MODULARITY ANALYSIS AND COMMONALITY OPTIMIZATION FOR THE TOP-DOWN PLATFORM-BASED PRODUCT FAMILY DESIGN

LIU ZHUO

(*B.Eng & M.Eng, Xian Jiaotong University*)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF MECHACNIAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2008

ACKNOWLEDGEMENTS

Firstly, I would like to express my most sincere gratitude to my advisors, Professor Wong Yoke San and Associate Professor Lee Kim Seng, for motivating and guiding me academically and professionally in this project over the past years. The breadth and depth of their knowledge, timely feedback, constructive insight, and constant encouragement have helped me to complete my research and will continue to benefit my future work.

I would like to thank Dr. Lu Cong, Dr. Maria Low Leng Hwa, Mr. Li Min, Mr. Fan Li Qing, and Miss Zhu Ya Da for their help and advice during my Ph.D research. And I would also like to thank Mr. Zhu Shi Yan for his kind assistance in the case study.

In addition, I would like to thank Associate Professor Zhang Yun Feng and Associate Professor Loh Han Tong, for their invaluable comments and suggestions on my research during my Ph.D qualification exam.

I would also like to thank National University of Singapore for offering me research scholarship. The first-class research facilities, abundant professional resource, and beautiful campus environment will leave me a strong impression for ever.

Finally, I would like to devote this thesis to my family and girlfriend for their self-giving love and constant support.

Table of Contents

Summary

With highly fragmented market and increased competition, platform-based product family design has been recognized as an effective method to construct a product line that satisfies diverse customer's demands while aiming to keep design and production cost- and time- effective. Recognizing the essentiality of modularity and commonality in the platform development, this thesis presents a systematic framework to implement top-down platform and product family development, which aims to achieve modularity for variety management at system-level design stage and rationalize commonality configuration for module instantiation at detailed design stage

Rather than just identifying module boundary and interface in the product architecture, the development of product family architecture (PFA) in this research incorporates customized requirements and constructs a flexible and robust product architecture to accommodate variations. Towards this, the implication of PFA can be viewed as a conceptual structure with three interrelated elements: module, variant, and coupling interface. Variants in term of different customer requirements act as the external drivers of architectural variation and meanwhile variation is propagated within the product architecture through module interaction. Based on this principle, a step-by-step method is proposed to systematically modularize the PFA, involving functional modularization and variety analysis. The generated product portfolio architecture provides an engineering insight to manage variety in terms of functional module configuration and also prepares the targets for further design.

To achieve economy of scales by increasing commonality during module instantiation, a scalable platform design method is adopted at the detailed design stage. Its success often relies on properly resolving the inherent tradeoff between commonality across the family and performance loss compared to individually tailored design. In this research, we propose a multi-platform product family (MPPF) approach to accomplish such balance. In the light of the basic premise that increased commonality enhances manufacturing efficiency, we present an effective platform decision strategy to quantify family design configuration using a commonality index. The proposed strategy takes into account the basic platforming elements and expected sharing degree by coupling design varieties with production variation. Meanwhile, unlike many existing methods that assume a single given platform configuration, the proposed method addresses the multi-platforming configuration across the family, and can generate alternative product family solutions with different levels of commonality. A modified genetic algorithm is developed to solve the aggregated multi-objective optimization using an efficient and dynamic weighted aggregation method.

In the case studies, a family of power tool design is used to demonstrate the proposed method at system-level and detailed design stages.

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Background

Today's turbulent market compels enterprises to redefine the paradigm of doing business. Customers express their preferences not only for product quality, but also for variety and personalization. Therefore, identification and fulfillment of customer's individual needs become imperatives to maintain competitiveness by integrating customer requirements in the value creation. Meanwhile, increasingly competitive intensity - arising in particular from unceasing technical renovation, globalization and convergence of industries – rapidly shortens the product life cycle from launch to disposal, and thus compels companies to reduce delivery time to market and expand product variety (Anderson, 1997).

In response to this customer-driven market, most manufacturers take advantage of the strategy of *mass customization* or *mass personalization* to increase customer satisfaction with a high variety of offerings. Contrary to the traditional one-at-a-time design, mass customization aims to deliver a variety of products and services simultaneously for various market niches without sacrificing efficiency, effectiveness and low costs. With effective planning and management of product development, mass customization enables manufacturers to quickly respond to market fluctuation and grasp latent opportunities. In addition, the emergence of e-business also relies on and facilitates the successful implementation of mass customization to maximize the customers' satisfaction through expansion of their product lines.

1.2 Platform-based Product Family Design

Currently, a popular strategy to effectively deliver a stream of products is to design multiple products as a *product family*, within which components, processes and technologies are effectively shared among the family members via a *product platform*. Then the individual product can be derived from the platform in an effectively planned manner to meet various requirements, which may come from space context as *spatial variety* and time context as *generational variety* (Martin and Ishii, 2002). Spatial variety refers to the variety that the company offers the market at a point in time, in terms of various combinations of features or cost segmentations. The generational variety involves the evolutional changes of a product family over time. Both spatial and generational varieties are very important and always synchronously implemented for product development

Clusters of examples from different industries have been reported that take advantage of platform-based product family development to cater for spatial and generational requirements, as shown with examples illustrated in Figure 1.1. Sony has used three platforms to successfully create hundreds of different portable stereo models in its Walkman line since 1980's (Sanderson and Uzumeri, 1997). This variety-intensive development pattern helps Sony to dominate worldwide market for more than a decade despite fierce competitions from other contenders. Black & Decker, the world's largest producer of power tools, built its product line around motor platform to meet different applications (Meyer and Lehnerd, 1997). Kodak is reported to win the market share of single-use cameras back from Fuji by effectively planning platform development (Robertson and Ulrich, 1998). Hewlett Packard successfully develops a series of printers and gains platform benefits by postponing the point of differentiation (Feitzinger and Lee, 1997). In the automotive industry, Volkswagen has shared its platform across several brands, such as Audi, Seat, Skoda, as well as Volkswagen (Simpson, 2004b&2006). These successful examples prove the feasibility and superiority of platform-based strategies to ensure companies' competitiveness by creating a consecutive line of product offerings. While adopting platform thinking in the product development, these companies present different platform definitions and strategies in their context due to the spectrum covered in the platform planning and development, as well as the nature of targeted products and marketplace (Halman *et al.*, 2003).

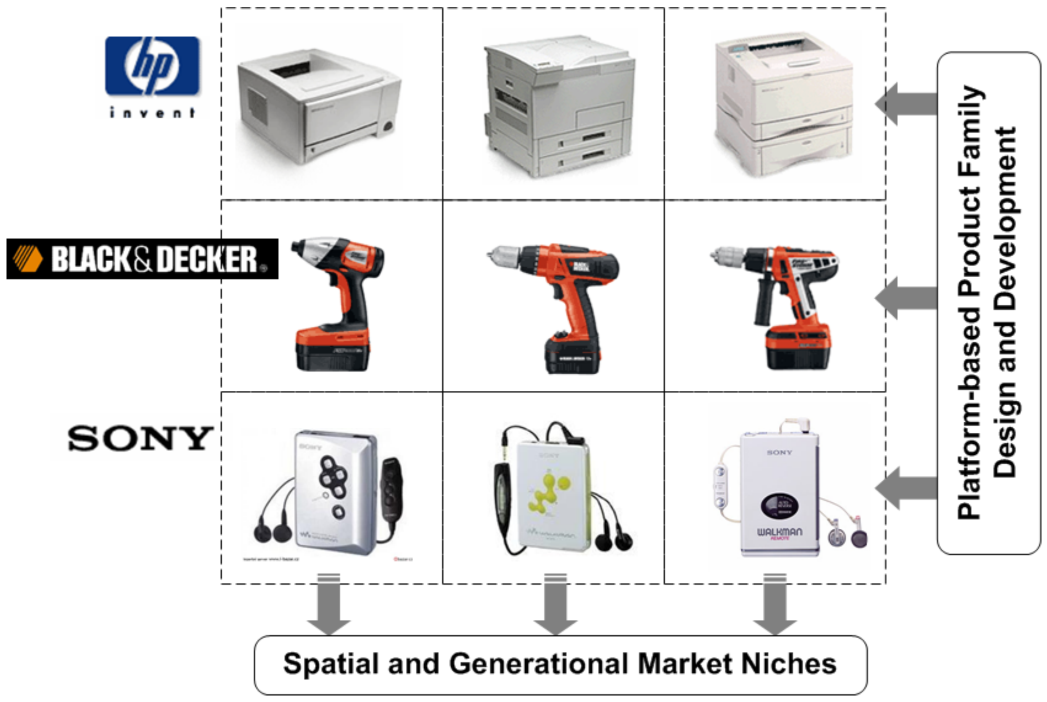

hp invent
BLACK&DECKER
SONY
Platform-based Product Family Design and Development
Spatial and Generational Market Niches

Figure 1.1: Industrial examples of platform-based product families

1.2.1 Product Architecture

To efficiently customize products for individual customers and help understand the complexity of product design at the conceptual design stage, the definition of product architecture is brought forward to decompose the complex system into subsystems or chunks. Ulrich (1995) defines product architecture as a scheme by which the function of a product is allocated to physical components. Modularity is referred to as the most important characteristics of product architecture and accordingly there are two types of architecture: modular and integral. A modular product architecture is one-to-one or many-to-one mapping relationship between functional elements and physical structure, and can easily create product variants by combinations of functional blocks, such as personal computers; otherwise, integral architecture is characterized by a complex or coupled mapping of functional elements to physical structures and it can acquire advantages of performance due to elimination of interfaces and integration of multi-functions into fewer parts (Gonzalez-Zugasti, 2000). While integral architectures aim to increase product performance and reduce cost, modular architectures are driven by variety, product change, and standardization (Cutherell, 1996).

Modularity or modular design enables firms to achieve many strategic advantages and has become a major focus for product realization (Baldwin and Clark, 2000; Jose and Tollenaere, 2005; Jiao *et al.*, 2007d). In terms of functional modularity, companies can easily create the variety of product offerings by changing the arrangement and adding new functional modules (Ulrich and Eppinger, 2000). Meanwhile, modular design provides a flexible and loosely coupled product structure and thus allows for reuse of the existing design with minor changes and reduced efforts for product upgrade (Sand *et al.*, 2002). Additionally, modularity can help designers to decompose the overall design into smaller tasks and achieve parallel product development to shorten time-to-market (Gershenson *et al.*, 2003).

1.2.2 Platform Strategies

Although various approaches to product family design are developed by many companies or researches to deliver a series of variants targeted to different market niches, there is still a strategic difference among them depending on whether or not the companies take proactive steps to mange the platform development and variety generation (Simpson *et al.*, 2001a). One is called the *top-down* approach wherein a company strategically develops a family of products based on a carefully tailed product platform, as illustrated by platform *B* in Figure 1.2 (Simpson, 2004b; Alizon *et al.*, 2007). Some industrial companies (e.g. *Sony, Kodak*) are reported to introduce a derivative based series on a product platform by carefully planning and managing the platform design (Sanderson and Uzumeri, 1997; Robertson and Ulrich, 1998). Another approach is the *bottom-up* approach or reactive redesign, wherein a company redesigns or consolidates a group of distinct products to improve economies of scale by standardizing the components, as illustrated by platform *A* in Figure 1.2 (Simpson, 2004b; Alizon *et al.*, 2007). For instance, Black & Decker is reported to benefit from component standardization by redesigning universal motor (Meyer and Lehnerd, 1997). Whether it is *top-down* or *bottom-up* approach, platform-based product development provides a lot of benefits, including reduced development complexity and cost, reduced production cost, improved response to market, and reduced risk for new product development (Meyer and Lehnerd, 1997; Simpson, 2004b&2006;).

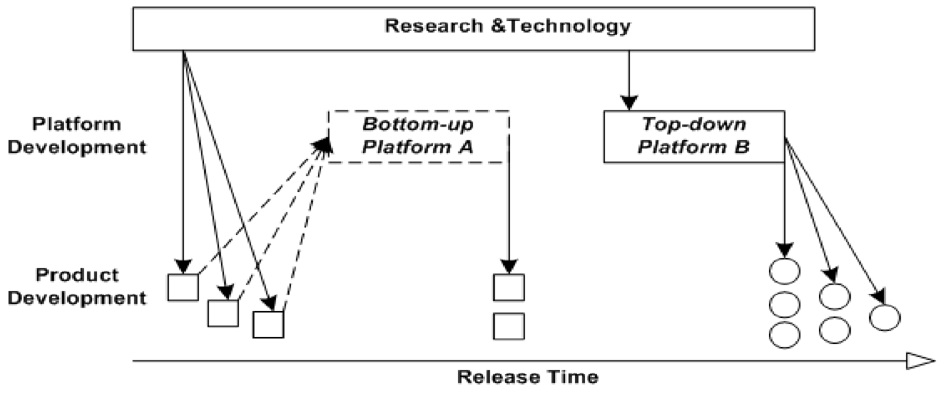


Figure 1.2: Illustrations of bottom-up platform A and top-down platform B

1.2.3 Modular and Scalable Product Platform

Depending on the hierarchical level in the product architecture, there are two different types of platform: modular and scalable platform. The former platform is through the development of modular product architecture and product family members are instantiated by adding, substituting, and/or removing one or more functional modules from the platform (Simpson, 2004b). For example, *Sony* builds all of its Walkmans around key modules and platforms by using the principle of modular design to deliver more than 250 models (Sanderson and Uzumeri, 1997).

The scalable platform is to "stretch" or "shrink" the platform in one or more dimensions to satisfy a variety of market niches (Simpson *et al.*, 2001a). Unlike module-based product platforms, scale-based platform focuses on the commonality issue at the lower level of product structure and provides an effective means to satisfy a variety of performance requirements by scaling one or more variables. For example, Simpson *et al.* (2001a) develop a family of electrical motors based on scaling optimization along various dimensions to produce a range of power outputs for diverse applications.

1.3 Research Objectives

Recognizing the essentiality of *modularity* and *commonality* in platform-based product development, this research aims to develop a *top-down* methodology for proactive product family design to aid in product differentiation for various market requirements and thereby facilitate the effective implementation of mass customization. More specifically, the necessary tasks in this study are identified as follows.

- The first task is to achieve *modularity* at the system-level design stage for variety generation and management. By extending the extent of the traditional product architecture, the product family architecture is approached as a conceptual structure with three important interrelated elements: module, variant and coupling interface. An integrated modularization approach is developed to translate the variety of requirements into a dynamic configuration of the conceptual product family architecture, involving variety analysis, functional modularization, and generation of product portfolio architecture.

- The second task is to tackle the *commonality* issue as a multi-objective optimization problem based on an effective platform decision. To enhance commonality at detailed module instantiation stage while maintaining certain economical efficiency, a manufacturing-biased platform decision strategy for scalable product family design is presented to coordinate design variety with production variation so that the family members can be derived in expected economical manner.
- The third task is to develop an effective optimizer to solve the inherent trade-off between performance and commonality. A modified genetic algorithm is developed to explore the alternative solutions with varying level of commonality based a dynamic weighted aggregation method.

The results of this study as a whole would serve as a guide tool to approach the platform-based product family design. The proposed methodology does not intend to replace the existing development process but assist in handling multi-product development while exploiting opportunities to achieve economy of scales with effective planning and optimization.

1.4 Organization of this thesis

The thesis is organized as follows.

Chapter 2 reviews the research work related to platform-based family design, as well as the gaps current approaches reported in the literature, and motivation for this research.

Chapter 3 presents the framework for platform-based family design in this thesis, which is viewed as a *top-down* development paradigm to achieve *modularity* at the system-level design stage and *commonality* at the detailed design stage.

Chapter 4 focuses on system-level modularization of product family architectures for variety generation based on functional modeling, and also develops a quantitative method to analyze the variety effect of customization on modules. A case study of power tool family design is used to demonstrate the proposed method.

Chapter 5 introduces a manufacturing-biased platform decision for detailed module instantiation and commonality optimization at the detailed design stage. The proposed platform strategy attempts to quantify family design configuration using a commonality index that couples design varieties with production variation. Then the measured commonality is incorporated into the family design model

Chapter 6 presents the development of a modified genetic algorithm for optimizing multi-objective product family design using dynamic weighted aggregation method.

Chapter 7 demonstrates the proposed approaches to scalable product family design and optimization on a case study of designing a family of transmission module.

Chapter 8 gives the conclusions, contributions and recommendations.

Chapter 2 Literature Review

2.1 Overview

An increasingly large but diverse body of research on platform-based product family design has been made over the last decade to address various aspects of product fulfillment, involving marketing, design, manufacturing, management and so on. The variety of methodologies stems from not only the particular aspects of family design addressed, but also the inherent nature of their case studies and assumptions made. Thus it is very difficult to capture the rationale behind the seemingly isolated issues without a conceptual structure and overall logical organization. Fortunately, the adoption of multi-domain views along the entire spectrum of product realization (Suh, 1990&2001) enables platform-based family design to be tackled from several coherent perspectives, namely customer, functional, physical, and process domains as shown in Figure 2.1 (Jiao *et al.*, 2007d). Although the platform-based approaches proposed in the literature share the same principle of *commonalization*, the platform in each domain exhibits different implication within the context.

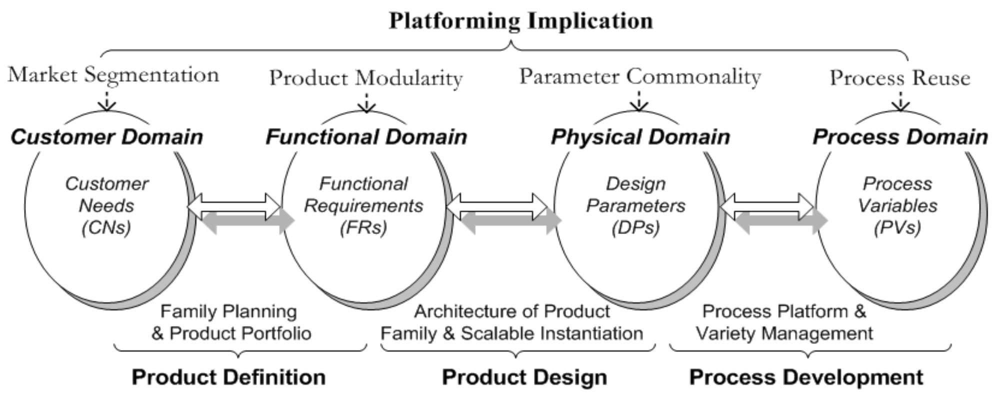


Figure 2.1: An overview of platform-based product family design

The customer domains can be described with a set of diverse customer needs (CNs), which represent different functions and performance characteristics towards the target product. Accordingly, the task is to plan the right product variety to the right market segment and then trigger the downstream stage of product design in a cascading manner (Jiao *et al.*, 2007d). In the functional domain, the CNs are first translated into functional requirements (FRs) in terms of available engineering technologies. Then a conceptual architecture for product family can be developed to assist in the variety generation and management. Subsequently, the detailed family solutions are generated in the physical domain by mapping FRs to design parameters (DPs) based on the effective platform basis. This stage not only involves decisions regarding family design and optimization to minimize the loss of performance or distinctiveness due to the platforming effect, but also maintain the manufacturing efficiency by coupling design varieties with product variation. At the back-end, the mapping from DPs to process variables (PVs) generate production planning to construct a standard process platform or infrastructure, around which variant processes can be derived to realize the production of the product family (Jiao and Tseng, 2004).

As a whole, the implementation of mass customization begins with the front-end customer domain and then spreads to the latter design stage in terms of various functional/physical entities, and then to the production stage in terms of re-allocation of processes and resources. To maintain the whole value chain in a cost- and time-effective manner, various platform-based approaches in each domain are developed to capture and utilize commonality for variety generation.

2.2 Product Family Architecture

2.2.1 Product Architecture and Modularity

The development of a product architecture, assigning forms to functional elements, is a critical phase at the conceptual design stage because the choice generated will strongly influence the product performance in several aspects, including later detailed design, manufacturability, product variety, and so on. Ulrich and Eppinger (1995) define a product architecture as consisting of three elements: (1) the arrangement of functional elements (2) the mapping relation between functions and physical elements, and (3) the specification of the interfaces among interacting physical components.

Most research in this field focuses on identification and representation of modular architecture using decomposition and clustering techniques. Pimmler and Eppinger (1994) decompose the product into elements and then cluster them into chunks by considering the generic interaction types: spatial, energy, information and material. Kusiak and Huang (1996) develop the modular product with the consideration of performance and cost, and later they develop a decomposition approach to solve modularity problem based a matrix representation (Huang and Kusiak, 1998). Gu and Sosale (1999) identify product modules from various life cycle engineering perspectives such as assembly, maintenance and recycling. Van Wie *et al.* (2001) address architectural issues from interface perspective and aims to reduce assembly cost by investigating component interactions. Later, he and co-authors (2003) present an architecture representation to link functional design and embodiment design.

Functional modeling or diagram in terms of available engineering technologies provides another effective means to modularize product architecture at the conceptual design stage. Stone *et al.* (2000b) combine functional model and a heuristic method to assist in identifying modules. Later, they propose a quantitative functional model to develop architecture with consideration of customer need ratings (Stone *et al.*, 2000c). Dahmus *et al.*, (2001) also adopt functional modeling method to modular product architecture for multi-product design. By incorporating functional structure, Holtta *et al.* (2005) present a method to measure redesign effort based on analysis of functional flows: material, energy, and information.

Additionally, modularity has been well studied from many perspectives (Fixson, 2003; Gershenson *et al.*, 2003&2004). Mikkola and Oliver (2003) introduces a mathematical modularization function to assess the degree of modularity in a given product architecture. Kusiak (2002) investigates the integration aspects of modularity of products, processes and resources. Sosa *et al.*, (2000) analyze the difference in the way modular and integrative design teams handle interface using design structure matrix (DSM).

2.2.2 Architecture for Product Family

The emergence of product family design to meet customized requirements imposes new challenges to define product architecture. As Fujita and Yoshida (2004) point out, the most important difference between the architecture of a product family and that of a single product is the simultaneous handling of multiple products. Thus, the concept and implication of product architecture have to be extended to manage the complexity of product family. Du *et al.* (2001) view a product family architecture (PFA) as the logical organization of a product family with a generic product structure. Then tailored product variants can be generated with several generic mechanism (e.g. module swapping, scaling). By capturing the functionally common and unique structures, Dahmus *et al.* (2001) develop a conceptual method to architecture the product family.

PFA has been studied from different perspectives along the product life cycle. Erens and Verhulst (1997) assert that the development of a product family requires the definition of product architecture in three domains: function, technological realization, and physical realization. The multi-view of PFA development is also supported by Jiao and Tseng (1999&2000), who present a method to rationalize product family development for mass customization from three aspects of functional, technical and physical views. Additionally, Du *et al.* (2001) investigate some fundamental issues regarding the architecture of a product family from both sales and engineering perspective. Muffatto and Roveda (2002) also study the multiple aspects of product architecture including functions, requirements, technological solutions, product concepts, product strategies and platforms. Serving multiple managerial purposes, Fixson (2005) investigates the multi-dimensional architecture issues, involving product development, process and supply chain design.

As a whole, the operation of modularity analysis at different development stage is the strategic result of a search for potential common technical solutions. The earlier modularization process provides more freedom to define architectural content, and allocates function-component mapping relationship. Function-based module definitions can explore conceptual product architecture and gain an early insight into common and unique functionality (Stone *et al.*, 2000a&2000b; Dahmus *et al*, 2001). Such functional modularization relaxes the constraint of the pre-definition of sub-module level components and offers a fundamental approach for proactive platform development. Assuming the basic physical element as fixed, physical modularization generates the modular product architecture by re-arranging these elements into larger units (modules), and is always adopted for product or platform redesign (Martin and Ishii 2002; Hsiao and Liu 2005). Parametric modularity considers the product structure as essentially fixed and product characteristics are varied only within boundaries of the individual elements or parameters. This kind of approach provides the least freedom to change product structure and only pursues certain commonality at detailed module/assembly design stage (Simpson *et al.*, 2001a).

Based on the previous review, the architectures for product and product family have been well studied from the perspectives of definition, representation, vocabulary, multi-view synchronization and so on. However, in a dynamic market environment with uncertainty, the modularization of product family architecture not only requires qualitative identification of module boundary and standardization of coupling interface, but also needs quantitative analysis to estimate the customization effect on product architecture and translate the external variety of requirements into a dynamic configuration. Unfortunately, few studies have been done so far with respect to this direction.

2.3 Platform-based Product Family Design

2.3.1 Platform Implications

The definitions of platform have been diverse due to the specific perspective and purpose (Halman *et al.*, 2003). Jiao *et al.* (2007d) divide them into two classes: namely physical platform and abstract platform. The former platform refers to a collection of common elements including features, parts, modules, subsystem (Meyer and Lehnerd, 1997). Then a stream of derivative products can be developed by operating (e.g. swapping, scaling, adding) elements. This kind of platform definition is easily understood and the range of products can be described with physical entities. The abstract one is broadly defined as the collection of functions, components, processes, knowledge, people, and even relationships that are shared by a set of products (Robertson and Ulrich, 1998). Accordingly, the major issues may not be limited to the scope of product definition and design, and can be extended to the front-end of marketing (Jiao *et al.*, 2005) and the back-end of process platform (Jiao *et al.*, 2007c) and supply chain (Fixson, 2005; Lamothe *et al.*, 2006;).

To assist in platform planning and development, the market segmentation grid is always used to represent the principal customer groups served by the offering products. Accordingly, Meyer and Lehnerd (1997) define three different platform leveraging strategies within the grid shown in Figure 2.2: horizontal leveraging, vertical leveraging, and the beachhead approach, which combines both. Although horizontal leveraging strategies always take advantage of modular platforms, scale-based platform design can be used for vertical leveraging strategies (Simpson *et al.*, 2001a).

Another interesting pattern observed from industrial and academic examples shows that most large corporations (e.g. Volkswagen, Boeing, Kodak, Sony, and HP) have started platform development in a systemic and planned manner, usually with effective multi-discipline coordination in platform thinking, involving marketing, design, production and even supply chain. On the other hand, small/medium enterprises (SME) run short of technical workforce and financial support so that the platform development can only be leveraged through reactive re-engineering to reduce unwanted internal varieties (Simpson, 2004b). Hence, despite the advantages in strategic and tactical terms, the content of top-down platform approach, particularly from an engineering design view, has not been completely understood and utilized by SME practitioners.

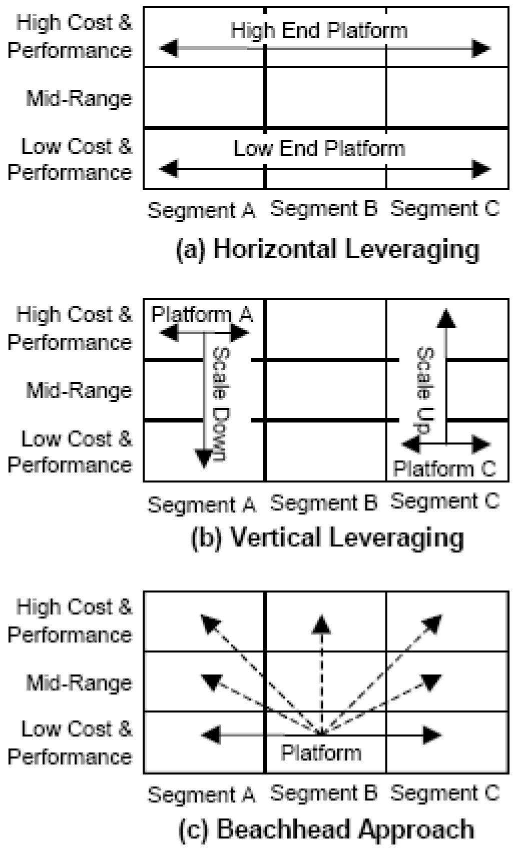


Figure 2.2: Three platform leveraging strategies (Meyer and Lehnerd, 1997)

2.3.2 Types of Platform Design

Corresponding to the scalable and modular product platforms, there are two types of approaches to platform-based product family design. One is referred to as configuration-based product family design. This higher-level method aims to develop modular product architecture and then construct a combinatorial design space. The individual product can be generated by adding, substituting, and/or removing one or more functional modules (Ulrich and Eppinger, 2000; Du *et al.*, 2001; Simpson, 2004b). So, it is also called module-based product family design and can achieve certain economic efficiency to produce custom-built product from standard models.

In academic community, some researches tackle module-based product family design by establishing mathematical models to capture the module/component combination and also optimize objectives of interest. Chakravarty *et al.* (2001) optimize module variation to achieve profit maximization. Given sets of module instances, Yigit and Allahverdi (2003) formulate modular design as an integer optimization problem and try to find a trade-off between quality loss and reconfiguration cost. Rai and Allada (2003) also tackle modular product family design as a multi-objective optimization problem and use agent-based techniques to determine Pareto-design solutions. Kreng and Lee (2004) develop QFD-based design method to model a linear optimization problem by capturing the modular drivers. Moon *et al.* (2007) adopt a dynamic multi-agent system to determine platform level selection. Jiao *et al.* (2007b) use a genetic algorithm based method to design a family of products while maximizing the customer-perceived benefit per-cost. In addition, module/component selection for a product family in a supply chain is also investigated as an integer-programming model (Gupta and Krishnan, 1999; Da Cunha *et al.*, 2007).

While modular elements are assumed *priori to* optimization of module-based family design, identification of modular product architecture is reported in several papers to discuss the mechanism to generate product family from functional or physical perspectives. Chandrasekaran *et al.* (2004) propose a template-based design method for product family generation based on patterns of functional flow. De Lit *et al.* (2003) develop a method to integrate the product family design and assembly system design using functional entities of product. With data mining and fuzzy clustering techniques, Moon *et al.* (2006) propose a method to cluster functional features into modules for product family. While functional modularization can help designers to proactively plan platform development at the conceptual design stage, physical component-based methods provide an effective means to redesign existing product structure for multiple product design. Martin and Ishii (2002) develop an index based method to develop a decoupled and standardized architecture for future generation of products. Similarly, Hsiao and Liu (2005) investigate the component interaction and redesign a product physical structure for variety generation.

The other lower-level one is called scalable or parametric product family design, which utilizes the principle of stretching or shrinking the product platform in one or more dimensions to meet diverse performance requirements (Simpson *et al.*, 2001a). Unlike module-based product platforms, scale-based platform focuses on the lower level of product architecture and provides an effective means to satisfy a variety of performance requirements by scaling one or more variables. Accordingly, balancing the trade-off between commonality and individual performance deviation is the core issue for scalable product family design. The detailed review of research on scalable platform design will be given in section 2.4.

Module- and scale-based platform designs always address only one aspect of product development because of simple assumption. However, some variety-oriented product development always requires the flexible mix of modular and scalable platform design. Accordingly, this type of product family entails greater actual complexity with its dynamics and uncertainty (Maier and Fadel, 2007). Fujita (2002) classifies the product variety design into three categories: attribute assignment, module combination and simultaneous design of both. Later, he and Yoshida (2004) optimize the simultaneous design of module combination and module attributes in multi-stage. Hernandez *et al.* (2003) develop product platform constructal theory method (PPCTM) and adopt two modes of dimensional customization and modular combination to deliver a series of differentiated product. Later, Williams *et al.* (2007) augment PPCTM for non-uniform market demand and extend its application to the domain of process parameter design. Li *et al.* (2007) develop a genetic algorithm based method to design adaptive platform involving structural and parametric optimization.

2.3.3 Commonality Metrics for Product Family Design

Commonality refers to the similarity extent of product characteristics from a particular point of view, such as requirements, design features, and even physical structures. The commonality measure of a generic BOM (bill-of-material) structure allows post-assessment of product family efficiency and also provides feedback information to redesign family members (Jiao and Tseng, 2000a; Kota *et al.*, 2000; Blecker and Abdelkafi, 2007). Most developed commonality indices include component-level information, such as the number of common components, the component cost, manufacturing process, and so on. Thevenot and Simpson (2006) have made a detailed comparison among several commonality indices existing in the literature as to consistency, repeatability, sensitivity, and then proposed a framework to redesign a product family using such indices. However, a component-level commonality measure overlooks the quality/performance aspect in the evaluation, and can not fully reflect the inherent trade-off existing in a product family design.

