A Thesis for the Degree of Ph.D. in Engineering

Optimal Product Platform Configuration Decision in Mass Customization

Ting Wang

Supervisor: Prof. Hiroaki Matsukawa

March 2023

Graduate School of Science and Technology
Keio University

Abstract

Platform-based product development (PPD) is a cost-efficient approach to achieve mass customization. Through PPD, manufacturers can develop various products to meet diverse customer preferences and requirements while maintaining production efficiency without compromising cost, quality, and delivery. One important problem in PPD is product platform configuration (PPC), which aims to identify and configure modules, components, or design variables on the product platform that can be shared across a product family. Two pertinent problems are: (1) how many and which type of product platform should be developed for a product family; (2) which product platform will be assigned to develop the product. The PPC decisions are endogenously linked to supply chain-related activities and will affect all the stages and sectors throughout the supply chain. Studying PPC problem from a supply chain management perspective is significant for manufacturers implementing the PPD approach. In this study, a modular platform configuration model is targeted, and various optimization methods are applied to determine the optimal platform configuration decision. More specifically, the following aspects will be targeted in this study.

In chapter 3, we study a modular platform configuration model. Modular design is recognized as the most important underlying architecture to support the product family design and product platform design. Two types of modular design approaches can be found, i.e., module selection and module integration. The platform configuration based on more module selections provides a broader solution space of possible platform configurations to meet the customer requirements exactly. However, it will increase the complexity of production process due to the proliferation of module types and part numbers. Module integration in the platform configuration facilitates the platform commonality to gain economic benefits. Traditional platform research focuses more on platform

configuration based on module selection without considering the module integration simultaneously. A new model is developed to determine the optimal platform configuration for a product family while considering both module selection and integration. A hybrid-search method (HSM) combining simulated annealing (SA) and variable neighbourhood search (VNS) is developed to solve the proposed model.

In chapter 4, we examine the optimal platform configuration decision considering platform design strategy and supplier selection. Different types of platform design strategies can be found to satisfy product requirements, i.e., matching-designed, over-designed, under-designed, and hybrid-designed platforms. The matching-designed platform has a higher platform development cost and a lower customization cost while the over-designed or under-designed platform contributes different performances in these two types of costs. Traditional research balances the cost trade-off within the design domain, and few studies include the relative procurement cost from suppliers. Involving the supplier selection problem at the earliest design stage has proven beneficial to companies. However, little attention has been paid to integrating supplier selection into the PPC problem. In this chapter, we propose a non-linear mixed-integer programming model to determine the optimal platform configuration decision while considering platform design strategy and supplier selection. A cost model including development cost, sourcing cost and customization cost is developed to illustrate the cost trade-off between platform development and customization. A solution method applying the linearization method with Gurobi solver is proposed to solve this model.

In chapter 5, we study the platform configuration problem considering demand uncertainty. Demand uncertainty is a huge challenge for supply chain management and product platform configuration. Generally, the development of product platform is ahead of new product introduction (NPI), which makes it difficult to forecast demand. Most existing research on platform configuration assumes that the demand is deterministic so that the problem can be easily dealt with. However, when considering the uncertain demand, the platform configuration decision may be changed, and the

optimization problem will become more complex. How to determine the optimal PPC decision under demand uncertainty is highly important for manufacturers to develop product platform. This research gap will be filled in chapter 5. The platform configuration problem under demand uncertainty is formulated as a two-stage stochastic programming model, including the platform configuration stage and platform customization stage. A cost model including the development cost of platform, production cost and material cost for two stages, customization cost, and penalty cost of excessive platforms is developed. A linear programming embedded genetic algorithm is developed to solve the proposed model. The proposed algorithm searches the binary variables for platform configuration by using GA and determines the integer variables by solving a linear programming subproblem using Gurobi solver. Numerical experiments are conducted to illustrate the proposed model and algorithm.

