as applied in the single-attribute case. The MV-SSD rule always results in a curve or surface in mean-variance space. As with statistical independence, the assumption of normally distributed attributes is not valid in general. However, it is an effective approximation in many practical cases—e.g., measurement errors tend to be normal, as is the sum of several random variables.

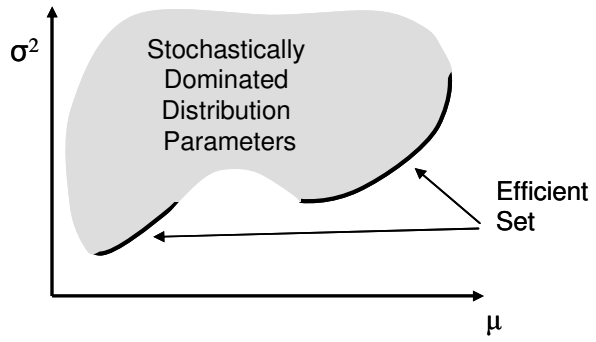


Figure 8.2: Illustration of second-degree stochastic dominance under the mean-variance assumptions. Note the orientation for comparisons of variance.

8.3 Parameterized Efficient Sets using Stochastic Dominance Rules

Just like the deterministic case from previous chapters, it may be difficult for designers to identify a universal preference ordering for each attribute for a particular component. Moreover, one still must deal with attributes such as gear ratio and cylinder bore, for which no problem-independent preferences exist, regardless of whether there is uncertainty in the attribute data. Consequently, designers need a parameterized version of the relevant stochastic dominance criterion. This section is a description of a parameterized version of the mean-variance SSD rule and how to formulate decisions using a generalized tradeoff model fit to the parameterized efficient set that results from applying the parameterized MV-SSD rule.

8.3.1 Assumptions

The stochastic dominance rule used in this study incorporates three main assumptions. They are summarized as follows.

Designers are Non-Risk-Taking

Designers are assumed to be non-risk-taking. Mathematically, this means that their utility function must be convex:

$$\frac{\partial^2 u(\mathbf{z})}{\partial z_i^2} \leq 0 \quad \forall i = 1 \ldots N,$$

where $\mathbf{z} = [z_1, z_2, \ldots, z_N]$ is a vector of decision attributes. Under this risk attitude, one is willing to sacrifice some amount of "upside" to avoid the risk of potential "downside." For situations in which the mean-variance assumptions hold, this means one is willing to trade mean attribute value for reductions in attribute variance.

Although this assumption about risk attitude will not hold for all design decisions, it is a common attitude among designers. For example, the desire to improve product

quality through reduced product variation, commonly called robust design, is one manifestation of this type of risk attitude.

Attributes are Statistically Independent

The uncertainty in attributes is assumed to be statistically independent. Whether this assumption is valid depends on both the data source and the characteristics of the underlying uncertainty. If the principal uncertainty is measurement error in determining the attribute values, statistical independence likely is a good model. However, there can exist underlying uncertainties that affect multiple component-level attributes and thereby lead to correlations. For example, manufacturing variations can influence multiple attributes in this way. Designers must consider whether this assumption holds on a case-by-case basis.

Attributes are Normally Distributed

The uncertainty in attributes is assumed to be normally distributed. Like the assumption about statistical independence, this assumption will hold only in some cases. Normal distributions are common when multiple uncertainties are added together, and so can be reasonable uncertainty models for component-level attributes in at least some cases.

8.3.2 Parameterized MV-SSD and Tradeoff Models

Under the preceding assumptions, one can formulate a parameterized version of the MV-SSD rule from Definition 8.1. Only attribute means may be parameters in this framework. This is due to the assumption that designers are non-risk-taking in every attribute. That assumption induces a preference ordering on all attribute variances (i.e.,

they all are dominators). Aside from this distinction between means and variances, the notions of dominators and parameters is the same as in Chapter 3.

Let $D \subseteq \{1,\ldots,N\}$ denote the set of indexes corresponding to dominator attributes and $P \subset \{1,\ldots,N\}$ denote the set of indexes corresponding to parameter attributes. As in the deterministic case, $P \cup D = \{1,\ldots,N\}$ and $P \cap D = \varnothing$. Thus, one can define parameterized MV-SSD as follows:

Definition 8.2 (Parameterized MV-SSD): If $\mathbf{z}_a \sim F_a(\cdot)$ *and* $\mathbf{z}_b \sim F_b(\cdot)$ *are independent and normally distributed, then* $\mathbf{z}_a$ *parametrically dominates* $\mathbf{z}_b$ *by parameterized MV-SSD if an only if,* $\mu_{i,a} = \mu_{i,b}\ \forall i \in P$, $\mu_{j,a} \geq \mu_{j,b}\ \forall j \in D$, $\sigma^2_{k,a} \leq \sigma^2_{k,b}\ \forall k = 1\ldots N$, *and at least one of the inequalities is strict.*

This parameterization of the mean-variance SSD rule is a direct analogy to the way in which parameterized Pareto dominance is an extension of classical Pareto dominance. As with the other dominance rules, the significance of this rule from a decision-making perspective is that a dominated alternative cannot be the most preferred alternative under the prevailing assumptions. This is formalized as follows:

Theorem 8.2: For an alternative a *with uncertain attributes* $\mathbf{z}_a \sim F_a(\cdot)$ *and an alternative* b *with uncertain attributes* $\mathbf{z}_b \sim F_b(\cdot)$, *if* $\mathbf{z}_a$ PMV-SSD $\mathbf{z}_b$ *then* $u(\mathbf{z}_a) > u(\mathbf{z}_b)\ \forall u \in U_2$.

Appendix A contains a proof of this statement.

By applying PMV-SSD, designers obtain a parameterized efficient set similar to the deterministic case. The main distinction is that one interprets the efficient set resulting from parameterized Pareto dominance directly in the space of decision attributes, whereas one interprets the set resulting from PMV-SSD in the space of distribution parameters—i.e., mean-variance space[1].

One also can fit a tradeoff model in this generalized tradeoff space. Like the deterministic case, one's choice of tradeoff model inputs and outputs is largely arbitrary. The main restriction is that attribute means that are parameters must be inputs to a tradeoff model. This avoids the possibility of a one-to-many mapping, which one cannot represent functionally. Beyond this, one should choose a model structure with accuracy in mind.

8.3.3 Formulating Decisions

For decisions under uncertainty, tradeoff models capture the association between attribute distribution functions. Designers can use this information to formulate requirements allocation and system selection decisions. However, unlike in the deterministic case, designers explore a generalized tradeoff space. For example, rather than resulting in target values for attributes, requirements allocation results in target values for attribute distribution parameters.

Although the general idea is similar to the deterministic case, the notation one requires to formalize a decision is somewhat more complicated in the case of decisions under uncertainty. The following definitions are used:

[1] For ease of visualization and interpretation, designers may prefer using a space constructed using attribute means and standard deviations. All the results of this chapter hold for both cases.

- Let y_i for $i = 1 \ldots N$ denote component-level attributes that are independent and normally distributed.
- Let $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_N] = E[\mathbf{y}]$ denote the vector of attribute means and $\boldsymbol{\sigma}^2 = \left[\sigma_1^2, \sigma_2^2, \ldots, \sigma_N^2\right]$ denote the vector of attribute variances.
- Let $\theta = \left[\boldsymbol{\mu}, \boldsymbol{\sigma}^2\right]$ denote a generalized coordinate vector in mean-variance space. The data one uses to fit a tradeoff model in this case consists of θ samples.
- Let $N\left(y_i \mid \left[\mu_i, \sigma_i^2\right]\right)$ denote a normal distribution with the mean and variance given and let $MN(\mathbf{y} \mid \theta) = \prod_{i=1}^{N} N\left(y_i \mid \left[\mu_i, \sigma_i^2\right]\right)$ denote a multivariate normal distribution constructed from independent normal marginal distributions with the given distribution parameters.
- Let $l = 1 \ldots L$ be an index indicating which one of L discrete alternatives are being considered.
- Let $\mathbf{S}_l(\cdot)$ denote a system model that relates component-level attributes to system-level attributes for the l^{th} discrete system alternative.
- Let $\mathbf{T}_l(\cdot)$ denote a tradeoff model fit to parameterized efficient set data in the mean-variance space as indicated by the PMV-SSD rule.
- Let $\tilde{\theta}_l$ denote the subset of generalized coordinates used as inputs to the tradeoff model, such that $\theta = \left[\tilde{\theta}, \mathbf{T}_l\left(\tilde{\theta}\right)\right]$ is a complete coordinate vector in the mean-variance space.
- Let $\tilde{\Theta}_l$ denote the valid input domain for tradeoff model $\mathbf{T}_l(\cdot)$.

Given this notation, one can predict the most preferred requirements allocation decision as

$$\theta^* = \left[\tilde{\theta}_l^*, \mathbf{T}_l\left(\tilde{\theta}_l^*\right)\right],$$

where

$$\begin{aligned}\tilde{\theta}_l^* &= \arg\max_{\tilde{\theta}_l \in \tilde{\Theta}_l} E\left[u\left(\mathbf{S}_l\left(\mathbf{y}_{\left[\tilde{\theta}_l, \mathbf{T}_l\left(\tilde{\theta}_l\right)\right]}\right)\right)\right] \\ &= \arg\max_{\tilde{\theta}_l \in \tilde{\Theta}_l} \int u\left(\mathbf{S}_l\left(\mathbf{y}\right)\right) MN\left(\mathbf{y} \mid \left[\tilde{\theta}_l, \mathbf{T}_l\left(\tilde{\theta}_l\right)\right]\right) d\mathbf{y}.\end{aligned}$$

This computational procedure is analogous to that of Equation (3.4). However, in this case is that one searches over distribution functions rather than directly over attribute values.

Given the preceding formalization of allocation decisions, one can state the selection decision formally as:

$$\begin{aligned}l^* &= \arg\max_{l=1\ldots L} E\left[u\left(\mathbf{S}_l\left(\mathbf{y}_{\theta_l^*}\right)\right)\right] \\ &= \arg\max_{l=1\ldots L} \int u\left(\mathbf{S}_l\left(\mathbf{y}\right)\right) f\left(\mathbf{y} \mid \theta_l^*\right) d\mathbf{y}.\end{aligned}$$

One should note that designers already will have evaluated the expected utility of their alternatives during requirements allocation. Consequently, this decision is computationally straightforward.

8.4 Gearbox Design Problem

For the purposes of demonstrating how to make decisions using tradeoff models under uncertainty, the gearbox design problem from Chapter 3 is revisited. The problem largely is the same, except that an uncertainty model is used when generating attribute data for the different gearbox configurations. The PMV-SSD rule is applied to the data

for each configuration and a tradeoff model is fit to the resulting parameterized efficient sets. The decision problem involves identifying the most preferred gearbox configuration.

8.4.1 Generating Gearbox Tradeoff Models under Uncertainty

Gearbox Concepts

This example involves the same concepts as the example from Chapter 3. To review, Figure 3.4 (page 73) is an illustration of the different configurations:

- Planetary Gearbox (PGB): Basic planetary gear system, with input on sun, output on arm and fixed ring. Depicted in Figure 3.4(a).
- Single-Sided Fully-Reverted Gearbox (SGB): Four-gear system with two identical pinions and two identical gears. Depicted in Figure 3.4(b).
- Double-Sided Fully-Reverted Gearbox (DGB): Similar to single-sided concept, but includes two paths for torque flow. Depicted in Figure 3.4(c).