2.4 Scalable Platform and Product Family Design

First proposed by Simpson *et al.* (2001a), scale-based product family design aims to synchronously design multiple products to maximize commonality across the whole family while minimizing impact on their individual performance. Accordingly, the challenge is to resolve the inherent tradeoff or balance between monetary and technical aspects: increasing commonality in the family and minimizing performance loss compared to individual design. Most existing approaches meet the challenge as a multi-conflicting-criteria problem and utilize multi-objective optimization techniques to solve the problem from the perspective of meeting performance variation (Nelson *et al.*, 2001; Simpson, 2004b). To simplify the optimization model, most approaches assume that maximizing commonality in terms of shared variable settings among products minimizes production cost. Although fulfilling the diverse functional requirements through a variety of design parameters is the major concern in design, it is the production stage that actually determines the final product costs, process complexity, and lead time (Jiao *et al.*, 2007c). Therefore, without explicitly investigating the associated manufacturing cost or coordination with production stage, the simple assumption may lead to sub-optimal family solutions (Simpson, 2004b). Recent research trend in family design is towards a more systematic process as shown in Figure 2.3, involving effective platform decision with coordination of the back-end production stage, multi-platforming configuration with varying level of commonality, and integration with the front-end marking research.

A number of product examples have been used as case studies to demonstrate the proposed approaches, including consumer products or components (such as transmission module for drills, universal electrical motor, and automobiles), industrial products (such as absorption chillers, flow control vales), conceptual products (such as cantilever beams, pressure vessels and nail guns) and complex systems (such as aircraft and spacecraft). Most examples involved in the case studies are described with analytical equations or computation simulations to capture the relationship between the input and output variables. When explicit equations are not given, design of experiments (DOE) is used to develop a response surface and then derive these equations as an approximation to the relationship between variables (Hernandez *et al.*, 2001; Jiang and Allada, 2005).

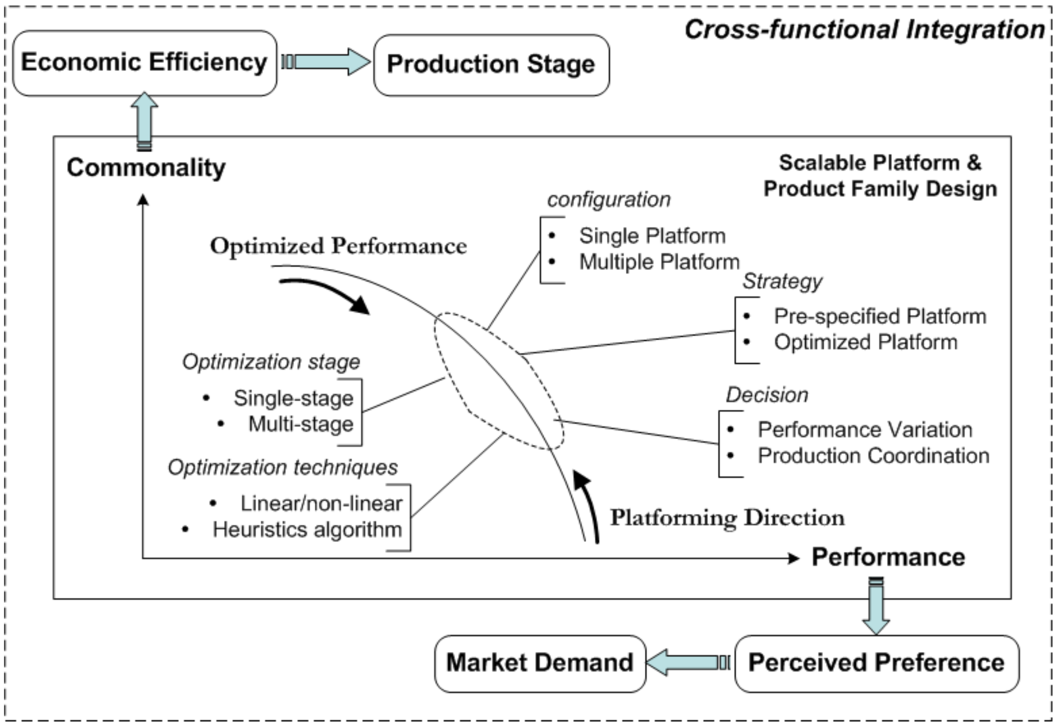


Figure 2.3: Research scope of scalable product family design

2.4.1 Platform Configuration and Decision

Platform decision in family design includes two different strategies to select appropriate shared elements of the platform: pre-specified platform and optimized platform configuration (Simpson, 2004b; Simpson *et al.*, 2007d). The former requires the specification of the elements (variables or components) to be shared *a priori* to the optimization, and aims to reduce the computational efforts and make the family design more tractable. Accordingly, this kind of approaches always involves only a single platform configuration, which makes the platform elements shared across the entire family and non-platform elements instantiate the individual products. However, it may lead to the local compromise of performance because the unique platform setting may not be ideal for every product in the family. Some low-end products may be over-designed or certain high-end products may be under-designed (Dai and Scott, 2007).

Subsequently, a separate optimization stage using robust design principles is employed to determine the platform settings, such that shared variables have the smallest impact on performance variation (Messac *et al.*, 2002a&2002b; Nayak *et al.*, 2002). The recent trend is to consider dynamic platform configuration or multiple platforms during optimization. Simpson and D'Souza (2004a) consider varying levels of platform commonality within the product family by setting a set of "switch" codes to control the commonality of the corresponding design variable. However, these approaches cannot remove the disadvantage of the single platform settings, in which variables are either shared across the entire family or not at all. Fellini *et al.* (2005&2006) attempt to explore partial component sharing between any two variants in the family using a heuristics algorithm. Dai and Scott (2007) develop sensitivity and cluster based method to construct multiple platforms, in which some design variables can be shared by any subset of variants within the family. Although these strategies pose many computational challenges, it enhances exploration of the design space and may yield better solutions.

Additionally, whether pre-specified or not, most current approaches decide platform settings primarily from the aspects of the design problems. They seek to fix those variables which have not made much contribution to the performance variation and thus may not result in much performance loss when consolidated (Messac *et al.*, 2002a&2002b; Nayak *et al.*, 2002; Fellini *et al.*, 2004&2005&2006). Unfortunately, this method overlooks a fact that commonality among these design variables cannot always generate great benefits from other product lifecycle activities. Accordingly, the results of product family hardly reduce the process complexity or manufacturing cost without linking to the back-end of product realization during the family design (Simpson, 2004b). Simpson *et al.* (2001a) discuss this issue in their case study of the electrical motor and explore possible benefit from the commonality settings from an engineering standpoint. Although their proposed optimization approach revealed that the motor platform should be scaled around the radius, the best choice in the practical situation was stack length from the perspective of production cost. Dai and Scott (2003) also propose a meaningful method to consider monetary and technical aspects of commonality in the platform decision. Therefore, there is a clear need to incorporate the impact of product platforms on the production stage into the model of product family design to derive an economical platform setting.

Although sharing of variable values is assumed to derive some benefits, another inevitable problem, but still unsolved, is that some design variables are coupled to jointly determine the dimensions of a sub-assembly or component (Scott *et al.*, 2006). It means that under specific manufacturing condition, there is no expected benefit to be generated from variable sharing unless we synchronously share all variables related to the component. This complicated or coupled design situation poses more challenges on the family design and requires an effective strategy in platform decision. Unfortunately, few studies have been done so far on the coupled design case for product family design.

2.4.2 Optimization Stages and Techniques

The optimization procedure for the family design problem can be classified into one-stage and multi-stage (Simpson, 2004b). One-stage approaches seek to optimize the platform settings and the corresponding members of family simultaneously, while multi-stage approaches optimize the platform first, and then instantiate the individual products during the second stage. Although the two approaches are about equally common in the literature, the choice often depends on the size of the product family design. Both platform settings and non-platform design variables are often solved in one stage when the number of derived products and design variables is relatively small (Simpson, 2001a; Messac *et al.*, 2002a&2002b; Simpson and D'Souza, 2004a; Fujita and Yoshida, 2004; Kumar and Allada, 2007). These methods yield the best overall performance of product family, but require huge computational expense. When the size of product family or the number of design variables increases, the dimensionality of the optimization problems can become so high that for the one-stage method it become difficult to deal with the complexity and computational expense. As a result, multi-stage approaches can provide an effective means to divide the task into two stages: platform configuration to decide which variables are shared and their settings, and instantiation to generate the optimal values for non-platform variables for all product variants (Nayak *et al.*, 2002; Dai and Scott, 2006&2007; Fellini *et al.*, 2004&2005&2006; Hernandez *et al.*, 2003).

Simpson (2004b) has given a detailed review on optimization algorithms used for family design. Some derivative-free methods, including genetic algorithms, simulated annealing, pattern search, and branch-and-bound techniques, are employed in many studies, in addition to linear and non-linear programming algorithms. The choice of optimization techniques depends on the size of the design space. When the design space is relatively small, exhaustive search techniques are used to generate all possible combinations. However, many researchers advocate the use of genetic algorithms (GA) for product family design due to the combinatorial nature of design problems and its high efficiency for one-stage optimization in exploring the design space (Simpson and D'Souza, 2004; Fujita and Yoshida, 2004; Li and Azarm, 2002; D'Souza and Simpson, 2003; Jiao *et al.*, 2007a&2007b; Li *et al.*, 2007; Huang *et al.*, 2007; Khajavirad *et al.*, 2007).

Due to the nature of multi-objective optimization, various GA based approaches are developed to deal with objective conflict, mainly including commonly-used weighted aggregation, goal programming, and non-dominated based methods. The classical weighted aggregation based approaches, which are conceptually easy to understand, provide an advantage of computational efficiency. However, they can obtain only one solution from one run and also have unsatisfactory performance when dealing with optimization problems with a concave Pareto front (Jin *et al.*, 2001). Goal programming technique is similar to the method of objective weighting except that it requires a goal vector for each objective prior to aggregation. The most profound drawback of the two kinds of approaches is their sensitivity to settings of weights or goals and the prerequisite of understanding the design problem comprehensively *a priori to* optimization (Srinivas and Deb, 1994). Towards this, non-dominated sorting approaches are adopted in a few researches to fully search solutions along the Pareto front (D'Souza and Simpson, 2003&2004; Akundi *et al.*, 2005). Compared to the conventional methods involving single overall objective function, non-dominated approaches handle multiple objectives synchronously and provide decision-makers an opportunity to explore a number of Pareto-optimal solutions from one run of optimization without pre-specifying any priority for objectives. But this kind of methods involves exhaustive non-dominated sorting among all the objectives throughout the population and thus imposes extremely high computation expense during optimization, especially for family design with larger design space or size of family.

2.5 Summary

The purpose of this chapter is to describe the related research background in the field of platform-based product family design and conduct a meaningful review of existing methodologies. Meanwhile, some research gaps or drawback existing in the current literature are identified and discussed.

From the above review, modularity and commonality are two essential issues for platform-based product development and play different roles in different context of product family design. For a bottom-up or assembly-to-order family design approach, combination or clustering of variants from a given collection of module instances becomes the main means to deliver a family of products and is always accomplished by optimizing objectives of interests, such as profit, cost, sales, or even customer preferences in terms of expected utilities. Instead of being design goals to be achieved, modularity and commonality always serve as pre-conditions or constraints in the model.

Otherwise, a top-down or proactive platform development approach requires definition of product architecture in terms of modularity first (if the end product is directly targeted for market) and then enhance the commonality across the family at detailed design stage of module instantiation. Unfortunately, these two topics are seldom captured together as a logically correlative manner to handle variety-oriented product development. Meanwhile, few studies have been done so far to help clarify the entire content of proactive platform development, which involves carefully planned management of modularity in response to external variety of requirements, and effective decision of commonality with coordination of product realization within an entire framework.

Chapter 3 A Framework for the Proactive Platform-based Product Family Design

3.1 Introduction

From the aforementioned literature review, it can be seen that a bottom-up platform approach is characterized by the enhancement of common elements among a group of distinct products without fundamentally redefining the product architecture; whereas, a top-down one is driven by the combinatorial platform of planned modular architecture and optimal physical configuration. Therefore, the two issues of modularity and commonality may co-exist for a proactive platform development, and are logically inseparable along product creation process.

As a whole, modularity and commonality are two essential dimensions to characterize varieties among family members and their correlation can be embodied in a class-member manner (Jiao *et al.*, 2000b). As shown in Figure 3.1, a product architecture is defined in terms of its modularity, through which module boundaries are specified according to technological feasibility of the design solutions. For each type of module (class), variety of design can further result from diverse instances (members) in response to variety of external requirements. As a result, derived product variants may share the same module boundaries but entail different instances of every module. In other words, a family of products is described by modularity, whereas product variants differentiate according to the commonality among module instances. The less commonality among module instances, the more differentiation among product variants. Furthermore, viewpoint-specific (e.g.

functional, physical, life-cycle) modularity operation on product architecture develops a structural design space, wherein the design tasks are broken into module/assemble-level instantiations with a less complicated commonality design space.

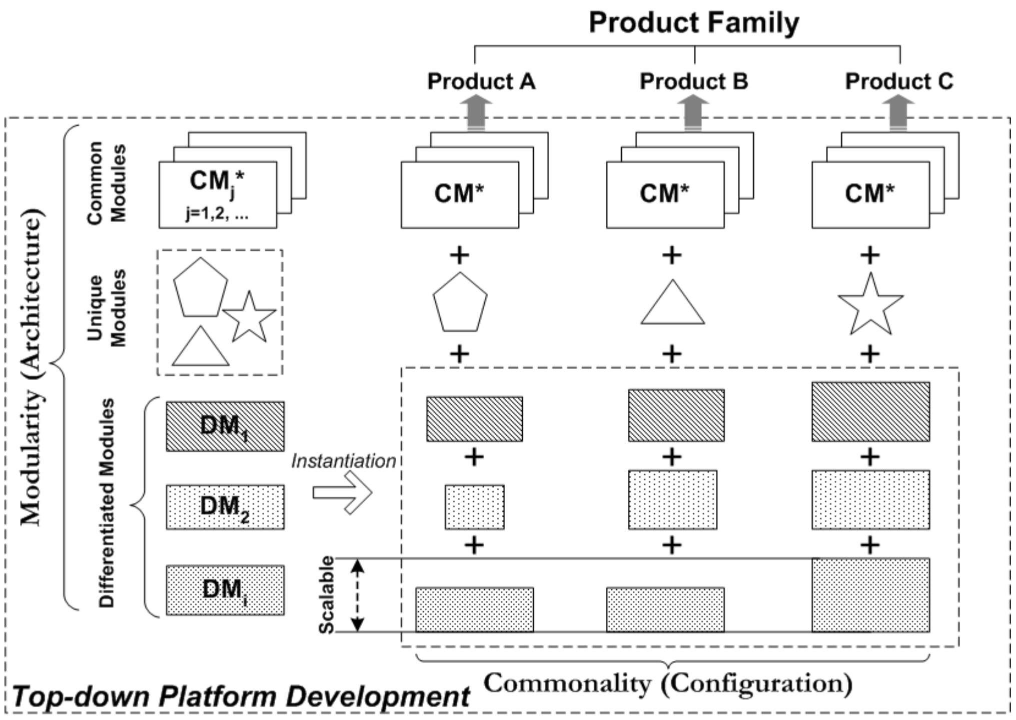


Figure 3.1: Modularity and commonality for platform development

Modularity and commonality design work under the same cost-effective platforming principle to meet mass customization, as illustrated in Figure 3.2. That is, elements that incur additional complexities or expense, but contribute less value in customer view, are to be stabilized and consolidated as a platform base, while elements that may offer more customer-perceived value ought to be customized with more emphasis and resources.

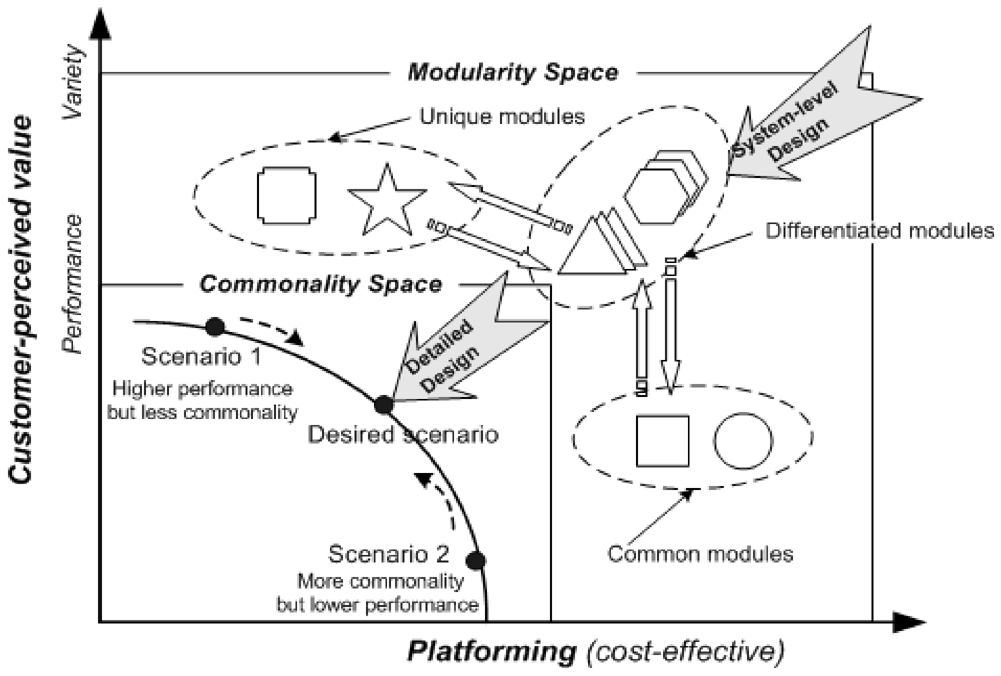


Figure 3.2: Platforming principles for modularity and commonality

With standardized interfaces, a suitably conceived modular architecture provides a platform basis for variety generation. As shown in Figure 3.2, in the form of common (functional and parametric) features, some elements address fewer customer-perceived varieties and can be shared as a common base. Otherwise, to respond to more external variety of requirements, some modules need to be instantiated in the dimensions of engineering specifications for different family members. These differentiated modules are basic elements making one product different from another. Sometimes, unique functional modules may exist to provide special customer-perceived distinctiveness. Meanwhile, one-time development cost acts as another important constraint to impact on the platforming decision. Modules with higher ratios of customer-perceived values (e.g. distinctiveness, variety) to initial investment may increase market coverage more efficiently and gain more customized designs (Zacharias and Yassine, 2008); otherwise, higher development costs incurred from some modules may depreciate the value in the employment of complete design differentiation on them although they address certain variety.

The same platforming direction can be applied to the detailed module instantiation, wherein reduction of recurring engineering cost (e.g. manufacturing expense) will be pursued by increasing commonality across the family configuration. A specific module type characterized by modularity involves less design complexities, and can be described by an explicit analytical or simulation-based model to reflect engineering relationship between controllable variables and performance responses (another term of customer-perceived value). Based on a scaling platform, these instances can be clustered to achieve the reuse of some elements (variables or parts) and gain certain economical efficiency from common settings while minimizing impact on their individual performance. Therefore, the core challenge is to resolve the inherent trade-off or balance between the desired commonality across the family, and allowable performance loss compared to certain benchmark design.

Based on this principle, the whole procedure for the top-down platform development can be divided into two levels: namely system-level design for modularity, and detailed design for commonality. The following section presents general steps involved in the top-down platform development process.

3.2 A Framework for Top-down Product Family Design

In this research, we view the top-down platform development as two different tasks, namely modularity at system-level design stage and commonality at detail design stage as shown in Figure 3.3. System-level design, which links with the front-end planning phase, aims to define conceptual product architecture by decomposing the complex product into sub-systems or chunks according to available technological solutions. In the context of multiple-product design, the external variation existing in the market should be taken into account to achieve flexible and robust *modularity*. While modularity resembles decomposition of product structures and can provide a platform basis for variety generation at the system level, commonality at the design stage of module instantiation embodies the difference among product variants. Therefore, pursuing *commonality* based on a platform decision is the main concern of detail design.

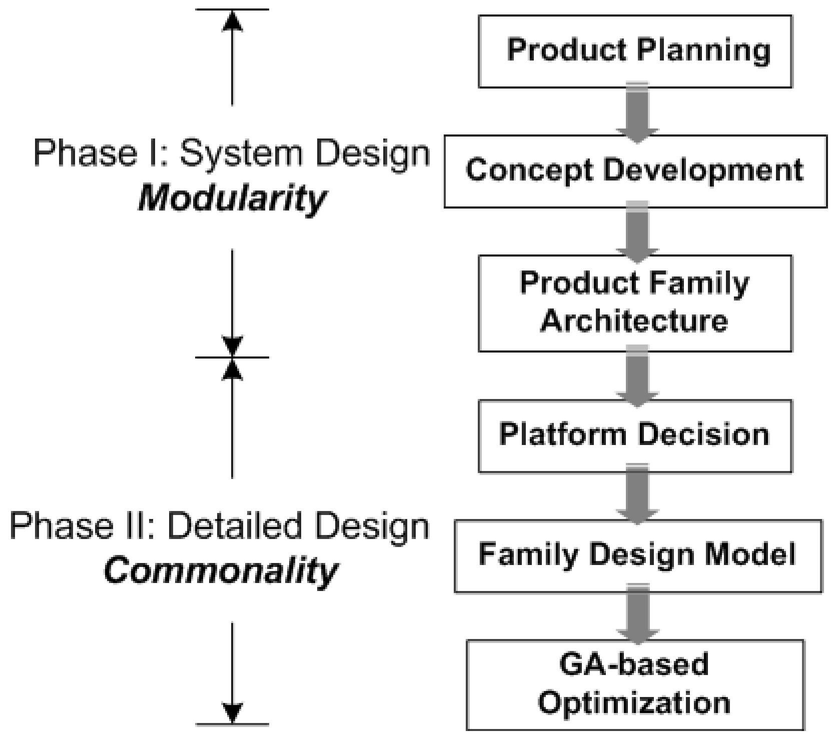


Figure 3.3: A proposed framework for product family design

3.2.1 System-level design: Modularization of PFA

This higher-level design centers on modularization of product architecture for variety generation. Rather than just identifying module boundary and interface in the product architecture, the development of product family architecture (PFA) in this research incorporates customized requirements and constructs a flexible and robust product architecture to accommodate variations. Towards this, the implication of PFA can be viewed as a conceptual structure with the following three interrelated elements: module, variant, and coupling interface. Variants in term of different customer requirements act as the external drivers of architectural variation and meanwhile variation is propagated within the product architecture through module interaction.

Step 1 in this phase is about product planning to identify the portfolio of products. A range of products are collected and described in terms of product attributes and their corresponding level. Several product family planning and strategy evaluation is available in the literature and results of market analysis are assumed to exist beforehand.

Step 2 is conceptual modularization and variety analysis, which involves the identification of conceptual modules based a functional modeling method and generates the variety index for each module using the derived attributed-module matrix (AMM). AMM is characterized by the engineering relation between product attributes in the customer domain and the conceptual module in the functional domain.

Step 3 is to finalize the product family architecture in term of common modules and differentiated modules. Meanwhile, engineering specifications are allocated to module instances to form an engineering view of product portfolio architecture and provide further goals for detailed design stage.

3.2.2 Detailed Design: Commonality Optimization of Scalable Product Family Design

The detailed design phase aims to address the commonality issue using a scaling platform method and achieve the manufacturing efficiency through an effective platform decision strategy.

Step 1 is about platform decision to decide the right elements to be shared during the family design. In order to access varying levels of commonality, this research adopts a mechanism of quantifying the level of commonality as commonality index. The developed commonality index (CI) is a measure of sharing degree regarding the design parameters throughout the entire product family. By coupling design varieties with production variation, the derived CI function for the whole family can be viewed as an efficiency indicator of reduced manufacturing complexity and cost savings.

Step 2 is to formulate the optimization model for family design. The scale-based family design involves two conflicting aspects: performance responses and commonality index. Based on preference aggregation method, the multi-objective optimization is aggregated into a single overall function by incorporating the quantified level of commonality. By varying weights for commonality objective, the proposed method can access alternative product family solutions with different level of commonality.

Step 3 is to develop a GA-based optimizer to solve the product family design and optimization problem. GA-based optimization method provides an effective means to explore the mixed-discrete non-linear problem behind the family design. By adopting the evolutionary dynamic weighted aggregation method, the modified optimizer can explore solutions along the Pareto front while maintaining the computational expense at the economical level.

3.3 Problem Boundary

Firstly, a generic product development process may include feasible study, conceptual design, detail design, process design and so on. This research mainly focuses on platform development from conceptual design to detail design. Although the proposed method involves co-ordination or link with front-end market research and back-end production stage, we assume the required information exists and is available.

Secondly, the proposed method principally aims to assist engineering designers in the handling multi-product design by exploiting the potential platform opportunities, and does not replace the existing design rules and flows. Meanwhile, the application or implementation of the proposed method requires some prerequisite, such as available analytical or computational model to predict the engineering relationship between variables and responses, and technological solutions to realize specific functional goals.

Finally, although the proposed framework involves decomposition of the product structures into several chunks or modules for further detailed design, this study does not provide decision support to decide whether a specific module and its instances should be designed and produced inside the firms or be outsourced to other organizations.

Chapter 4 Modularization of Conceptual PFA

Successful platform development always begins with the system-level design of product architecture (Ulrich, 1995), including decomposition of the product into modules and classification of module types. A functional diagram provides designers opportunities to modularize product structures at the early design stage due to the functional nature of modularity. In this chapter, we present an integrated method to modularize the product architecture for variety generation by incorporating the external variety of requirements.

4.1 Introduction

The success of mass customization lies in the manufacturer's ability to cater for the potential market niches by providing suitably customized varieties based on a rationally technical framework in an effective and timely manner. Since most relevant decisions about the cost and schedule of components or parts are made in the design phase, it is believed that mass customization can be approached from the perspective of design, particularly the early stage of architecture design (Erens and Verhulst, 1997; Jiao and Tseng, 1999&2000).

Here we look at the product family architecture (PFA) as a conceptual structure consisting of three elements: modules, coupling interface, and variants. Figure 4.1 illustrates the interrelation among the three elements of PFA. The traditional product architecture only consists of module and interface because product development is always stably driven by producers' marketplace. However, the paradigm of mass customization, which is incurred by buyers' market, impose a necessity on manufacturers to suit individual customer needs and thus poses new requirements on the form of product architecture (Anderson, 1997). Being the external factors in terms of scattered customer requirements, variants result in the spatial and generational varieties, and act as a new source of product development complexity. Because of variants some module boundaries have to be redefined or reconfigured to form new modules corresponding to the variant attributes of products. Some modules may become a common platform basis to support the whole product family, and some modules need to be differentiated in specific dimensions for variety generation. Meanwhile, the higher risk and complexity involved in product family design require such information as design efforts for architectural variation to make an early evaluation and operational decision. So, the modularization process for a family of products includes not only the identification of module boundary, but also the classification of modules according to the variety of requirements, as well as quantitative analysis of customization effects on the product architecture.

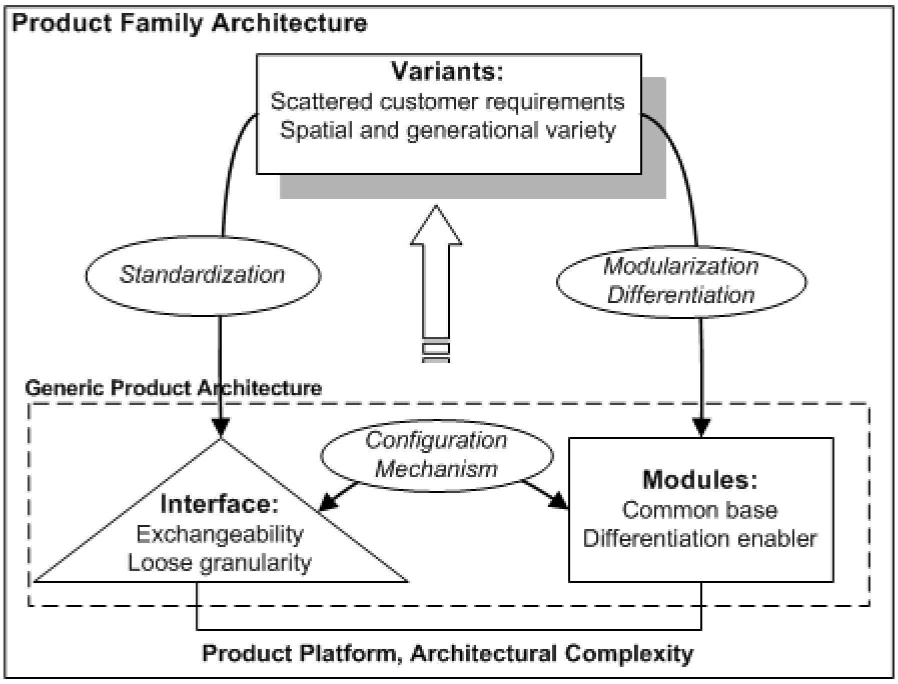

Product Family Architecture
Variants:
Scattered customer requirements
Spatial and generational variety
Standardization
Modularization
Differentiation
Generic Product Architecture
Configuration
Mechanism
Interface:
Exchangeability
Loose granularity
Modules:
Common base
Differentiation enabler
Product Platform, Architectural Complexity

Figure 4.1: Three elements of Product Family Architecture

At the same time, variants impose certain technical standardization of the coupling interface to achieve exchangeability among modules (Chen and Liu, 2005), and loosely coupled granularity on architectural elements so that the resultant product architecture can accommodate customized requirements without too many changes of the modules and their interface. Therefore, interface definition and strategy is another important issue in modularizing product architecture, especially for a family of products (Chen and Liu, 2005), and its complexity directly affects the final cost (Van Wie *et al.*, 2001). As a result, the realization process for variety-oriented design moves toward the module/component configuration mechanism based on the common platform basis and differentiation enabler (Du *et al.*, 2001), as shown in figure 4.1.

4.2 Variety Analysis

In Axiomatic Design introduced by Suh (2001), product design can be viewed from different domains: customer, functional, physical, and process domain. Each domain is characterized by the needs or attributes which provide solution for the preceding domain while giving new requirements for the next domain. To conceptualize the solution, we need the mapping process between the domains and also can mathematically model this mapping process in terms of the characteristic vectors that define the design goals and design solution (Suh, 2001). Similarly, this method can be applied in the modularization of PFA to capture the architectural variation due to the customized variants.

At the conceptual design stage, product design can be considered as the mapping between the functional domain and customer domain, and written as follows:

$$\{M\} = \{C\}[AMM] \tag{4.1}$$

where C=[C_1, C_2 … C_n] is the vector of product attribute requirements in the customer domain and may be characterized with a finite number of engineering metrics (e.g. voltage, power, weight), and M=[M_1, M_2 … M_p] is the conceptual module vector and can be described as functional entities based on the feasible engineering technologies. For example, the functionality of power module is to provide electricity and can be physically realized by a battery module. *AMM* is design matrix, namely the attribute-module matrix that characterizes the relation between product attribute requirements and the conceptual module. If one product attribute or metric is implemented or affected by one or more modules, there will be an engineering relationship between this attribute and its corresponding modules. The attribute-module matrix has following form.

$$AMM = \begin{bmatrix} a_{11} & \dots & a_{1i} & \dots & a_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ a_{j1} & \dots & a_{ji} & \dots & a_{jp} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & \dots & a_{ni} & \dots & a_{np} \end{bmatrix} \qquad (4.2)$$

$$i = 1,\dots,p \quad j = 1,\dots n$$

Usually the products catering for heterogeneous market niches will be launched at different levels along the dimensions of product attributes to result in various product offerings. When this variety of requirements {ΔC} in customer view needs corresponding realization, the variation will spread to the functional domain and generate the variant module configuration {ΔM}. This customization process can be written as

$$\{\Delta M\} = \{\Delta C\}[AMM] \qquad (4.3)$$

Here we develop a variation mapping method, namely *Variety Index* (VI), to investigate how variants affect the conceptual modules. Similar to quality function deployment (QFD) method in transferring customer requirements into design characteristics, VI transfers variance from customer requirements into architectural elements **M**= [M_1, M_2... M_p] in the function domain, as shown in Figure 4.2. Thus, VI can be viewed as an indicator of design variations or efforts on conceptual modules to meet customer-perceived variation.