Keywords: platform-based product development, product platform configuration, supply chain cost model, mass customization, modular design, platform commonality, demand uncertainty, stochastic programming

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Background and motivation

The diversity of customer preferences and requirements enables manufacturers to offer a larger product variety. Product variety can be defined as the different number of products for a particular class (ElMaraghy et al., 2013). Accordingly, manufacturers today are no longer focused on a single product or service but offer a wide range of products with different functions, features, requirements, and specifications. The increasing product variety can offer the potential to expand the market and increase sales volume and revenue (ElMaraghy et al., 2013). However, the proliferation of product variety is not always beneficial for a company. The complexity from product design to production, inventory, selling, and services increases as more products and more parts numbers, which enables the relative supply chain to become inefficient and incur substantial cost within the company (Simpson, 2004). Therefore, one imperative challenge today is to fulfill increasingly diverse customer needs while achieving internal efficiencies in designing, manufacturing, and delivering those products.

To manage this challenge in product variety management, mass customization has gained increasing attention in the past decades. Mass customization emerges in the early 1990s with the goal of satisfying increased product variety with near-mass production efficiency (ElMaraghy et al., 2013; Pirmoradi et al., 2014). As one of the effective tools to implement mass customization, platform-based product development (hereafter abbreviated as PPD) is proposed to develop different products for a product family based on common platform (AlGeddawy and ElMaraghy, 2013; Wang et al., 2022). A product family refers to a group of similar products that are derived from a common platform and possess specific features and functions to meet customer requirements (Meyer and Lehnerd, 1997;

Jiao et al., 2007). A product platform is a set of sub-systems and interfaces that form a common structure from which a stream of derivative products can be efficiently produced and developed (Meyer and Lehnerd, 1997; Jiao et al., 2007).

Many manufacturers, such as Toyota, Volkswagen, Philips, Airbus, Ford, IBM, and LG, have adopted the PPD approach to produce their various products (Wang et al., 2022). Through PPD, manufacturers can bless multiple benefits including reduced development time and improved ability to upgrade products (Simpson, 2004), increased efficiency and reduced cost in manufacturing (Liu et al., 2010; Ben-Arieh et al., 2009), improved product quality, and reduced waste (Pirmoradi et al., 2014). For example, Toyota leverages Toyota New Global Architecture (TNGA) platform to produce various models, such as the medium-sized (e.g., Corolla), the large (e.g., Crown and Lexus LS), and the compact vehicles (Corporation, 2021). The Volkswagen Group uses the Modular Transverse Matrix (MQB) platform to produce various vehicle models (ElMaraghy et al., 2013).

The key point in PPD is product platform configuration (hereafter abbreviated as PPC). Product platform configuration aims to identify and configure modules, parts, components, or design variables on the product platform that can be shared across multiple products within a product family. Two types of product platform configuration have been widely discussed in the existing research, namely module-based platform configuration and scale-based platform configuration (Simpson, 2004; Jiao et al., 2007). A module-based platform configuration consists of functional modules that can be added, substituted, or removed to derive unique products while a scale-based platform uses the scaling variables to stretch or shrink design parameters so as to satisfy product requirements (Simpson, 2004). Regardless of whether the platform is modular and scalable, the main problem in the PPC relies on the selection of modules and design variables and the realization of platform commonality.

Existing research on PPC ranges from an engineering design perspective to a product management perspective. The PPC problem in the engineering design domain aims to identify and

evaluate the platform commonality by constituting the shared modules and design variables. For example, various commonality matrixes are developed to assess the similarity of modules in the product architecture. Classification and group technology are introduced in the design and manufacturing process. Research on PPC from a product management perspective treats the PPC problem as an optimization problem. Through searching the possible combinations of modules or design variables, the optimal platform configuration can be obtained under different evaluation criteria such as quality, cost, and utility.