As in the earlier gearbox example, each concept has its own parametric structure. These low-level design parameters control the number of teeth on each gear, the gear face widths and the gear module. This low-level design space is sampled in order to generate component-level attribute data about feasible implementations of each concept.

Gearbox Attributes and Preference Classifications

The gearbox tradeoff models account for three attributes.

- Cost: The cost of constructing the gearbox, computed as a function of the material and parts involved.
- Reliability: The probability that the gearbox operates without failure, considering both static and dynamic loading phenomena.
- Gear ratio: The ratio of transformation from input to output.

Of these attributes, reliability and cost involve significant uncertainties, which are assumed normally distributed and independent. However, one can compute gear ratio with certainty. Thus, the resulting tradeoff space consists of five dimensions:

- Gear ratio, N_g,
- Mean reliability, μ_R,
- Variance of reliability, σ_R^2,
- Mean cost, μ_C, and
- Variance of cost, σ_C^2.

Of these, gear ratio is a parameter and the rest are dominators. The rationale for this is straightforward. The variances must be dominators since it is assumed that designers are non-risk-taking (and thus less variance is preferred to more, all things being equal). Mean cost is a dominator because designers generally prefer to minimize cost. Similarly, mean reliability is a dominator because designer prefer to maximize it. No universal preference exists for gear ratio, and so it must be a parameter (see Section 3.4.1 for an elaboration).

Data Generation and Dominance Analysis

As with the earlier gearbox example, a model-based data gathering approach is used. The distinction is that this example assumes uncertainty exists on the low-level design parameters. Table 8.3 is a summary of the low-level uncertainties used as inputs to the engineering analysis models.

Implementations of each concept are generated by sampling the design parameter space systematically. Each implementation is evaluated under uncertainty using Latin hypercube sampling (LHS) (McKay, et al. 1979). The LHS procedure results in

Table 8.3: Uncertainty models for low-level parameters used during gearbox data generation.

Parameter	Uncertainty Model Normal(mean, std dev)
Gearing	
Application Factor	N(1.7, 0.1)
Gear Quality Factor	N(8, 0.25)
Bending Strength Geometry Factor	N(0.24,0.025)
Gear Material, Bending Fatigue Strength	N(200e6, 40e6) [Pa]
Gearbox cost model	N(0, 5) [$]

attribute samples from which distribution statistics (means and variances) are computed for the cost and reliability attributes (Figure 8.3). The result of this procedure is data samples in the mean-variance space, each of which represents one feasible gearbox implementation. The PMV-SSD rule (Definition 8.2) is applied to the sample data using gear ratio as a parameter and the reliability and cost statistics as dominators.

Model Fitting and Validation

For each concept, the tradeoff model inputs are N_g, μ_R, σ_R^2 and σ_C^2; the output is μ_C. Kriging interpolation methods and the DACE Matlab Kriging Toolbox are used for

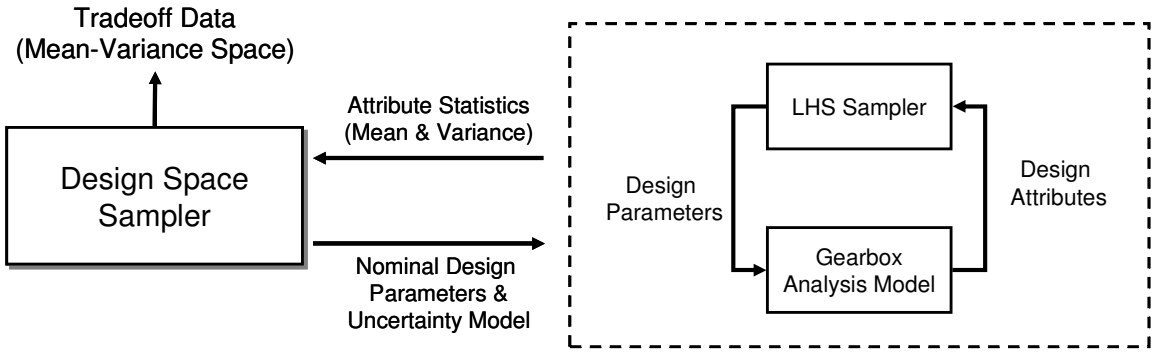


Figure 8.3: Procedure for generating and evaluating gearbox implementations.

fitting the tradeoff models (Lophaven, et al. 2002). Hold-out validation is conducted using fifty non-dominated implementations of each concept. Tradeoff models are fit to the rest of the parameterized efficient set data for each concept.

8.4.2 Design Problem Scenario

System and Environment

The system is the same as from the example of Chapter 3 (see Figure 3.8). However, the parameters used to evaluate the system differ in this case, with several of them being modeled as random variables. Table 8.4 is a summary of the uncertainty models.

Decision Problem Formulation

Designer objectives are the same for this problem as they are in the design problem of Chapter 3: to maximize profits from competing in a race. However, because system attributes are uncertain in this case—by virtue of uncertain gearbox attributes as well as the uncertainties indicated in Table 8.4—designers formulate the decision problem in terms of expected profit. The utility function in this case is

$$u(R,W,C) = RW - C\,,$$

Table 8.4: Uncertainty models for parameters used in the gearbox design problem.

Parameter	Value
System and Environment	
Total Vehicle Mass	N(240, 15) [kg]
External Drag Coefficient	N(0.4, 0.05) [N/(m/s^2)]
Internal Drag Coefficient	N(0.01, 0.0025) [N/rpm]
Course Roughness Coefficient	N (4, 0.5)
Winnings model uncertainty	N(0,5.5) [$]

where R is the reliability of the gearbox, W is the anticipated winnings assuming perfect reliability and C is the cost of building the gearbox. Details on how these attributes are computed from the gearbox attributes are given in Chapter 3.

The first step in the solution process is to perform requirements allocation for each gearbox configuration. Let $\theta = \left[N_g, \mu_R, \sigma_R^2, \mu_C, \sigma_C^2 \right]$ denote a coordinate vector in the generalized attribute space for the gearbox component. Also let $\tilde{\theta}_l = \left[N_g, \mu_R, \sigma_R^2, \sigma_C^2 \right]$ denote a vector of inputs to the gearbox tradeoff model, $T_l(\cdot)$, for concept $l = \{SGB, DGB, PGB\}$, where PGB, SGB and DGB denote planetary gearbox, single-sided gearbox and double-sided gearbox, respectively. Finally, let ϕ denote the random vector of system and environmental parameters that influence system-level attributes via the system model, $\mathbf{S}_l(\cdot)$. Thus, the requirements allocation decision is to find $\theta_l^* = \left[\tilde{\theta}_l^*, T_l\left(\tilde{\theta}_l^*\right) \right]$ such that

$$\begin{aligned} \tilde{\theta}_l^* &= \arg\max_{\tilde{\theta}_l \in \tilde{\Theta}_l} E\left[u\left(\mathbf{S}_l\left(\mathbf{y}_{\left[\tilde{\theta}_l, T_l(\tilde{\theta}_l)\right]}, \phi \right) \right) \right] \\ &= \arg\max_{\tilde{\theta}_l \in \tilde{\Theta}_l} \iint u\left(\mathbf{S}_l(\mathbf{y}), \phi\right) MN\left(\mathbf{y} \mid \left[\tilde{\theta}_l, T_l\left(\tilde{\theta}_l\right) \right]\right) f(\phi)\, d\mathbf{y}\, d\phi, \end{aligned}$$

where $\tilde{\Theta}_l$ is the domain description for the l^{th} tradeoff model and $f(\cdot)$ is the multivariate distribution function for the random vector ϕ. Since the component-level attributes, $\mathbf{y}$, and the uncertain system parameters, ϕ, are statistically independent, one can express the multivariate distributions as products of the nominal distributions. One can solve this problem using standard methods optimization under uncertainty.

Given the requirements allocation solutions for each gearbox configuration, one can select the configuration that is most preferred:

$$l^* = \underset{l \in \{PGB, SGB, DGB\}}{\arg\max} \; E\left[u\left(\mathbf{S}\left(\mathbf{y}_{\theta_l^*}, \phi \right) \right) \right].$$

8.4.3 Results

Table 8.5 (next page) contains results from the gearbox concept selection problem. It contains the tradeoff criteria and expected utilities corresponding to the most preferred implementation of each gearbox configuration as predicted using the tradeoff models. The final decision is to design the PGB concept. To continue development of this concept, designers can use the tradeoff criteria indicated in the table as design-to targets.

The problem also is solved using classical design optimization techniques, the results of which are listed in Table 8.5. These confirm the results based on tradeoff space search. The decision approach formulated using predictive tradeoff models yields the same decision—to develop the PGB concept—as one would reach using the more computationally taxing extensive search method. Furthermore, one can observe that the approach based on tradeoff models yields accurate predictions of the tradeoffs corresponding to the most preferred implementation of each concept. This indicates that the tradeoff models yield the right decision for the right reason.

The results from the tradeoff modeling approach are remarkably close to those from the reference solution. This is a consequence of using a large amount of data during tradeoff model generation, both in terms of the numbers of points in the design space that

Table 8.5: Results from gearbox example.

				Tradeoff Modeling Results	*Reference Solution*	*Percent Difference*
PGB	*Expected Utility*			1185	1185	0.0
	Gear Ratio		N_g	5.733	5.733	0.0
	Reliability	*Mean*	μ_R	0.99	0.99	0.0
		Std	σ_R^2	0.012	0.013	7.7
	Cost ($)	*Mean*	μ_C	275	275	0.0
		Std	σ_C^2	7.4	7.5	1.3
SGB	*Expected Utility*			1138	1138	0.0
	Gear Ratio		N_g	5.76	5.76	0.0
	Reliability	*Mean*	μ_R	0.989	0.989	0.0
		Std	σ_R^2	0.014	0.014	0.0
	Cost ($)	*Mean*	μ_C	320	320	0.0
		Std	σ_C^2	7.5	7.5	0.0
DGB	*Expected Utility*			1078	1081	-0.3
	Gear Ratio		N_g	5.76	5.76	0.0
	Reliability	*Mean*	μ_R	0.993	0.985	0.8
		Std	σ_R^2	0.008	0.02	60.0
	Cost ($)	*Mean*	μ_C	389	372	4.6
		Std	σ_C^2	7.3	7.5	2.7

are explored and the number of samples taken at each design site to estimate the means and variances. One obtains good estimates of the distribution parameters at many locations, which lead to accurate tradeoff models. Such accuracy is unlikely to be the case in general, particularly when sample data is more limited.

8.5 Conclusions and Chapter Summary

The main conclusion one can draw from this chapter is that it is possible for designers to deal with uncertain data rigorously in a tradeoff modeling framework. To do

so, designers must adopt an expanded interpretation of what constitutes a tradeoff space and use parameterized efficient sets that are based on a suitable stochastic dominance criterion. This is demonstrated through the use of the PMV-SSD criterion defined in Section 8.3 and the successful solution of a gearbox design problem.