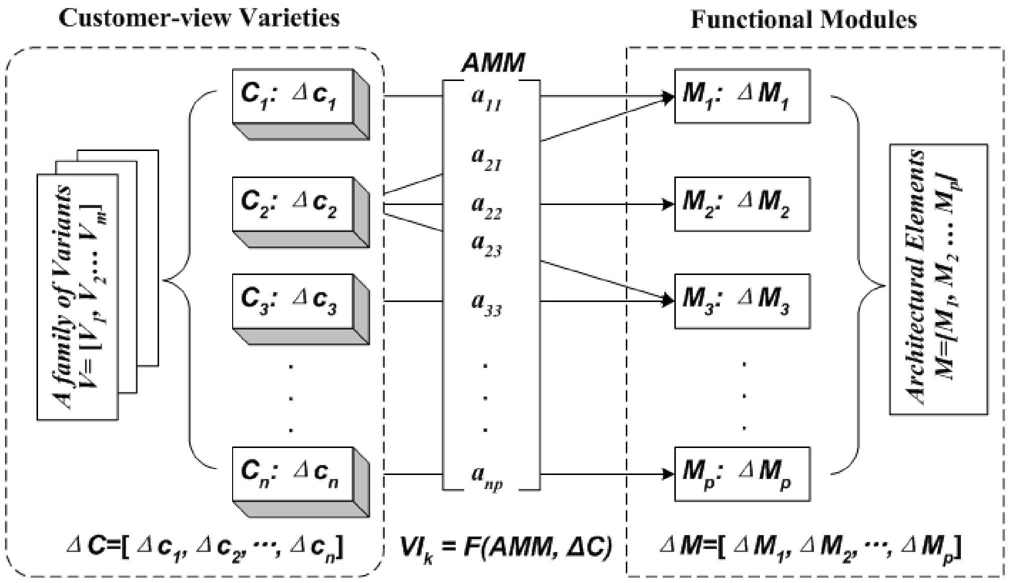


Figure 4.2: Illustration of Variety Index

The range of the product family is a collection of product variants **V**= [V_1, V_2... V_m] with customized value or level for each product attribute **C**= [C_1, C_2... C_n]. VI can be represented as *VI:* $\Delta C \rightarrow \Delta M$, where ΔC= [Δc_1, Δc_2...Δc_n] is variance of attribute and ΔM= [ΔM_1, ΔM_2...ΔM_p] is variance of module. To balance the attribute in customer choice, preferences for product attributes are normalized and given through assigning weights by market analysis. For module *k*, VI (ΔM) can be mathematically represented as follows

$$VI_k = \sum a_{jk} \Delta c_j w_j \tag{4.4}$$

where w_j is the weight for attribute j to normalize the customer preference.

For all modules, VI can be represented as the following matrix form.

$$\begin{Bmatrix} \Delta M_1 \\ \Delta M_2 \\ \ldots \\ \Delta M_p \end{Bmatrix} = \{\Delta C_1 w_1 \quad \Delta C_2 w_2 \quad \ldots \quad \Delta C_n w_n\} \begin{bmatrix} a_{11} & \ldots & a_{1i} & \ldots & a_{1p} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{j1} & \ldots & a_{ji} & \ldots & a_{jp} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{n1} & \ldots & a_{ni} & \ldots & a_{np} \end{bmatrix} \tag{4.5}$$

$i = 1, \ldots, p \quad j = 1, \ldots n$

VI provides a simple and straightforward tool to analyze the variation of the conceptual product architecture due to the customized requirements. Although depending on the specific design context, the derived variety index, together with other constraints of initial investment and feasibility of over-design, can assist designers in identifying crucial modules and further determining platforming direction in the product architecture. Smaller VI means less variety value, from the customer viewpoint, delivered by the corresponding modules, which may be over-designed and settled down as a platform base of standard components; larger VI denotes greater customer-perceived value in terms of varieties and requires more design efforts to differentiate them. However it is still desirable to impose suitable over-design on module instantiation to reduce the number of instances and thus developing cost

Meanwhile, uncoupled design, in which AMM is diagonal and each attribute can be satisfied independently by means of one module, maintains the independence of the functional requirements and is more suitable for mass customization than decoupled design with interlaced relationship between modules and attributes (Dan and Tseng, 2007).

4.3 Integrated Method to modularize Conceptual PFA

To better understand the variety effect on product architecture and support platform-based product family development, the modularization procedure of the conceptual PFA is studied here to achieve system-level modularity. In our proposed method, there are primarily three steps to modularize the conceptual PFA, as illustrated in Figure 4.3. Step 1 is about product family planning. Step 2 utilizes a functional modeling method to identify functional modules and generate the variety index (VI) for each module based on the estimated Attribute-Module matrix. Step 3 generates engineering specification for each module instance, and then integrates them into a product portfolio architecture (PPA). The three steps are described in detail in the following sections that aim to guide designers through the modularization process.

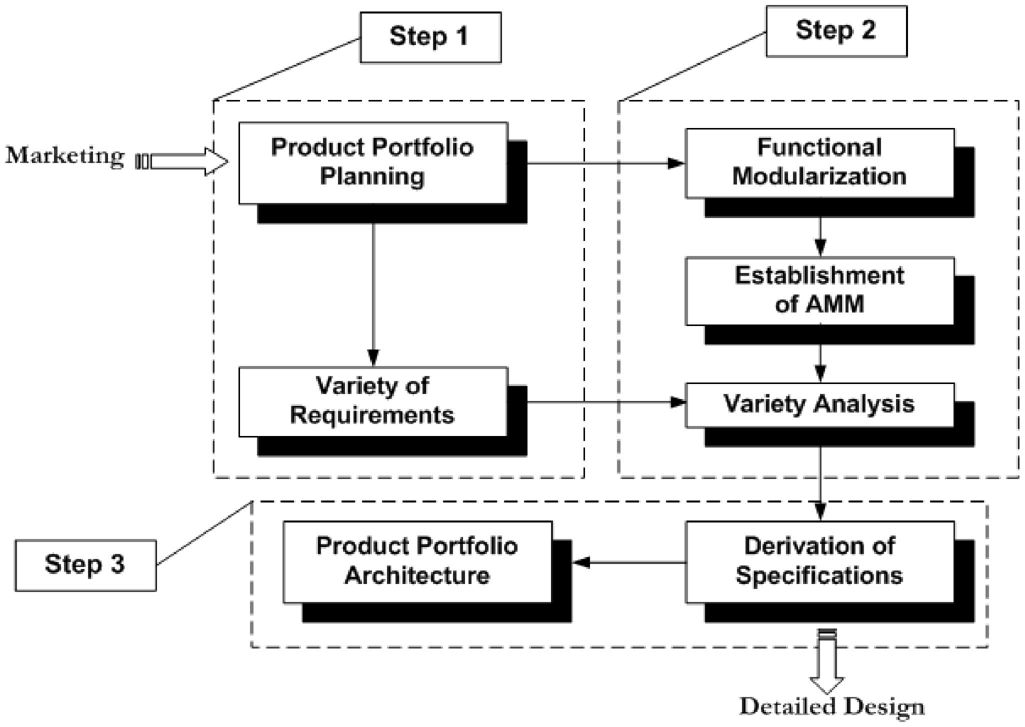

Step 1
Step 2
Marketing
Product Portfolio Planning
Variety of Requirements
Functional Modularization
Establishment of AMM
Variety Analysis
Step 3
Product Portfolio Architecture
Derivation of Specifications
Detailed Design

Figure 4.3: Three steps for modularizing the conceptual PFA

4.3.1 Product Family Planning

Since variety comes from various market segments, it is very necessary to rationally plan the whole product family offered to the market. In this step, product specifications existing in the market are collected and prepared for analysis. Although the original mindset of family design is to provide variety of products for market, immoderate customization may constrain customers' satisfaction and even lead to mass confusion (Huffman and Kahn, 1998). To finalize the range of target offerings, the company must choose the optimal level of product attributes (engineering metrics) for each product variant, and the optimal amount of product offerings. Such decisions may involve effective product family positioning or product portfolio planning (PPP) to maximize profit, sales or share of choices. Among many methods developed, conjoint analysis is one of the most popular preference-based techniques to decide product variety (Moore *et al.*, 1999). In particular, portfolio decision with customer-engineering interaction can effectively balance trade-offs between the benefits derived from providing variety and cost savings that can be achieved within firms (Jiao *et al.*, 2005).

In this research, we assume the range of product family and their specification in terms of engineering metrics are available for further investigation. From the collection of product specifications, we can use equation (4.6) to simply estimate the variance degree of product attribute ΔC.

$$\left[\Delta \mathrm{c}_{\mathrm{j}}, j=1,2...n\right] = \begin{cases} (N_j - 1)/m, & \text{for the discrete attribute} \\ \\ 0 & \text{, for the binary attribute} \end{cases} \qquad (4.6)$$

where Δc_j is the variance degree for product attribute j; N_j is the number of levels offered for product attribute j; m is the number of variants offered in the product family. For example, the voltage metric need to be customized at the 4 different levels for 6 variants and thus the variance degree is (4-1)/6=0.5. For binary attribute (i.e. yes or no), variance degree is always 0 since it can be viewed as an auxiliary feature. If derived variance degree of specific product attribute is closer to 1, this attribute may require more customization efforts to implement it.

4.3.2 Function-based Product Modularization

For product modularization process, there are three different types of approaches reported in the literature: customer-, function- and structure-based approaches as summarized in table 4.1. Each type of modularization process occurs at different product development stage and defines its own rule in its scope.

Table 4.1: Comparison of various modularization processes

Orientation	Approach	Methodology	Scope	Case Study
Customer	Moore *et al.*, 1999	Conjoint analysis	Product family	Electrical equipment
	Yu *et al.*, 1999	Market analysis	Product family	Leg room of car
Function	Stone *et al.*, 2000b	Functional modeling & heuristic method	Product	Electrical equipment
	Dahmus *et al.*, 2001	Functional modeling & heuristic method	Product family	Power tools
	Kurtadikar *et al.*, 2004	Functional modeling & heuristic method	Product family	Shop vacuum
	Zhang *et al.*, 2006	Functional modeling	Product family	Assembly device
Structure	Newcomb *et al.*, 1998	Modularity measure	Product	Center console
	Gershenson *et al.*, 1999	Modularity measure	Product	Mechanical pencil
	Hsiao and Liu, 2005	Interpretive Structural Model	Product family	Coffee maker

Since the definition of product architecture begins with the arrangement of functional element, functional modularization method provides an early schematic view of architectural exploration for product family while linking customer needs with engineering design. Meanwhile, the implication of modularity is to form basic configurable elements for functional sharing and variety generation. Therefore we adopt functional modeling method to modularize the product.

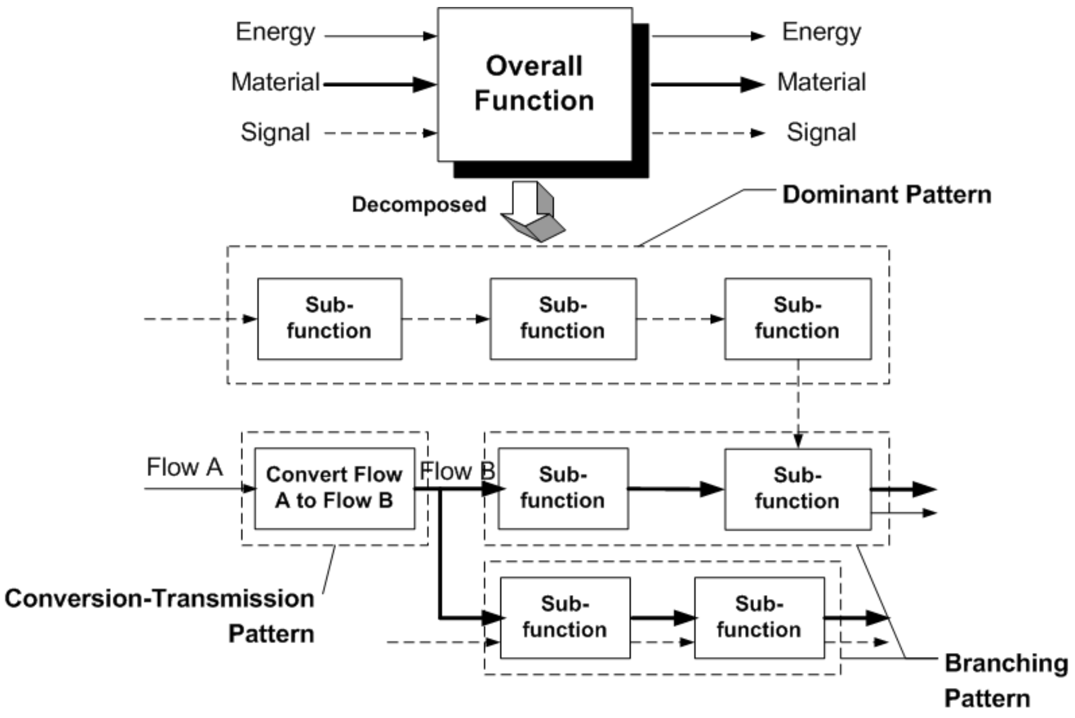


Figure 4.4: Illustration of functional modeling

Functional modeling is a key step at the conceptual design stage (Pahl and Beitz, 1996; Stone and Wood, 2000a), whether original or redesign. It provides an engineering view of how the sub-functions work together (based on feasible technological solutions) to achieve the desired functional requirements, and is independent of how the function is performed. This model uses a graph-based functional design language to form the product conceptual structure, where the product function is characterized in a standardized verb-object (function-flow) format and decomposed further into sub-functions (Stone and Wood, 2000a), as shown in Figure 4.4. Then a modular architecture is formed by grouping sub-functions together to form modules based on three heuristic methods: dominant flow, branching flow, and transmission/conversion (Stone *et al.*, 2000b). The modules identified can be used for concept generation and embodiment design. Using the functional model can significantly contribute to the product architecture development by moving the product architecture decision earlier in the conceptual design stage, particular for a series of similar products.

4.3.3 Variety Analysis

To fully understand the engineering relation between product attributes and conceptual modules for variety analysis, the source of architectural variation should be identified first. Several researches existing in the literature investigate the mechanism of variation transmission between segmented market and product architecture. They view the variety of requirements as the external drivers of variation, and the coupling interactions among components as the internal variation propagation (Martin & Ishii, 2002; Hsiao & Liu, 2005). Similarly, this thesis also approaches the generation of the Attribute-Module Matrix (AMM) from two perspectives: namely specification implementation and specification propagation.

Specification Implementation

Specification flows can be viewed as the design information that must be passed among designers to design their respective modules. Generally, each product attribute has its engineering metric to be customized for different levels and each conceptual module has the specification output to implement the corresponding product attribute. Thus, by directly mapping the product attribute to the module implementing the specification, we can establish their mapping relationship from the perspective of specification implementation. Table 4.2 presents an example showing the engineering relation between metrics and modules regarding power drill.

Then we use a 9/6/3/1/0 rating system for estimating the relation values, as shown in Table 4.3. The used ratio or proportional rating scale has been investigated in the field of cognitive psychology and seems preferable to the traditional linear interval scale, for example 8/6/4/2/0 (Franceschini and Rupil, 1999). For each relation node in the matrix, the design team estimates the implementation degree, which can be viewed as an indicator of redesign efforts to meet attribute change. Higher value means a stronger implementation relationship and results in greater redesign on the corresponding module. For example, metrics of voltage and charger time are completely realized by electricity supply module and any varieties of these two attributes will incur great changes in designing. Accordingly their relation values are assigned the highest value of 9.

Table 4.3 VI rating system

Rating	**Description**
9	Has a crucial relation between attribute and module
6	Has a strong relation between attribute and module
3	Has a partial relation between attribute and module
1	Has a minor relation between attribute and module
0	No relation exists

Table 4.2: Specification implementation between modules and attributes

Modules (Input/Output Flow)	**Product Attributes and Engineering Metrics**							
	Voltage (V)	Max. torque (in-lbs)	Number of variable speed	Rotation Speed (rpm)	Hammer capacity	Chuck size (mm)	Charger time (minutes)	Clutch setting
Elec. Supply Module (Electricity, V)	X						X	
Actuator Module (Signal/Electricity, V)	X							
Elec.-to-torque Module (Electricity, V/ Torque, in-lbs &rpm)		X		X	X			
Transmission Module (Torque/Torque, in-lbs &rpm)		X	X	X				
Hammer Module (Torque/Torque, in-lbs &rpm)					X			X
Secure Module (Torque/Torque, in-lbs &rpm)						X		
Handle Module (Signal/Torque, in-lbs &rpm)								

Specification Propagation

The coupling interaction among modules in term of specification flow is another important aspect of product architecture according to the definition by Ulrich (1995). Thus the specification change drawn from the various customer requirements may spread within the product architecture by module interaction. Here, since we focus on the functional aspects of product architecture, the functional modeling can provide a visual tool to identify the specification propagation within the product architecture by tracing the flows.

For the specifications propagated among modules, the design team should estimate the sensitivity of each module (based on feasible solutions) to a change in those propagated specification flows. Here we also use the 9/6/3/1/0 rating system to quantify the sensitivity. If a small change in the propagated specification requires a large change in the realization of the module, this module has a high sensitivity to the change of the attribute with that specification and thus their relation is given a higher rating value. For example, the electricity flow associated with voltage is propagated to actuator module. Although the electrical flow can range from 12V to 24V, a small change occurs to the actuator (switch) since most of switch designs can accommodate the different voltage settings from 12V to 24V. So the relation between voltage and actuator module is rated to be 1. But for electricity-to-torque module (motor), the change of electricity flow will require a moderate design change and thus the relation between voltage and electricity-to-torque module is rated to 6.

Variety Index

Traditionally individual product design captures customer needs and transfers them into engineering design. However current multi-product development usually involves designing a series of similar products/components synchronously and continuously. This compels designers to understand and evaluate the variation effect from customers on product architecture and later detailed design in the early design phase. Given a collection of the specifications for a product family, product architecture, and attribute-module relation, we can estimate VI for each module according to equation (4.5), which provides an indicator of the extent of efforts we need to design module instances.

4.3.4 Product Portfolio Architecture

The modules with low VI can be the common modules of product platform and will be used across the entire product family. On the other hand, the modules with high VI can be the differentiating modules, which need more instantiation for the specific variant product. Differentiating modules and their instances can be viewed as class-member relations. The instantiation design usually involves the scale-based optimization to achieve commonality with respect to variables/parts.

In order to further facilitate product development, product variants with their corresponding module instances in terms of engineering specification should be generated. Du *et al.* (2001) have investigated this process, namely variant derivation, which may involve four important steps: selection constraints, parameter propagation, include condition, and variety generation. In this thesis, parameter propagation is mainly used to accomplish the derivation of module instances.

At the end of this step, a product portfolio architecture (PPA) is formed to guide subsequent design activities and early product evaluation, which provide an engineering view of product family configuration characterized by combinations of the common modules $\{CM_j{}^*, j=1, 2 \ldots k\}$, and a set of differentiating modules $\{DM_i$, *i*=1, 2 … *n*} associated with their customized instances. To represent a family and its product variants, a decomposition /classification structure can be adopted to represent PPA as shown in Figure 4.5. In the vertical direction, PFA is constructed in a hierarchical form with the decomposition in each sub-level and with instances attached to the end modules. In the horizontal direction, PPA is organized into two categories of conceptual modules: common and differentiating ones. Meanwhile, the coupling interfaces among modules will be committed between different design team, and defined in terms of consistent engineering specifications (e.g. dimension, torque).

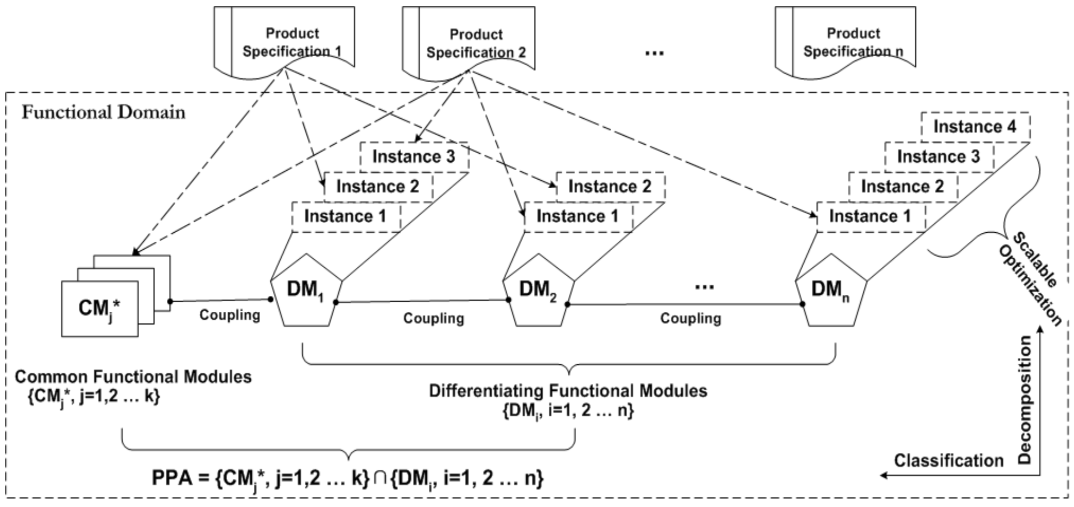


Figure 4.5: Engineering view of product portfolio architecture

4.4 Case Study

For the case study, a family of power tools is used here to illustrate the aforementioned approach because the target market is increasingly segmented and filled with clusters of products with different functional features and engineering metrics, such as drill and jigsaw. Meanwhile, a lot of research work has used different types of power tools in their case study and provided plentiful technical support. In the following section, a step-by-step method is illustrated to investigate the architectural variation due to customized requirements and identify platforming direction at the system-level design.

4.4.1 A Product Family of Cordless Drills/Drivers

The investigated power tools are centered on a cordless drill/driver series, which is assumed to share the same functionality but target at different price-performance levels. According to the market strategy, the firm aims to serve different application in the vertical market segments (e.g. household, workshop, construction). Based on the market analysis using quality function deployment (QFD), eight relevant product attributes that are important in the customer choice are identified: voltage, maximum torque, the number of variable speed, rotation speed (no load), clutch settings, chuck size, charging time, and hammer capacity. However, potential customers express different extents of preference toward these attributes. To balance the attribute importance in customer choice, preferences for different attributes are normalized based on questionnaire survey and given through weight assignments by experienced designer, as shown in Table 4.6.

Since product family planning is not our research focus and several researches have provided support to accomplish it (Jiao *et al.*, 2005&2007a), the resultant range of product family (6 variants) is assumed to be available at this stage, as shown in Table 4.4. Three common attributes and five differentiating attributes for variants are identified, together with their corresponding attribute levels.

Table 4.4 Collection of product family specifications

Common Attributes		
Clutch Settings (C_8)	23	For all variants
Charging Time (C_7)	1 hour	For all variants
Clutch Size (C_6)	13mm	For all variants
Differentiating Attributes for Variants		
Voltage (C_1)	12 V	V1
	14.4 V	V2, V4
	18 V	V3, V5
	24 V	V6
Max. Torque * (C_2)	22 N.m	V1
	27 N.m	V2
	34 N.m	V3, V4
	40 N.m	V5
	46 N.m	V6
Number of Variable Speeds (C_3)	2	V1,V3,V4
	3	V2,V5,V6
No Load Speed (C_4)	400 rpm	For all variants
	1400 rpm	V1,V2,V3,V4,V5
	2000 rpm	V2,V5,V6
Hammer (C_5)	yes	V4,V5, V6
	no	V1,V2,V3

According to equation (4.5), the vector of variance degree for each product attribute (C_1, …, C_8) in the family of cordless drills is normalized as ΔC= [3/6, 4/6, 1/6, 2/6, 0, 0, 0, 0]. For example, 5 desired levels for torque attribute will result in a variance degree of (5-1)/6.

4.4.2 Functional Modularization

Although physical components and structures for target product may exist, a functional view can acquire early modularity in the architecture, especially when some components/modules require orders from other firms. In addition, borrowing functional vocabulary from design repositories can facilitate the concept generation and design reuse (Stone and Wood, 2000a).

The functional diagram of cordless drills/drivers is shown in Figure 4.6 and Table

4.5 lists the candidate modules identified by functional modeling. By tracing the flow status through the whole function chain, we can divide the cordless drills/drivers into four sub-systems: electrical, conversion, mechanical and support sub-systems. Each sub-system is decomposed into modules, which address the implementation of related product attributes. In this example, electrical sub-system includes electricity supply module and actuator module. Conversion sub-system only has electricity-to-torque module. Mechanical sub-system contains transmission, hammer, and secure module. Handle module is accessorial support sub-system.

Since the target product family has the same product attributes but different levels at some attributes, they can share a common functional model. But for each product variant, some modules in the functional diagram differ in the flow intensity. For instance, although electricity-to-torque module provides the same functionality for each product, the required input flow (electricity) and generated output flow (torque) hold different intensities according to engineering metrics. As a result, the identified architecture of PFA at this stage provides the same module type but may require further instantiation for each variant.

Table 4.5: Modules of the cordless drill family

Module	**Module Name**	**Used Heuristic Method**
M_1	Elec. Supply module	Dominant Flow
M_2	Actuator module	Dominant Flow
M_3	Elec.-to torque module	Transformation/Conversion
M_4	Transmission module	Dominant Flow
M_5	Hammer module	Dominant Flow
M_6	Secure module	Dominant Flow
M_7	Handle module	Branching Flow

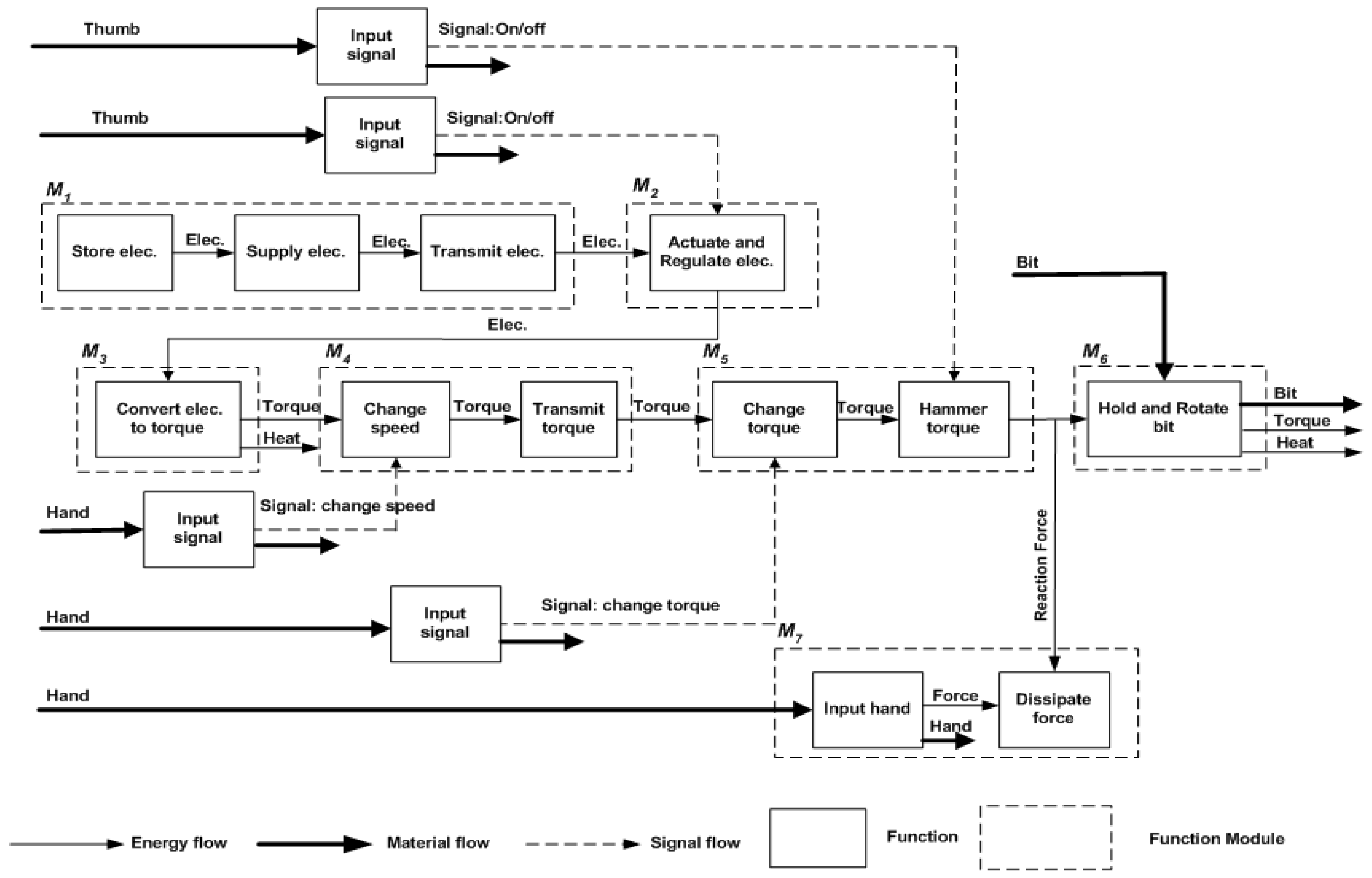


Figure 4.6: Functional modeling of power tool family

4.4.3 Attribute-Module Matrix and Variety Analysis

In this example, AMM is determined by interviewing the experienced designers. Two perspectives of attribute-module relation will be investigated according to the method mentioned earlier. Figure 4.7 illustrates the final mapping relation result regarding specification implementation and specification propagation. By tracing the flow status in functional diagram, the relation between modules and engineering metrics can be easily identified. For instance, electricity flow is always related to such metrics as voltage and charging time; torque flow determines maximum torque and rotation speed.

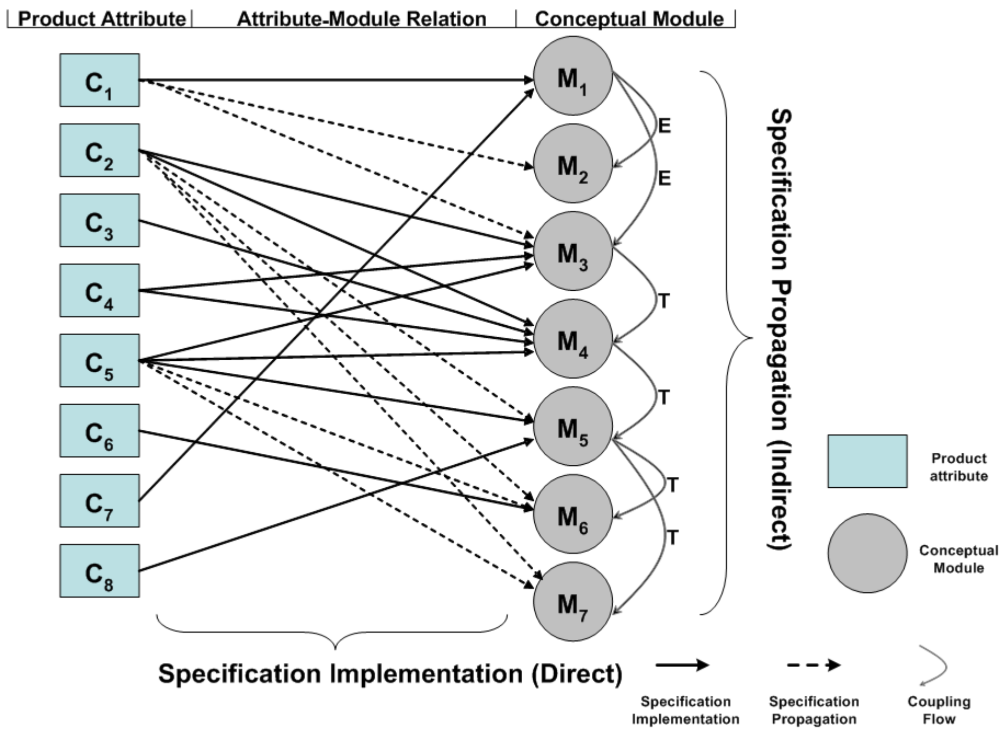

Product Attribute
Attribute-Module Relation
Conceptual Module
C1
C2
C3
C4
C5
C6
C7
C8
M1
M2
M3
M4
M5
M6
M7
E
E
T
T
T
T
Specification Propagation (Indirect)
Specification Implementation (Direct)
Product attribute
Conceptual Module
Specification Implementation
Specification Propagation
Coupling Flow

Figure 4.7: Two perspectives of Attribute-Module relation

Table 4.6 lists the estimated value for attribute-module matrix, together with preference weights and variance degrees for product attributes. Then the VI for each module can be derived by incorporating the rated relation between the attribute and the module, with variance degree and preference weight. For instance, electricity supply module is responsible to two attributes: voltage and charging time with different preference weights [0.14, 0.08]. Voltage and charging time will vary for the whole product family with normalized variance degree 3/6 and 0, respectively. Thus according to equation 4.4, VI for electricity supply module will be

$$VI_1 = \sum a_{j1} \Delta c_j w_j = 9 \times 0.14 \times 3/6 + 9 \times 0.08 \times 0 = 0.63 .$$

The derived variety index can assist designers in identifying crucial modules. Smaller index means less variety value in customer view and may require less attention; larger index means possible greater customer-perceived value and more design complexity delivered by the differentiation. In order to identify the modules and their corresponding components/assembly on which we should focus in the latter design, module-component categorization is given in Table 4.7. This evaluation process also lists the estimated nonrecurring engineering (NRE) costs, which refers to the one-time cost of researching, designing, and testing a new product and can be viewed as an effort indicator of designing the differentiating components. In this case, the NRE cost mainly includes payment for designers, prototyping cost, and tooling cost and has been collected from the investigated company in China. In addition, the feasibility of over-design for each module is evaluated by the design team and represented as "bubble", as plotted in Figure 4.8, to indicate how well the component design can accommodate the specification change through over-design without much cost increase and performance loss. If the component/module has a higher VI and NRE cost, balance needs to be made to decrease the developing cost incurred from the increasing number of instances; otherwise, those modules with lower VI and NRE cost may be potential platform elements for reuse among variants. The threshold values of 5000, 0.5 are given as reference line and derived based on the mean value of NRE and VI, respectively.