Meanwhile, successful supply chain management is critical to forming the competitive advantage of manufacturers. The PPC decision will affect all the stages and sectors throughout the supply chain, e.g., new product development, procurement, production, manufacturing, inventory management, and distribution. It is generally known that approximately 80% of manufacturing costs are determined during the design phase (Mikkola, 2007). Studying PPC problems focused on supply chain management is highly important for implementing the PPD approach.

Two pertinent problems are targeted in this study: (1) how many and which type of product platform should be developed for a product family; (2) which product platform should be assigned to develop the products within a product family. The first problem is to determine the platform configuration by selecting modules, components, and elements to be shared. The second problem is related to the platform design strategy and customization, i.e., a platform should be configured as under-designed, over-designed or matching-designed regarding its dedicated product. We solve these two problems by proposing mathematical optimization models while considering supply chain-related costs.

1.2 Research scope

Research on PPD and PPC covers a wide range of areas, including engineering design, business model-oriented, marketing-driven, customer satisfaction, and supply chain domains. We identify the

research scope of this study from four perspectives, i.e., platform configuration type, targeted domain, associated design idea, and targeted product family type.

Firstly, two types of product platform configurations have been widely discussed in the existing research, namely module-based platform configuration and scale-based platform configuration (Simpson, 2004; Jiao et al., 2007). A module-based platform configuration consists of functional modules that can be added, substituted, or removed to derive unique products. A scale-based platform uses the scalable variables to stretch or shrink design parameters so as to satisfy different product requirements. The module-based platform configuration is to be examined in this study because of the targeted mass customization scenario. As one of the main enablers of mass customization, modular design has been widely used in academia and industry. With modularity, a complex product can be decomposed into independent modules or parts, which makes it possible to fulfill various functions and product variety through different combinations of modules and module options. Moreover, the optimization problem of platform configuration can be formulated as a 0-1 combinatorial optimization model through modelling the module selection.

Secondly, product platform configuration affects the entire spectrum of product realization from customer analysis, function identification, production, manufacturing, logistics and selling. Besides the product design considerations, our study focuses more on the area of supply chain management (SCM) and product lifecycle management (PLM). The purpose of this study is to configure the product platform based on the requirements and constraints of different processes in the supply chain and product lifecycle. For example, the cost related to the production and inventory will be included in our study, e.g., procurement cost, inventory cost, and ordering cost.

Thirdly, this study also supports the design idea related to design for variety (DFV) and variety reduction program (VRP). The DFV and VRP aim to reduce the internal variety and complexity so as to reduce the total supply chain cost. In this study, we mainly analyze how the economic benefits of platform and module commonality can be realized through the risk-pooling in the process of

development, procurement, manufacturing, and inventory.

Finally, a broad range of modular products and platform products can be targeted as an illustration in this study, e.g., consumer electronic products, computers, and printers, vehicles, as defined by Jacobs and Chase (2018). As a typical modular product, the motherboard design of personal computer product family will be introduced to illustrate the proposed model and solution method. The motherboard can be regarded as the platform conception of PC and the components of PC can be understood as modules, e.g., processor, RAM, wireless network card, and speaker. The proposed model and method are going to support the best platform configuration decision for case studies.

1.3 Research objective and plan

The objective of this study is to determine the optimal platform configuration decision focused on supply chain management. The specific objective and research plan are presented as follows.

(1) To evaluate the overall performance of platform approach, a supply chain cost model is developed. Several cost items throughout the supply chain are included, i.e., development cost, procurement cost, ordering cost, inventory cost, shortage cost and salvage cost. Through quantifying the total supply chain cost model, the impact of PPC on the major supply chain activities, e.g., sourcing, production, and manufacturing will be examined. The specific supply chain cost model will be described in chapter 3.