The ideas presented in this chapter can serve as the foundation for tradeoff modeling under uncertainty, but this is not the final word on the subject. As one can observe from Table 8.2, dominance rules are more varied in the case of decisions under uncertainty. This research is based on one rule that is appropriate under fairly specific assumptions—that designers are risk averse and all uncertainty is statistically independent and normally distributed. Although these assumptions are met in some cases, they are restrictive in general and open questions remain about how to generalize the approach. Interesting questions include: "what assumptions about design risk attitude and data uncertainty are most representative of reality?" and "which stochastic dominance criteria apply to those situations?"

One interesting observation about the PMV-SSD rule is that it preserves a Pareto-like representation as a frontier in the mean-variance space. Much of the prior literature on tradeoff modeling is motivated by visualization and visual decision making, and it would seem that the frontier one obtains by applying PMV-SDD would be useful in that context. However, visualization is cumbersome when the number of dimensions is large. For a problem having N decision attributes, the number of tradeoff dimensions for the PMV-SSD rule is $2N$. This limits the viability of straightforward visualization techniques for tradeoff modeling under uncertainty.

CHAPTER 9:

CONTRIBUTIONS, LIMITATIONS AND OPEN QUESTIONS

This chapter is a review and synthesis of material from the preceding chapters. The main objective is to reexamine the research questions and hypothesis first introduced in Section 1.4 in light of the results reported in the other chapters. Section 9.1 is a recap of the questions, the corresponding hypotheses and the results of efforts to validate these hypotheses. Section 9.2 is a review of the main research contributions made during the course of this investigation. Section 9.3 is a summary of the main limitations of this research. Section 9.4 is brief exploration of potential avenues for future research relating to this problem or stemming from the ideas developed in this research. Section 9.5 contains a summary and closing remarks.

9.1 Review of the Research

The broad motivation for this research is summarized in Chapter 1 in terms of the following question:

> *How can designers model system-level decision alternatives quantitatively in order to support sound and effective system-level decision making?*

The research objective is to study a particular approach to modeling system-level decision alternatives. Section 1.3 is an outline of this approach, in which designers compose a system-level model using component-level models that they (or other designers) generate from attribute data about efficient implementations of a component.

These component-level models are predictive in the sense that designers can use them to answer the following question:

> *What would be the resulting attribute vector if a designer implemented the component in question with particular preferences for making tradeoffs among the component-level attributes?*

The presumption is that having a prediction of this attribute vector for each component in a system would enable designers to evaluate the system with confidence. By extension, they can choose between different system-level alternatives by repeating this prediction-and-evaluation procedure for each alternative. The specific research questions for this investigation are aimed at determining whether this approach is fundamentally correct.

9.1.1 Component-Level Dominance Analysis

The first research question and hypothesis are about identifying when a particular component is inferior to others in a manner that is independent of any specific design problem:

RQ1. *How can designers conclude that one implementation of a component dominates another when they lack specific knowledge of the system in which the component will be used?*

H1. *Designers can use the parameterized Pareto dominance rule to eliminate attribute data about dominated implementations of a component.*

The main evidence supporting Hypothesis 1 is as follows:

- Classical Pareto dominance is insufficient in many cases because there may be no problem-independent preference for one or more component-level attributes. The

assumptions required for classical Pareto dominance are expressed in Definition 3.1 and Theorem 3.1 and the basic argument in favor of a new approach is given in Section 3.1.3.

- Parameterized Pareto dominance is defined and shown to be mathematically sound in Section 3.1.4 (Definition 3.2 and Theorem 3.2). Although both the parameterized and the classical rules are sound, the parameterized rule is applicable under a broader set of assumptions that better suit the concept-level tradeoff modeling problem.
- Section 3.2.2 is a description of a procedure for formulating decisions by abstracting a single tradeoff model from multiple tradeoff models of a component using the parameterized Pareto dominance rule. The procedure is demonstrated successfully in the gearbox design example of Section 3.4. This further supports the correctness and appropriateness of the dominance rule.
- Tradeoff models based on parameterized Pareto set data are used to solve design problems in Chpater 3 (concept selection for a gearbox design problem), Chapter 6 (requirements allocation for a log splitter) and Chapter 7 (architecture selection for a hydraulic hybrid vehicle). If parameterized Pareto dominance was incorrect in some way—e.g., it leads one to eliminate solutions that could be most preferred—then one would expect to have difficulty on these examples. To the contrary, the tradeoff modeling approach is successful in each example.

9.1.2 Tradeoff Model Domain Description

Although phrased in terms of the tradeoff modeling problem, the second research question and hypothesis relate to a practical challenge of any data-driven modeling procedure:

RQ2. How can designers describe the set of valid inputs to a tradeoff model mathematically?

H2. Designers can use a domain description procedure based on kernel-based support vector domain description and clustering methods.

The main evidence in support of this hypothesis is as follows:

- A mathematical framework exists in the literature for describing the domain of a data set using a technique inspired by kernel-based support vector machines. The mathematical foundations of this is reviewed in Section 4.2.
- An extension of this framework is reported in the literature that allows one to identify distinct clusters in a data set. This theory is reviewed in Section 4.3 and the technique is demonstrated on synthetic data sets (Figures 4.5-4.7) and an actual data set (Figure 4.10).
- A procedure for applying support-vector domain description and support-vector clustering in the context of tradeoff modeling is given in Section 4.4. This fits within the procedure for generating tradeoff models described in Chpater 3..
- It is difficult to validate positively the use of the tradeoff model domain description approach. However, the example problems of Chapter 6 (log splitter) and Chapter 7 (hydraulic hybrid vehicle) would be difficult to complete if this approach was not effective. If the domain descriptions were deficient, one could expect either nonsensical solutions to the design problems (e.g., because the

domain would be failing to constrain the optimization search) or a failure to detect reasonable solutions (e.g., because the domain description rules out reasonable solutions).

9.1.3 Composing Tradeoff Models

The third research question is about making the component-level tradeoff models useful at a systems level:

RQ3. Under what conditions can designers compose component-level tradeoff models in order to model a system-level decision alternative soundly?

H3. One can compose predictive tradeoff models soundly if the tradeoff models are based on parameterized Pareto sets and all induced preferences for any component-level dominator attribute are monotonic in the same direction.

The main evidence supporting this hypothesis is as follows:

- Section 5.3 contains a mathematical analysis of composition problem. Assuming one is dealing with parameterized Pareto sets (as opposed to tradeoff models fit to that data), then the composition procedure is mathematically sound: if a an implementation of a component is dominated by parameterized Pareto criterion then it cannot be part of the most preferred system. Theorem 5.2 and Corollaries 5.3 and 5.4 are the relevant results. One can conclude that if a tradeoff model is a perfect representation of the actual parameterized efficient set of a component, then composition is valid.
- The design problems of Chapter 6 (log splitter) and Chapter 7 (hydraulic hybrid vehicle) are evidence that it is valid to compose tradeoff models that approximate the true parameterized efficient sets.

9.1.4 Tradeoffs under Uncertainty

The final research question and hypothesis are about the problem of dealing with uncertainty in the attribute data one would use to generate a tradeoff model:

RQ4. How should designers identify and visualize the (parameterized) efficient set of tradeoffs when the attribute data is uncertain?

H4. Designers can identify the (parameterized) efficient set of tradeoffs under uncertainty using (parameterized) stochastic dominance criteria and can visualize this set as a surface in mean-variance space.

The evidence in support of this hypothesis is as follows:

- From the survey of Section 8.2, one can conclude that stochastic dominance criteria are the appropriate mathematical constructs for performing dominance analysis under uncertainty in a multi-attribute utility theory setting.
- The multivariate mean-variance second-order stochastic dominance (MV-SSD) rule is defined and shown to be a consequence of other results from the literature (see Definition 8.1 and Theorem 8.1). Under the assumptions associated with MV-SSD, designers can visualize the efficient set as a curve or surface in the mean-variance space.
- A parameterized version of MV-SSD, denoted PMV-SSD, is introduced in Section 8.3 and shown to be sound under the prevailing assumptions (Definition 8.2 and Theorem 8.2).
- Section 8.3.1 contains explanations of the main assumptions underlying PMV-SSD. Although these assumptions do not hold in general, one can conclude that

they are reasonable at least some of the time and correspond to assumptions designers commonly make.

- The PMV-SSD rule is demonstrated successfully on a gearbox design problem (Section 8.4). The example is conducted under nearly ideal circumstances, and so questions remain about the general viability of the approach. However, the results are sufficient to conclude that the approach is internally consistent.

9.2 Summary of Contributions

9.2.1 Domain Description using Support Vector Machines

The mathematical framework for support-vector domain description and support-vector clustering existed in the literature. However, this research extends the basic framework to a complete procedure for constructing a domain description. It also is a novel application of domain description and clustering techniques in a systems design context.

The procedure for support-vector domain description is significant in that it is an indispensable part of the overall approach to tradeoff modeling. Numerical search routines—optimizers and design exploration codes—lack the intuition of a designer and therefore require a formalized expression of what constitutes a valid input to a model.

The domain description procedure also has significance beyond the tradeoff modeling approach to decision making. Interest in data-driven predictive modeling is on the rise within the design community as evidenced by publications in leading journals

and a recent workshop on the subject[m]. The domain description procedure described in Section 4.4 can be useful in this context.

9.2.2 Contributions to Dominance Analysis

This research includes several contributions relating to dominance analysis:

- A new dominance criterion, called parameterized Pareto dominance, is defined and validated mathematically.
- The soundness of composing efficient sets is studied mathematically for both classical and parameterized Pareto dominance.
- A new approach to decision making based on composing approximations of parameterized efficient sets, called tradeoff models, is demonstrated and shown to be effective.
- A new stochastic dominance criterion, called parameterized mean-variance second-order stochastic dominance, is defined and validated mathematically.

The new dominance criteria are sound extensions of existing criteria. From this research, one can conclude that they are useful constructs for systems design and have value from an applications perspective. The modeling approach studied in this research would not be possible without them and their general nature is conducive to broader application.

The general approach to system-level decision making studied in this research—based on abstracting problem-independent models of components from parameterized efficient set data—is a new and noteworthy departure from prior research. Prior work on tradeoff analysis in design is monolithic and problem-specific, requiring a full system

[m] Workshop on Performance Prediction in System-Level Design, held at the ASME International Design Engineering Technical Conferences (IDETC2008), Brooklyn, NY, 3 August 2008

model and precluding the reuse and composition of the resulting tradeoff models. Prior work on hierarchical decision-based design and optimization is based on tightly-coupled system and component models and generally requires more information than designers have available during system-level decision making. The approach investigated in this research addresses these limitations. Design processes are not large monolithic decision processes in practice, and it is important that research reflects this reality.

The soundness results for composing efficient and parameterized efficient sets are an important part of this research in terms of understanding when it is valid to eliminate a component from consideration. It also is interesting from the standpoint of understanding the system-level implications of component-level actions and has applications beyond this research. Section 9.4 includes some speculation about applying these dominance results to combinatorial optimization problems. Preliminary results are encouraging and further study is warranted.

9.2.3 Tradeoff Analysis under Uncertainty

Although there is much prior work in the design community on tradeoff analysis, nearly all of it is devoted to the case in which attribute values are known with certainty. The current research contributes a new perspective on how designers can model and visualize tradeoffs under uncertainty. The mathematics at the core of this perspective are established results from the decision theory and operations research communities. However, the application of these ideas to systems design problems is novel. They provide a mathematical basis for identifying efficient sets under uncertainty, an understanding of which previously was lacking in the design community.