Table 4.6: Attribute-Module Matrix and Variety Index

Sub-system	**Module**	**Product Attributes**								**Architectural Variety (VI)**
		Voltage (C_1)	Max. torque (C_2)	Number of variable speed (C_3)	Speed (C_4)	Hammer capacity (C_5)	Chuck size (C_6)	Charger time (C_7)	Clutch Settings (C_8)	
Electrical	Elec. Supply module (M_1)	9	0	0	0	0	0	9	0	**0.63**
	Actuator module (M_2)	*3*	0	0	0	0	0	0	0	**0.21**
Conversion	Elec.-to torque module (M_3)	*6*	9	0	6	3	0	0	0	**2.08**
Mechanical	Transmission module (M_4)	0	6	9	9	6	0	0	0	**1.54**
	Hammer module (M_5)	0	*1*	0	0	9	0	0	9	**0.17**
	Secure module (M_6)	0	*1*	0	0	*3*	9	0	0	**0.17**
Support	Handle module (M_7)	0	*1*	0	0	*3*	0	0	0	**0.17**
Weight of attribute (w_j)		0.14	0.25	0.20	0.08	0.14	0.03	0.08	0.08	$\sum w_j = 1$
Variance Degree (ΔC)		3/6	4/6	1/6	2/6	0	0	0	0	

Table 4.7: Module-component categorization of cordless drill

Module	Component/assembly	VI	Feasibility of Over-design	NRE ($)*
M_1	Battery	0.63	Low	3,000
M_2	Switch	0.21	High	500
M_3	D.C Motor	2.08	Low	5,000
M_4	Gear Assembly	1.54	Medium	18,000
M_5	Clutch (Cam)	0.17	High	7,000
M_6	Chuck	0.17	High	1,500
M_7	Handle	0.17	High	1,000

* NRE is nonrecurring engineering cost and cannot be repetitive

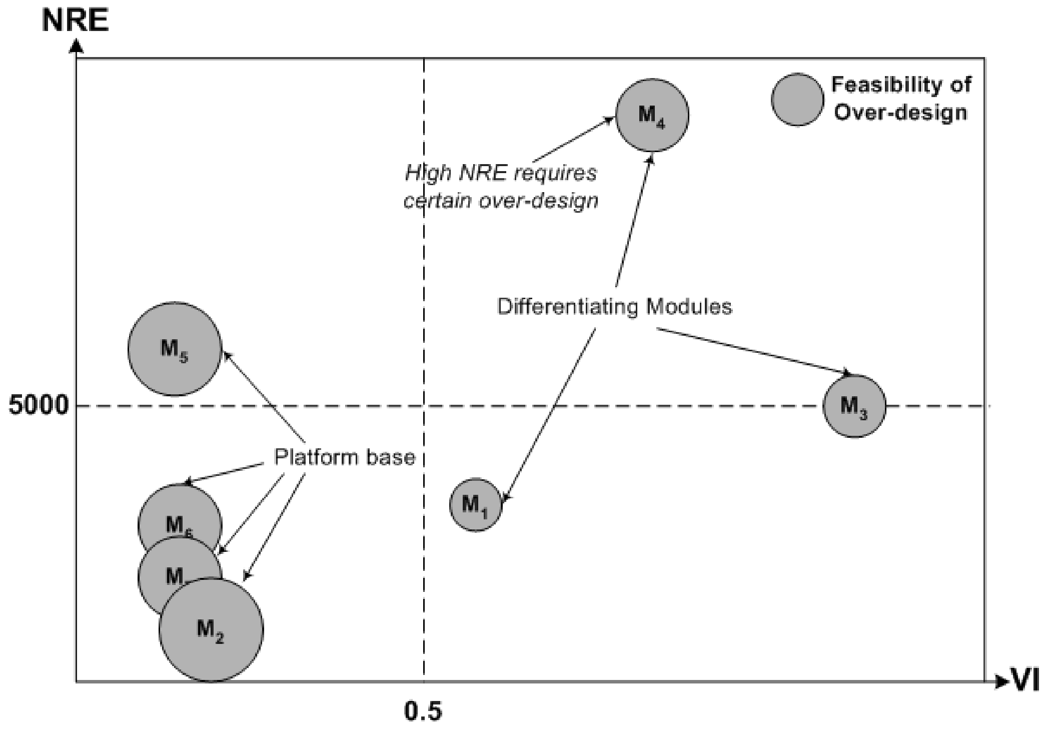


Figure 4.8: VI versus NRE versus feasibility of over-design

Over-design may be an effective design strategy to improve standardization and commonality across the family of products. For those modules with lower VI and higher feasibility of over-design, they can be considered “fixed” for the product family through standardization. This implies standardizing these modules so that they can accommodate a small or even moderate change in the specification. For example, the actuator module (switch) may require different working voltage for each variant. However standardizing the switch that works at different voltages raging from 12V to 24V can keep only one instance for all variants since there is little cost increase and performance lost. In this example, Actuator (M_2), Hammer (M_5), Secure (M_6) and Handle (M_7) modules compose the platform basis and can be reused across the whole family. On the other hand, those modules with higher VI, such as electricity supply (M_1), electricity-to-torque (M_3), and transmission (M_4) modules, will be differentiating modules and need more instances to meet different specifications. Gear assembly (M_4) has a highest VI and NRE cost, which means the module instantiation for each product requires greatest complexity and cost. To reduce cost saving from repetitive design and process reuse, suitable over-design will be adopted to reduce the number of instances in the detailed design stage, which will be explored in Chapter 5.

4.4.4 Instance Derivation and Product Portfolio Architecture

After variety analysis for each module, the suitable instance specification for each module should be determined according to variant requirements. This process is typically based upon a set of selected options from customers (Du *et al.*, 2001). Selected attribute levels are transformed to the variety parameters of the end-product and then propagated down the hierarchy of product architecture. Through this parameter propagation, all parameters of module instances obtain specific values. However the allocation of parameter value to each instance requires domain knowledge (e.g. industrial standard) about mapping functional requirements to design parameters (Suh, 2001). Table 4.8 lists the 6 variants at the end-product level and the corresponding instances for each module with the aid of engineer designers.

Table 4.8: Engineering specification for module instances

	Module/Component	**Instance**	**Specification Description**	**Variants**
Differentiating	M_1/ Battery Package (Ni-Cd)	$A^1{}_1$	Output:12V; Capacity: 1.7 Amp-Hour	V1
		$A^1{}_2$	Output:14.4V; Capacity: 1.7 Amp-Hour	V2,V4
		$A^1{}_3$	Output:18V; Capacity: 1.7 Amp-Hour	V3,V5
		$A^1{}_4$	Output:24V; Capacity: 1.7 Amp-Hour	V6
	M_3/D.C. Motor Assembly (D.C.)	$A^3{}_1$	Input:12V; Max. Speed:20k rpm, Torque (stall)=0.55 Nm	V1
		$A^3{}_2$	Input:14.4V; Max.Speed:20k rpm, Torque (stall)=0.675 Nm	V2
		$A^3{}_3$	Input:14.4V; Max.Speed:20k rpm, Torque (stall)=0.85 Nm	V3
		$A^3{}_4$	Input: 18V; Max. Speed:20k rpm, Torque (stall)= 0.85 Nm	V4
		$A^3{}_5$	Input: 18V; Max. Speed:20k rpm, Torque (stall)=1.0 Nm	V5
		$A^3{}_6$	Input: 24V; Max. Speed:20k rpm, Torque (stall)=1.15 Nm	V6
	M_4/Planetary Gear Train	$A^4{}_1$	Output: Speed=500/1500 rpm, Max Torque =22 Nm	V1
		$A^4{}_2$	Output: Speed=500/1500/2000rpm, Max Torque =27 Nm	V2
		$A^4{}_3$	Output: Speed=500/1500 rpm, Max Torque =34 Nm	V3,V4
		$A^4{}_4$	Output: Speed=500/1500/2000rpm, Max Torque =40 Nm	V5
		$A^4{}_5$	Output: Speed=500/2000 rpm, Max Torque =46 Nm	V6
Common	M_2/Switch	$A^2{}_1$	D.C; Voltage:12-24; Switch type: on/off/Variable speed	All
	M_5/Clutch(Cam)	$A^5{}_1$	Coil Clutch; (Number of cam tooth:18)	All
	M_6/Chuck	$A^6{}_1$	Chuck Size:13mm; Keyless; Single sleeve; Metal	All
	M_7/Handle	$A^7{}_1$	Material: plastic	All

4.5 Summary

This chapter investigates the modularization of the conceptual PFA. Based on traditional definition of product architecture and the proposed variety analysis, the proposed method provides designers a step-by-step procedure to achieve early modularity at the system-level design stage, integrating product family planning, functional modularization, and variety analysis. Incorporating the external variety of requirements into definition of product architecture and constructing a PFA at the conceptual design stage not only can benefit early product family evaluation and conceptual design, but also address other operational issues (Fixson, 2005), such as supply chain, collaborative development, variety management, and so on. A family of cordless drills/drives is used as case study to demonstrate the proposed modularity analysis and finalize the platforming direction in the product architecture. Eventually, the product portfolio architecture with detailed design specifications for each module is formed to guide subsequent module instantiation and commonality design at phase 2.

It is very interesting to note that AMM matrix indicates the interrelated coupling between customer domain and functional domain, and its characteristics (e.g. diagonal and non-diagonal) determine the complexity in modularizing PFA. For AMM with a diagonal matrix, each functional module in the product implements the corresponding attribute without interacting with other modules, and accordingly modularization of PFA can be easily tackled in either domain. However, most practical product systems are characterized by non-diagonal matrix, which make modularization of PFA more complicated and involve interaction between two domains. For the highly coupled design, AMM might be a complexity indicator to re-construct functional architecture by grouping some highly interrelated elements into big chunks. Moreover, a 9/6/3/1/0 rating scale is used in this thesis to quantify AMM based on subjective judgment of coupling degree between modules and attributes. A further investigation of rating scale on the effect of variety analysis has not been carried on at this stage because the use of rating scales itself is an interesting topic in the field of cognitive psychology and beyond the current research scope in the thesis. More comparison and illustration about rating scale in QFD can be found in (Franceschini and Rupil, 1999).

Suitable over-design can be an effective design strategy to reduce the number of instances and increase commonality across a family of products by over-designing component dimensions to accommodate the specification variation. However some disadvantages can be incurred from over-design. One of them is that the individual performance may be impaired. Also the increased material costs due to over-designed dimension of component may limit the commonality of this component. In addition, the achieved economical benefit from commonality configuration is another factor to determine the extent of over-design. In the next chapter, these issues will be investigated to achieve commonality at the detailed design stage.

Chapter 5 A Manufacturing-biased Platform Decision and Product Family Design

5.1 Introduction

While *modularity* identified at the system-level design stage aims to help designers decompose product structures and manage variety generation, *commonality* is another important issue for platform development during the detail design stage and determines the final economies of scale. Towards this end, most researches attempt to balance the trade-off between reducing variety or increasing commonality of design, and performance deviation of individual variant due to platforming effect. Varieties of design that incur additional complexity but contribute less value in customer view need to be standardized or commonalized, but those varieties to provide more customer-perceived values (functional and performance characteristics) should be emphasized and require more resource to realize them. However, it is always assumed that maximizing commonality across the family minimizes the production cost and minimizing performance deviation maximizes its demand, since it is very difficult and not practicable to directly capture the relationship between commonality and cost during early design stage, as well as the relationship between performance and market demand.

To meet external variety of requirements characterized by functional features and engineering specification, the corresponding internal varieties of design and production have to be offered (Anderson, 1997). Variety of design refers to the diversity of product definition in terms of various functional and technical varieties.

Similarly, the direct consequence of product variety on production is observed as an exponentially increased number of process variation such as customized operations, and resource variation, such as diverse machines, tools, inventories and labors. Although most decisions related to product definition are made during the design stage, the major portion of product costs is actually committed and determined at production stage, as shown in Figure 5.1. This implies that product family design can only be rationalized by effectively coordinating product design and production processes and exploiting potential platforming opportunities.

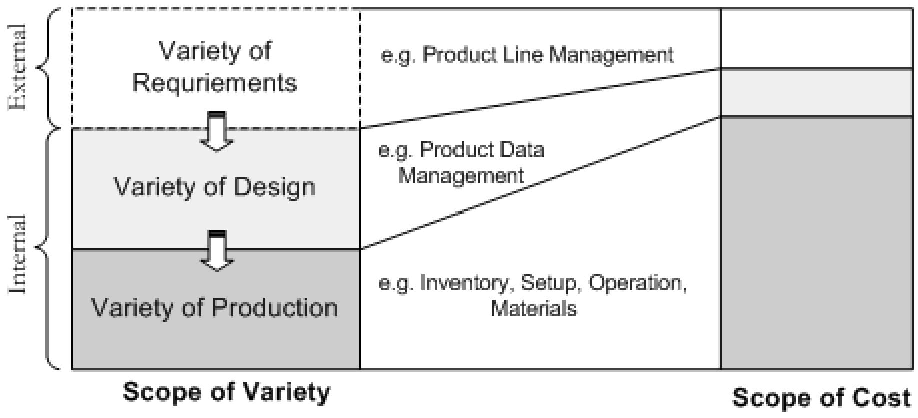


Figure 5.1: Cost contribution of different varieties

5.2 Multi-platforming Configuration

Mathematically, the family design problem with *p* products can be formulated as follows:

$$\begin{aligned} &Find \quad X = (x_1^1,...,x_1^p,x_2^1,...,x_2^p,...,x_n^1,...x_n^p) \\ &Max. \quad \{\sum f^P(X), f^C(X)\}, \\ &s.t. \quad g_j(X) \le 0, \qquad j = 1,...,k \\ &\qquad X_L \le X \le X_U, \;\; X = [X^d, X^c]^T \end{aligned} \tag{5.1}$$

where the vector of design variables *X* describes the target product, including possible platform settings and unique variables. $\sum f^P(X)$ is a set of design objectives or responses for the whole family. $f^C(X)$ is the commonality aspects of the whole family, and may be pre-specified (*e.g.* single-platform) or optimized during the family design. $g_j(X)$ represents the constraint function in the design model. A design space is the range that the design variables can take. Currently, most methods existing in the literature only involve continuous types of variables. However, for conformance to industrial standards and manufacturing requirements, the design space for some design variables is not always continuous (X^d), but discrete (X^c) or even binary.

A successful family design always begins with an effective platform configuration, which provides designers with various platforming directions. Despite the advantage of simplicity, single-platform configurations may make some low-end products over-designed and some high-end products under-designed because the unique common value for one variable may not be ideal for each product in the family (Dai and Scott, 2007). As shown in Figure 5.2 (a), one platform variable x is shared by all of the products in single-platform configuration. However, variant p_1 and p_4 have more performance loss because the uniquely converged point deviates more from optimum points of variant p_1 and p_4. To avoid the drawback, one possible way is to partially share the variable value among subset of variants. As shown in Figure 5.2 (b), x for variant p_1 and p_2 can be clustered into x^{1**} and meanwhile x for variant p_3 and p_4 can be clustered into x^{2**}.

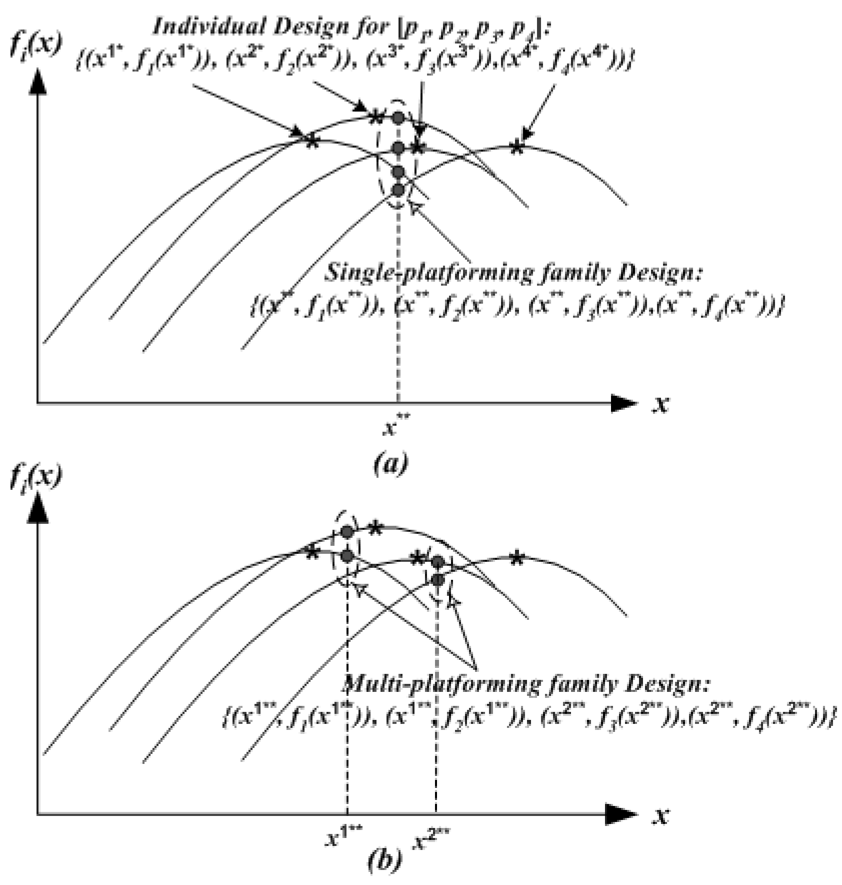

$f_i(x)$
Individual Design for $[p_1, p_2, p_3, p_4]$:
$\{(x^{1*}, f_1(x^{1*})), (x^{2*}, f_2(x^{2*})), (x^{3*}, f_3(x^{3*})), (x^{4*}, f_4(x^{4*}))\}$
Single-platforming family Design:
$\{(x^{**}, f_1(x^{**})), (x^{**}, f_2(x^{**})), (x^{**}, f_3(x^{**})), (x^{**}, f_4(x^{**}))\}$
x
x^{**}
(a)
$f_i(x)$
Multi-platforming family Design:
$\{(x^{1**}, f_1(x^{1**})), (x^{1**}, f_2(x^{1**})), (x^{2**}, f_3(x^{2**})), (x^{2**}, f_4(x^{2**}))\}$
x
x^{1**}
x^{2**}
(b)

Figure 5.2: (a) Single-platform configuration (b) Multi-platforming configuration

This multi-platforming configuration, in which variables may be partially shared among variants in any possible combination of subsets, offers opportunities for more superior overall design but presents a more difficult computation problem. Suppose the target product is described by a set of 6 design variables and there are 4 variants in the family, the total number of possible combinations is 15^6, compared to 2^6 possible combinations of the single-platform configuration. Detailed derivation can be found in (Dai and Scott, 2007). Moreover, this number of multi-platforming combinations increases rapidly with both number of design variables and the number of product variations.

5.3 Manufacturing-biased Commonality Index

In order to access multi-platforming family solutions with varying levels of commonality, we adopt a mechanism of quantifying the level of commonality as *commonality index* (CI), and incorporating it into the family design to dynamically generate the platform configuration. CI is a measure of sharing degree regarding the design parameters throughout the entire product family. It can be viewed as an efficiency indicator of reduced manufacturing complexity or cost savings from family design, and also as a means to control the level of commonality across the family during optimization. Although several component-based commonality indices have been proposed in the literature, they emphasize on the bill-of-material assessment of product family based on their particular standpoint (Thevenot and Simpson, 2006; Blecker and Abdelkafi, 2007), which is also mathematically difficult to be implemented at the parametric design stage.

The motivation behind the scale-based family design is to derive certain economic benefits from sharing some variable values over more than one product. Thus an effective quantification of commonality regarding design space determines the final outcome. However, some unsolved challenges remain and should be met before the final formulation is given. The first is that of resolving the coupled design case, where several variables jointly determine the final dimensions of a single component. Consider the simple example of a structural component with cross-section characterized by three variables: b, h and t as shown in Figure 5.3. By sharing b and h and holding different t, two instances are derived to meet different requirements. If stamping is used to produce it, certain manufacturing benefit can be generated from reuse of the bottom die. However, when powder injection molding method is used, partial sharing cannot produce any benefit at all because the three variables jointly determine the component dimensions and the corresponding molds used by this process. In other words, only complete sharing of b, h and t between the two products will reuse the molds to be tooled and thus reduce cost. Thus, the first challenge is *to provide for benefit from variable sharing with regard to manufacturing requirements*. In the simple or uncoupled design case, each variable corresponds to one component or process, and the sharing of variables generates the expected advantages at the manufacturing stage. However, the coupled design situation does not necessarily ensure benefit from direct sharing of variables. Toward this, analysis of sharing pattern linked to the manufacturing stage is required to decide on the *basic platforming elements*, whose reuse will reduce manufacturing complexity and thus improve economic efficiency. In this case, the basic elements may not be single variables, but subset of variables or even a set of all variables related to one component. In the previous example, the basic element to be shared is the subset of *b* and *h* for stamping method; for injection molding method, all variables (*b*, *h* and *t*) related to the component form the basic element.

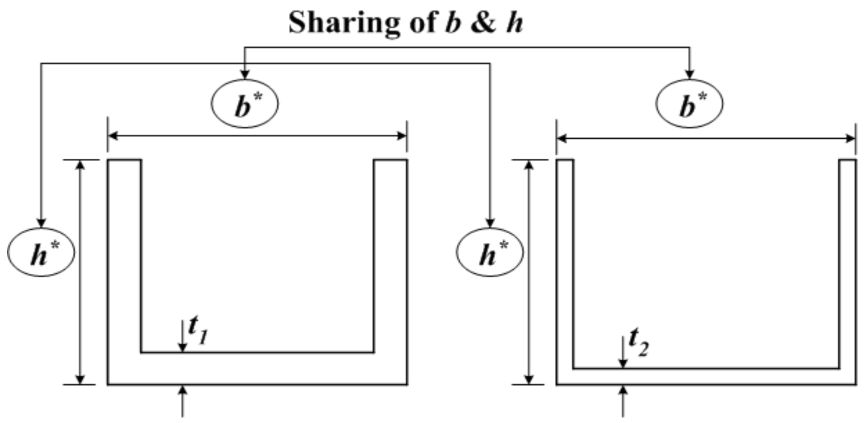


Figure 5.3: Platform decision affected by manufacturing consideration

More generally, the identification of the *basic platforming elements* is not a separate task at design phase and requires careful coordination with that of the production stage. Commonality in terms of shared variable values must be consistent with technical commonality in fulfillment.

Another challenge is *to determine the relative extent that the platforming element* *can provide the desired benefits from sharing among family products*. This situation can be illustrated with the family design of electric motors used earlier by Simpson *et al.* (2001a). Although their proposed optimization approach revealed that the motor platform scaled on the outer radius of stator can achieve better specified performance compared to that scaled on the stack length, the manufacturers still choose the latter due to lower production cost. Therefore, when measuring the level of commonality regarding the shared elements, we need to pay special attention on the derived benefits on the manufacturing cost from their sharing. Toward this, we propose the *expected sharing degree* (*ESD*) for each platforming element to reflect their contribution in cost saving. Those elements which generate more cost saving from being shared will be assigned a higher ESD. Accordingly they will contribute more value to the overall *CI* and have a higher probability of being shared among all the candidate platform elements.

Finally, for a product family with *p* products, the formulation of *CI* is proposed in this research and can be defined as follows:

$$\mu_C = \sum_{j=1}^{k} w_j^{\,e} C_j = \sum_{j=1}^{k} w_j^{\,e} \left(\frac{p - S_j}{p-1}\right) \qquad (5.2)$$

where w_j^e is the *expected sharing degree* (ESD) for the *basic platforming element j*; S_j is the number of different instances for element *j* and range from 1 to *p*; C_j is the commonality index for element *j* and treated as a linear normalization as shown in Figure 5.4, which means a fixed increase of C_j from one less instance of element *j*; μ_c is the overall commonality index for the family by aggregating commonality index of all *k* elements according to their ESD. μ_c has a range of [0, 1] and the larger μ_c means higher level of commonality regarding the design space throughout the whole family.

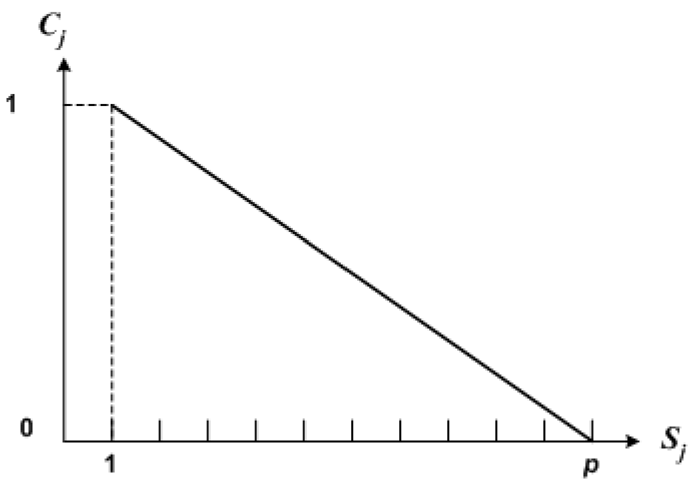


Figure 5.4: A linear relation between CI and the number of instance

Unlike the pre-specified platform configuration mentioned in the literature, the use of commonality index (μ_c) enables dynamic platform configuration during the family design and can potentially provide various family solutions with different level of commonality. In the meantime, the platform decision in this paper focuses on manufacturing efficiency so that those platforming elements with higher ESD will have a higher possibility of being reused, and thus generate a cost-effective platform configuration.

5.4 Systematic Scalable Product Family Design

Based on the aforementioned platform decision strategy, a systematic method for scale-based family design is proposed and developed, with its framework illustrated in Figure 5.5. This newly proposed method begins with an individual design and takes its overall design requirements (e.g. performance specification from system-level modularity analysis) as the input. The output is the specifications of the product family design (e.g. physical parameters to describe each product) with varying levels of commonality. Each step is explained in detail as follows and serves to guide the engineering designer to formulate and solve the family design problem.

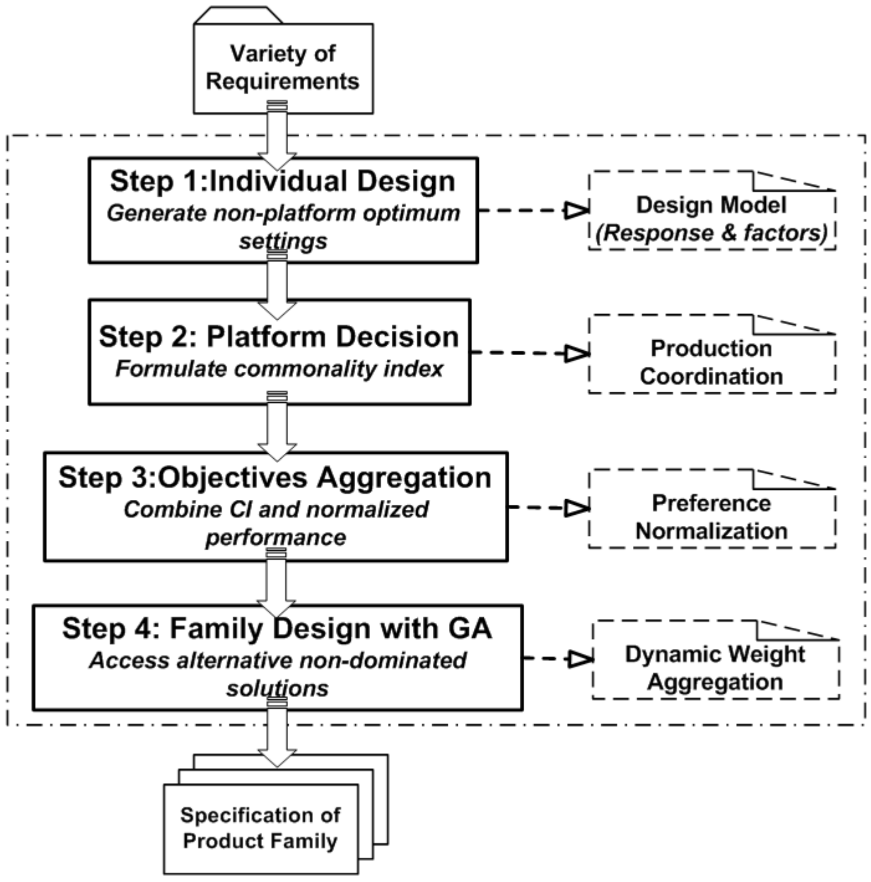

Variety of Requirements
Step 1:Individual Design
Generate non-platform optimum settings
Design Model (Response & factors)
Step 2: Platform Decision
Formulate commonality index
Production Coordination
Step 3:Objectives Aggregation
Combine CI and normalized performance
Preference Normalization
Step 4: Family Design with GA
Access alternative non-dominated solutions
Dynamic Weight Aggregation
Specification of Product Family

Figure 5.5: Framework of systematic product family design

5.4.1 Individual Design

The first step is to generate the optimal null-platform parameter configuration for each product by solving the design problem individually. The result will serve as the design target to normalize the performance objective for the corresponding family design and also for comparison as a benchmark. This step may be not necessary if the benchmark can be collected from existing counterparts by other providers. An explicitly mathematical model or computational simulation is first required to reflect the relationship between the input variables and the output performance response. Although most approaches assume an existing design model, whether mathematical or simulation-based, it is necessary to carefully prepare the model and identify two important factors as follows:

- *Response*: performance output of the product family to distinguish from products in the market. Performance response may be a single feature or be a multi-objective function if more than one aspects of product behavior are involved. In the latter case, the different responses have to be normalized first and combined into an aggregated objective function. In addition, it is assumed that the performance responses of a family of products will never be improved by increasing commonality.
- *Control factors*: design variables that can be freely specified within the boundaries by a designer to characterize the product. These variables may include the platform elements, pre-specified or determined during optimization, and scaling variables by which a product platform is leveraged through a scaling method to derive the family members.

Assuming that all products in the family share the same constraints but goals may differ depending on the market segments served by the each variant; the following optimization is used to obtain optimal design solutions for each family member.

$$\begin{aligned}
&For\ product\ v_i, i = 1,2,3,...,p \\
&Find \quad X^i = (x_1^i, x_2^i, ..., x_n^i) \\
&Max. \quad \{\sum f^P(X^i)\}, \\
&s.t. \quad g_j(X^i) \le 0, \qquad j = 1,...,k \\
&\qquad\quad X_L{}^i \le X^i \le X_U{}^i
\end{aligned} \tag{5.3}$$

where X^i is a vector of design variables for product *i*. Although only the performance aspect is involved in the individual design, several performance criteria with different dimensions may be normalized and aggregated, to be explored in section 5.3.3. This individual design is repeated by *p* times, each time with the design goals for the selected product.

5.4.2 Platform Decision

This step centers on determining the right components/variables to be shared or reused during the family design so that the family members can be derived in a cost-effective manner. Although the economic efficiency (e.g. manufacturing cost) has been widely accepted as the core issue driving the reuse of variables, components, and resources, most family design approaches relying on scale-based platform in the literature determined the combination of common and scaling parameters only from the aspects of design problem (e.g. minimizing performance variation), as mentioned in the literature review of Chapter 2. Accordingly, the resultant family design marginally achieves the desired economical advantage. In order to resolve the drawback existing in the platform decision, we adopt a systematic means to analyze the impact of product platforming on production activities by coordinating design varieties with process variations and while quantifying the level of commonality as an index of manufacturing efficiency.

Modeling of Production System

The first task in the platform decision is to investigate cost implications of production system. Several different approaches have been developed to model production flow based on their particular standpoint. The Activity-Based Costing (ABC) method is reported in many researches to support product family design due to its effectiveness and accuracy in tracing costs in the context of mass customization (Hundal, 1997; Park and Simpson, 2005; Zhang and Tseng, 2007). Based on the ABC principle, the production system can be modeled as a hierarchy of activities, in which various activities or processes are identified and associated resources are assigned as shown in Figure 5.6. Generally, activities can be classified into type of direct costs at unit-level, and type of indirect costs at batch-, product-, and facility-level. Due to the fact that scale-based platform approaches are always dedicated to design a family of component/module-level products, the complexity involving the production model is much reduced, and unit- (e.g. casting, machining) and batch-level (e.g. setup, work-in-process) activities can suffice to help to analyze the platforming effect. However, for a large-scale design problem with complicated production system, a comprehensive cost estimation system for product family can be developed to assist in the platform decision (Park and Simpson, 2005). Since it is a broad topic and beyond our research scope, we only focus on those activities potentially affected by platform and product family design.