(2) Modular design plays an important role in platform configuration and product family design. The platform configuration based on more module selections can provide a broader solution space of platform selection to meet the customer requirements for a niche market exactly. However, it also increases system complexity and brings negative effects on the production process due to the proliferation of module types and part numbers. Module integration in the platform configuration facilitates the platform commonality to gain economic benefits. How to balance module selection

and integration in the platform configuration will be studied with the goal of minimizing the total supply chain cost in chapter 3. The notion of integration in this research refers to replacing a lower specification module with one that has a higher specification. Therefore, the term integration is different from the term “integral” which appears in product development comparing it with “modular” based product development in Japanese research papers.

(3) Another key issue for PPC is platform design strategy when leveraging product platforms to develop multiple products within a product family. Different types of design strategies can be found, i.e., matching-design, under-design, over-design, and hybrid-design. The different platform design strategies contribute different impacts on the cost items. For example, a product platform can be configured as a matching-designed platform that exactly complies with product requirements. It may require more platforms to be developed for one product family, thus incurring a higher platform development cost. On the other hand, product platform can be configured as under-designed or over-designed platform, thereby reducing the number of platforms. However, additional platform customization will be needed to update the required functions when deriving a high-end product from an under-designed platform. When deriving a low-end product from an over-designed platform, some functions on the platform may be wasted. Considering different platform design strategies with customization will affect PPC decisions. Furthermore, involving the supplier selection problem at the earliest design stage has proven beneficial to companies. However, little attention has been paid to the PPC problem. To fill this gap, the supplier selection problem will be integrated into the platform configuration model to investigate the impact of outsourcing decision. This problem will be studied in chapter 4.

(4) The demand uncertainty is a great challenge for supply chain management and platform configuration. It is difficult to forecast demand of each platform type during several months ahead of the new product introduction (NPI). However, the platform development always takes a long time. A platform configuration decision under demand uncertainty is imperative for implementing the PPD.

This problem will be studied in chapter 5. A two-stage stochastic programming model is proposed to handle the platform configuration problem under demand uncertainty.

(5) Research method

The platform configuration optimization is a combinatorial optimization problem. For a small-scale problem, we may find an optimal solution to satisfy the given customer requirements. However, it is hard to be solved directly through existing software when the problem is large-scale. An efficient solution method is crucial for solving PPC problems. Three different solution methods are developed in this study, including a hybrid methodology combining simulated annealing (SA) and variable neighborhood search (VNS) in chapter 3, a linearization method in chapter 4 and a linear programming embedded genetic algorithm (GA) in chapter 5.

1.4 Thesis structure

This thesis is organized as follows.

Chapter 2 systematically reviews the relevant research on the PPD, PPC, product family design and their relevant supply chain issues and identifies the research gap.

In chapter 3, a supply chain cost model is developed as the evaluation criteria to assess the overall platform performance. A new model for product platform configuration considering module selection and module integration is proposed. A hybrid algorithm combining simulated annealing (SA) and variable neighborhood search (VNS) is developed to solve the proposed model.

In chapter 4, we examine the different platform design strategies while considering platform customization and supplier selection. Four different platform design strategies are identified, i.e., matching-design, under-design, over-design, and hybrid-design. Some linearization methods are applied to transform the proposed nonlinear mixed-integer programming model into linear ones.

In chapter 5, we propose a two-stage stochastic programming model for platform configuration to study the optimal PPC decision under demand uncertainty. The impact of different demand

distributions on PPC decision is also investigated. A linear programming embedded genetic algorithm (GA) is developed to solve the proposed model.

Finally, we summarize the conclusions and present the limitations and future work in chapter 6.

Chapter 2 Literature review

Platform-based product development has been recognized as a cost-efficient way to offer the required product variety without increasing costs and time-to-market (Simpson et al., 2014; Andersen et al., 2022; Wang et al., 2022). Various methods and tools have been developed to help manufacturers implement the PPD approach. Jiao et al. (2007) divided the PPD approach into three stages: (1) translating identified customer needs (CN) into functional requirements (FR) based on a product portfolio; (2) mapping those functional requirements (FR) into proper design variables (DV) based on the shared product platform; and (3) enabling the design variables to correspond to the process variables and logistic variables.