This investigation includes only a preliminary foray into stochastic dominance and its potential applications to design. The achievements in the context of the tradeoff modeling approach studied in this research are modest. However, the more valuable contribution may be the introduction of ideas about mathematically-sound stochastic dominance criteria to the design community.

9.3 Limitations

There are several limitations or caveats associated with the approach to making system-level decisions using predictive tradeoff models. The following is a summary of and response to the more notable of these.

Scalability in the Number and Types of Attributes

Scalability could be a problem for certain types of components. In the examples presented here, most components had fewer than four or five attributes. Tradeoff model generation and validation is tractable under these circumstances, but it could become practically difficult if the number of attributes becomes large. It is unclear how many attributes would be too many, and such a threshold probably is component-dependent.

Another consideration is that some components have attributes that are not readily quantifiable or are otherwise difficult to work with in this setting. For example, the engine models in this investigation consisted of fairly abstract "peak" attributes—e.g., maximum power, maximum torque and the speeds at those points. These capture only a small aspect of engine behavior and it would be preferable from a simulation standpoint to have complete engine and fuel-consumption maps. However, it is unclear how one could incorporate information like this into a tradeoff model in a meaningful way.

The Attributes for a Component are Not Unique

There is no unique set of decision attributes for a given decision problem. One designer may select an engine primarily based upon its maximum power output and cost while another also considers mass and service life. The complication from a tradeoff modeling perspective is that a tradeoff model the first designer generates (with power and cost as attributes) is inadequate for the second designer.

Although this fact does impact the practical value of the tradeoff modeling approach, it is not indicative of a fundamental flaw. All models involve assumptions and it is a practical impossibility to create any model that is universally valid. In truth, there is remarkable commonality in the attributes systems designers use to describe a particular component. This is evidenced by parts catalogs that report a certain set of information for each type of component. It seems reasonable to infer that the data set reported for each component type is reflective of the attributes that designers use most commonly. Designers who generate tradeoff models can focus on those types of attributes, thereby maximizing the applicability of the models.

Preferences Can Change over Time / Exceptions to Preference Rules

Tradeoff modeling is based on the notion that for at least some component-level attributes there is a general and identifiable preference structure. Designers always want less cost, more reliability, less noise, and so forth. One criticism of the approach is that preferences may not be so fixed—that designers could change their mind during a project or simply have unconventional preferences at the outset.

The response to this is twofold. First, a change in preferences is not necessarily the same as changing a general property about preferences, such as monotonicity. There is little doubt that preferences can shift during the course of a design project—e.g.,

designers receive new marketing data that prompts them to rethink what they should design. However, one should think it rare for a designers to completely reverse their objectives—e.g., to go from wanting to maximize reliability to wanting to minimize it. Provided preferences do not undergo such a structural change, a tradeoff model will remain valid.

Second, it is impossible to account for all possible exceptions to a rule and some designers may be unable to reuse a preexisting tradeoff model. For example, although most designers would minimize mass (all other factors being equal), dynamics considerations might lead designers to treat the mass of particular components more like a target-type attribute. Designers should treat target-type attributes as parameters rather than dominators because target-matching preferences are incompatible with the monotonicity assumptions associated with dominator attributes.

The Impact of Environmental or Boundary Conditions

One challenge that has been ignored to this point is the problem of dealing with the influence of environmental and boundary conditions. For example, in the gearbox example one must assume loading conditions in order to estimate reliability.

There are two main ways to deal with this issue. One is to include some or all of the environmental variables in the tradeoff model. This would enable designers to account for their impact, but would complicate the models considerably. Another approach is to evaluate attributes impacted by external considerations under agreed-upon conditions. Although it is difficult for an industry to identify standard test conditions, this probably is the better approach and certainly reflects how parts catalogs deal with similar issues today. For example, the L10 lifetime rating for bearings works this way. Another

example is fuel consumption rates for engines, which usually are reported only at open throttle.

Data Availability

Data availability is a major practical consideration. Without data or a validated model with which one can generate data, it is impossible to generate a valid tradeoff model. Presently, data sources are limited. The examples in this research were made possible in part because corporate sponsors agreed to share some of their data. Other data was collected through a variety of means—on the internet, via phone conversations with vendors, etc.—but this process is painstaking and, at times, frustrating.

However, this does not mean research into data-driven modeling techniques lacks merit. If designers have data-driven methods that are effective, manufacturers and vendors will get the message and publish more of their data.

Generalizing across Discrete Alternatives

The tradeoff modeling approach is based on creating continuous models from discrete data samples. This has two principal advantages: it permits designers to make inferences about potential implementations of a component or system and it allows them to use search and optimization algorithms that perform better on continuously-valued models. The first advantage makes sense if designers are able to custom design their components (or have them custom built on their behalf). However, one can question the wisdom of doing this when designers ultimately will select off-the-shelf components for their system.

The response to this is that it depends. In some cases, many different sizes of a particular component are on the market and designers can be reasonably confident that one will exist that is "close enough" to the solution they obtain using their continuous

tradeoff model. However, if system utility is highly sensitive to the attributes of this component or there are few sizing choices available on the market, designers may be well advised to avoid fitting a tradeoff model and instead to work directly with the efficient set data.

9.4 Open Questions and Opportunities for Future Research

Practical Aspects and Extensions of Tradeoff Modeling

This research focuses on studying the foundations of tradeoff modeling rather than investigating the practical aspects of the approach. This leaves several open questions:

- Are tradeoff models really a general and reusable representations of a type of component? Although this seems to be the case in principle, it may be that practical considerations such as the logistics of model documentation and upkeep may detract significantly from their value. Another problem may be that the required dimensionality is unmanageably large for many types of components.
- Can designers update an existing tradeoff model easily given new data? This is important in determining whether tradeoff modeling is a practical approach to representing engineering knowledge. Intuitively, it would seem that such procedures are possible and practically reasonable. However, this remains to be demonstrated conclusively.
- Can companies use tradeoff models as an effective knowledge representation among a multidisciplinary team? While much about a detailed simulation of an engine would be meaningless to a marketing analyst, a tradeoff model that expresses a relationship between a company's technical capabilities and other enterprise concerns (manufacturing, pricing, environmental impact, etc.) might be

a useful mode of communication. The same idea applies to communication among designers with different engineering backgrounds.

- Can designers use tradeoff models to create technology forecasts? In the current research, tradeoff models are assumed to be static entities that describe a current state of affairs. However, as technology improves over time, a tradeoff model too must change. Designers may be able to achieve higher power outputs from their engines at less cost. This means a parameterized efficient frontier will move through the tradeoff space over time. Designers may be able to infer trends useful for strategic planning or identify opportunities in the marketplace by analyzing this motion over time.
- Can tradeoff models replace design catalogs (at least for some components)? One interpretation of a tradeoff model is that it is a computable representation of the information contained within a design catalog. Because it is abstract of design details it naturally shields a manufacturer from divulging trade secrets that lead to its competitive advantage. This is not necessarily the case for classic engineering analysis models, which compute higher-level attribute from lower-level design information. This approach may be useful in the context of mass customization, where manufacturers offer customers nearly infinite variety.
- Can designers use tradeoff models to account for attributes relating to environmental impact? In principle, there is no reason why this should not be possible provided the relevant attributes are quantitative measures. The larger challenge may lie in how to assess the environmental impact of a particular component. This issue goes well beyond system-level decision making, but

tradeoff modeling could be a useful connecting point between research on environmental impact and systems decisions.

Implications of the Decision Chain Perspective

The role in this research of the decision chain concept is to motivate a perspective on decision making that is broader than engineering optimization. The core idea is that design problems are too complex to be formulated as one large optimization problem. In the current research, this leads to a data-driven predictive modeling approach. The basic idea is that designers can infer much about the other decisions in a decision chain from this data without having to model the decision processes explicitly. However, this perspective has other implications.

A common definition for a decision is that it is "an irrevocable allocation of resources" (Hazelrigg 1996). From a detail design perspective, one can interpret this as a designer making choices about the product: How large should it be? What should it be made of? The allocated resources are the materials and labor that go into making the product. However, a decision-chain perspective implies a broader interpretation of what this allocation means. A systems designer makes relatively few choices about the engineered artifact that are not revocable. For example, even after choosing one particular system architecture, a design team can reevaluate and backtrack to go with another alternative. What, then, has the system designer allocated irrevocably?

The only commitment of resources dictated by the systems designer in this example relate to the design process itself: the time spent by the design team developing the system according to a particular architecture, including any computational resources devoted to the project. Designers require a better understanding of their decisions from this perspective. One of the basic arguments from the set-based design literature is that it

can be advantageous for designers to develop a large number of alternatives in parallel (Sobek II, et al. 1999). However, it remains unclear how designers should determine how many alternatives to consider or when to narrow the field down to a single solution strategy. Although some research exists on characterizing the relationship between a design process and its outcome (e.g.,(Sobek II and Jain 2007)), it remains a largely open topic.

Combinatorial Optimization

The parameterized Pareto dominance criterion may be useful for certain types of combinatorial optimization problems. Table 9.1 is a condensed version of Table 6.2 that contains the numbers of components in the hydraulics database used for the log splitter problem before and after dominance analysis. What is striking about this table is the last column: the number of valid component combinations. By taking a purely combinatorial approach, one could construct over 24 million valid log splitter systems from the original database. Even after eliminating a number of components using rudimentary outlier analysis, one is left with over 10 million combinations. However, after dominance analysis one has only a half-million combinations—a rather manageable number.

Table 9.1: Components in database before and after dominance analysis.

Component	*Engine*	*Pump*	*Cylinder*	*Control Valve*	*Total # of Combinations*	*Hours to Evaluate**
Total # in DB	59	61	188	36	24,358,032	6.8
# aft. outlier analysis	49	43	158	32	10,652,992	2.9
# aft. dom. analysis	19	24	137	8	499,776	0.14

**Assuming 1ms computation time per combination.*

To put these results in perspective, suppose each configuration requires one millisecond to evaluate. To consider all combinations of the original database, one would require almost seven hours while the evaluations would be a matter of minutes after applying the parameterized Pareto dominance criterion—a speedup of 98%! The potential value of dominance analysis is even more apparent for longer computation times: if evaluations instead take one-tenth of a second, the full combinatorial search would take over 28 days, while the post-dominance analysis search would require about a half day (14 hours).

What makes this speedup possible are the theoretical results from Chapter 5, specifically the soundness results for composing parameterized Pareto sets. Those results guarantee that eliminating components via parameterized Pareto dominance is sound from a systems perspective—i.e., designers will not eliminate a component that would have been part of the best system configuration.

Although these preliminary results are very encouraging, the true value of this approach depends on how typical it is for designers to eliminate large numbers of components using parameterized Pareto dominance. It is likely that this particular example is atypical, but the effect would be notable even if the average speedup were more modest. Future research is required to ascertain what kind of improvements are likely and to characterize the search approach mathematically.

Applications of Support-Vector Domain Description

The support vector-domain description and clustering techniques introduced in Chapter 4 have many potential applications in design.