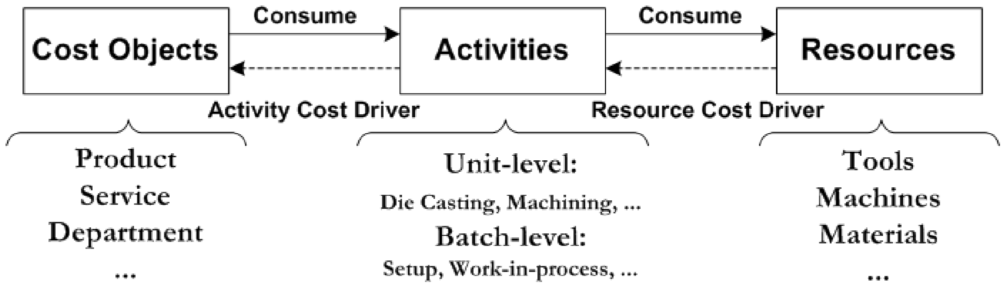


Figure 5.6: The principle of activity-based costing

Variety Coordination

Secondly, mapping design variety to manufacturing process variation in the context of manufacturing requirement is analyzed to investigate the platforming influence on production. Generally, parts or components in a product with various features usually involve several manufacturing processes and require different resources to achieve part differentiation. However, not all related processes will be affected by platform design. To identify the platforming effect, the sets of design parameters are mapped to the resources consumed by processes in the production flow to determine the affected activities and basic platforming elements, the reuse of which can create economy of scale by reducing the required resource. Consider a cylindrical component with a coaxial hole described by t (thickness) and r (inner radius) for example, and assume die casting process (affected activity) to produce it. To reduce cost from reuse of common die (resource), both variables t and r have to hold common values among different products. Accordingly, the basic element in this case will be a set of t and r since they are coupled together in the process. However, the coupled relationship between r and t can be released in case of other types of processes. Two different machining processes (cutting and drilling), for example, are employed sequentially to finish the whole cycle. Variable r is related to drilling process (affected activity) and common use of r among products can produce savings from tool sharing and setup labor reduction (resource). Thus, the single variable r is the basic platforming element and its reuse or sharing can generate certain manufacturing benefit.

Allocation of Expected Sharing Degree

Thirdly, *expected sharing degree* (ESD) will be assigned to all basic platforming elements for reuse to reflect their contribution in cost savings. To achieve this, the additional cost ($C^a_{j,\, j=1\ldots k}$) incurred from one more instance for element j needs to be calculated from allocated resources, or estimated based on historical data. Then the additional cost of each element is normalized by the aggregated additional cost of all elements and assigned as ESD.

$$w_j^e = \frac{C_j^a}{\sum C_i^a} \qquad i, j = 1, 2, \ldots \tag{5.4}$$

For the previous example, the set of r and t is a basic platforming element e_1 and affects the number of molds consumed by die casting process. The additional cost from differentiating such the element mainly includes the expense to make the mold. So, the expected degree for sharing e_1 can be obtained by computing their relative additional cost of tooling one mold to overall additional cost of all platforming elements. For those elements whose varieties result in more than one additional cost from allocated resources in the production flow, the aggregated additional costs ($\sum C^a_j$) are used to reflect the overall influence of element platforming on the production system.

Formulation of Commonality Index

Finally, the formulation of commonality index (CI) for the whole family can be derived according to equation (5.2) with identified set of *basic platforming elements* and associated *expected sharing degree* (ESD). Instead of single variable, platforming elements represent the basic reusing entities under the specific manufacturing environment; ESD denotes their relative contributions of the corresponding elements to cost savings. When incorporated into the stage of family design and optimization as commonality objective, CI can direct the platforming in a desired economical manner as illustrated in Figure 5.7.

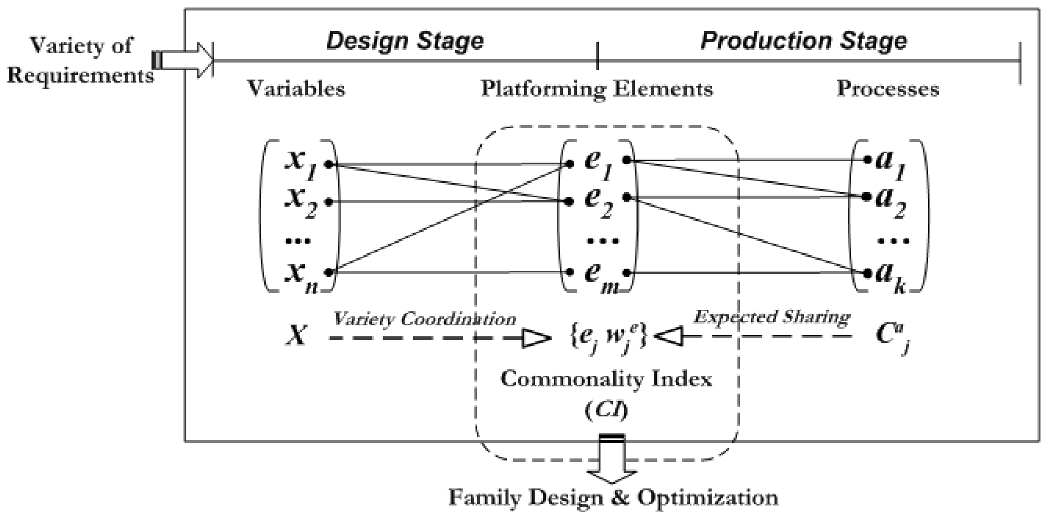


Figure 5.7 Illustration of manufacturing-biased platform decision

However, pre-selection of the common variables, proven to have no impact on performance variation by engineering experience or separated optimization stage, can be included to reduce the computational efforts, especially for a large-scale design problem. Several optimization approaches using robust design principles are available in the literature to investigate those variables "insensitive" to performance variation (Nayak *et al.*, 2002; Sopadang *et al.*, 2001).

5.4.3 Aggregation of Multiple Objectives

Scale-based product family design is a multi-objective optimization problem, involving two different aspects: performance and commonality. For some design, the aspect for performance even includes a weighted set of multiple objectives. Thus, the results of product family are always sensitive to the particular method used for normalizing and aggregating the objectives. In this paper, we use preference based method to aggregate objectives as follows, where preferences are expressed as normalization of performance measures (Dai and Scott, 2006).

$$\begin{aligned} f(X) &= \gamma_P f^P(X) + \gamma_C f^C(X) \\ &= \gamma_P \sum \omega_i \theta_i / p + \gamma_C \sum_{j=1}^{k} w_j{}^e \left(\frac{p - S_j}{p - 1}\right) \end{aligned} \qquad (5.5)$$

where γ_P and γ_C are importance weights for normalized performance and commonality objectives. To access different levels of commonality across the family, we can vary different weights γ_C to the commonality objective. $\sum\omega_i\theta_i/p$ is an aggregated set of averaged preferences or normalized values based on performance priority (ω_i), and $f^C(X)$ is the commonality index.

The performance preference functions θ_i take values from a range of [0, 1], with 0 indicating completely unacceptable and 1 indicating totally satisfactory. They can be derived through normalization of performance measure based on the design targets. For example, the mass objective of family design can be simply normalized as shown in equation (4), where the mass target m_T is determined from individual design since family design will definitely compromise the response of mass. Similarly, safety factor is normalized as a simple linear preference and over-design above pre-specified target SF_T is not expected.

$$\theta_{mass} = \begin{cases} 1; & m \le m_T \\ m_T / m; & otherwise \end{cases} \quad \theta_{SF} = \begin{cases} 1; & SF \ge SF_T \\ (SF-1)/(SF_T-1); & otherwise \end{cases} \qquad (5.6)$$

Although higher weights can be assigned to objectives of higher priority, the weight assignment among performance objectives is problem-dependent and requires a trial-and-error step to generate the expected results. On the other hand, by adjusting the weight ratio among the performance criteria, the weight-based aggregation method to formulate a single overall performance measure provides decision makers strategic opportunities to deliver families of products with different preferences for diverse market environments.

5.5 Discussion

A commonality index is developed to measure the shared or reused elements across the whole family from the perspective of manufacturing efficiency. If a comprehensive cost model exists, it can provide a more explicit insight on what can be achieved from economic efficiency. However, traditional costing method by allocating fixed costs and variable costs across multiple products may produce distorted cost analysis due to possible sunk costs associated with investment into product and process platforms (Jiao *et al.*, 2007d). Meanwhile, the related information, such as manufacturing and inventory cost, is not always available or complete during the early design stage. Although several ABC-based costing methods are developed in the literature to assist in estimating the cost in the context of product family design, there is no uniform agreement on tracing the relationship between product variety and cost.

Focusing on the affected manufacturing activities or processes due to the family design, we only capture the additional cost incurred by varieties of design on the activities or consumed resources. Therefore, the measured commonality across the family can reflect the potential platforming benefit without costing the whole process flow. Another advantage of commonality index is its capability of numerical operation, e.g. easy normalization with range of [0, 1]. On the contrary, the cost model cannot be normalized and aggregated into the multi-objective family design without the pre-specified cost goal. Additionally, in terms of a function with respect to parameter settings, CI can provide a straightforward view on the reuse status of the identified *basic platforming elements*.

Though the proposed platform decision confines the costs and activities to manufacturing aspect, the methodology can be extended into the assessment of other types of costs such as product design and development cost, logistic cost and purchasing cost. Accordingly, the basic platforming elements may not be limited to intra-organizational processes, but intra-organizational activities.

5.6 Summary

This chapter introduces a platform decision strategy to maintain the manufacturing efficiency due to the increasing commonality. Considering that increased commonality implies reduced complexity and cost of manufacturing, we take into account expected sharing degree and sharing pattern with coordination between design and production stages. Then an effective quantification to measure the reuse degree of basic platforming elements across the whole family is developed to capture the level of commonality and reflect the enhanced manufacturing efficiency. Meanwhile, a step-by-step method to design a product family based on scalable platform is given to help designers guide the whole design procedure, involving individual design, platform decision and aggregation of multi-objectives.

Chapter 6 Product Family Design Using a Modified GA-based Optimizer

6.1 Introduction

As discussed in chapter 5, scalable product family design involves two tasks: to minimize performance deviation from the individually tailored design and to maximize measured commonality across the family. This multi-objective optimization not only involves mix-discrete non-linear programming, which is intractable for most general optimization approaches, but also requires a flexible and efficient algorithm to explore design space along the Pareto front and access alternative solutions with varying level of commonality. Genetic algorithm (GA) has been reported in recent literature and appears well suited for optimizing a product platform and the corresponding family members due to the combinatorial nature of the family design problem. Towards this end, this chapter presents the development of a modified GA-based optimizer to solve product family design and optimization.

6.2 Evolutionary Weight Aggregation for Multi-objective Optimization

6.2.1 Non-dominated Solution

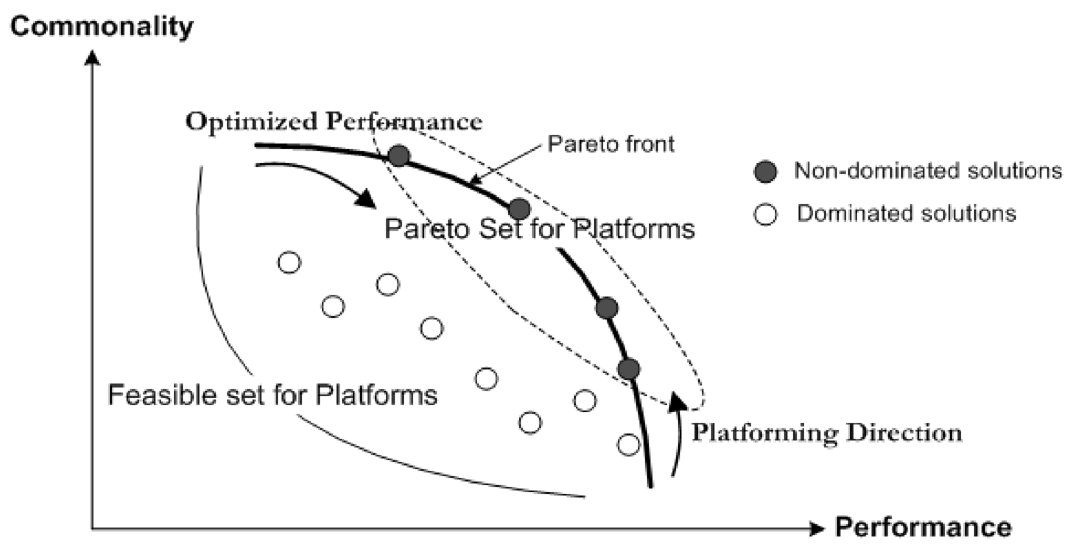


Figure 6.1: Multi-objective optimization for family design

Multi-conflicting-objective optimization (e.g. product family design) tends towards a set of solutions that are not superior to one another according to each objective. These solutions are known as non-dominated solutions or Pareto-optimal solutions, as illustrated in Figure 6.1. The bold curve of these non-dominated solutions is called Pareto front. The rest of solutions are called as dominated solutions.

In a minimization or maximization problem, the non-dominated solutions can be defined as follows. Given a multi-objective optimization problem with m (m>1) objectives, a solution $X^{(1)}$ is said to dominate the other solution $X^{(2)}$ if both the following condition are true:

1) The solution $X^{(1)}$ is no worse than $X^{(2)}$ in all objectives.

2) The solution $X^{(1)}$ is strictly better than $X^{(2)}$ in at least one objective.

If any of the above condition is violated, the solution $X^{(1)}$ does not dominate the solution $X^{(2)}$. Similarly, this concept can be extended to find a non-dominated set of solutions in a population of solutions. Consider a set of N solutions, each having m (m>1) objective function values. The following algorithm is used to find the non-dominated set of solutions:

***Algorithm 1*: Non-dominated Sorting**

Step 0: Begin with i=1.

Step 1: For all $j \neq i$, compare solutions $X^{(i)}$ and $X^{(j)}$ for domination using the above two conditions for all m objectives.

Step 2: If for any j, $X^{(i)}$ is dominated by $X^{(j)}$, mark $X^{(i)}$ as “dominated”. Increment i by one and go to Step 1.

Step 3: If all solutions in the set are considered, go to Step 4; else increment i by one and go to Step 1.

Step 4: All solutions that are not marked “dominated” are non-dominated solutions.

Obviously, $N \times N$ (N^2) iterations of comparison among solutions are required to find non-dominated solutions among N population, if only two objectives are involved. A population of solutions can be classified into groups of different non-dominated levels. When Algorithm 1 is applied for the first time in a population, the generated set is the non-dominated set of first level. To have further classification, these non-dominated solutions can be temporarily omitted from the original set and the algorithm 1 can be applied again to generate non-dominated solutions of second level. This procedure can be continued until all population members are classified into a non-dominated level. Based on this procedure, several non-dominated sorting methods existing in the literature are reported to solve the product family design problem (D’Souza and Simpson, 2003; Akundi and Simpson, 2005). However, the computational expense for non-dominated sorting at all levels is tremendously large, and for the worst case where there exists only one solution at each level, the complexity is $O(mN^3)$, where m is the number of objectives and N is the number of populations. For instance, Simpson and D'Souza (2003) optimize a family of 3 variants with 6 design variables using non-dominated sorting GA. When implemented on the workstation, the execution time is reported to be 6-8 hours. Akundi and Simpson (2005) find that as many as 25,000 generations and a population size of 1500 are need to obtain a good spread solutions for a family of 10 universal motor described by 8 design variables. Although detailed running time has not been published, the computational expense seems unreasonably large.

6.2.2 Dynamic Weighted Aggregation

To make the optimization tractable and efficient, the weight-based aggregation method is adopted to aggregate the multiple objectives into single overall objective function. For product family design, two objectives of performance and commonality are aggregated based on the assigned weight. However, the fixed weight setting always generates one solution during one optimization run. Moreover, the discrete nature of some objectives (e.g. commonality index) imposes certain noisy effect on the exploration of design space so that the optimization may not work effectively (Srinivas and Deb, 1994). To solve such drawbacks, we adopt evolutionary dynamic weighted aggregation (EDWA) method, as proposed by Jin *et al.* (2001). In EDWA the weight for different objectives are changed during optimization so that the optimizer can access all points along the Pareto front. This dynamic weighted method not only avoids the drawback of conventional weighted aggregation, but also maintains the computational expense of exploring multiple non-dominated solutions at the economical level (Jin *et al.*, 2001).

As illustrated in Figure 6.2, the weights $\{w_1(t), w_2(t)\}$ for the two objectives are changed gradually so that the optimizer will go along the Pareto front from one stable non-dominated point to another. However, the dynamic weighted aggregation function need experimental running before it can deliver satisfying results. Meanwhile, it is necessary to record the Pareto solutions that have been found so far.

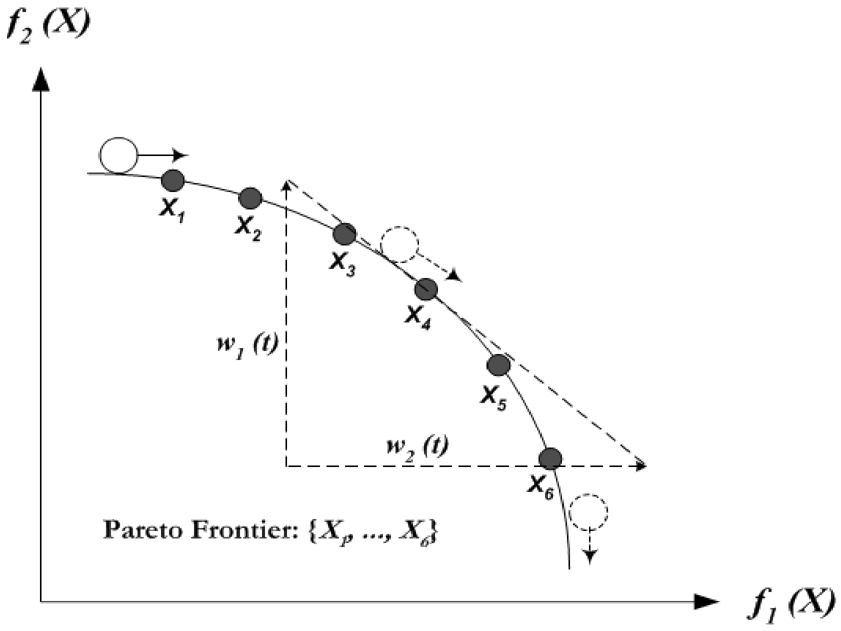
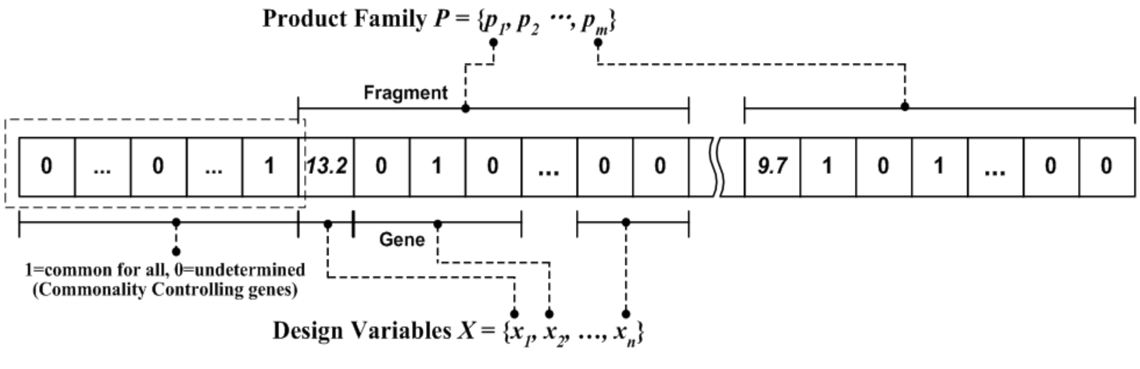


Figure 6.2: Dynamic weighted aggregation toward Pareto front

6.3 GA-based Optimizer for Product Family Design

6.3.1 Generic Coding



Product Family $P = \{p_1, p_2, \cdots, p_m\}$
Fragment
0 … 0 … 1 13.2 0 1 0 … 0 0 9.7 1 0 1 … 0 0
Gene
1=common for all, 0=undetermined
(Commonality Controlling genes)
Design Variables $X = \{x_1, x_2, \ldots, x_n\}$

Figure 6.3: Example of coding scheme for product family

An important procedure in the implementation of GA involves the representation of a problem to be solved with a finite-length string called chromosome. Figure 6.3 shows one example of the structure of a mixed chromosome representing the product family. A chromosome consists of one to many fragments corresponding to the individual product in the family. Every fragment of the chromosome comprises many genes, each of which can assume one possible value of design variables in the product model. Then the fitness for each solution is evaluated according to the single aggregated function, including performance and commonality aspects. Similarly, the individual design can be represented by only one fragment as shown in Figure 6.4 and the fitness evaluation only involves the aggregated performance objectives.

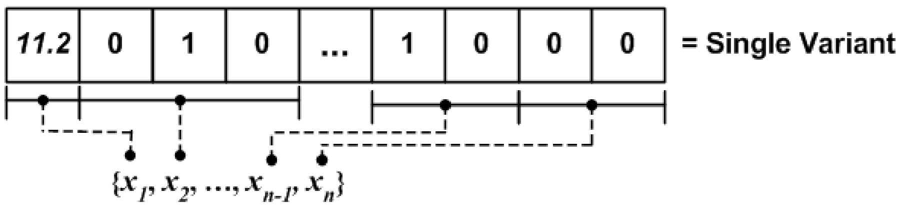


Figure 6.4: Example of generic representation for individual design

Since the optimization in our approach is a mixed-discrete nonlinear problem involving discrete and continuous variables, different coding schemes are used for representation. For the discrete design variable X^d, the possible values can only take from a set of pre-defined values and therefore they are coded as a binary string. The number of bits for coding one variable is determined by the number of possible values for the target variable. If, for example, a variable x_2 has no more than 8 (2^3) possible values, 3 bits are sufficient to represent x_2, as shown in Figure 6.4. For the continuous design variables X^c, we use a floating point representation as it is conceptually closest to the problem space and also allows for easy and efficient implementation of dynamic operators (Michalewicz, 1994).

For those variables proven for effective platform settings from engineering experience, pre-specified controlling genes may be put in front to help determine the sharing status of the corresponding variables during optimization. If one gene takes the value of 1, the corresponding variable is a proven platform variable, and will be shared across the whole family during optimization; otherwise, it is still undetermined and requires the optimization procedure to finalize the sharing status.

6.3.2 Generic Operator: Mutation and Crossover

Crossover and mutation are two main generic operators to realize the population evolution in GA. Crossover can help to explore new regions of the design space by exchanging the digits at a randomly chosen position. Even though selection and crossover effectively search and recombine extant solution, they occasionally lose some useful genetic features. Therefore, mutation is needed to avoid this situation and meanwhile increase the diversity of the population for the global search.

To increase computational efficiency, two-point crossover is adopted, which involves randomly choosing two crossover sites in the mix-coded strings of parents and exchanging the digits between the selected two sites, as illustrated in Figure 6.5.

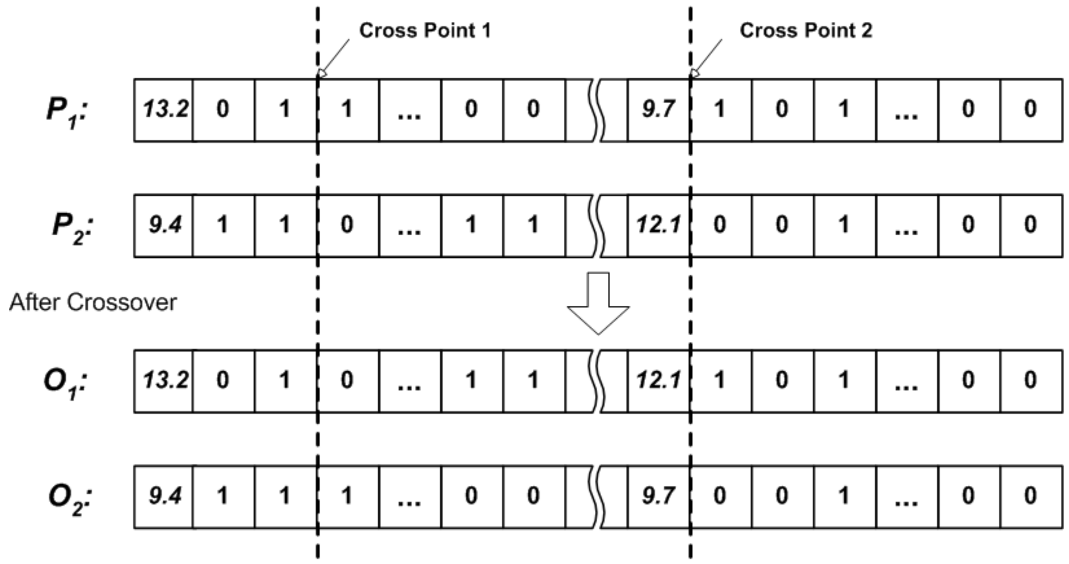

Cross Point 1
Cross Point 2
P1: 13.2 0 1 1 ... 0 0 9.7 1 0 1 ... 0 0
P2: 9.4 1 1 0 ... 1 1 12.1 0 0 1 ... 0 0
After Crossover
O1: 13.2 0 1 0 ... 1 1 12.1 1 0 1 ... 0 0
O2: 9.4 1 1 1 ... 0 0 9.7 0 0 1 ... 0 0

Figure 6.5: Illustration of two-point crossover

After crossover, the mutation operator is applied to the whole population in a specified probability. Since there are two different coding schemes in this work, two different mutation operators are adopted for binary and floating codes, respectively, as shown in Figure 6.6. For binary codes, a simple switch between 0 and 1 is applied; for floating codes, a non-uniform mutation is applied to aim at both improving single-element tuning and reducing the disadvantage of random mutation in the floating point implementation (Michalewicz, 1994). The new operator is defined as follows: if $s^t_v=\{v_1, \ldots, v_m\}$ is a chromosome (t is the generation number and m is the population size) and the element v_k is selected for this mutation, the result is a vector $s^{t+1}_v=\{v_1, \ldots, v'_k, \ldots, v_m\}$, where

$$v_k' = \begin{cases} v_k + \Delta(t, UB - v_k) & if\ a\ random\ digit\ is\ 0 \\ v_k - \Delta(t, v_k - LB) & if\ a\ random\ digit\ is\ 1 \end{cases} \tag{6.1}$$

and UB and LB are upper and lower bounds of the variable v_k. The function $\Delta(t, y)$ returns a value in the range $[0, y]$ such that the probability of $\Delta(t, y)$ being close to 0 increases as t increases. This property causes this operator to search the space uniformly initially when t is small and very locally at later stages, thus increasing probability of generating the new number closer to its successor which is a random choice. We use the following function:

$$\Delta(t, y) = \Delta p \bullet Round(\frac{y}{\Delta p} \bullet (1 - r^{(1-\frac{t}{G})^b})) \tag{6.2}$$

where Δp is the required precision for the floating value; $Round(x)$ returns the nearest integer; r is a random number from [0, 1]; G is the maximal generation number; b is a system parameter determining the degree of dependency on iteration number (Michalewicz, 1994) and we use b=0.5.

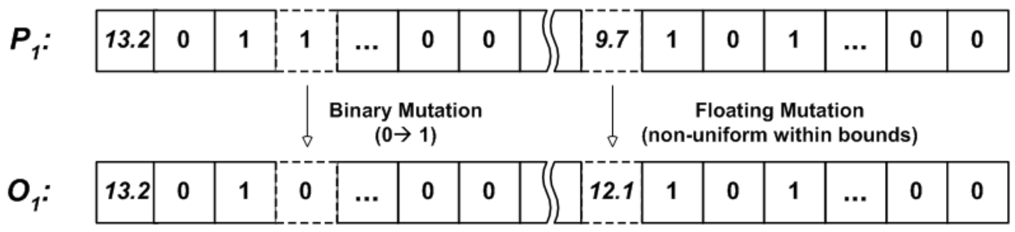


Figure 6.6: Illustration of two mutation operators

6.3.3 Fitness Evaluation

Fitness evaluation of product family design involves two aspects of performance and commonality. Fitness evaluation regarding performance can be easily obtained by averaging the individually normalized product performance according to equation 5.2. One of the challenges is constraint handling, in which genetic operation tends to generate the new chromosome randomly and often yield infeasible offspring. Toward this, we accomplish constraint handling from two aspects of range checking and compatibility. A penalty strategy is implemented after a new generation to penalize the infeasible chromosomes violating certain constraints. This technique does not simply reject the infeasible solutions which may contain much useful information about the optimal solutions and thus acquire a balance between information preservation and selection capability. However, those chromosomes that do not satisfy compatibility constraints are rejected right away and the new chromosomes will be produced and supplemented in the population.

Another remaining challenge lies in the evaluation of commonality index, which requires dynamic calculation of the number of instance for each basic platforming element. Before computing the commonality index for each solution, we need to build the matrix to characterize the relationship between the basic platforming elements and design variables as follows:

$$I = \begin{array}{c} \\ e_1 \\ e_2 \\ \cdots \\ e_m \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & \cdots & x_n \end{array} \\ \begin{bmatrix} I_{11} & I_{12} & \cdots & I_{1n} \\ I_{21} & I_{22} & \cdots & I_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ I_{m1} & I_{m2} & \cdots & I_{mn} \end{bmatrix} \end{array} \tag{6.3}$$

I_{mn} is assigned a value of 1 if there is a relation between variable n and element m; otherwise, I_{mn} is given 0. Therefore, the variable set X^{e_m} related to element m can be described as follows.

$$\begin{bmatrix} X^{e_1} \\ X^{e_2} \\ \dots \\ X^{e_m} \end{bmatrix} = \begin{bmatrix} I_{11} \bullet x_1 & I_{12} \bullet x_2 & \dots & I_{1n} \bullet x_n \\ I_{21} \bullet x_1 & I_{22} \bullet x_2 & \dots & I_{2n} \bullet x_n \\ \dots & \dots & \dots & \dots \\ I_{m1} \bullet x_1 & I_{m2} \bullet x_2 & \dots & I_{mn} \bullet x_n \end{bmatrix}$$

or (6.4)

$$X^{e_m} = [I_{m1} \bullet x_1 \quad I_{m2} \bullet x_2 \quad \dots \quad I_{mn} \bullet x_n]$$

For example, a planetary gear train to be investigated as a case study in the next chapter can be described with variables [*F*, *Z_r*, *Z_s*, *Z_p*, *M_d*, *N_p*] and four basic platforming elements are identified: sun gear (e_1), planet gear (e_2), ring gear (e_3), and planet carrier (e_4). The engineering relation between variables and platforming elements can be found in (Roos and Spiegelber, 2004) and characterized by a matrix as follows.

$$I = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \end{array} \begin{array}{c} \begin{array}{cccccc} F & Z_r & Z_s & Z_p & M_d & N_p \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array} \Rightarrow \begin{bmatrix} X^{e_1} \\ X^{e_2} \\ X^{e_3} \\ X^{e_4} \end{bmatrix} = \begin{bmatrix} F & 0 & Z_s & 0 & M_d & 0 \\ F & 0 & 0 & Z_p & M_d & 0 \\ F & Z_r & 0 & 0 & M_d & 0 \\ 0 & Z_r & 0 & 0 & M_d & N_p \end{bmatrix} \quad (6.5)$$

Based on this matrix representation, each instance for one basic platforming element can be derived by assigning the corresponding parameter values, and then the following algorithm can be used to calculate the number of instances.

***Algorithm 2*: Calculation of the number of instances *N_ins* for the basic platforming element e_i**

Step 0: Begin with *j*=2, *N_ins* =1, *temp_CI* (status variable)=1.

Step 1: If *j* <=*N_p* (family size), begin with *k*=1; else go to Step 6

Step 2: If *k*<=*j*-1, compare instance *j* and *k*; else go to Step 4

Step 3: If same, *temp_CI*=0, increment *j* by 1 and go to step 1; else, *temp_CI*=1, increment *k* by 1 and go to step 2.

Step 4: If *temp_CI*==1, *N_ins* for element *i* increase by 1;

Step 5: Increment *j* by 1 and go to Step 1.

Step 6: The number of instances for element *i* is recorded as *N_in*s.

6.3.4 Selection

To improve the efficiency of computation, a modified elitism-replacement-based strategy is adopted to select the next new generation from the parent and the offspring solutions generated (Michalewicz, 1994). The steps of the algorithm are listed as follows.

***Algorithm 3*: Elitism-Replacement-based selection**

Step 1: Select *t* (10%) chromosomes with best fitness as elitism to be preserved for next generation.