Although our study focuses more on supply chain management, design-related research, e.g., engineering design of module and product architecture, is also inextricably linked. The relevant design specifications and requirements are the prerequisites for product platform configuration. Therefore, three literature streams will be reviewed, namely (1) product platform configuration, (2) product portfolio and product family design, and (3) supply chain issues for PPD and product family design.

2.1 Basic conceptions

In this section, the relevant conceptions including mass customization, modularity, product platform, product family design, and supply chain management are introduced.

2.1.1 Mass customization

Mass customization emerges in the early 1990s in response to increasing product variety. Mass customization means providing customized goods and services that best meet individual customer

needs with near mass production efficiency (Pine, 1993; ElMaraghy et al., 2013; Pirmoradi et al., 2014; Fogliatto et al., 2012). The purpose of MC is to achieve the economy of scope at a cost approaching that of economy of scale by delaying product differentiation and capitalizing on commonality and similarity between variants within a product family (ElMaraghy et al., 2013). The key feature of mass customization is to integrate the product varieties derived from individual customer needs and the efficiency of mass production (Tseng and Jiao, 2001).

2.1.2 Modularity

Modular design and product modularity have been regarded as effective approaches to achieve mass customization. A module is a physical or conceptual grouping of components that correspond to particular function (Jiao et al., 2007), while its structural elements are strongly interconnected and weakly connected to elements in other units or modules (AlGeddawy and ElMaraghy, 2013). Modular design allows the rapid development of new products by using alternate modules or module instances. Common modules can be shared across different products within a product family, thus achieving commonality and reducing costs. The main work on modular design focuses on how to group the partition of functional carriers or components into one module and standardize the interface (Bonvoisin et al., 2016; Liu et al., 2010). For example, the quality function deployment (QFD) method and design structure matrix (DSM) are the common tools for forming and illustrating product modules (Pirmoradi et al., 2014).

2.1.3 Product platform

Product platform is defined diversely ranging from being general and abstract to being industry and product specific (Jiao et al., 2007). Two types of product platform are widely discussed in the existing literature. The first one refers to a product platform as a physical conception, i.e., a collection of elements shared by multiple products (Ericsson and Erixon, 1999). The other one defines a product

platform as a common structure that contains a set of subsystems and interfaces. A stream of derivative products can be efficiently developed based on the product platform (Meyer and Lehnerd, 1997).

2.1.4 Product family design

A product family is a set of similar products that are derived from a common platform with differentiated features to meet particular customer requirements (Jiao et al., 2007). Each individual product within a product family can be defined as a product variant or instance. A product family targets a certain market segment, and each product variant corresponds to a specific customer need in the market segment. All product variants within a product family will share common platforms, modules, and components to achieve commonality. Product portfolio is the most important problem in product family design, which aims to find the best combination of product variants to respond to diverse customer needs.

From the marketing and sales perspective, the functional structure of product family is represented by product lines and product portfolios and thus is characterized by various sets of functional features for different customer groups. The product family from the engineering view is embodied by different product technologies and associated manufacturability, and thereby is characterized by various design parameters, components, and assembly structures.

2.1.5 Supply chain management

Supply chain management (SCM) deals with the coordination and integration of various businesses involved in the realization of products throughout the supply chain (ElMaraghy et al., 2013). SCM encompasses the integrated planning and execution of processes required to manage the movement of materials, information, and financial capital. It contains various supply chain activities such as design, planning, sourcing, production, inventory management, logistics, and customer

service. The goal of SCM is to deliver the right products to the right place at the right time, with the right quantity and quality while optimizing the total cost. Various supply chain issues, e.g., sourcing, procurement, assembly, manufacturing, and distribution, will have an important impact on the product platform development (Pirmoradi et al., 2014).