- The most critical application may be in the context of validating models generated from observational data. Although often an afterthought, it is important for model

developers to describe the context in which others may validly use their models (Malak and Paredis 2007).

- The SVDD and SVC techniques also may be useful in the context of strategic planning for design projects. Domain description and clustering may be useful in identifying different types of customers or in identifying unexploited niches in the product landscape.
- Support-vector domain description is applicable to outlier detection and therefore can be useful for quality control in manufacturing. Typical classification algorithms require examples of both positive and negative results (i.e., good and bad parts) to learn the desired associations. In contrast, the SVDD algorithm requires only positive examples to learn a decision boundary. This could be advantageous.
- A SVDD can serve as a compact representation for a very large set of data. It may require only a handful of support vectors to represent thousands of data points. This can be useful in the context of data mining applications relating to design.

Tradeoffs under Uncertainty

Ample room exists for further research into applications of stochastic dominance rules in systems design. Just within the context of tradeoff modeling, several open questions exist.

- Are there other distributions for which the mean-variance space is a rigorous tradeoff representation? As noted in Chapter 8, this is known to be the case for normal and log-normal distributions. Other types of distributions are common on engineering problems, including the uniform, triangular and exponential. Are any

of these compatible with this representation? If not, what is a suitable representation for them?

- When is the benefit of modeling tradeoffs under uncertainty worth the added complexity? This is a difficult, but important question. Even using somewhat aggressive assumptions, the PMV-SSD rule requires twice as many dimensions as parameterized Pareto dominance. At what point to other uncertainties in the problem dominate those due to the underlying data?
- What is the practical implication of assuming attributes are statistically independent? This assumption simplifies the representation considerably, but probably is, strictly speaking, incorrect in most cases. What is the impact of modeling the distributions as independent when they really are not? What recourse is available to designers when independence is a very poor assumption?
- Is there practical value in visualizing efficient tradeoffs in a generalized tradeoff space (e.g., mean-variance space)? Such a space has high dimensionality and it can be problematic to visualize relationships in such a space. For this to be practical, researchers must consider the meaning of these relationships and provide practicing designers with insight into how to interpret them. Research into advanced visualization tools also may be useful.

9.5 Summary

System-level decision making requires both insight and foresight about the design problem and the potential solutions to it. This type of knowledge is difficult for designers to formalize. In fact, they commonly rely on qualitative approaches due to the difficulty of evaluating their decision alternatives using quantitative models. This is not because it

is difficult for designers to model the interactions between and behaviors of individual system components mathematically. Instead, it is due to the difficulty in arriving at reasonable estimates for the likely attributes of those components. Put in the simplest of terms, it would be invalid to conclude that an all-electric vehicle is superior to a hybrid one based on exaggerated assumptions about how much electrical power storage one can have for a given cost and mass.

This research is an investigation of a predictive approach to modeling system-level decision alternatives. The core premise is that designers can make inferences about how they or other designers would implement a particular component based on attribute data about prior implementations of that type of component. They can use this information at the systems level by composing together the component-level models. The component-level models are based on a new decision-theoretic construct, called a parameterized efficient set. The internal consistency of this approach is validated using mathematical analysis and practical applicability is demonstrated on representative design problems.

The contributions from this research are significant in the context of this modeling approach and there is reason to believe they also have broader significance. Several potential extensions to the modeling approach and alternative applications of these research ideas have been explored. Whether this particular approach to modeling system-level alternatives is the best solution to the problem remains unclear. However, one can conclude that the contributions made in this research are useful waypoints on the path to a lasting and effective solution to the problem.

APPENDIX A:

PROOFS OF MATHEMATICAL STATEMENTS

A.1 Theorem 1

The theorem follows directly from the definitions of classical Pareto dominance and monotonicity. From Definition 3.1 (page 54), $\mathbf{z}'$ DOM $\mathbf{z}''$ means $z_i' \geq z_i'' \quad \forall i = 1 \ldots N$ and $z' > z_i'' \quad \exists i = 1 \ldots N$. From Definition 5.1, if a scalar function, $V(\cdot)$, is monotonically increasing in $\mathbf{z} = [z_1, z_2, \ldots, z_N]$ then the condition $z_i' \geq z_i'' \quad \forall i = 1 \ldots N$ and $z' > z_i'' \quad \exists i = 1 \ldots N$ implies $V(\mathbf{z}') > V(\mathbf{z}'')$ ■

A.2 Theorem 2

The theorem follows directly from the definitions of parameterized Pareto dominance and monotonicity. From Definition 3.2 (page 57), $\mathbf{z}'$ PDOM $\mathbf{z}''$ means if $z_i' = z_i'' \; \forall i \in P$, $z_i' \geq z_i'' \; \forall i \in D$ and $z_i' > z_i'' \; \exists i \in D$ which is special case of $z_i' \geq z_i'' \quad \forall i = 1 \ldots N$ and $z' > z_i'' \quad \exists i = 1 \ldots N$ since $P \cup D = \{1 \ldots N\}$ and $P \cap D = \varnothing$. From Definition 5.1, if a scalar function, $V(\cdot)$, is monotonically increasing in $\mathbf{z} = [z_1, z_2, \ldots, z_N]$ then the condition $z_i' \geq z_i'' \quad \forall i = 1 \ldots N$ and $z' > z_i'' \quad \exists i = 1 \ldots N$ implies $V(\mathbf{z}') > V(\mathbf{z}'')$ ■

A.3 Theorem 5.1

The theorem is proved using a direct approach. One can state Theorem 5.1 as:

$$\mathbf{y}_k' \text{ DOM } \mathbf{y}_k'' \;\; \forall \mathbf{y}_k', \mathbf{y}_k'' \in \mathbf{Y}_k \rightarrow \mathbf{S}(\mathbf{y}') \text{ DOM } \mathbf{S}(\mathbf{y}'') \;\; \exists \mathbf{y}', \mathbf{y}'' \in \mathbf{Y},$$

where it is understood that $\mathbf{y}' = [\mathbf{y}'_k, \mathbf{y}'_{-k}]$ and $\mathbf{y}'' = [\mathbf{y}''_k, \mathbf{y}''_{-k}]$ such that $\mathbf{y}'_{-k}, \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$, and the stated monotonicity condition holds. Using the definition for classical Pareto dominance (Definition 3.1) yields, for all $\mathbf{y}'_k, \mathbf{y}''_k \in \mathbf{Y}_k$:

$$\begin{gathered}\left(y'_{kj} \geq y''_{kj} \;\; \forall j = 1 \ldots M_k\right) \wedge \left(y'_{kj} > y''_{kj} \;\; \exists j = 1 \ldots M_k\right) \rightarrow \\ \left(\begin{array}{l} S_i\left(\mathbf{y}'_{(i)}\right) \geq S_i\left(\mathbf{y}''_{(i)}\right) \;\; \forall i = 1 \ldots N \;\; \wedge \\ \quad S_i\left(\mathbf{y}'_{(i)}\right) > S_i\left(\mathbf{y}''_{(i)}\right) \;\; \exists i = 1 \ldots N \end{array}\right) \exists \mathbf{y}', \mathbf{y}'' \in \mathbf{Y}\end{gathered} \tag{A1}$$

where $\mathbf{y}_{(i)}$ denotes the vector of attributes used in the i^{th} system composition model. There always exists a $\mathbf{y}' = [\mathbf{y}'_k, \mathbf{y}'_{-k}]$ and $\mathbf{y}'' = [\mathbf{y}''_k, \mathbf{y}''_{-k}]$ such that $\mathbf{y}', \mathbf{y}'' \in \mathrm{Y} = \mathrm{Y}_1 \times \mathrm{Y}_2 \times \cdots \times \mathrm{Y}_K$ and $\mathbf{y}'_{-k} = \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$. Thus, to prove the proposition, one need only prove that the premise implies the conclusion for attribute vectors constructed this way. The conclusion of Equation (A1) involves two comparisons joined by a conjunction, and so both must hold. The first inequality must hold for all system composition models. Let $I_k \in \{1 \ldots N\}$ denote the set of indices corresponding to system composition models with an attribute from component k as an input and I_{-k} be its compliment. Strict equality holds for $S_i(\cdot) \;\; \forall i \in I_{-k}$, since their inputs come from $\mathbf{y}'_{-k} = \mathbf{y}''_{-k}$—i.e., $S_{\mathrm{i}}\left(\mathbf{y}'_{(i)}\right) = S_i\left(\mathbf{y}''_{(i)}\right) \;\; \forall i \in I_{-k}$. The $S_i(\cdot) \;\; \forall i \in I_k$ have inputs such that $y'_{(i)m} \geq y''_{(i)m} \;\; \forall m = 1 \ldots M_i$ because either the attribute is not from component k (in which case $y'_{(i)m} = y''_{(i)m}$) or it is from component k, in which case $y'_{(i)m} \geq y''_{(i)m}$ is assumed via the theorem's premise. Every $S_i(\cdot)$ for $i \in I_k$ is strictly increasing in any attribute from component k and $y'_{(i)m} \geq y''_{(i)m} \;\; \forall m = 1 \ldots M_i, i \in I_k$, so it follows from the definition of

strict monotonicity that $S_{\mathrm{i}}\left(\mathbf{y}'_{(i)}\right) > S_i\left(\mathbf{y}''_{(i)}\right)\ \forall i \in I_k$. Thus, since $I_k \cup I_{-k} = \{1 \ldots N\}$, one has $S_{\mathrm{i}}\left(\mathbf{y}'_{(i)}\right) \geq S_i\left(\mathbf{y}''_{(i)}\right)\ \forall i = 1 \ldots N$. The second inequality requires only existence to hold and was proved in the course of proving the first inequality—i.e., because $S_{\mathrm{i}}\left(\mathbf{y}'_{(i)}\right) > S_i\left(\mathbf{y}''_{(i)}\right)\ \forall i \in I_k$ it follows that $S_{\mathrm{i}}\left(\mathbf{y}'_{(i)}\right) > S_i\left(\mathbf{y}''_{(i)}\right)\ \exists i = 1 \ldots N$ ■

A.4 Corollary 5.1

Since $\mathbf{Y}^*_{-k} \subset \mathbf{Y}_{-k}$, one can substitute the statement $\mathbf{y}'_{-k} = \mathbf{y}''_{-k} \in \mathbf{Y}_{-k}$ for $\mathbf{y}'_{-k} = \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$ in the proof for Theorem 5.1 and the proof still works. Thus, it follows that the corollary is true ■

A.5 Corollary 5.2

It follows that if every system composition model is strictly increasing in every variable, then every component satisfies the scenario for Theorem 5.1. Since one can apply Theorem 5.1 to every component, it follows that the corollary holds ■

A.6 Theorem 5.2

The logic for this proof follows that for Theorem 5.1. One can restate Theorem 5.2 as:

$$\mathbf{y}'_k \,\mathrm{PDOM}\, \mathbf{y}''_k\ \ \forall \mathbf{y}'_k, \mathbf{y}''_k \in \mathbf{Y}_k \rightarrow \mathbf{S}(\mathbf{y}')\,\mathrm{DOM}\,\mathbf{S}(\mathbf{y}'')\ \ \exists \mathbf{y}', \mathbf{y}'' \in \mathrm{Y}.$$