Step 2: Select *r* (40%) parents from current generation based on universal sampling method to be preserved for next generation. Each selected chromosome is marked as applicable to exactly one fixed genetic operation.

Step 3: Selection *pop_size* (the size of population)-*r-2t* distinct chromosomes from current generation and copy them to next generation.

Step 4: Let *r+t* parent chromosomes breed to produce exactly *r+t* offsprings

Step 5: Insert these *r+t* new offsprings into the next generation.

The above selections (Steps 2 and 3) are done according to stochastic universal sampling method (Baker, 1987). The elitism-replacement-based algorithm reserves the best solutions in the current generation and at the same time replaces those solutions with lower fitness with new offspring. Experimental results show the modified elitism-based strategy can acquire more stable and effective results when compared to the traditional approaches only involving evaluation and selection.

6.3.5 Overall Workflow

The overall workflow for the proposed multi-objective optimization is summarized in Figure 6.7. To improve searching efficiency, the optimization program can be run without considering commonality objective and the generated family solutions are collected as seeds for later family design with varying level of commonality.

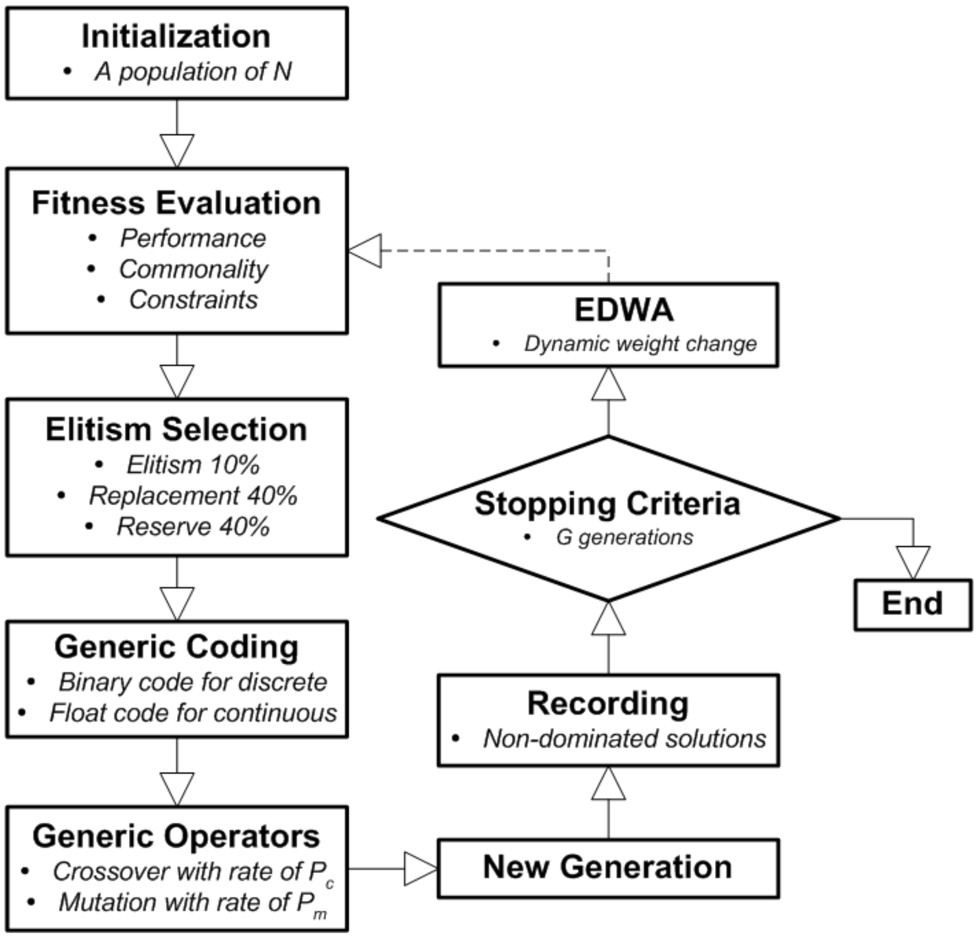


Figure 6.7: Overall procedure of GA-based optimization

6.4 Summary

This chapter presents the development of a GA-based optimizer to assist in solving the multi-objective product family design and optimization problem. To solve the drawback of conventional weighted aggregation and maintain the computational expense at the economical level, we adopt an evolutionary dynamic weighted aggregation method in the GA-based optimization to help explore multiple non-dominated solutions.

In addition, different from the conventional methods with pre-specified platform or optimized single platform, the proposed product family design method determines the platforming direction during the evolving process by using computed commonality index of chromosomes as another objective function. On the other hand, normalized performance functions ensure the performance deviation from benchmark at the acceptable range. Those chromosomes with less performance loss, which also achieve higher index regarding the defined commonality, will have a larger possibility of being preserved from selection. This scheme enables designers to access the optima of multi-platforming settings at one stage.

Chapter 7 Case Study: A family of transmission module design

In this chapter, we illustrate the proposed product family design method using a case study of designing a family of transmission modules for drills/drivers. Given explicitly analytical equations for the planetary gear train and the derived engineering metrics from system-level design phase, the simultaneous design of multiple transmission instances are constructed as a multi-objective optimization model, involving normalized performance and measured commonality.

7.1 Introduction

An industrial example of transmission modules for a family of cordless drills/drivers with different applications, e.g. household, workshop, construction, is used to demonstrate the proposed method. The function of a transmission module is to transmit power and speed (ω_{in}) generated from the motor to the chuck with the output speed (ω_{out}) at a targeted transmission ratio of (ω_{out}/ω_{in}). As shown in an explored view in Figure 7.1, each transmission generally consists of three coupled layers of planetary gears and in each layer, one driving sun gear meshes with several planet gears coupled to planet carrier simultaneously, while these planet gears engage the inside of the ring gear.

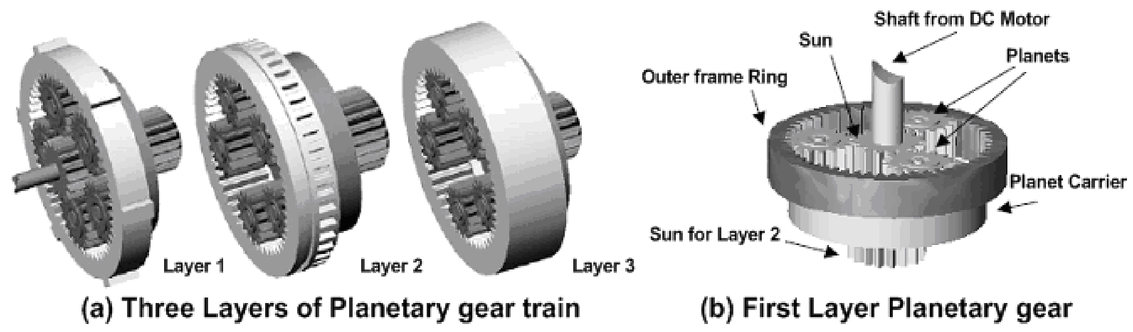


Figure 7.1: Explored structure of planetary gear train (Simpson et al., 2001b)

One of the main factors to distinguish different market niches for drills/drivers is the torque output, ranging from 20 Nm to 50 Nm. Heavier drills transmit more torque and require larger dimensions of gears to meet allowable bending and contact stresses on the meshing teeth of each gear. Although this can be accomplished by increasing the number of planets, modules or the face width of gears, the proliferated components from unplanned individual design will impose undesirable manufacturing complexity and thereby diminish the benefits of providing the additional variety. So, it is very necessary to impose certain platform commonality on the family of planetary gear trains for different drills/drivers to reduce the average production cost and efforts, whilst minimizing performance loss when compared to individual design.

In this case the family of planetary gear transmissions includes 5 variants [v_1, v_2, v_3, v_4 , v_5], which are characterized by different torque outputs ([22, 27, 34, 40, 46] Nm) for the corresponding market niches according to market analysis in Table 4.4. The performance requirements include: minimum mass, targeted transmission ratio, and safety factor (*SF*). The targets and market preferences for all performance criteria are presented in Table 7.1. The transmission mass is designed to be minimized to lower the moment exerted on the user's wrist. The mass target of each variant may be decided by analyzing the counterparts from the highly competitive providers existing in the market. However the related information is not available in this case and the mass targets will be determined from individually optimal design of each product. *SF* for strength requirements is the most important functional characteristic, and targeted at higher value for larger torque output, as shown in Table 7.1, due to the heavy-duty requirement from harsh operating condition and additional hammering function. The transmission ratio is another important performance requirement for drills/drivers to generate certain rotational speed for various applications. In this case, the transmission module is a three-layer planetary gear where the second planetary layer can be engaged/disengaged to provide two different transmission ratios. Generally, cordless drills have 1,400 and 400 rpm for high and low output speeds, and the speed (at no load) of the standard electrical motor is around 20,000 rpm. Then the high to low ratio can be derived to be 14.29 (20,000/1,400) and 50.0 (20,000/400), respectively and the transmission ratio for the three layers may be targeted for 4.29, 3.5, and 3.33 respectively. Detailed ratio derivation can be found in (Gear Manual, 2000).

In this case, multiple performance responses are involved and their relative preference varies in the different market situation. *SF* is the crucial factor as it affects life expectancy of tools and determines the market domination. For a highly competitive market, it should be given highest priority. That means that *SF* should be compromised to the least extent by platform leveraging since family design results in members of product family losing performance to different extent. On the other hand, minimum mass of the gear train is not the best desirable performance characteristic because mass variation of gear train is generally smaller compared to the total mass of the drill. To reflect the relative importance of the respective performance criterion in the market preference, a weighted aggregation method is used in this case to derive a single overall performance measure, and the setting of weight ratio is problem-dependent and a trial-and-error step.

Table 7.1: Performance criteria with target and preference

Performance Criteria	**Family Member (p_i, i=1,2,3,4,5)**					**Preference**
	p_1 (22Nm)	p_2 (27Nm)	p_3 (34Nm)	p_4 (40Nm)	p_5 (46Nm)	
Mass	Based on individual design of each layer					Moderate priority
Safety Factor	1.2	1.2	1.25	1.25	1.3	High priority
Ratio	4.29/3.5/3.33 for layer 1, 2, 3 respectively					Moderate priority

7.2 Individual Design

The application of the proposed method begins with individual design of each gear train. In other cases, this step may be not necessary if the benchmark can be collected from existing counterparts by other providers. Figure 7.2 shows the simplified design model for the planetary gear. The derivation of the mathematical model for mass, *SF* and transmission ratio is given in the Appendix A and detailed formulas can be found in (Roos and Spiegelber, 2004; Gear Manual, 2000). Table 7.2 lists information of 6 design variables in the model for the planetary gear, in which the variable range and type are obtained from experienced designers and (Gear Manual, 2000). Three different types of variables are involved in this case: continuous, integer and discrete. The gear material is assumed to be consistent for the three layers.

Table 7.2: Information of design variables

Variables	**Description**	**Range and type**
F	Face width	1.5 mm $\leq F \leq$15 mm, continuous
Z_s	Num. of teeth on sun gear	10 $\leq Z_s \leq$25, integer
Z_r	Num. of teeth on ring gear	37 $\leq Z_r \leq$52, integer
Z_p	Num. of teeth on planet gear	6 $\leq Z_p \leq$21, integer
M_d^*	Gear module	[0.6, 0.7, 0.8, 0.9]mm, discrete
N_p	Number of planet gears	3$\leq N_p \leq$5, integer

*Gear module describes the gear size and differs from the module extensively used in this thesis

The individual design model for the planetary gear train is shown in Figure 7.2. In summary, for each layer there are six variables, five constraints, and a single overall objective based on three aggregated performance responses. Dimension constraint (g_1) imposes the outer diameter of ring gear to be within certain range to ensure that the ring gear of each layer can be assembled into the gear housing within required clearance. Relation (g_2) and assembly constraints (g_3) provide the acceptable conditions for assembly. Interference constraint (g_4) ensures no interference occurring during operation. Strength constraint (g_5) makes sure that the basic requirements of bending and contact stresses can be met. Detailed explanation of constraints can be referred in (Gear Manual, 2000). The gear material is powdered alloy steel for all layers. To aggregate the three performance objectives into an overall measure, weight ratio needs to be assigned to reflect their relatively importance. After trial test of different weight combinations on the design results, a ratio of (0.2/0.5/0/3) for mass, *SF* and ratio objectives, respectively, is applied to maintain high priority for *SF*. Meanwhile, estimated values of less than the possible minimum mass are assigned as the mass targets for each individual design.

Given :

Material Selection : *Powdered Alloy Steel*

$\sigma_{F,all} = 500MPa,\ \sigma_{H,all} = 1500MPa,\ \rho = 7800kg/m^3$

Find :

$X = \left[x_1, x_2, x_3, x_4, x_5, x_6\right]^T = \left[F, Z_r, Z_s, Z_p, m, N_p\right]^T$

subject to :

$\left| r_{\text{target}} - m \times Z_r \right| \le r_{dev}$ *... Dimension const.*

$Z_r - Z_s = 2Z_p$ *... Relation const.*

$(Z_r + Z_s)/N_p = \text{int}$ *... Assembly const.*

$(Z_p + Z_s) \times \sin(\pi / N_p) > Z_p + 2$ *... Interference const.*

$\sigma_{F,all} \ge \sigma_{F,sun};\ \sigma_{F,all} {*} 0.7 \ge \sigma_{F,planet};\ \sigma_{H,all} \ge \sigma_H$ *... Strength const.*

Maximize :

$f(F, Z_r, Z_s, Z_p, m, N_p) = \omega_1 \theta_{mass} + \omega_2 \theta_{SF} + \omega_3 \theta_{ratio}$

Figure 7.2: Simplified design model for planetary gear train

The individual design is implemented with GA-based programming. Based on preliminary tests, a population size of 200, a mutation rate of 0.08, a crossover rate of 0.7, and a maximum generation limit of 400 are adopted during optimization. When run on the IBM T43 notebook with the 1.86 GHz processor, execution time for the whole GA optimization is approximately 30 seconds, allowing us to analyze the results very quickly. Figure 7.3 shows one running examples of individual design. GA optimizer converges and returns the optimal result after 100 generations.

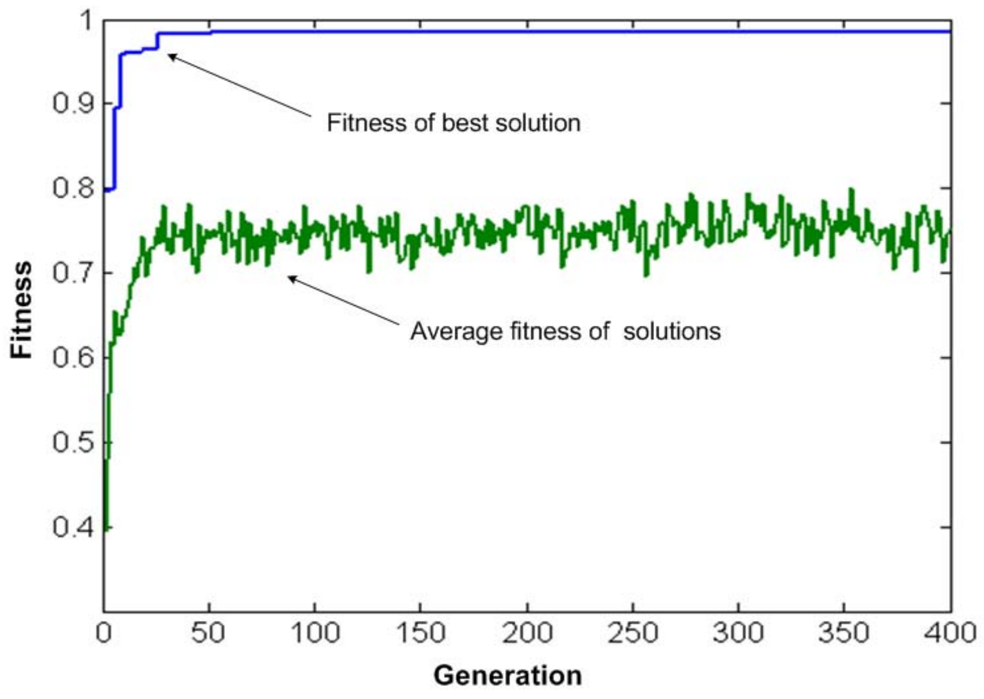


Figure7.3: Running samples of individual deign

Table 7.3 shows the results of the individual design of each layer for each gear train, which is solved 15 times (5 different torque requirements in 3 layers) by genetic algorithm. Note that not much commonality exists in the variable settings. Accordingly in each layer both *SF* and transmission ratios reach the design targets, and each variant achieves the lowest mass while meeting constraints. Also obvious is that the average mass of layer 3 plays a dominant part in the whole train due to the augmented torque output exerted on this layer. After totalizing the three layers of each transmission, we derive a line of products with gradually increased weights ([67.1, 75.9, 96.8, 109.0, 132.6] g) and with completely targeted requirements of *SF* and transmission ratio.

Table 7.3: Results of individual design (benchmark)

Variant		Design Variables						Performance		
		F	Z_s	Z_r	Z_p	M_d	N_p	*m*(g)	SF	Ratio
Layer 1	v_1^1	1.5	14	46	16	0.6	3	10.5	1.32	4.29
	v_2^1	1.6	14	46	16	0.6	3	10.6	1.20	-
	v_3^1	1.6	14	46	16	0.6	4	11.6	1.25	-
	v_4^1	1.9	14	46	16	0.6	4	12.6	1.25	-
	v_5^1	2.3	14	46	16	0.6	4	14.2	1.30	-
Layer 2	v_1^2	2.8	16	40	12	0.7	4	19.0	1.20	3.50
	v_2^2	2.4	20	50	15	0.6	5	21.4	1.20	-
	v_3^2	3.3	20	50	15	0.6	5	25.3	1.25	-
	v_4^2	5.5	16	40	12	0.7	4	28.4	1.25	-
	v_5^2	6.9	16	40	12	0.7	4	33.0	1.30	-
Layer 3	v_1^3	9.0	18	42	12	0.6	5	37.6	1.20	3.33
	V_2^3	11.0	18	42	12	0.6	5	43.8	1.20	-
	v_3^3	11.1	21	49	14	0.6	5	59.9	1.25	-
	v_4^3	13.0	18	42	12	0.7	5	68.0	1.25	-
	v_5^3	12.4	18	42	12	0.8	5	85.5	1.30	-
Totals:	v_1							67.1	* Meet SF requirements	
	v_2							75.9		
	v_3							96.8		
	v_4							109.0	* High Ra.=14.29 Low Ra=50	
	v_5							132.6		

Further observation of results shows that the main difference of variable settings in each layer is the various combinations of different face width (*F*), module (M_d), and number of planets (N_p). The range of face width changes from 1.5mm to 12.4mm in order to meet stress requirements. The module also varies from uniform setting of 0.6mm for layer 1 to mix of [0.6, 0.7, 0.8] mm for layer 2 and 3. The tooth numbers of the planet, sun, and ring gears (Z_s, Z_r, Z_p) appear quite similar since the requirements of transmission ratio for all products are the same and completely determined by the function regarding Z_s, Z_r, Z_p. Engineering experience from gear designers also provides a strong support that the three variables (*F*, M_d, and N_p) are very effective scales that can be varied to meet different torque requirements; on the other hand, other variables (Z_s, Z_r, Z_p) have no impact on performance variation and can be settled into platform setting.

7.3 Platform Decision

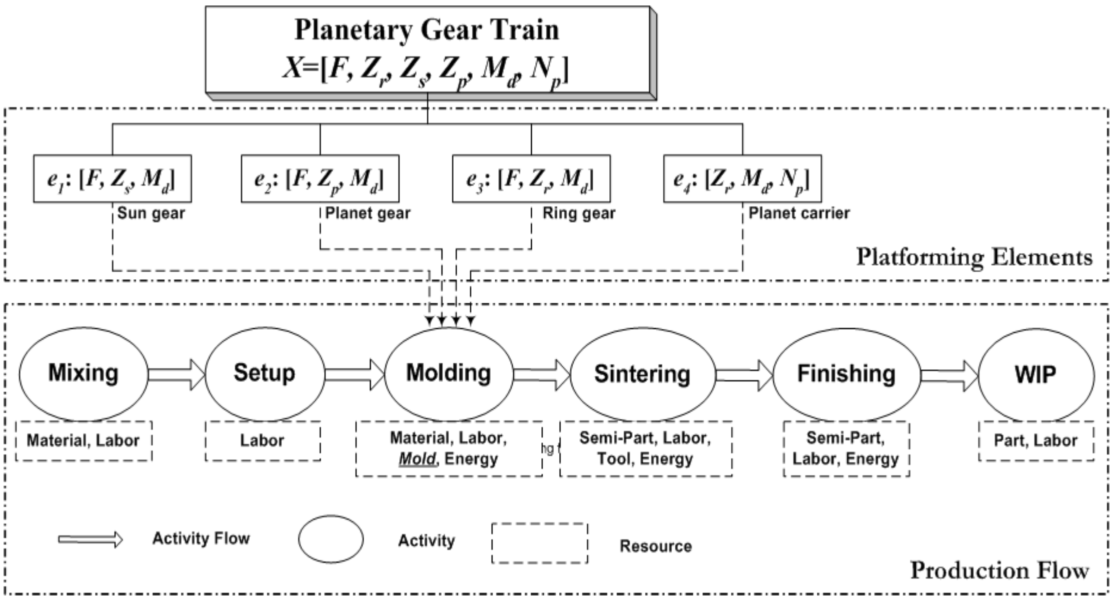


Figure 7.4: A scheme of the influence of platforming elements on processes

The task of platform decision is to determine the basic platforming elements from views of manufacturing efficiency. Beforehand, an explicit analysis of production flow for planetary gears is required to fully understand the platforming effect on the manufacturing cost. The investigated planetary gear train is produced using powder injection molding method and the process flow is modeled in Figure 7.4 based on ABC principle. Several processes are required to fulfill the production of gear train; however, the molding process is most affected by family design and accordingly the greatest platforming benefit will be the reduced tooling cost by reusing molds among the family members. Typically, each layer has four basic parts to be manufactured. Although the sun gears in layers 2 and 3 and planet carriers in layers 1 and 2 are compound parts, we still view them as basic parts to avoid design confusion. If the dimensions of one part are all identical, the same mold can be reused and the tooling cost will be reduced. Otherwise, partially sharing dimensions for one component still generate various combinations and require different molds to be tooled. As stated previously in chapter 5, simply using variable sharing cannot guarantee the expected benefits from platform settings, and therefore, the basic platforming elements in this case are sun gear (e_1), planet gear (e_2), ring gear (e_3), and planet carrier (e_4). Other cost savings based on platform design may come from inventory management and experimental prototyping. However these indirect benefits are relatively less significant compared to tooling cost savings and are not considered in this example. The design variables are mapped into vectors of variables corresponding to the four basic elements as follows:

$X^{e_1}=[x_1^{e_1}, x_2^{e_1}, x_5^{e_1}]^T=[F, Z_s, M_d]^T$

$X^{e_2}=[x_1^{e_2}, x_4^{e_2}, x_5^{e_2}]^T=[F, Z_p, M_d]^T$

$X^{e_3}=[x_1^{e_3}, x_3^{e_3}, x_5^{e_3}]^T=[F, Z_r, M_d]^T$

$X^{e_4}=[x_3^{e_3}, x_5^{e_4} , x_6^{e_4}]^T=[Z_r, M_d, N_p]^T$

The ESD (w_i^e) can be determined from variety cost of resource since one more instance of platforming element will incur certain additional cost of affected process. Table 7.4 shows the normalized ESD for the four basic platforming elements. In this case the additional cost simply comes from the molds to be tooled and corresponding tooling cost for the basic parts is collected based on historic data from the investigated company. For a more complicated production system, the variety cost for certain platforming element may stem from different processes and require aggregation of the additional costs of consumed resources to reflect its overall platforming effect on production.

Table 7.4: ESD for the *basic platforming elements*

Platforming elements ($e=[e_1, e_2, \ldots]$)	**Additional Cost** ($C^a_{j,\, j=1\ldots k}$)	**Normalized Weight** (w_i^e)
Sun gear (***e**₁*)	5,000 *RMB*	5,000/28,000=0.18
Planet gear (***e**₂*)	5,000 *RMB*	5,000/28,000=0.18
Ring gear (***e**₃*)	10,000 *RMB*	10,000/28,000=0.35
Planet carrier (***e**₄*)	8,000 *RMB*	8,000/28,000=0.19
Total	28,000 *RMB*	1

In addition, to reduce the computational efforts, Z_s, Z_r, Z_p are pre-selected as common variables to be shared by all variants since they have relatively little impact on performance variation. The following equation (7.1) is the formulated function of commonality index regarding the four basic platforming elements, and will be employed during optimization to calculate the commonality level across the family.

$$\begin{aligned}\mu_C &= \sum_{j=1}^{k} w_j^{\,e} \left(\frac{p - S_j}{p-1}\right) \\ &= \frac{0.18\cdot(5-S_1) + 0.18\cdot(5-S_2) + 0.35\cdot(5-S_3) + 0.19\cdot(5-S_4)}{4}\end{aligned} \tag{7.1}$$

where $S_{i,\ i=1,2,3,4}$ denotes the number of instances for the basic platforming elements and will be computed by comparing vector of variable values related to the target element among the product family.

7.4 Aggregation of Multiple Objectives

The model for the entire product family can be achieved by incorporating the commonality index as another objective. Equation (7.2) is the aggregated family design model with averaged performance normalization and commonality index.

$$\begin{aligned}
&Find \quad X=(Z_s,Z_p,Z_r,F^1,\ldots,F^5,M_d{}^1,\ldots,M_d{}^5,N_p{}^1,\ldots,N_p{}^5)\\
&Max.\ \ f(X)=\gamma_P(\omega_1\sum\theta_{mass}{}^i+\omega_2\sum\theta_{SF}{}^i+\omega_3\sum\theta_{ratio}i)/p+\gamma_C\mu_C\\
&s.t.\\
&\theta_{mass}{}^i=m_T{}^i/m^i, if\ m^i\ge m_T{}^i;\ otherwise\ \theta_{mass}{}^i=1\\
&\theta_{SF}{}^i=(SF^i-1)/(SF_T{}^i-1), if\ SF^i\le SF_T{}^i;\ otherwise\ \theta_{SF}{}^i=1\\
&\theta_{ratio}{}^i=1-\left|Ra_T{}^i-Ra^i\right|/(0.1\cdot Ra_T{}^i),\ if\ \left|Ra_T{}^i-Ra^i\right|\le 0.1\cdot Ra_T{}^i;\\
&otherwise\ \theta_{ratio}{}^i=0
\end{aligned} \tag{7.2}$$

where $\theta_{mass}{}^i$, $\theta_{SF}{}^i$, and $\theta_{ratio}{}^i$ are the preference function to normalized mass, *SF*, and transmission ratio (*Ra*) based on the design target as shown in Figure 7.5. $m_T{}^i$ is the mass target derived from individual design results for product *i*. $SF_T{}^i$, $Ra_T{}^i$ is the respective target for *SF* and ratio, and will be fixed for all variants according to Table 7.1. The preference function for mass indicates that any mass less than or equal to target from individual design is considered completely satisfactory with a preference of 1. *SF* is normalized in similar manner. A triangular function is used to normalize the preference for ratio with a tolerance range of 10% deviation from the target based engineering experience. In this case we only adopt simple or linear preference functions for simplicity. More sophisticated functions could be used to precisely reflect performance satisfaction with respect to performance output.

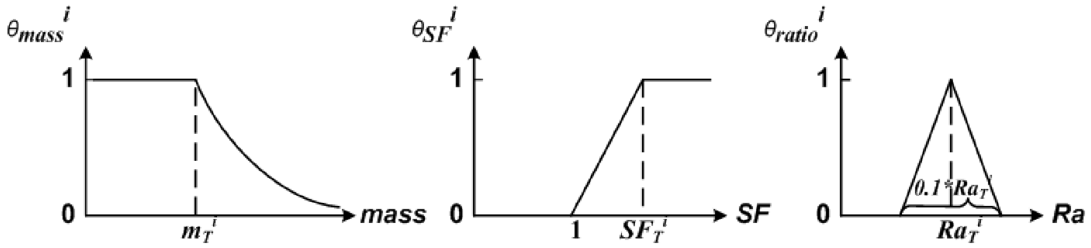


Figure 7.5: Preference functions for performance normalization

7.5 Family Design using GA

The family design and optimization for each layer is implemented in a Matlab® environment with the modified genetic algorithm, as detailed previously in chapter 6.

The aggregated two-objective function, including commonality and performance aspects, will provide an overall fitness value for solution selection. Here a maximum number of 1000 generations is specified as the criterion for termination. Based on the experimental test, a mutation rate of 0.08 and a crossover rate of 0.7 are used during optimization. For the family size in this case, a population size of 400 is found to give computational effectiveness, beyond which further improvement is marginal. Initially, the optimization program is run without considering commonality objective (γ_C=0) and the generated results are collected as seeds for later family design to improve searching efficiency.

Multi-objective optimization using the conventionally weighted aggregation is conceptually straightforward and computationally efficient. However, there are some drawbacks which hamper accessing all Pareto fronts in the design space. For example, only one Pareto solution can be achieved from one single run of optimization; the discrete nature of some objectives (e.g. commonality index) might restrict the searching direction into the local optima (Jin *et al.*, 2001). To overcome these drawbacks, the evolutionary dynamic weight aggregation method can be used to explore all the solutions along the Pareto front. It is suggested the change of weight should be smooth to allow the optimizer to move from one stable point to another (Jin *et al.*, 2001). Based on trial test, it is found that a dynamic change of weights can be realized in the following way for the two-objective optimization:

$$\begin{aligned} \gamma_P(t) &= (1 - t/G)^{1/2} \\ \gamma_C(t) &= 1 - (1 - t/G)^{1/2} \end{aligned} \tag{7.3}$$

where t is the generation index; G is the maximum number of generations (1000) to be run as stopping criterion. γ_P and γ_C are dynamically weighted functions for performance and commonality objectives, respectively.

Figure 7.6 shows the running examples of product family design based GA-optimizer. Implemented on the same running platform with individual design, family design requires more computational resource and execution time as much as 12 minutes. In view of the discrete nature of measured commonality index, the fitness curve regarding commonality displays ladder-shape increase during evolutionary optimization. On the other hand, the normalized performance fitness decreases with increasing generations because of its gradually lower weight assignment.