2.2 Product platform configuration

Product platform configuration aims to identify and configure the shared modules or design variables on the product platform to satisfy customer requirements. Various methodologies from an engineering design perspective and product management perspective are applied to configure product platforms for a product family.

2.2.1 PPC in engineering design domain

The PPC problem from the engineering design perspective aims to determine the engineering technical configuration for the module, platform, and product architecture. The main concern is the realization of commonality. Various design methods, e.g., commonality matrixes, classification, and group technology, are developed to assess the similarity of modules and components in the architecture design.

Chen and Wang (2008) proposed a method to design a product platform through clustering analysis and information theoretic approach. Askhøj and Mortensen (2020) applied a DNA method (deciding the number of architectures) to determine the total number of product architectures. Their method consists of four stages, i.e., market segmentation, mapping new generation with an existing architecture strategy, architecture changes, and the new architecture strategy. Colombo et al. (2020) developed a value analysis method to rank alternative platform configurations according to customer preferences. A case study based on the Google ARA Spiral-2 modular smartphone concept was presented. Otto et al. (2016) introduced a generic platform design approach with 13 steps, including

market segment definition, market attack plan, customer need gathering, system requirement definition, functional requirement definition, etc., for developing a modular product platform within the development process. Zhao et al. (2022) developed a module clustering approach to form the product platform taking into account the design structure and the relationship between product architecture and manufacturing process. Okpoti et al. (2019) presented an agent-based collaborative design model and proposed a decentralized coordination mechanism to facilitate the design variables for product platforms in a product family. Galizia et al. (2020) proposed a decision support system (DSS) to design and select product platforms in high-variety manufacturing. A median joining phylogenetic network was applied to design the platforms and a phylogenetic tree decomposition was used for platform selection with the analysis of platform variety and customization in the DSS.

2.2.2 PPC in product management domain

Another research stream from the product management perspective treats the platform configuration problem as a combinatorial optimization problem while focusing on production and manufacturing. The product platform can be configured through different combinations of modules and components. Qu et al. (2011) developed a two-stage platform development approach for mass customization using a genetic algorithm. The common components can be identified according to the structure commonality in the first stage while a parametric optimization can be conducted in the second stage. Ben-Arieh et al. (2009) formulated a mixed integer programming model to configure multiple modular platforms for a given product family while minimizing overall production cost. The products in a product family can be derived by assembling and disassembling components on the product platforms. A genetic algorithm was developed to solve their proposed model. Hanafy and ElMaraghy (2015) further considered the assembly sequence constraints in the multiple platform configuration model based on Ben-Arieh's model. They solved their model by using the commercial solver CPLEX and illustrated a case study of touch screen tablet family. ElMaraghy and Moussa

(2019) expanded the platform design by utilizing additive and subtractive manufacturing conception and developed a genetic algorithm-based model to design the optimal or near-optimal platform for a large set of products and features (Moussa and ElMaraghy, 2020). Moreover, Moussa and ElMaraghy (2021) proposed a multiple-platform design model that utilizes additive and subtractive manufacturing to customize products from platforms. A genetic algorithm was developed and applied to case studies of guiding bushings and gear shafts to demonstrate this model.

Most of the above studies assumed a deterministic product demand, while a few studies considered uncertainty and allowed more complex cost structures. For example, Van den Broeke et al. (2015) formulated a supply chain cost model for product-platform assignment decision, including the development cost, purchasing cost, inventory cost, and transformation cost while considering the demand uncertainty. A simulated annealing algorithm was proposed to determine the number of platforms and from which platform the product is derived. Furthermore, Van den Broeke et al. (2017) formulated two fathoming rules to improve the algorithm efficiency and illustrate the applicability in the branch-and-bound algorithm, simulated annealing algorithm, and genetic algorithm. Song and Ni (2019) formulated a fuzzy programming model to design a product platform with a modularity strategy under fuzzy environment. In their model, the cost savings of designing a modular platform, the demand quantity of products, the parameters representing economies of scale, and product quality improvement were characterized as fuzzy variables. An efficient algorithm combining fuzzy simulation and simulated annealing was proposed to solve the model.