Recall that $D \subset \{1 \ldots M\}$ is the set of indices for the dominator attributes and $P \subset \{1 \ldots M\}$ is the set of indices for the parameter attributes and each are defined such that $D \cap P = \varnothing$, $D \cup P = \{1 \ldots M\}$ and $D \neq \varnothing$. Applying the dominance definitions yields, for all $\mathbf{y}'_k, \mathbf{y}''_k \in \mathbf{Y}_k$:

$$\left(y'_{kj} = y''_{kj} \;\; \forall j \in P\right) \wedge \left(y'_{kj} \geq y''_{kj} \;\; \forall j \in D\right) \wedge \left(y'_{kj} > y''_{kj} \;\; \exists j \in D\right) \rightarrow$$
$$\left(\begin{array}{l} S_i\left(\mathbf{y}'_{(i)}\right) \geq S_i\left(\mathbf{y}''_{(i)}\right) \;\; \forall i = 1 \ldots N \;\; \wedge \\ \quad S_i\left(\mathbf{y}'_{(i)}\right) > S_i\left(\mathbf{y}''_{(i)}\right) \;\; \exists i = 1 \ldots N \end{array}\right) \exists \mathbf{y}', \mathbf{y}'' \in \mathbf{Y}$$

Let $I_k \neq \varnothing$ denote the set of indices for system composition models, $S_i(\cdot)$, with one or more attributes from component $k \in \{1 \ldots K\}$ as an input and I_{-k} denote its compliment. As in the case for Theorem 5.1, there always exists tradeoffs $\mathbf{y}' = [\mathbf{y}'_k, \mathbf{y}'_{-k}]$ and $\mathbf{y}'' = [\mathbf{y}''_k, \mathbf{y}''_{-k}]$ such that $\mathbf{y}'_{-k} = \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$. Let $\mathbf{y}_k = [\mathbf{y}_{kD}, \mathbf{y}_{kP}]$, where $\mathbf{y}_{kP}$ is a vector consisting of all the attributes of subsystem k treated as parameters (the y_{kj} for $j \in P$) and $\mathbf{y}_{kD}$ is a vector of the remaining attributes (the y_{kj} for $j \in D$). Thus, one has $\mathbf{y}'_k = [\mathbf{y}'_{kD}, \mathbf{y}'_{kP}, \mathbf{y}'_{-k}]$ and $\mathbf{y}''_k = [\mathbf{y}''_{kD}, \mathbf{y}''_{kP}, \mathbf{y}''_{-k}]$. According to the premise of Theorem 5.2, $\mathbf{y}'_{kP} = \mathbf{y}''_{kP}$ and $\mathbf{y}'_{kD}$ DOM $\mathbf{y}''_{kD}$. By letting $\mathbf{y}_= = [\mathbf{y}_{kP}, \mathbf{y}_{-k}]$, one has $\mathbf{y}' = [\mathbf{y}'_{kD}, \mathbf{y}'_=]$ and $\mathbf{y}'' = [\mathbf{y}''_{kD}, \mathbf{y}''_=]$ such that $\mathbf{y}'_= = \mathbf{y}''_=$, which is equivalent to the case proved for Theorem 5.1. Therefore, it follows that Theorem 5.2 holds ■

A.7 Corollary 5.3

Since $\mathbf{Y}^*_{-k} \subset \mathbf{Y}_{-k}$, one can substitute the statement $\mathbf{y}'_{-k} = \mathbf{y}''_{-k} \in \mathbf{Y}_{-k}$ for $\mathbf{y}'_{-k} = \mathbf{y}''_{-k} \in \mathbf{Y}^*_{-k}$ in the proof for Theorem 5.2 and the proof still works. Thus, it follows that the corollary is true ■

A.8 Corollary 5.4

It follows that if every system composition model is strictly increasing in every dominator attribute, then every component satisfies the scenario for Theorem 5.2. Since one can apply Theorem 5.2 to every component, it follows that the corollary holds ■

A.9 Theorem 8.1

Several authors prove that the mean-variance assumptions hold for the single-attribute case (e.g., (Tobin 1958, Hanoch and Levy 1969, Baron 1977)). Combining this result with Theorem 2 in (Huang, et al. 1978) yields a complete proof of the multi-attribute case of Theorem 8.1 ■

A.10 Theorem 8.2

Proof follows directly from the definition of MV-SSD. Since $\mu_{i,a} = \mu_{i,b} \ \forall i \in P$, $\mu_{j,a} \geq \mu_{j,b} \ \forall j \in D$ and $P \cup D = \{1,\ldots,N\}$, one can conclude that $\mu_{k,a} \geq \mu_{k,b} \ \forall k = 1\ldots N$. This yields the conditions for MV-SSD (Definition 8.1) and so dominance holds and the theorem is proved ■

REFERENCES

Achten, P. A. J. (2008). "A Serial Hydraulic Hybrid Drive Train for Off-Road Vehicles." 51st National Conference on Fluid Power (NCFP 2008). 515-21.

AIAA (1998). "Guide for the Verification and Validation of Computational Fluid Dynamics Simulations." AIAA-G-077-1998. American Institute of Aeronautics and Astronautics, Reston, VA.

Aird, F. (2000). *Automotive Math Handbook*. MotorBooks/MBI Publishing.

Aizerman, M., Braverman, E. and Rozonoer, L. (1964). Theoretical Foundations of the Potential Function Method in Patter Recognition Learning. *Automation and Remote Control*, 25, 821-37.

Akao, Y. (2004). *Quality Function Deployment: Integrating Customer Requirements into Product Design* (G. H. Mazur, Trans.). New York: Productivity Press.

Alexandrov, N. M. and Lewis, R. M. (1999). "Comparative Properties of Collaborative Optimization and Other Approaches to MDO." technical Report NASA/CR-1999-209354. Institute for Computer Applications in Science and Engineering, NASA Langley, Hampton, VA.

Aronszajn, N. (1950). Theory of Reproducing Kernels. *Transactions of the American Mathematics Society*, 68(3), 337-404.

ASME (2006). "Guide for Verification and Validation in Computational Solid Mechanics." ASME Standard American Society of Mechanical Engineers, New York.

Baccelli, F. and Makowski, A. M. (1989). Multidimensional Stochastic Ordering and Associated Random Variables. *Operations Research*, 37(3), 478-87.

Balci, O. (1997). Principles of Simulation Model Validation, Verification and Testing. *Transactions of the Society for Computer Simulation International*, 14(1), 3-12.

Balling, R. (1999). "Design by Shopping: A New Paradigm." Third World Congress of Structural and Multidisciplinary Optimization (WCMSO-3), Buffalo, NY. 295-7.

Barber, C. B., Dobkin, D. P. and Huhdanpaa, H. T. (1996). The Quickhull Algorithm for Convex Hulls. *ACM Transactions on Mathematical Software*, 22(4), 469-83.

Baron, D. P. (1977). On the Utility Theoretic Foundations of Mean-Variance Analysis. *Journal of Finance*, 32(5), 1683-97.

Barton, J. A. and Love, D. M. (2000). Design Decision Chains as a Basis for Design Analysis. *Journal of Engineering Design*, 11(3), 283-97.

Barton, J. A., Love, D. M. and Taylor, G. D. (2001). Evaluating Design Implementation Strategies using Enterprise Simulation. *International Journal of Production Economics*, 72(3), 285-99.

Bascaran, E., Bannerot, R. B. and Mistree, F. (1989). Hierarchical Selection Decision Support Problems in Conceptual Design. *Engineering Optimization*, 14(3), 207-38.

Ben-Hur, A., Horn, D., Siefelmann, H. T. and Vapnik, V. (2001). Support Vector Clustering. *Journal of Machine Learning Research*, 2(2), 125-37.

Berezkin, V. E., Kamenev, G. K. and Lotov, A. V. (2006). Hybrid Adaptive Methods for Approximating a Nonconvex Multidimensional Pareto Frontier. *Computational Mathematics and Mathematical Physics*, 46(11), 1918-31.

Bernhard, B. (2004). "Hybrid Drives for Off-Road Vehicles." FISITA World Automotive Congress, Barcelona, Spain, 23-27 May 2004.

Boehm, B. W. (1981). *Software Engineering Economics*. Upper Saddle River, NJ: Prentice Hall.

--- (1988). "A Spiral Model of Software Development and Enhancement." *IEEE Computer*, 21(5), 61-72.

Bras, B. and Mistree, F. (1993). Robust Design using Compromise Decision Support Problems. *Engineering Optimization*, 21, 213-39.

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.

Bruns, M. (2006). *Propagation of Imprecise Probabilities through Black Box Models*. M.S. thesis, Georgia Institute of Technology.

Buede, D. M. (2000). *The Engineering Design of Systems*. New York: John Wiley & Sons.

Chen, W., Allen, J. K., Marvis, D. N. and Mistree, F. (1996). A Concept Exploration Method for Determining Robust Top-Level Specifications. *Engineering Optimization*, 26, 137-58.

Chen, W., Allen, J. K. and Mistree, F. (1997). A Robust Concept Exploration Method for Enhancing Productivity in Concurrent Systems Design. *Concurrent Engineering*, 5(3), 203-17.

Clemen, R. T. (1996). *Making Hard Decisions: An Introduction to Decision Analysis* (2nd ed.). Pacific Grove, CA: Duxbury Press.

Cramer, E. J., Dennis Jr., J. E., Frank, P., Lewis, R. M. and Shubin, G. R. (1994). Problem Formulation for Multidisciplinary Optimization. *SIAM Journal of Optimization*, 4, 754-76.

Daschbach, J. M. and Apgar, H. (1988). Design Analysis through Techniques of Parametric Cost Estimation. *Engineering Costs and Production Economics*, 14(2), 87-93.

Davis, A. M. (1993). *Software Requirements: Objects Functions and States*. Upper Saddle River, NJ: Prentice Hall.

Dean, J. (1976). *Statistical Cost Estimation*. Bloomington, IN: Indiana University Press.

Dewulf, W. (2003). *A Pro-Active Approach to Ecodesign: Framework and Tools*. Ph.D. thesis, Katholieke Universiteit Leuven.

Donndelinger, J. (2006). "A Decision-Based Perspective on the Vehicle Development Process." In K. Lewis, W. Chen and L. C. Schmidt (Eds.), *Decision Making in Engineering Design* (pp. 217-25). New York: American Society of Mechanical Engineers.

Eaton Corp. (1998). "Pump and Motor Sizing Guide." Eaton Corporation Hydraulics Division, Eden Prarie, MN.

Farineau, T., Rabenasolo, B., Castelain, J. M., Meyer, Y. and Duverlie, P. (2001). Use of Parametric Models in an Economic Evaluation Step During the Design Phase. *International Journal of Advanced Manufacturing Technology*, 17(2), 79-86.

Ferguson, S., Gurnani, A., Donndelinger, J. and Lewis, K. (2005). A Study of Convergence and Mapping in Preliminary Vehicle Design. *International Journal of Vehicle Systems Modelling and Testing*, 1(1/2/3), 192-215.

Fernandez, M. G., Seepersad, C. C., Rosen, D. W., Allen, J. K. and Mistree, F. (2005). Decision Support in Concurrent Engineering - The Utility-based Selection Decision Support Problem. *Concurrent Engineering: Research and Applications*, 13(1), 13-27.