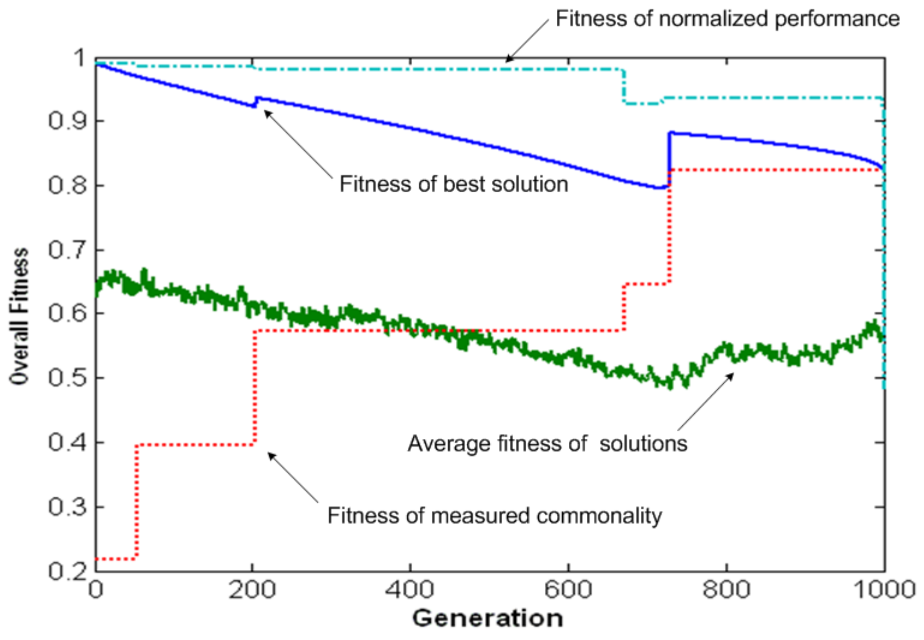
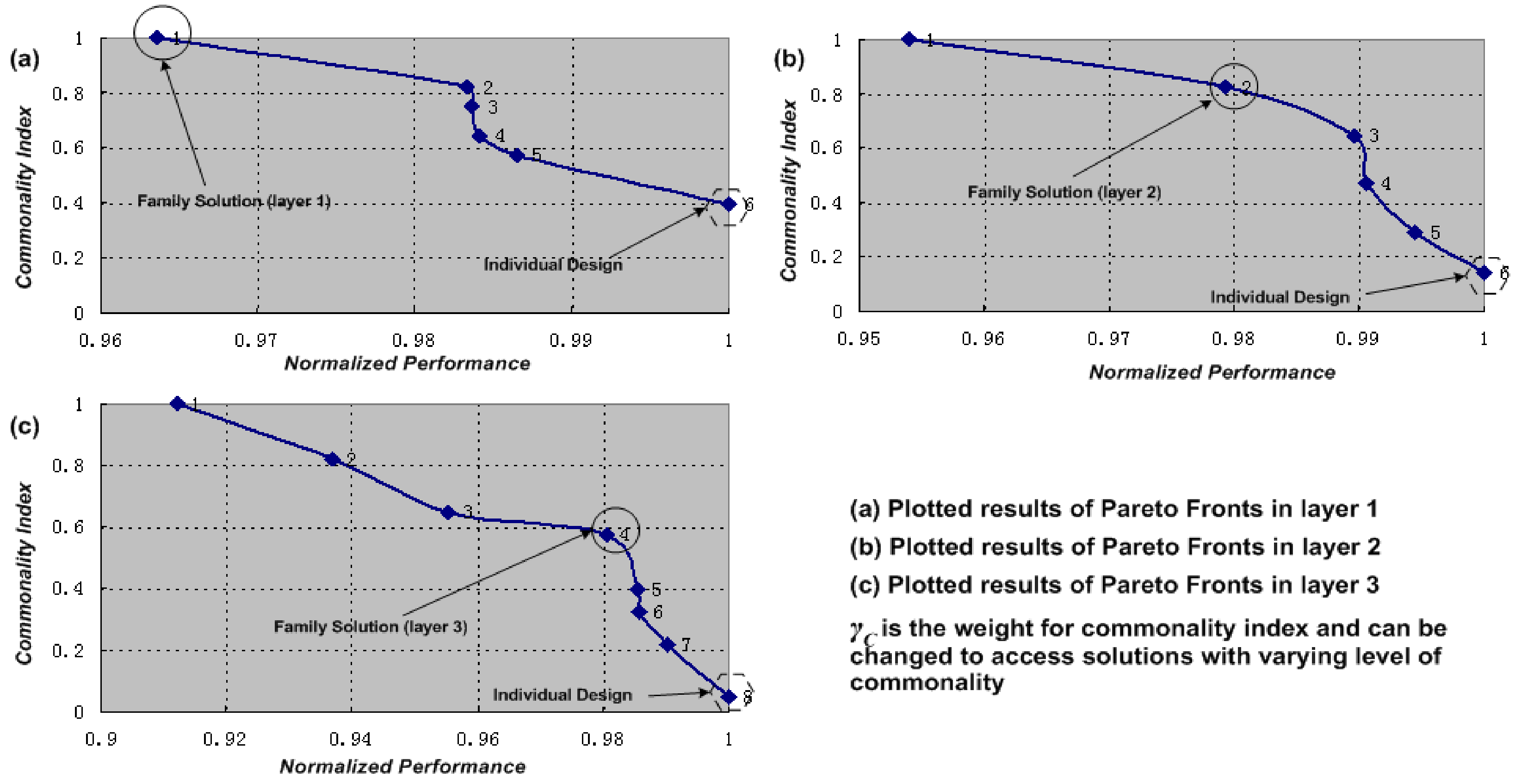


Figure 7.6 Running sample of product family design

Due to the nature of the multi-objective problem, the GA optimizer generates some non-dominated solutions, none of which is superior to each other regarding both normalized performance criteria and commonality index. Thus, during optimization it is necessary to archive all non-dominated Pareto solutions. The detailed data of all the non-dominated solutions found during optimization (3 times running) can be referred at Appendix B and the plotted results for each layer regarding commonality index and normalized performance are shown in Figure 7.7.



(a) Plotted results of Pareto Fronts in layer 1

(b) Plotted results of Pareto Fronts in layer 2

(c) Plotted results of Pareto Fronts in layer 3

γ_C is the weight for commonality index and can be changed to access solutions with varying level of commonality

Figure 7.7: Plotted family solutions with varying level of commonality

With the plotted tradeoff charts in Figure 7.7, the engineering designers can be provided with various platform-based families with different level of commonality. However, the final decision involves mutual coordination from market and manufacturing, and cannot be achieved in a quantitative manner so far. The desirable family solutions should reduce average production cost for effective margin of platform settings while reducing the impact (e.g. market demands, overall product performance) of performance deviation to benchmark. In this case, the three layers of gear train are viewed as three different products and the decision of family solution can be accomplished individually. The first layer with small torque exerted on it requires small dimensions and mass to meet stress requirements. For a part with mass less than 15 g, the tooling cost is always a dominant factor (German, 2003). On the other hand, performance loss resulting from family design mainly lies in the increased mass due to the assigned high priority for *SF*. However, compared to the whole transmission system, mass of layer 1 is much smaller that higher performance loss is acceptable from market view. Accordingly, the family solution at point 1, as shown in Figure 7.7 (a), is picked to provide one unique setting for all variant in the family. The selected product family offers 60% increase in defined commonality over the benchmark with only 3.64% performance deviation from targets. Similarly, the family solution for layer 2 can be picked at point 2, as shown in Figure 7.7 (b), to provide two different settings for the whole family since the increased mass augments the proportion of material cost and reduces the acceptable extent of performance deviation. Layer 3 transmits the largest torque and the increased mass plays the most important role in both overall cost and performance aspects. Thus the decision needs careful balance between performance loss and increased commonality. In Figure 7.7 (c), the commonality decreases rapidly from

60% at point 4 to about 5% at point 8 while the performance only loses about 2%; however, increasing commonality above 60% causes performance to decrease very rapidly. Thus the family solution for layer 3 can be picked at point 4 for the effective margin of commonality. The final family specification for each layer (point 1, 2, and 4, respectively) with multi-platforming configuration is tabulated as follows.

Table 7.5: Specification of multi-platforming family design

Variant		**Design Variables**						**Performance**		
		F	Z_s	Z_r	Z_p	M_d	N_p	***m*(g)**	**SF**	**Ra.**
Layer 1	v_1^1	2.3	14	46	16	0.6	4	14.2	1.88	4.29
	v_2^1	-	-	-	-	-	-	-	1.70	-
	v_3^1	-	-	-	-	-	-	-	1.51	-
	v_4^1	-	-	-	-	-	-	-	1.39	-
	v_5^1	-	-	-	-	-	-	-	1.30	-
Layer 2	v_1^2	3.5	16	40	12	0.7	4	21.2	1.33	3.50
	v_2^2	3.5	-	-	-	-	-	21.2	1.20	-
	v_3^2	6.9	-	-	-	-	-	33.0	1.51	-
	v_4^2	6.9	-	-	-	-	-	33.0	1.39	-
	v_5^2	6.9	-	-	-	-	-	33.0	1.30	-
Layer 3	v_1^3	8.1	18	42	12	0.7	5	47.5	1.33	3.33
	V_2^3	8.1	-	-	-	0.7	5	47.5	1.20	-
	v_3^3	13.0	-	-	-	0.7	5	68.0	1.36	-
	v_4^3	13.0	-	-	-	0.7	5	68.0	1.25	-
	v_5^3	12.4	-	-	-	0.8	5	85.5	1.30	-
Totals:	v_1							82.9	* Meet SF requirements	
	v_2							82.9		
	v_3							115.2		
	v_4							115.2	* High Ra.=14.29 Low Ra=50	
	v_5							132.6		

Direct observation of parameter configuration in Table 7.5 indicates that there is one unique setting for layer 1 and thus all the products have one instance. For layer 2, *F* is treated as the scalable variable and there are two different settings for the family. For layer 3, more variable settings with scalable *F* and M_d are generated, and three settings are needed to differentiate the individual product. A further comparison with result from individual design, as shown in Table 7.6, finds that MPPF design achieves or exceeds the targets of *SF* and transmission ratio due to assigned higher priorities and can be considered to be equivalent to the benchmark; however, each individual product has a mass increase with different extents (23.5%~5.4%) except variant v_5 designed for the heaviest drill. Obviously, this multi-platform configuration results in the mass increase of each variant due to over-design of lower-end products, but achieves a higher level of commonality in the design space and makes it possible to reduce production cost by maintaining economies of scale.

Family solutions are characterized by certain performance deviation from individually tailored design and increased commonality or reduced variety across family. However, based on different perspectives, commonality/variety encompasses different implications and thus exhibits evolutionary track during the family design. Figure 7.8 depicts three different scenarios of the average performance deviation with respect to: (a) measured commonality index, (b) number of variable instances, and (c) number of part instances. For example, in terms of the same number of variable instances as shown in Figure 7.8 (b), family solutions 6 and 7 result in different number of part instances, as shown in Figure 7.8 (c), due to the coupled relationship between design domain (variables) and physical entities (parts). Similarly, family solutions 4 and 5 have the same number of part instances but different measured commonality since the proposed index incorporates cost factors. In other words, inconsistent results of family design may be generated if the commonality is defined in a different way.

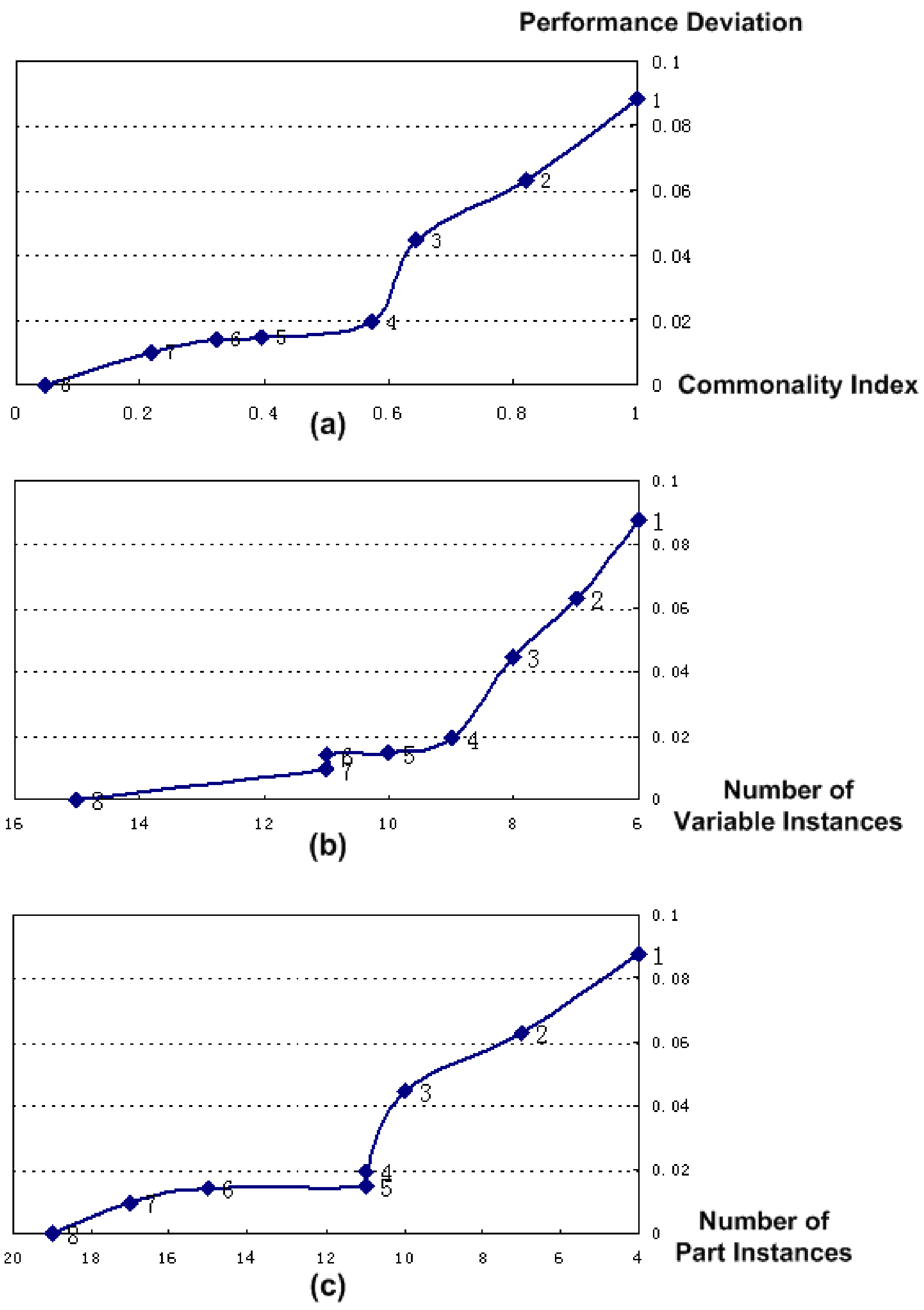


Figure 7.8: Performance deviation in layer 3 with respect to commonality (a), number of variable instances (b), and number of part instances (c)

Table 7.6: Performance comparison of non-platform and platform designs

v_i	**Mass (m)**			**Safety Factor**			**Transmission Ratio**		
	Individual Design	Family Design	Difference	Individual Design	Family Design	Difference	Individual Design	Family Design	Differenc e
v_1	67.1	82.9	23.5%	1.32/1.20/1.20	1.88/1.33/1.33	Equiv.	4.29/3.5/3.33	4.29/3.5/3.33	Equiv.
v_2	75.9	82.9	9.2%	1.20/1.20/1.20	1.70/1.20/1.20	-	-	-	-
v_3	96.8	115.2	19.0%	1.25/1.25/1.25	1.51/1.51/1.36	-	-	-	-
v_4	109.0	115.2	5.7%	1.25/1.25/1.25	1.39/1.39/1.25	-	-	-	-
v_5	132.6	132.6	0	1.30/1.30/1.30	1.30/1.30/1.30	-	-	-	-

Table 7.7: Results of families with pre-specified single platforms

V_i	**Design Variables**																		**Performance**		
	Layer 1						**Layer 2**						**Layer 3**						m**(g)**	**SF**	**Ra.**
	F	Z_s	Z_r	Z_p	M_d	N_p	F	Z_s	Z_r	Z_p	M_d	N_p	F	Z_s	Z_r	Z_p	M_d	N_p			
Product Family scaled around F																					
v_1	2.3	14	46	16	0.6	4	6.9	16	40	12	0.7	4	*5.0*	18	42	12	0.8	5	92.5	* Meet SF requirements * High Ra.=14.29 Low Ra.=50	
v_2	-	-	-	-	-	-	-	-	-	-	-	-	*6.2*	-	-	-	-	-	98.8		
v_3	-	-	-	-	-	-	-	-	-	-	-	-	*8.5*	-	-	-	-	-	111.2		
v_4	-	-	-	-	-	-	-	-	-	-	-	-	*10.0*	-	-	-	-	-	119.4		
v_5	-	-	-	-	-	-	-	-	-	-	-	-	*12.4*	-	-	-	-	-	132.6		
Product Family scaled around N_p																					
v_1	2.3	14	46	16	0.6	4	6.9	16	40	12	0.7	4	12.4	18	42	12	0.8	*3*	118.7	* Meet SF requirements * High Ra.=14.29 Low Ra.=50	
v_2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	*3*	118.7		
v_3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	*4*	125.7		
v_4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	*4*	125.7		
v_5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	*5*	132.6		

7.6 Verification and Discussion

The result from proposed MPPF approach is verified by comparing with the ones based on the Product Platform Concept Exploration Method (PPCEM) (Simpson *et al.*, 2001a). In their method, the platform variables are pre-selected as the scale factors from which the family members are derived. As suggested by Simpson (2001b), the two families with different scaling factors, N_p and F respectively, are tested. Similarly, we implement the optimization with GA and Table 7.7 shows the final results. For each family, it is suggested that layers 1 and 2 should use the same set of variable configuration since their mass is much smaller compared to layer 3. In layer 3, one variable, F and N_p respectively, is identified as a scale factor that can be varied to meet the individual performance requirements for each variant while keeping other variables common across the family.

Considering the computational expense, the pre-specified single platform only explores the optimum regarding the performance objective. Thus it needs a smaller number of populations in GA optimization and relatively less computational time, compared to the MPPF method. With regard to the performance aspects, both *SF* and ratio achieve their targets in the two family solutions due to the high priorities for *SF* and transmission ratio. The consequence is that the individual product exhibits non-uniform performance loss in mass due to much over-design of low-end products in the family as shown in Figure 7.9. Compared to the family scaled on F, the solution scaled on N_p is found to have more mass deviation from the benchmark. The reason behind this may be that increasing number of planets for large torque based on platform design is not an effective method to maintain compact dimensions of gear train.

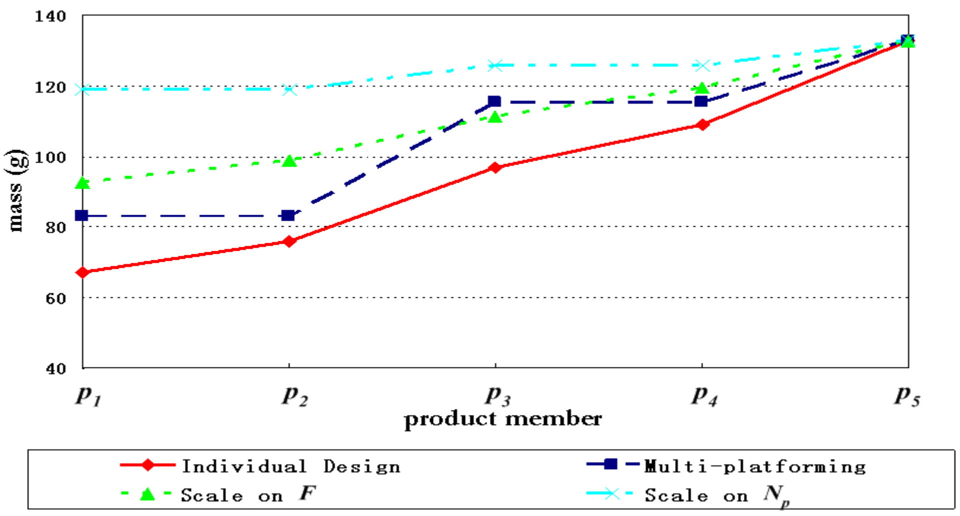


Figure 7.9: Mass comparison among the four family solutions

Family design will generate less variety of parts to be produced. On the other hand, it can result in the over-design of dimensions of low-end products and thus increase the average material cost for each variant. Thus, it is necessary to evaluate the family design and its effect on production cost. Here we do not attempt to model the full cost of manufacturing planetary gear, but only investigate some important cost factors: powder, tooling, and overhead cost, which are potentially affected by the family design. The tooling cost (C_T) for one set of molds including planet/sun gear, carrier and ring gear is approximately 28,000 *RMB* based on a case study of power tool company in China. Here we assume the product volume (Q) across the family is uniform and equal to 50,000 for each product. Molding cost (C_M) is a fixed part of final cost since molding process has not affected by family design. Then the average cost for a family of products can be estimated using equation (7.4).

$$C = C_P \cdot m_{aver} + (N_T \cdot C_T + C_O) / Q + C_M$$

where C_p is the material cost based on China market and m_{aver} refers to the average mass for a family of products; N_T is the number of tooling sets; C_O is the overhead cost and can be approximated from tooling cost.

Table 7.8: Cost comparison among individual and family designs

Scheme	**Manufacturing Cost Factors**			**Cost Comparison** (Molding cost is fixed and 2.5 *RMB*/piece)
	Powder cost (C_P=0.04g/*RMB*)	**Tooling cost** (C_T=28,000/set)	**Overhead cost** (C_O)	
Individual Design	[67.1, 75.9, 96.8, 109.0, 132.6]g Average 3.85 *RMB*/piece	15 sets of molds to be tooled Average 1.68 *RMB*/piece	* More parts increase experiment, prototyping, and inventory cost * inverse proportional to production volume * Approximately 4% more tooling cost increase for one more set of molds	Average 9.00 *RMB*/piece
Multi-platforming Family Design	[82.9, 82.9, 115.2, 115.2, 132.6]g Average 4.23 *RMB*/piece	6 sets of molds to be tooled Average 0.67 *RMB*/piece		Average 7.56 *RMB*/piece (16%) decrease
Family Design scaled around *F*	[92.5, 98.8, 111.2, 119.4, 132.6]g Average 4.44 *RMB*/piece	7 sets of molds to be tooled Average 0.78 *RMB*/ piece		Average 7.94*RMB*/piece (12%) decrease
Family Design scaled around N_p	[118.7, 118.7, 125.7, 125.7, 132.6]g Average 4.97 *RMB*/piece	3 sets of molds to be tooled Average 0.34 *RMB*/Piece		Average 7.85 *RMB*/piece (13%) decrease

Table 7.8 shows the cost comparison of individual and the proposed family designs. It is estimated that, despite the material cost increase with larger dimensions, MPPF design can still reduce the production cost per piece for a line of transmissions by approximately 16% from tooling and overhead cost saving. When the other two families with the pre-specified platforms are taken into comparison, we find that the family with scale F is not the expected effective method to construct such a line. This is attributed to the fact that the sharing of variable F cannot generate the expected cost saving from tooling stage since the face width is coupled with other variables (e.g. Z_s, Z_r, M_d) to jointly determine the dimensions of parts (sun, ring, and planet gears) to be tooled. The family with scale N_p generates most cost saving from tooling since the number of planets (N_p) is hardly coupled with other variables to affect the component dimensions. However, it will require much extra cost in material and consequently, this platform is also not a best choice for the product family in our case. But when the production volume for each product is reduced to 10,000 as shown in Figure10, this platform scaled around N_p will provide a most economical solution to meet various requirements since tooling cost of other platforms accounts for more percentage of production cost in a low volume. Otherwise, when the production volume increases to 200,000 units for each product, individual design requires lowest cost due to material saving from individually optimized design.

Furthermore, we can find that production quantity (Q) and material cost (C_P) are two important factors affecting the appropriate use of family design. Smaller quantity of production needs certain component reuse to share the tooling and overhead cost, and move approximately the economical point to the direction of family design with higher level of commonality as shown in Figure 7.10; whereas, larger quantity can stabilize the production cost and enable individual design to be a better choice. On the other hand, the higher material cost will depreciate the use of family design because of extra expense for over-design of low-end products.

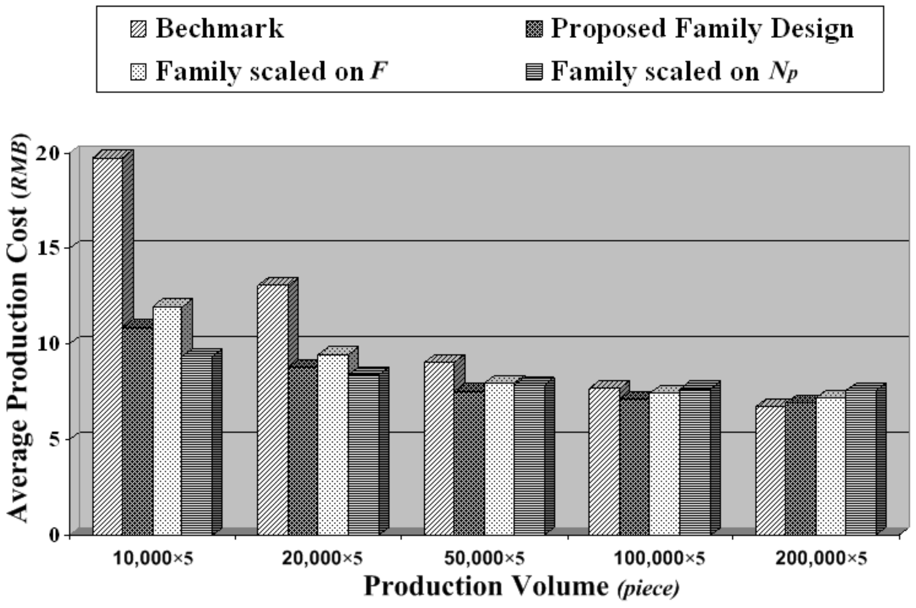


Figure 7.10: Comparison of the average production cost by four methods with varying volume

7.7 Summary

Through design of planetary gear transmission for power tool family, the proposed method for scale-based platform and product family design has provided an effective means to balance the tradeoff between commonality across the family and performance loss. Compared to the traditional scalable product family design, the proposed MPPF approach incorporates analysis of the platforming effect on the production cost and thus enables manufacturers to deliver a variety of products in cost-effectively manner. Meanwhile, the measured commonality index provides designers opportunity to explore alternative family solutions with varying level of commonality for various market demands.

Although this study has dealt with detail design of commonality as multi-objective optimization problem, successful implementation of scalable platform for practical application involves a few factors (e.g. availability of analytical or simulation model). It would be desirable to develop an effective and efficient optimizer with standard interface to easily build and solve different family design problems. In addition, how to integrate the proposed family design approach into computer-aided support system (e.g. CAD, CAE) need to be addressed in the future study.

Chapter 8 Conclusion

This chapters gives a summary of the methods presented in this dissertation, discusses the contribution of research, and proposes several recommendations for future work,

8.1 Conclusion

The main objective in this research is to provide a systematic top-down method to design a family of platform-based products based an umbrella of *modularity* and *commonality.*

Effective product family design begins with a systematic modularization procedure of product architecture in the context of mass customization to achieve modularity. The proposed method begins with product family planning, and then adopts the functional modeling method to identify module boundary. To estimate variance degree of each module due to different customer requirements, *Variety index* is developed at the early stage of product family development by establishing the attribute-module relation from two perspectives: specification implementation and specification propagation. Finally, instance specifications for all modules are derived and integrated to form product portfolio architecture, which provides an engineering view of variety generation and develops targets for further product family design.

Commonality is another important issue at the detailed design stage of module instantiation, which aims to concurrently design a series of product members through scaling platform settings. The proposed method for scale-based platform and product family design has provided an effective means to balance the tradeoff between commonality across the family and performance loss. A manufacturing-biased platform decision strategy is developed to measure the level of commonality as commonality index, which is aggregated into the family design with normalized performance objective. To solve this multi-objective optimization problem, a modified GA optimizer is developed.

Although both modularity and commonality are recognized as two essential issues for platform development, the implementation of the proposed framework can be scaled down to fit reality of target design. For highly standardized design, the platforming direction may be confined to modular aspect of product architecture; for highly integrated design, only commonality may be the desirable issues.

8.2 Contributions

The contribution of the proposed platform-based family design approach can be captured as follows:

8.2.1 Modularity Analysis for Variety Generation

Recognizing the necessity to explore the customization effect on product architecture, this study investigates the architectural robustness of product family architecture by extending the concept of the traditional product architecture and modeling the product family architecture (PFA) as a conceptual structure with three important interrelated elements: module, variant and coupling interface. Variants in terms of different customer requirements act as the external drivers of architectural variation, which is then propagated within the product architecture through module interaction. Within this framework, a step-by-step method to systematically modularize the *PF*A has been proposed. Rather than just identification of the module boundary, the proposed modularization methods translates the variety source generated from requirements analysis into a dynamic configuration of the conceptual PFA, involving variety analysis, functional modularization and generation of product portfolio architecture (PPA). The PPA provides an engineering insight to understand product variety in terms of conceptual module configuration and meanwhile develops the targets for the further development of product family.

8.2.2 Manufacturing-biased Platform Decision

In the light of the basic premise that increased commonality implies reduced complexity and cost of manufacturing, we present an effective commonality decision strategy to help dynamically determine the shared elements and thus generate an economical platform configuration. By coordinating design with production stages, we first propose the concept of the basic platforming elements, whose reuse will reduce manufacturing complexity and thus improve economic benefit. Then the proposed strategy takes into account *expected sharing degree* to reflect their contribution of those elements to cost savings and adopts an effective quantification to measure the reuse degree of platforming elements across the whole family as commonality index (CI). The proposed CI can serve as an efficiency indicator of reduced manufacturing complexity or cost savings from family design, and also as a means to control the level of commonality and assure the platforming direction in a desired economical manner when incorporated in the family design.

8.2.3 Effective GA-based Optimizer for Product Family Design

Unlike most existing methods that assume a given single platform, our research attempts to address the multi-platforming configuration across the family by incorporating the quantified level of commonality into the family design and optimization. The proposed method not only yields better solutions through more effective exploration of design space, but also achieves alternative product family settings with different levels of commonality.

The product family design problem with two conflicting objectives: performance and commonality, is normalized and aggregated into one single overall function. A modified genetic algorithm is developed to solve the mixed-type optimization problem. The evolutionary weighted aggregation is adopted to dynamically change the weight assignment among objectives so that the optimizer can access all points along the Pareto front. This dynamic weighted method not only avoids the drawback of conventional weighted aggregation, but also maintains the computational expense at the economical level.

8.3 Recommendation for Future Work

While the methods developed in this thesis can assist engineering designers in the development of product family, there are still several opportunities for further investigation and improvement.

8.3.1 Interface Design for PFA

Although this study highlights some important issues regarding the variety analysis and modularization of the conceptual product family architecture, there are still some limitations that need to be solved in future research. For example, interface can be another important aspect in product family architecture. How to evaluate or quantify the degree of coupling interfaces existing among the modules and their effect on product family development may be a challenge for further research.

8.3.2 Integration of Market Research

It is always assumed that the reduced performance loss from family design and optimization increases customer satisfaction or sales, but a more realistic model should take into account some other factors (e.g. volume, distribution) existing in the market and allow designers to gain a comprehensive insight into issues regarding family design. As illustrated in the case study of transmission module, the demand volume determines the final evaluation of family design and in turn affects the decision on the desired level of commonality. Moreover, demand distribution information would help designers focus on particular sets of products. Those products that have a higher volume of sales or are sensitive to performance variation should be given more attention or favor with respect to their performance. However, such information as volume distribution and performance preference is not available in this case study, so a more systematic study is not possible at this stage. Therefore, future work should improve on the decision model and incorporate the cross-functional information consisting of design, manufacturing, and marketing into the more precise modeling of family design to realize a successful product platform.

8.3.3 Improvement of Computational Efficiency

Improvement of the computational efficiency for a large-scale family design problem can be included in future studies to make optimization tractable. Although GA-based optimizer provides an effective means to access the non-dominated Pareto front, the design space may expand increasingly when the size of product family or number of design variables increases so that one-stage method cannot deal with the complexity and computational expense. Instead, two-stage approaches seem well suitable for optimizing a product platform and the corresponding family members separately. Detailed implementation of the new optimization procedure may be investigated in the future work.

Reference:

Akundi, V.K., Simpson, T.W., Reed, P.M., 2005, Multi-objective design optimization for product platform and product family design using genetic algorithms, *ASME Design Engineering Technical Conference*, Long Beach, California, USA.

Alizon, F., Khadke, K., Thevenot, H.J., Gershenson, J.K., Marion, T.J., Shooter, S.B., Simpson, T.W., 2007, Frameworks for product family design and development, *Concurrent Engineering: Research and Applications*, 15(2): 187-199.

Anderson, D.M., 1997, *Agile Product Development for Mass Customization*, Irwin Professional Pub., Chicago.

Baker, J.E., 1987. Reducing bias and inefficiency in the selection algorithm, in *Proceedings of the Second International Conference on Genetic Algorithm and their Application*, Hillsdale, USA.

Baldwin, C.Y., Clark, K.B., 2000, *Design rules*, MIT Press, Cambridge.

Blecker, T., Abdelkafi, N., 2007, The development of a component commonality metric for mass customization, *IEEE Transactions on Engineering Management*, 54(1): 70-85.

Chakravarty, A.K., Balakrishnan, N., 2001, Achieving product variety through optimal choice of module variations, *IIE Transactions*, 33: 587-598.

Chandrasekaran, B., Stone, R.B., Mcadams, D.A., 2004, Developing design templates for product family focused design, *Journal of Engineering Design*, 15(3): 209-228.

Chen, K.M., Liu, R.J., 2005, Interface strategies in modular product innovation, *Technovation*, 25: 771-782.

Cutherell, D., 1996, Product Architecture, The PDMA handbook of new product development, Rosenau, M., *et al.*, (Edited), Wiley, New Yoke.

Da Cunha, C., Agard, B., Kusiak, A., 2007, Design for cost: module-based mass customization, *IEEE Transactions on Automation Science and Engineering*, To be pressed.

Dahmus, J.B., Gonzalez-Zugasti, J.P., Otto, K.N., 2001, Modular product architecture, *Design Studies*, 22: 409 – 424.

Dan, B., Tseng, M.M., 2007, Assessing the inherent flexibility of product families for meeting customization requirements, *International Journal of Manufacturing Technology and Management*, 10(2-3): 227-246.

Dai, Z., Scott, M.J., 2003, Meaningful tradeoffs in product family design considering monetary and technical aspects of commonality, *ASME Design Engineering Technical Conference*, Long Beach, USA.

Dai, Z., Scott, M.J., 2006, Effective product family design using preference aggregation, *Journal of Mechanical Design*, 128: 659-667.

Dai, Z., Scott, M.J., 2007, Product platform design through sensitivity analysis and cluster analysis, *Journal of Intelligent Manufacturing*, 18: 97-113.

De Lit, P., Delchambre, A., Henrioud, J., 2003, An integrated approach for product family and assembly system design, *IEEE Transactions on Robotics and Automation*, 19(2):324-334.

D'Souza, B., Simpson, T.W., 2003, A genetic algorithm based method for product family design optimization, *Engineering Optimization*, 35(1): 1-18.

Du, X., Jiao, J., Tseng, M.M., 2001, Architecture of product family: fundamentals and methodology, *Concurrent Engineering: Research and Application*, 9(4): 309 – 325.

Erens, F., Verhulst, K, 1997, Architectures for product families, *Computers in*

Industry, 33: 165-178.

Feitzinger, E., Lee, H.L., 1997, Mass customization at Hewlett Packard: the power of postponement, *Harvard Business Review*, 75(1): 116-121.

Fellini, R., Kokkolaras, M., Michelena, N., Papalambros, P., Perez-Duarte, A., Saitou, K., Fenyes, P., 2004, A sensitivity-based commonality strategy for family products of mild variation, with application to automotive body structures, *Structural and Multidisciplinary* Optimization, 27: 89-96.

Fellini, R., Kokkolaras, M., Papalambros, P., Perez-Duarte, A., 2005, Platform selection under performance bounds in optimal design of product families, *Transactions of the ASME*, 127: 524-535.

Fellini, R., Kokkolaras, M., Papalambros, P., 2006, Quantitative platform selection in optimal design of product families, with application to automotive engine design, *Journal of Engineering Design*, 17(5): 429-446.

Fixson, K., 2003, The multiple faces of modularity: a literature analysis of a product concept form assembled hardware products, technical report, available at http://ioe.engin.umich.edu/techrprt/pdf/TR03-05.pdf.

Fixson, K., 2005, Product architecture assessment: a tool to link product, process and supply chain design decisions, *Journal of Operations Management*, 23: 345 – 369.

Franceschini, F., Rupil, A., 1999, Rating scales and prioritization in QFD, *International Journal of Quality & Reliability Management*, 16(1): 85-97.

Fujita, K., 2002, Product variety optimization under modular, *Computer-Aided Design*, 34:953-965.

Fujita, K., Yoshida, H., 2004, Product variety optimization simultaneously designing module combination and module attributes, *Concurrent Engineering: Research* *and Application*, 12(2): 105-118.

Gear Manual, 2000, China Machine Press.

German, R.M., 2003, *Powder Injection Molding: Design and Applications*, Innovative Material Solutions Stage College, PA.

Gershenson, J.K., Prasad, G.J., Allamneni, S., 1999, Modular product design: a life-cycle view, *Transactions of the Society for Design and Process Science*, 3(4): 13-26.

Gershenson, J.K., Prasad, G.J., Zhang, Y., 2003, Product modularity: definitions and benefits, *Journal of Engineering Design*, 14(3): 295-313.

Gershenson, J.K., Prasad, G.J., Zhang, Y., 2004, Product modularity: measures and design methods, *Journal of Engineering Design*, 15(1): 33-51.

Gonzalez-Zugasti, J.P., 2000, Models for platform-based product family design, Ph.D. thesis, Massachusetts Institute of Technology.

Gu, P., Sosale, S., 1999, Product modularization for life cycle engineering, *Robotics and Computer Integrated Manufacturing*, 15: 387-401.

Gupta, S., Krishnan, V., 1999, Integrated component and supplier selection for a product family, Production and Operations Management, 8(2): 163-182.

Halman, I.M., Hofer, A.P., Vuuren, W.V., 2003, Platform-driven development of product families: linking theory with practice, *Journal of Product Innovation Management*, 20: 149-162.

Hernandez, G., Allen, J.K., Woodruff, G.W., Simpson, T.W., Bascaran, E., Avila, L.F., Salinas, F., 2001, Robust design of families of products with production modeling and evaluation, *Journal of Mechanical Design*, 123: 183-190.

Hernandez, G., Allen, J.K., Mistree, F., 2003, Platform design for customizable products as a problem of access in geometric space, *Engineering Optimization*, 35(3): 229-254.

Holtta, K.M., Otto, K.N., 2005, Incorporating design effort complexity measures in product architectural design and assessment, *Design Studies*, 26: 463-485.

Hsiao, S.W., Liu, E., 2005, A structural component-based approach for designing product family, *Computers in Industry*, 56: 13-28.

Huang, C., Kusiak, A., 1998, Modularity in design of products and systems, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 28(1): 66-77.

Huang, G.Q., Li, L., Cheng, X., 2007, A tandem evolutionary algorithm for platform product customization, *Journal of Computing and Information Science in Engineering*, 7: 151-159.

Huffman, C., Kahn, B., 1998, Variety for sale: mass customization or mass confusion? *Journal of Retailing*, 74(4): 491-513.

Hundal, M.S., 1997, Product costing: a comparison of conventional and activity-based costing methods, *Journal of Engineering Design*, 8(1): 91-103.

Jiang, L., Allada, V., 2005, Robust modular product family design using a modified Taguchi method, *Journal of Engineering Design*, 15(5): 443-458.

Jiao, J., Tseng, M.M., 1999, A method of developing product family architecture for mass customization, *Journal of Intelligent Manufacturing*, 10: 3-20.

Jiao, J., Tseng, M.M., 2000, Fundamentals of product family architecture, *Integrated Manufacturing Systems*, 11(7): 469-483.

Jiao, J., Tseng, M.M., 2004, Customizability analysis in design for mass customization, *Computer-Aided Design*, 36: 745-757.

Jiao, J., Zhang, Y., Wang, Y., 2005, Product portfolio planning with customer-engineering interaction, *IIE Transactions*, 37(9): 801-814.

Jiao, J., Zhang, Y., Wang, Y., 2007a, A heuristic genetic algorithm for product portfolio planning, *Computers and Operation Research*, 34: 1777-1799.

Jiao, J., Zhang, Y., Wang, Y., 2007b, A generic genetic algorithm for product family design, *Journal of Intelligent Manufacturing*, 18: 233-247.

Jiao, J., Zhang, L., Pokharel, S., 2007c, Process platform planning for variety coordination from design to production in mass customization manufacturing, *IEEE Transactions on Engineering Management*, 54(1): 112-129.

Jiao, J., Simpson, T.W., Siddique, Z., 2007d, Product family design and platform-based product development: a stage-of-the-art review, *Journal of Intelligent Manufacturing*, 18: 5-29.

Jin, Y., Olhofer, M., Sendhoff, B., 2001, Evolutionary dynamic weighted aggregation for multiobjective optimization: Why does it work and how? *Genetic and Evolutionary Computation Conference*, San Francisco, CA.

Jose, A., Tollenaere, M., 2005, Modular and platform methods for product family design: literature analysis, *Journal of Intelligent Manufacturing*, 16: 371-390.

Khajavirad, A., Michalek, J.J., Simpson, T.W., 2007, A decomposed genetic algorithm for solving the joint product family optimization problem, *3rd AIAA Multidisciplinary Design Optimization Specialist Conference*, Honolulu, USA.

Kreng, V.B., Lee, T., 2004, QFD-based modular product design with linear integer programming-a case study, *Journal of Engineering Design*, 15(3):261-284.

Kumar, R., Allada, V., 2007, Scalable platforms using ant colony optimization, *Journal of Intelligent Manufacturing*, 18: 127-142.

Kurtadikar, R.M., Stone, R.B., Van Wie, M.J., McAdams, D.A., 2004, A customer needs motivated conceptual design methodology for product portfolio, *ASME Design Engineering Technical Conference*, Salt Lake City, USA.

Kusiak, A., Huang C., 1996, Development of modular products, *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, 19(4): 523-538.

Kusiak, A., 2002, Integrated product and process design: a modularity perspective, *Journal of Engineering Design*, 13(3): 223-231.

Lamothe, J., Hadj-Hamou, K., Aldanondo, M., 2006, An optimization model for selection a product and designing its supply chain, *European Journal of Operational Research*, 169(3): 1030-1047.

Li, H., Azarm, S., 2002, An approach for product line design selection under uncertainty and competition, *Journal of Mechanical Design*, 124: 385-392.

Li, L., Huang, G.Q., Newman, S.T., 2007, Interweaving genetic programming and genetic algorithm for structural and parametric optimization in adaptive platform product customization, *Robotics and Computer-Integrated Manufacturing*, 23: 650-658.

Maier, J.R.A., Fadel, G.M., 2007, A taxonomy and decision support for the design and manufacture of types product families, *Journal of Intelligent Manufacturing*, 18: 31-45.

Martin, M.V, Ishii, K., 2002, Design for variety: developing standardized and modularized product family architectures, *Research in Engineering Design*, 13: 213 – 235.

Messac, A., Martinez, M.P., Simpson, T.W., 2002a, Introduction of a product family penalty function using physical programming, *Transactions of the ASME*, 124: 164-172.

Messac, A., Martinez, M.P., Simpson, T.W., 2002b, Effective product family design using physical programming, *Engineering Optimization*, 34: 245-261.

Meyer, M.H., Lehnerd, A.P., 1997, *The Power of Product Platform: Building Value* *and Cost Leadership*, Free Pressed, New York.

Michalewicz, Z., 1994, *Genetic Algorithms+Data Structures= Evolution Programs*, Springer-Verlag, Berlin.

Mikkola, J.H., Oliver, G., 2003, Managing modularity of product architectures: toward an integrated theory, *IEEE Transactions on Engineering Management*, 50(2): 204-218.

Moon, S.K., Kumara, S.R.T., Simpson, T.W., 2006, Data mining and fuzzy clustering to support product family design, *ASME Design Engineering Technical Conference*, Philadephia, USA.

Moon, S.K., Park, J., Simpson, T.W., Kumara, R.T., 2007, A dynamic multi-agent system based on a negotiation mechanism for product family design, *IEEE Transactions on Automation Science and Engineering*, In Press.

Moore, W.L., Louviere, J.J., Verma, R., 1999, Using conjoint analysis to help design product platforms, *Journal of Product Innovation Management*, 16: 27-39.

Muffatto, M., Roveda, M., 2002, Product architecture and platforms: a conceptual framework, *International Journal of Technology Management*, 24(1): 1-16.

Nayak, R., Chen, W., Simpson, T.W., 2002, A variation-based method for product family design, *Engineering Optimization*, 34(1): 65-81.

Nelson, S.A., Parkinson, M.B., Papalambros, P.Y., 2001, Multi-criteria optimization in product platform design, *Journal of Mechanical Design*, 123: 199-204.

Newcomb, P.J., Bras, B., Rosen, D.W., 1998, Implications of modularity on product design for life cycle, *Journal of Mechanical Design*, 120: 483-490.

Pahl, G., Beitz, W., 1996, *Engineering Design: A systematic approach*, Design Council, London.

Park, J., Simpson, T.W., 2005, Development of a production cost estimation framework to support product family design, *International Journal of Production Research*, 43(4): 731-772.

Pimmler, T.U., Eppinger, S.D., 1994, Integration analysis of product decompositions, *ASME Design Theory and Methodology Conference*, Minneapolis, USA.

Rai, R., Allada, V., 2003, Modular product family design: agent-based Pareto-optimization and quality loss function-based post-optimal analysis, *International Journal of Production Research*, 41(17): 4075-4098.

Robertson, D., Ulrich, K., 1998, Planning for product platforms, Sloan Management Review, 39(4): 19-31.

Roos, F., Spiegelberg, C., 2004, Relations between size and gear ratio in spur and planetary gear trains, Technical Report, Available at: http://www.md.kth.se/~fredrikr/AM2D/gearReport.pdf.

Sand, J.C., Gu, P., Watson, G., 2002, HOME: house of modular enhancement- a tool for modular product redesign, *Concurrent Engineering: Research and Applications*, 10(2): 153-164.

Sanderson, S.W., Uzumeri, M., 1997, *Managing Product Families*, Irwin Professional Pub., Chicago.

Scott, M.J., Simpson, T.W., Allada, V., 2006, Towards a suite of problems for comparison of product platform design methods: a proposed classification, *ASME Design Engineering Technical Conference*, Philadephia, USA.

Simpson, T.W., Maier, J., Mistree, F., 2001a, Product family design: method and application, *Research in Engineering Design*, 12: 2-22.

Simpson, T.W., Seepersad, C.C., Mistree, F., 2001b, Balancing commonality and performance within the concurrent design of multiple products in a product family, *Concurrent Engineering: Research and Application*, 9(3): 177-190.

Simpson, T.W., D'Souza, B., 2004a, Assessing variable levels of platform commonality within a product family using a multi-objective genetic algorithm, *Concurrent Engineering: Research and Application*, 12(2): 119-129.

Simpson, T.W., 2004b, Product platform design and customization: status and promise, *AI EDAM*, 18(1): 3-20.

Simpson, T.W., Siddique, Z., Jiao J., 2006 *Product Platform and Product Family Design: Methods and Applications*, Springer, New York.

Sopadang, A., Cho, B.R., Lenard, M.S., 2001, Development of a scaling factor identification method using design of experiments for product-family-based product and process design, *Quality Engineering*, 14(2): 319-329.

Sosa, M.E., Eppinger, S.D., Rowles, C.M., 2000, Designing modular and integrative systems, *ASME Design Engineering Technical Conference*, Baltimore, USA.

Srinivas, N., Deb, K., 1994, Multiobjective optimization using non-dominated sorting in genetic algorithm, *Evolutionary Computation.*, 2(3): 221-248.

Stone, R., Wood, K., 2000a, Development of a Functional Basis for Design, *Journal of Mechanical Design*, 122(4): 359-370.

Stone, R., Wood, K., Crawford, R., 2000b, A heuristic method for identifying modules for product architectures, *Design Studies*, 21: 5 – 31.

Stone, R., Wood, K., Crawford, R., 2000c, Using quantitative functional models to develop product architectures, Design Studies, 21: 239-260.

Suh, N.P., 1990, *The Principles of Design*, Oxford University Press, New York.

Suh, N.P., 2001, *Axiomatic Design: Advances and Application*, Oxford University Press, New York.

Thevenot, H.J., Simpson, T.W., 2006, Commonality indices for product family design: a detailed comparison, *Journal of Engineering Design*, 17(2): 99-119.

Ulrich, K., 1995, The role of product architecture in the manufacturing firm, *Research Policy*, 24: 419 – 440.

Ulrich, K.T., Eppinger, S.D., 2000, *Product Design and Development*, Mc-Graw-Hill, Boston.

Van Wie, M., Greer J. L., Campbell, M.I, Stone, R.B., Wood, K.L., 2001, Interfaces and product architecture, *ASME Design Engineering Technical Conference*, Pittsburgh, Pennsylvania, USA.

Van Wie, M., Rajan, P., 2003, Representing product architecture, *ASME Design Engineering Technical Conference*, Chicago, Illinois, USA.

Williams, C.B., Allen, J.K., Rosen, D., Mistree, F., 2007, Designing platforms for customizable products and processes in markets of non-uniform demand, *Concurrent Engineering: Research and Applications*, 15(2): 201-216.

Yigit, A.S., Allahverdi, A., 2003, Optimal selection of module instances for modular products in reconfigurable manufacturing systems, *International Journal of Production Research*, 41(17): 4063-4074.

Yu, J.S., Gonzalez-Zugasti, J.P., Otto, K.N., 1999, Product architecture definition based upon customer demands, *Journal of Mechanical Design*, 121: 329-335.

Zacharias, N.A., Yasshine, A.A., 2008, Optimal platform investment for product family design, *Journal of Intelligent Manufacturing*, 19: 131-148.

Zhang, M., Tseng, M.M., 2007, A product and process modeling based approach to study cost implications of product variety in mass customization, *IEEE Transactions on Engineering Management*, 54(1): 130-144.

Zhang, W.Y., Tor, S.Y., Britton, G.A., 2006, Managing modularity in product family design with functional modeling, *International Journal of Advanced Manufacturing Technology*, 30: 579-588.

Appendix A

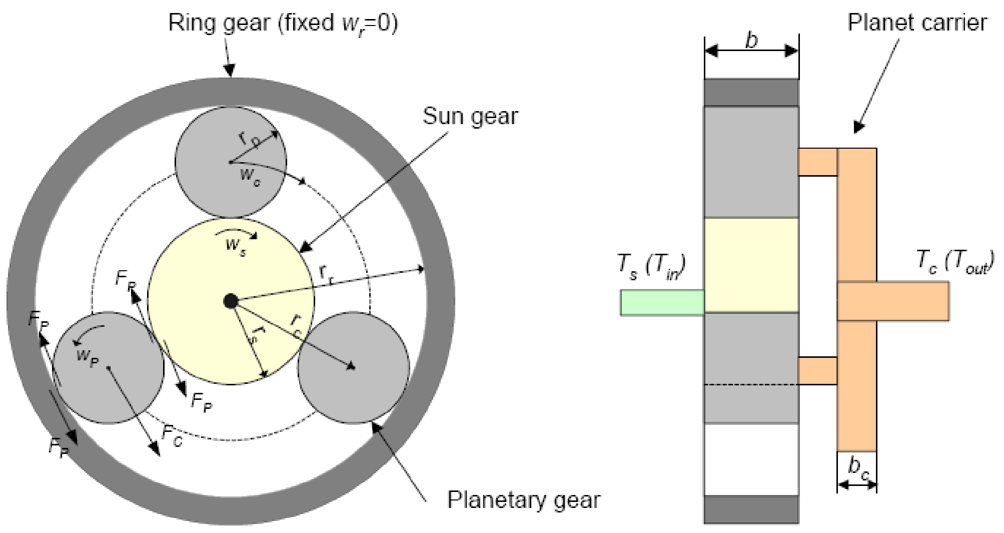


Figure A1: Sketch of a three wheel planetary gear train (Roos and Spiegelber, 2004)

The formulae for planetary gear train are presented here. Only those final equations for mass, SF and stress analysis, and transmission ratio are given because of the limited pages. Detailed derivation can be found in (Roos and Spiegelber, 2004; Gear Manual, 2000). Since the planetary gear consists of one internal gear pair (the ring and planet gears) and one external gear pair (the sun and planet gears), both pairs have to be checked with respect to Hertzian pressure and bending fatigue as follows.

Mass of planetary gear for one layer:

$$\begin{aligned} m &= m_s + m_r + N_p \times m_p + m_c \\ &= F {M_d}^2 \pi \rho (Z_s^2 + Z_r^2(k_{ro} - 1) + N_p \times Z_p^2 + b_c (Z_s + Z_p)^2 / F) / 4 \end{aligned} \tag{A.1}$$

where m_s, m_r, m_p and m_c are the masses of the sun gear, ring gear, planet gears and planet carrier respectively.

Transmission ratio:

$$Ra = \frac{\varpi_{out}}{\varpi_{in}} = \frac{Z_s + Z_r}{Z_s} \tag{A.2}$$

where ω_{out} is the output rotating speed and ω_{in} is the input rotating speed.

Safety Factor and Stress Analysis:

$$SF = Min\{SF_F, SF_H\},\ \ SF_F = \frac{\sigma_{F,all}}{\sigma_F},\ \ SF_H = \frac{\sigma_{H,all}}{\sigma_H}$$

$$\sigma_F = Y_F Y_\beta Y_\varepsilon K_a K_v K_{F\alpha} K_{F\beta} \frac{2T_{out}}{N_p F M_d^{\,2}(Z_r + Z_s)} \qquad \text{(A.3)}$$

$$\sigma_H = Z_H Z_M Z_\varepsilon \sqrt{K_a K_v K_{H\alpha} K_{H\beta} \frac{2T_{out}}{N_p F M_d^{\,2}(Z_r - Z_s) Z_s}}$$

Assuming the same material in all wheels, the maximum allowed stresses may be found out in the gear design manual depending on the material property. There is however one exception to this, the maximum root bending stress of the planet wheels. Since the peripheral (load) force changes sign every second contact, it is necessary to reduce the allowed bending stress with 30% (Gear Manual, 2000).

$$SF_F = \frac{\sigma_{F,all} \times 0.7}{\sigma_F}$$

The following table lists the design factors used in equations of planetary gear train, include their descriptions, constant values and equations to derive values.

Table A.1: Design Factors for planetary gear design (α=20^o)

Design Factor	Description	Value
Y_F	Form factor	Approximately Y_F=2.2+3.1e$^{-z/14}$
Y_β	Helix angle factor	Y_β=1 for spur gear
Y_ε	Contact ratio factor	$Y_\varepsilon=1/\varepsilon_a$
Z_H	Form (Zone) factor	$Z_H=(4/sin2a)^{1/2}$ for spur gear, Z_H=2.50
Z_M	Material factor	$Z_M = \sqrt{\dfrac{1}{\pi(\dfrac{1-v_1^2}{E_1} + \dfrac{1-v_2^3}{E_2})}}$
Z_ε	Contact ratio factor	$Z_\varepsilon = \sqrt{\dfrac{4-\varepsilon_a}{3}}$
K_a	Application factor	K_a=1
K_v	Dynamic factor	K_v=1
K_{Fa}, K_{Ha}	Load distribution factor	K_{Fa}, K_{Ha} =1
$K_{F\beta}$, $K_{H\beta}$		$K_{F\beta}$, $K_{H\beta}$=1.3

Table A2: Radial Contact Ratio of Standard Spur Gears, ε_α (α=20^o)

	12	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	110	120
12	1.420																				
15	1.451	1.481																			
20	1.489	1.519	1.557																		
25	1.516	1.547	1.584	1.612																	
30	1.537	1.567	1.605	1.633	1.654																
35	1.553	1.584	1.622	1.649	1.670	1.687															
40	1.567	1.597	1.635	1.663	1.684	1.700	1.714														
45	1.578	1.609	1.646	1.674	1.695	1.711	1.725	1.736													
50	1.588	1.618	1.656	1.683	1.704	1.721	1.734	1.745	1.755												
55	1.596	1.626	1.664	1.691	1.712	1.729	1.742	1.753	1.763	1.771											
60	1.603	1.633	1.671	1.698	1.719	1.736	1.749	1.760	1.770	1.778	1.785										
65	1.609	1.639	1.677	1.704	1.725	1.742	1.755	1.766	1.776	1.784	1.791	1.797									
70	1.614	1.645	1.682	1.710	1.731	1.747	1.761	1.772	1.781	1.789	1.796	1.802	1.808								
75	1.619	1.649	1.687	1.714	1.735	1.752	1.765	1.777	1.786	1.794	1.801	1.807	1.812	1.817							
80	1.623	1.654	1.691	1.719	1.740	1.756	1.770	1.781	1.790	1.798	1.805	1.811	1.817	1.821	1.826						
85	1.627	1.657	1.695	1.723	1.743	1.760	1.773	1.785	1.794	1.802	1.809	1.815	1.821	1.825	1.830	1.833					
90	1.630	1.661	1.699	1.726	1.747	1.764	1.777	1.788	1.798	1.806	1.813	1.819	1.824	1.829	1.833	1.837	1.840				
95	1.634	1.664	1.702	1.729	1.750	1.767	1.780	1.791	1.801	1.809	1.816	1.822	1.827	1.832	1.836	1.840	1.844	1.847			
100	1.636	1.667	1.705	1.732	1.753	1.770	1.783	1.794	1.804	1.812	1.819	1.825	1.830	1.835	1.839	1.843	1.846	1.850	1.853		
110	1.642	1.672	1.710	1.737	1.758	1.775	1.788	1.799	1.809	1.817	1.824	1.830	1.835	1.840	1.844	1.848	1.852	1.855	1.858	1.863	
120	1.646	1.676	1.714	1.742	1.762	1.779	1.792	1.804	1.813	1.821	1.828	1.834	1.840	1.844	1.849	1.852	1.856	1.859	1.862	.867	1.871
RACK	1.701	1.731	1.769	1.797	1.817	1.834	1.847	1.859	1.868	1.876	1.883	1.889	1.894	1.899	1.903	1.907	1.911	1.914	1.917	1.922	1.926

Appendix B

The following tables show alternative family solutions with varying level of commonality for layer 1, 2 and 3.

Table B1: Specification of multi-platforming family design (Layer 1)

Variant		Design Variables						Performance		
		F	Z_s	Z_r	Z_p	M_d	N_p	m(g)	SF	Ra.
Solution 1 CI=1.000	v_1^1	2.3	14	46	16	0.6	4	14.2	1.88	4.29
	v_2^1	-	-	-	-	-	-	-	1.70	-
	v_3^1	-	-	-	-	-	-	-	1.51	-
	v_4^1	-	-	-	-	-	-	-	1.39	-
	v_5^1	-	-	-	-	-	-	-	1.30	-
Solution 2 CI=0.8225	v_1^1	1.8	14	46	16	0.6	4	12.5	1.67	4.29
	v_2^1	1.8	-	-	-	-	-	12.5	1.51	-
	v_3^1	1.8	-	-	-	-	-	12.5	1.35	-
	v_4^1	1.8	-	-	-	-	-	12.5	1.24	-
	v_5^1	2.3	-	-	-	-	-	14.2	1.30	-
Solution 3 CI=0.7500	v_1^1	1.8	14	46	16	0.6	3	11.5	1.45	4.29
	v_2^1	1.8	-	-	-	-	3	11.5	1.31	-
	v_3^1	1.8	-	-	-	-	4	12.5	1.35	-
	v_4^1	1.8	-	-	-	-	4	12.5	1.24	-
	v_5^1	2.3	-	-	-	-	4	14.2	1.30	-
Solution 4 CI=0.6450	v_1^1	1.5	14	46	16	0.6	3	10.5	1.32	4.29
	v_2^1	1.5	-	-	-	-	3	10.5	1.19	-
	v_3^1	1.8	-	-	-	-	4	12.5	1.35	-
	v_4^1	1.8	-	-	-	-	4	12.5	1.24	-
	v_5^1	2.3	-	-	-	-	4	14.2	1.30	-
Solution 5 CI=0.5725	v_1^1	1.5	14	46	16	0.6	3	10.5	1.32	4.29
	v_2^1	1.5	-	-	-	-	3	10.5	1.19	-
	v_3^1	1.6	-	-	-	-	4	11.9	1.29	-
	v_4^1	1.8	-	-	-	-	4	12.5	1.24	-
	v_5^1	2.3	-	-	-	-	4	14.2	1.30	-

Table B2: Specification of multi-platforming family design (Layer 2)

Variant		Design Variables						Performance		
		F	Z_s	Z_r	Z_p	M_d	N_p	m(g)	SF	Ra.
Solution 1 CI=1.000	v_1^2	5.4	16	40	12	0.7	4	28.1	1.67	3.50
	v_2^2	-	-	-	-	-	-	-	1.51	-
	v_3^2	-	-	-	-	-	-	-	1.35	-
	v_4^2	-	-	-	-	-	-	-	1.24	-
	v_5^2	-	-	-	-	-	-	-	1.16	-
Solution 2 CI=0.8225	v_1^2	3.5	16	40	12	0.7	4	21.2	1.33	3.50
	v_2^2	3.5	-	-	-	-	-	21.2	1.20	-
	v_3^2	6.9	-	-	-	-	-	33.0	1.51	-
	v_4^2	6.9	-	-	-	-	-	33.0	1.39	-
	v_5^2	6.9	-	-	-	-	-	33.0	1.30	-
Solution 3 CI=0.6450	v_1^2	3.5	16	40	12	0.7	4	21.2	1.33	3.50
	v_2^2	3.5	-	-	-	-	-	21.2	1.20	-
	v_3^2	5.5	-	-	-	-	-	28.4	1.36	-
	v_4^2	5.5	-	-	-	-	-	28.4	1.25	-
	v_5^2	6.9	-	-	-	-	-	33.0	1.30	-
Solution 4 CI=0.4675	v_1^2	3.5	16	40	12	0.6	4	21.2	1.33	3.50
	v_2^2	3.5	-	-	-	-	-	21.2	1.20	-
	v_3^2	4.7	-	-	-	-	-	25.7	1.25	-
	v_4^2	5.5	-	-	-	-	-	28.4	1.25	-
	v_5^2	6.9	-	-	-	-	-	33.0	1.30	-
Solution 5 CI=0.2900	v_1^2	2.8	16	40	12	0.6	4	19.1	1.20	3.50
	v_2^2	3.5	-	-	-	-	-	21.2	1.20	-
	v_3^2	4.7	-	-	-	-	-	25.7	1.25	-
	v_4^2	5.5	-	-	-	-	-	28.4	1.25	-
	v_5^2	6.9	-	-	-	-	-	33.0	1.30	-

Table B3: Specification of multi-platforming family design (Layer 3)

Variant		Design Variables						Performance		
		F	Z_s	Z_r	Z_p	M_d	N_p	m(g)	SF	Ra.
Solution 1 CI=1.000	v_1^3	12.9	18	42	12	0.7	5	67.6	1.68	3.33
	v_2^3	-	-	-	-	-	-	-	1.52	-
	v_3^3	-	-	-	-	-	-	-	1.35	-
	v_4^3	-	-	-	-	-	-	-	1.25	-
	v_5^3	-	-	-	-	-	-	-	1.16	-
Solution 2 CI=0.8225	v_1^3	8.1	18	42	12	0.7	5	47.5	1.33	3.33
	v_2^3	8.1	-	-	-	-	-	47.5	1.20	-
	v_3^3	13.0	-	-	-	-	-	68.0	1.36	-
	v_4^3	13.0	-	-	-	-	-	68.0	1.25	-
	v_5^3	13.0	-	-	-	-	-	68.0	1.17	-
Solution 3 CI=0.6450	v_1^3	8.1	18	42	12	0.7	5	47.5	1.33	3.33
	v_2^3	8.1	-	-	-	-	-	47.5	1.20	-
	v_3^3	11.1	-	-	-	-	-	60.0	1.25	-
	v_4^3	15.0	-	-	-	-	-	76.4	1.34	-
	v_5^3	15.0	-	-	-	-	-	76.4	1.25	-
Solution 4 CI=0.5725	v_1^3	8.1	18	42	12	0.7	5	47.5	1.33	3.33
	v_2^3	8.1	-	-	-	0.7	-	47.5	1.20	-
	v_3^3	13.0	-	-	-	0.7	-	68.0	1.36	-
	v_4^3	13.0	-	-	-	0.7	-	68.0	1.25	-
	v_5^3	12.4	-	-	-	0.8	-	85.5	1.30	-
Solution 5 CI=0.3950	v_1^3	6.6	18	42	12	0.7	5	41.2	1.20	3.33
	v_2^3	8.1	-	-	-	0.7	-	47.5	1.20	-
	v_3^3	13.0	-	-	-	0.7	-	68.0	1.36	-
	v_4^3	13.0	-	-	-	0.7	-	68.0	1.25	-
	v_5^3	12.4	-	-	-	0.8	-	85.5	1.30	-
Solution 6 CI=0.3225	v_1^3	8.1	18	42	12	0.7	4	44.0	1.19	3.33
	v_2^3	8.1	-	-	-	0.7	5	47.5	1.20	-
	v_3^3	11.1	-	-	-	0.7	5	60.0	1.25	-
	v_4^3	13.0	-	-	-	0.7	5	68.0	1.25	-
	v_5^3	12.4	-	-	-	0.8	5	85.5	1.30	-
Solution 7 CI=0.2175	v_1^3	6.6	18	42	12	0.7	5	41.2	1.20	3.33
	v_2^3	8.1	-	-	-	0.7	-	47.5	1.20	-
	v_3^3	11.1	-	-	-	0.7	-	60.0	1.25	-
	v_4^3	13.0	-	-	-	0.7	-	68.0	1.25	-
	v_5^3	12.4	-	-	-	0.8	-	85.5	1.30	-

Publications

Liu Zhuo, Wong Yoke San and Lee Kim Seng, An approach to conceptualize product family architecture, *International Conference on Manufacturing and Material Processing*, Kuala Lumpur, Malaysia, Mar., 2006.

Liu Zhuo, Wong Yoke San and Lee Kim Seng, A Systematic Approach to Optimize the Scale-based Product Family Design with Genetic Algorithm, *International Conference on Manufacturing Automation*, Singapore, May, 2007.

Liu Zhuo, Wong Yoke San and Lee Kim Seng, Towards effective multi-platforming design of product family using genetic algorithm, *IEEE Conference on Automation Science and Engineering*, Scottsdale, USA, Sep., 2007

Liu Zhuo, Wong Yoke San and Lee Kim Seng, Modified GA-based optimizer for multi-objective product family design, *The 4th International Conference on Autonomous Robots and Agents*, Wellington, New Zealand, Feb., 2009

Liu Zhuo, Wong Yoke San and Lee Kim Seng, Integrated Approach to modularize the conceptual product family architecture, *International Journal of Advanced Manufacturing Technology*, 36: 83-96, 2008.

Liu Zhuo, Wong Yoke San and Lee Kim Seng, Modularity analysis and commonality design: a framework for the top-down platform and product family design, Accepted for publishing in *International Journal of Production Research*..