Filippone, M., Camastra, F., Masulli, F. and Rovetta, S. (2007). A Survey of Kernel and Spectral Methods for Clustering. *Pattern Recognition*, 41(1), 176-90.

Finch, W. W. and Ward, A. C. (1997). "A Set-based System for Eliminating Infeasible Designs in Engineering Problems Dominated by Uncertainty." 1997 ASME Design Engineering Technical Conferences, Sacramento, CA, Sept 14-17. American Society of Mechanical Engineers, Paper No. DETC97/DTM-3886.

Fishburn, P. C. (1965). *Decision and Value Theory*. New York: Wiley.

--- (1978). Stochastic Dominance without Transitive Preferences. *Management Science*, 24(12), 1268-77.

Forsberg, K. and Mooz, H. (1992). The Relationship of Systems Engineering to the Project Cycle. *Engineering Management Journal*, 4(3), 36-43.

Geisser, S. (1982). Aspects of the Predictive and Estimative Approaches in the Determination of Probabilities. *Biometrics*, 38, 75-85.

--- (1993). *Predictive Inference: An Introduction*. New York: Chapman & Hall.

Gu, X., Renaud, J. E., Ashe, L. M., Batill, S. M., Budhiraja, A. S. and Krajewski, L. J. (2002). Decision-based Collaborative Optimization. *Journal of Mechanical Design*, 124(1), 1-13.

Gurnani, A., Ferguson, S., Lewis, K. E. and Donndelinger, J. (2006). A Constraint-based Approach to Feasibility Assessment in Preliminary Design. *Artificial Intelligence in Engineering Design, Analysis and Manufacturing*, 20(4), 351-67.

Han, J. and Kamber, M. (2001). *Data Mining: Concepts and Techniques*. San Francisco, CA: Morgan Kaufmann.

Hand, D. J., Mannila, H. and Smyth, P. (2001). *Principles of Data Mining*. Cambridge, MA: MIT Press.

Hanoch, G. and Levy, H. (1969). The Efficiency Analysis of Choices Involving Risk. *The Review of Economic Studies*, 36, 335-46.

Harada, K., Sakuma, J. and Kobayashi, S. (2007). "Uniform sampling of local pareto-optimal solution curves by pareto path following and its applications in multi-objective GA." 9th annual conference on Genetic and evolutionary computation, London, England. 813-20.

Hazelrigg, G. A. (1996). *Systems Engineering: An Approach to Information-based Design*. Upper Saddle River, NJ: Prentice-Hall.

--- (1998). A Framework for Decision-Based Engineering Design. *ASME Journal of Mechanical Design*, 120, 653-8.

--- (2003). Validation of Engineering Design Alternative Selection Methods. *Engineering Optimization*, 35(2), 103-20.

Herrmann, J. W. and Schmidt, L. C. (2006). "Product Development and Decision Production Systems." In K. E. Lewis, W. Chen and L. C. Schmidt (Eds.), *Decision Making in Engineering Design* (pp. 227-42). New York: American Society of Mechanical Engineers.

Hsu, W. and Woon, I. (1998). Current and Future Research in the Conceptual Design of Mechanical Products. *Computer Aided Design*, 30(5), 377-89.

Huang, C. C., Kira, D. and Vertinsky, I. (1978). Stochastic Dominance Rules for Multi-Attribute Utility Functions. *The Review of Economic Studies*, 45(3), 611-5.

Huang, G. Q. (Ed. (1996). *Design for X: Concurrent Engineering Imperatives*. Springer.

Hume, D. (1965). *A Treatise of Human Nature*. Oxford: Clarendon Press. (Original work published in 1739-40).

Hunt, B. J., Blouin, V. Y. and Wiecek, M. M. (2007). Modeling Relative Importance of Design Criteria with a Modified Pareto Preference. *Journal of Mechanical Design*, 129(9), 907-14.

Jin, R., Du, X. and Chen, W. (2003). The Use of Metamodeling Techniques for Optimization under Uncertainty. *Structural and Multidisciplinary Optimization*, 25(2), 99-116.

Jun, S.-H. and Oh, K.-W. (2006). "A Competitive Co-Evolving Support Vector Clustering." *Neural Information Processing* (pp. 864-73). Berlin: Springer.

Karandikar, H., Srinivasan, R., Mistree, F. and Fuchs, W. J. (1989). Compromise: An Effective Approach for the Design of Pressure Vessels using Composite Materials. *Computers & Structures*, 33(6), 1465-77.

Keeney, R. L. and Raiffa, H. (1993). *Decisions with Multiple Objectives* (2nd ed.). Cambridge, UK: Cambridge University Press.

Kim, H. M., Michelena, N. F., Papalambros, P. Y. and Jiang, T. (2003). Target Cascading in Optimal System Design. *Journal of Mechanical Design*, 125(3), 474-80.

Kleindorfer, G. B., O'Neill, L. and Ganeshan, R. (1998). Validation in Simulation: Various Positions in the Philosophy of Science. *Management Science*, 44(8), 1087-99.

Kokkolaras, M., Louca, L. S., Delagrammatikas, G. J., Michelena, N. F., Filipi, Z. S., Papalambros, P. Y., Stein, J. L. and Assanis, D. N. (2004). Simulation-based Optimal Design of Heavy Trucks by Model-based Decomposition: An Extensive Analytical Target Cascading Case Study. *International Journal of Heavy Vehicle Systems*, 11(3/4), 402-32.

Kroo, I. and Manning, V. (2000). "Collaborative Optimization: Status and Directions." 8th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA. AIAA, Paper No. AIAA-2000-4721.

Krus, P. (2005). "Estimation Models for Concept Optimisation of Power Transformation and Transmission." The Ninth Scandinavian International Conference on Fluid Power (SICFP'05), Linkoping, Sweden.

Kuppuraju, N., Ittimakin, P. and Mistree, F. (1985). Design through Selection: A Method that Works. *Design Studies*, 6(2), 91-106.

Kutner, M. H., Nachtsheim, C. J., Neter, J. and Li, W. (2005). *Applied Linear Statistical Models* (5th ed.). New York: McGraw-Hill/Irwin.

Kyburg, H. E. and Pittarelli, M. (1996). Set-based Bayesianism. *IEEE Transactions on Systems, Man and Cybernetics*, 26(3), 324-39.

Law, A. M. and McComas, M. G. (2001). "How to Build Valid and Credible Simulation Models." Winter Simulation Conference (B. A. Peters, J. S. Smith, D. J. Medeiros and M. W. Rohrer, Eds.).

Lee, J. and Lee, D. (2005). An Improved Cluster Labeling Method for Support Vector Machines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3), 461-4.

Lee, S.-H. and Daniels, K. (2005). "Gaussian Kernel Width Generator for Support Vector Clustering." In M. He, G. Narasimhan and S. Petoukhov (Eds.), *Advances in Bioinformatics and Its Applications* (pp. 151-62). Fort Lauderdale, FL: World Scientific.

--- (2006). "Cone Cluster Labeling for Support Vector Clustering." 6th SIAM International Conference on Data Mining, Bethesda, MD (J. Ghosh, D. Lambert, D. Skillicorn and J. Srivastava, Eds.). 484-8.

Levhari, D., Paroush, J. and Peleg, B. (1975). Efficiency Analysis for Multivariate Distributions. *The Review of Economic Studies*, 42(1), 87-91.

Levy, H. (1973). Stochastic Dominance Among Log-Normal Prospects. *International Economic Review*, 14, 601-14.

--- (1990). "Stochastic Dominance." In J. Eatwell, M. Milgate and P. Newman (Eds.), *Utility and Probability* (pp. 251-4). New York: Norton & Company.

--- (1992). Stochastic Dominance and Expected Utility: Survey and Analysis. *Management Science*, 38(4), 555-93.

Lewis, K., Chen, W. and Schmidt, L. C. (Eds.). (2006). *Decision Making in Engineering Design*. New York: American Society of Mechanical Engineers.

Liker, J. K., Sobek II, D. K., Ward, A. C. and Cristiano, J. J. (1996). Involving Suppliers in Product Development in the United States and Japan: Evidence for Set-based

Concurrent Engineering. *IEEE Transactions on Engineering Management*, 43(2), 165-78.

Lophaven, S. N., Nielsen, H. B. and Sondergaard, J. (2002). "DACE: A Matlab Kriging Toolbox." Technical Report IMM-TR-2002-12. Technical University of Denmark.

Lotov, A. V., Bushenkov, V. A. and Kamenev, G. K. (2004). *Interactive Decision Maps*. Boston: Kluwer Academic Publishers.

Luce, R. D. and Raiffa, H. (1957). *Games and Decisions*. New York: Wiley.

Malak, R. J., Aughenbaugh, J. M. and Paredis, C. J. J. (2008). Multi-Attribute Utility Analysis in Set-Based Conceptual Design. *Computer Aided Design*, In Press.

Malak, R. J. and Paredis, C. J. J. (2007). Validating Behavioral Models for Reuse. *Research in Engineering Design*, 18(3), 111-28.

Marler, R. T. and Arora, J. S. (2004). Survey of Multi-Objective Optimization Methods for Engineering. *Structural and Multidisciplinary Optimization*, 26(6), 369-95.

Martin, J. D. and Simpson, T. W. (2006). A Methodology to Manage System-Level Uncertainty During Conceptual Design. *Journal of Mechanical Design*, 128(4), 959-68.

Mattson, C. A. and Messac, A. (2003). Concept selection using s-Pareto frontiers. *AIAA Journal*, 41(6), 1190-8.

--- (2005). Pareto Frontier Based Concept Selection Under Uncertainty, with Visualization. *Optimization and Engineering*, 6(1), 85-115.

McCandlish, D. and Dorey, R. E. (1984). The Mathematical Modeling of Hydrostatic Pumps and Motors. *Proceedings of the Institution of Mechanical Engineers, Part B: Management and Engineering Manufacture*, 198(10), 165-74.

McKay, M. D., Beckman, R. J. and Conover, W. J. (1979). A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 22(2), 239-45.

Messac, A. and Mattson, C. A. (2004). "Normal Constraint Method with Guarantee of Even Representation of Complete Pareto Frontier." 45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference, Palm Springs, California, 19-22 April 2004.

Michelena, N. F., Park, H. and Papalambros, P. Y. (2003). Convergence of Analytical Target Cascading. *AIAA Journal*, 41(5), 897-905.

Mistree, F., Hughes, O. F. and Bras, B. (1993a). "Compromise Decision Support Problem and the Adaptive Linear Programming Algorithm." In M. P. Kamat (Ed. *Structural Optimization: Status and Promise* (pp. 251-90). Washington, D.C.: AIAA.

Mistree, F., Smith, W. F. and Bras, B. (1993b). "A Decision-Based Approach to Concurrent Engineering." In H. R. Paresai and W. Sullivan (Eds.), *Handbook of Concurrent Engineering* (pp. 127-58). New York: Chapman & Hall.

Mistree, F., Smith, W. F., Bras, B., Allen, J. and Muster, D. (1990). Decision-based Design: A Contemporary Paradigm for Ship Design. *Transactions, Society of Naval Architects and Marine Engineers*, 98, 565-97.

Mosler, K. C. (1984). Stochastic Dominance Decision Rules when the Attributes are Utility Independent. *Management Science*, 30(11), 1311-23.

Mullur, A. A., Mattson, C. A. and Messac, A. (2003). "Pitfalls of the Typical Construction of Decision Matrices for Concept Selection." 41st Aerospace Sciences Meeting and Exhibit, Reno, NV. Paper No. AIAA 2003-0466.

Muster, D. and Mistree, F. (1988). The Decision Support Problem Technique in Engineering Design. *International Journal of Applied Engineering Education*, 4(1), 23-33.

Nahm, Y.-E. and Ishikawa, H. (2004). Integrated Product and Process Modeling for Collaborative Design Environment. *Concurrent Engineering: Research and Applications*, 12(1), 5-23.

Norton, R. L. (2000). *Machine Design: An Integrated Approach* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

O'Brien, G. and Scarsini, M. (1991). Multivariate Stochastic Dominance and Moments. *Mathematics of Operations Research*, 16(2), 382-9.

Oberkampf, W. L. and Trucano, T. G. (2002). Verification and Validation in Computational Fluid Dynamics. *Progress in Aerospace Sciences*, 38, 209-72.

Otto, K. N. and Antonsson, E. K. (1991). Trade-off Strategies in Engineering Design. *Research in Engineering Design*, 3(2), 87-104.

Pacheco, J. E., Amon, C. H. and Finger, S. (2001). "Developing Baysian Surrogates for Use in Preliminary Design." AMSE Design Theory and Methodology Conference, Pittsburgh, PA. American Society of Mechanical Engineers, Paper No. DETC2001/DTM-21701.

Pahl, G. and Beitz, W. (1996). *Engineering Design: A Systematic Approach* (2^{nd} ed.) (K. Wallace, L. Blessing and F. Baurt, Trans.). London: Springer-Verlag.

Paredis, C. J. J. (2008). "An Open-Source Modelica Library of Fluid Power Models." Bath/ASME Symposium on Fluid Power and Motion Control, Bath, UK, September 10-12.

Pareto, V. (1971). *Manual of Political Economy* (A. S. Schwier, Trans.). New York: Macmillan. (Original work published in 1906).

Parunak, H. V. D., Ward, A. C., Fleischer, M. and Sauter, J. (1997). "A Marketplace of Design Agents for Distributed Concurrent Set-based Design." ISPE/CE97: Fourth ISPE International Conference on Concurrent Engineering: Research and Applications, Troy, MI, Aug 20-22. 287-93.

Prasad, B. (1996). *Concurrent engineering fundamentals : integrated product and process organization*. Upper Saddle River, NJ: Prentice Hall PTR.

Pugh, S. (1991). *Total Design: Integrated Methods for Successful Product Engineering*. Reading, MA: Addison-Wesley.

Rekuc, S., Aughenbaugh, J. M., Bruns, M. and Paredis, C. J. J. (2007). Eliminating Design Alternatives based on Imprecise Information. *Society of Automotive Engineering Transactions*.

Royce, W. W. (1970). "Managing the Development of Large Systems: Concepts and Techniques." 9th International Conference on Software Engineering. ACM, 328-38.

Russell, W. R. and Seo, T. K. (1978). Ordering Uncertain Prospects: The Multivariate Utility Functions Case. *The Review of Economic Studies*, 45(3), 605-10.

Rygielski, C., Wang, J.-C. and Yen, D. (2002). Data Mining Techniques for Customer Relationship Management. *Technology in Society*, 24, 483-502.

Saari, D. G. (2000). Mathematical Structure of Voting Paradoxes: 1. Pairwise Votes. *Economic Theory*, 15(1), 1-53.

Saaty, T. L. (1990). How to Make a Decision: The Analytical Hierarchy Process. *European Journal of Operations Research*, 48(1), 9-26.

Sage, A. P. and Armstrong Jr., J. E. (2000). *Introduction to Systems Engineering*. Wiley and Sons.

Sargent, R. G. (2001). "Some Approaches and Paradigms for Verifying and Validating Simulation Models." Winter Simulation Conference (B. A. Peters, J. S. Smith, D. J. Medeiros and M. W. Rohrer, Eds.). Piscataway, NJ: Institute of Electrical and Electronics Engineers, 106-14.

Sauer-Sundstrand Co. (1997). "Selection of Driveline Components." Sauer-Sundstrand Company, Ames, IA.

Scarsini, M. (1988). Dominance Conditions for Multivariate Utility Functions. *Management Science*, 34(4), 454-60.

Scholkopf, B. and Smola, J. A. (2002). *Learning with Kernels*. MIT Press.

Seo, K.-K., Park, J.-H., Jang, D.-S. and Wallace, D. (2002). Approximate Estimation of the Product Life Cycle Cost using Artifical Neural Networks in Conceptual Design. *International Journal of Advanced Manufacturing Technology*, 16(6), 461-71.

Shabani, M. R. and Yekta, R. B. (2006). Chemical Process Equipment Cost Estimation using Parametric Models. *Cost Engineering*, 48(5), 26-32.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Shupe, J. A. (1988). *Decision-Based Design: Taxonomy and Implementation*, University of Houston.

Shupe, J. A., Mistree, F. and Sobieszczanski-Sobieski, J. (1987). Compromise: An Effective Approach for the Hierarchical Design of Structural Systems. *Computers & Structures*, 26(6), 1027-37.

Simpson, T. W., Mauery, T. M., Korte, J. J. and Mistree, F. (2001a). Kriging Models for Global Approximation in Simulation-Based Multidisciplinary Design Optimization. *AIAA Journal*, 39(12), 2233-41.

Simpson, T. W., Peplinski, J. D., Koch, P. N. and Allen, J. K. (2001b). Metamodels for Computer-based Engineering Design: Survey and Recommendations. *Engineering with Computers*, 17(2), 129-50.

Sobek II, D. K. (1996). "A Set-based Model of Design." *Mechanical Engineering*, 118(7), 78-81.

Sobek II, D. K. and Jain, V. K. (2007). Relating Design Process to Quality: A Virtual Design of Experiments Approach. *Journal of Mechanical Design*, 129(5), 483-90.

Sobek II, 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.

Sobieszczanski-Sobieski, J. and Haftka, R. T. (1997). Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments. *Structural and Multidisciplinary Optimization*, 14(1), 1-23.

Steward, D. V. (1981). The design structure system- A method for managing the design of complex systems. *IEEE Transactions on Engineering Management*, 28, 71-4.

Stump, G., Yukish, M., Simpson, T. W. and O'Hara, J. J. (2004). "Trade Space Exploration of Satellite Datasets using Design by Shopping." IEEE Aerospace Conference, 6-13 March 2004. 3885-95.

Tarassenko, L., Hayton, P., Cerneaz, N. and Brady, M. (1995). "Novelty Detection for the Identification of Masses in Mammograms." 4th International Conference on Artificial Neural Networks, Cambridge, UK. 442-7.

Tax, D. M. J. and Duin, R. P. W. (1999a). "Data Domain Description using Support Vectors." European Symposium on Artificial Neural Networks, Bruges, Belgium, 21-23 April. 251-6.

--- (1999b). Support Vector Domain Description. *Pattern Recognition Letters*, 20, 1191-9.

Thurston, D. L. (1991). A Formal Method for Subjective Design Evaluation with Multiple Attributes. *Research in Engineering Design*, 3(2), 105-22.

--- (2001). Real and Misconcieved Limitations to Decision Based Design with Utility Analysis. *Journal of Mechanical Design*, 123, 176-82.

Tobin, J. (1958). Liquidity Preferences as Behavior Toward Risk. *The Review of Economic Studies*, 25, 65-86.

U.S. EPA (2008) Environmental Protection Agency Urban Dynamometer Driving Schedule http://www.epa.gov/nvfel/testing/dynamometer.htm (accessed 31 October 2008).

Ulrich, K. T. (2005). Estimating the Technology Frontier for Personal Electric Vehicles. *Transportation Research Part C*, 13, 448-62.

United States Department of Defense. (2003). *DoD Modeling and Simulation (M&S) Verification, Validation and Accreditation (VV&A)*. DoD Instruction Number 5000.61. By US DoD, 13 May. Washington, D.C.: GPO. Accessed 24 Feb 2005 at <http://www.dtic.mil/whs/directives/corres/pdf/i500061_051303/i500061p.pdf>.

Vadde, S., Allen, J. K. and Mistree, F. (1994). Compromise Decision Support Problems for Hierarchical Design Involving Uncertainty. *Computers & Structures*, 52(4), 645-58.

Van de Ven, J., Olson, M. W. and Li, P. Y. (2008). "Development of a Hydro-Mechanical Hydraulic Hybrid Drive Train with Independent Wheel Torque Control for an Urban Passenger Vehicle." Proceedings of the International Fluid Power Exposition, Las Vegas, NV, March 11-15.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer.

von Neumann, J. and Morgenstern, O. (1980). *Theory of Games and Economic Behavior* (3rd ed.). Princeton, NJ: Princeton University Press. (Original work published in 1944).

Wang, G. G. and Shan, S. (2007). Review of Metamodeling Techniques in Support of Engineering Design Optimization. *Journal of Mechanical Design*, 129(4), 370-80.

Ward, A. C. (1989). *A Theory of Quantitative Inference Applied to a Mechanical Design Compiler*. Ph.D. thesis, MIT.

Ward, A. C., Liker, J. K., Cristiano, J. J. and Sobek II, D. K. (1995). The Second Toyota Paradox: How Delaying Decisions Can Make Better Cars Faster. *Sloan Management Review*, 36(3), 43-61.

Wertz, J. R. and Larson, W. J. (1999). "The Space Mission Analysis and Design Process." In J. R. Wertz and W. J. Larson (Eds.), *Space Mission Analysis and Design* (3rd ed.). Torrance, CA: Microcosm Press.

Wilson, B., Cappelleri, D., Simpson, T. W. and Frecker, M. (2001). Efficient Pareto Fronier Exploration using Surrogate Approximations. *Optimization and Engineering*, 2(1), 31-50.

Winer, E. H. and Bloebaum, C. L. (2002). Development of Visual Design Steering as an Aid in Large-Scale Multidisciplinary Design Optimization: Part I: Method Development. *Structural and Multidisciplinary Optimization*, 23(6), 412-24.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Academic Press.

Wood, W. H. and Agogino, A. M. (2005). Decision-based Conceptual Design: Modeling and Navigation of Heterogeneous Design Spaces. *Journal of Mechanical Design*, 127(1), 2-11.

Wood, W. H. and Dong, H. (2006). "Generating Design Alternatives across Abstraction Levels." In K. E. Lewis, W. Chen and L. C. Schmidt (Eds.), *Decision Making in Engineering Design* (pp. 61-72). New York: ASME Press.

Xu, R. and Wunsch II, D. (2005). Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks*, 16(3), 645-78.

Yang, J., Estivill-Castro, V. and Chalup, S. K. (2002). "Support Vector Clustering Through Proximity Graph Modeling." Ninth International Conference on Neural Information Processing. 898-903.

Yu, P. L. (1974). Cone Convexity, Cone Extreme Points, and Nondominated Solutions in Decision Problems with Multiobjectives. *Journal of Optimization Theory and Applications*, 14(3), 319-77.

Yukish, M., Stump, G. and Lego, S. (2007). "Visual Steering and Trade Space Exploration." 2007 IEEE Aerospace Conference, Big Sky, MT. 1-9.