2.3 Product portfolio and product family design

The platform configuration problem aims to find the optimal mix of platforms with modules or design variables, while the product portfolio problem seeks the right mix of product attributes to offer to the market (Van den Broeke et al., 2017; Jiao and Zhang, 2005). Mapping customer needs to functional requirements is essential for generating the product portfolio. The customer-perceived

utility of products has often been used as an objective to maximize customer satisfaction.

Jiao and Zhang (2005) proposed a shared-surplus model to address the product portfolio planning problem. Customer preferences, choice probabilities, and platform-based product costing were considered in this model. A heuristic genetic algorithm was developed for solving the product portfolio planning problem effectively (Jiao et al., 2007). Sadeghi et al. (2010) introduced a simulated annealing algorithm to compare with the genetic algorithm proposed by Jiao et al. (2007). They found that the SA algorithm is more efficient than GA in solving product portfolio problems. Goswami et al. (2016) formulated an integrated Bayesian-Game theoretic approach for multi-attributed product portfolio planning. The function-based cost estimating framework and multi-linear regression methodology were applied to estimate the manufacturing cost and product premiums for different product portfolios. Fujita and Yoshida (2004) proposed a method combining the genetic algorithm and nonlinear mixed-integer programming method to simultaneously optimize module selection and module attribute parameters. Zhang et al. (2008) developed a mixed-integer programming model to determine the optimal product portfolio that considers the module selection. Du et al. (2014) proposed a bi-level mixed nonlinear programming model to optimize the module selections and scalable module parameters for product family design. The upper-level optimization seeks an optimal configuration of modules and module attributes by maximizing customer-perceived utility, while lower-level optimization entails parametric optimization by maximizing the design parameters of each selected module. Zhang et al. (2020) proposed a progressing modelling method for feature-centered product family development. This model synthesized the product family information to support features-based knowledge modelling, hybrid innovation and time-dependent holistic product development.

Moreover, Yang et al. (2018) proposed a stochastic programming model to determine the optimal component selections and combinations for product architecture while handling uncertainty in component replenishment lead time. Li et al. (2018) formulated a stochastic mixed-integer

programming model to deal with product architecture problems with uncertain demand and uncertain supply. The model further considered four different carbon emission regulations to investigate the impact of carbon emission on the product architecture.

2.4 Supply chain issues in PPD and product family design

Some recent studies consider the relevant supply chain decisions, e.g., sourcing, manufacturing, assembly, and distribution decision in the product design decision. For example, Salvador et al. (2002) explored the impact of manufacturing characteristics on the modularity decisions for product family design. Huang et al. (2005) proposed a mathematical model to consider the material requirement planning and supplier selection problem in product design. Zhang et al. (2008) developed a mixed-integer programming model to simultaneously optimize the modular product design and the supplier selection decision. Furthermore, Zhang et al. (2010) studied the impact of different supply chain coordination schemes on product and supply chain configuration decision. Three coordination schemes were considered, including non-interactive suppliers, non-cooperative suppliers, and coordinative suppliers. Luo et al. (2011) formulated a linear programming model to integrate modular product family design and the supplier selection problem while considering customer purchasing behavior, supplier availability and related costs. Fujita et al. (2013) proposed a mixed integer programming model to handle a concurrent design problem of module commonality strategies under a given product architecture. Nepel et al. (2012) studied the product family design strategy and the supply chain design by employing a multi-objective programming model with the goal of minimizing costs and maximizing supply chain compatibility. Tan et al. (2022) proposed a concurrent optimization approach to integrate the manufacturing process and supplier selection into the personalized product architecture design problem. The objective is to maximize the potential profit of a product family based on a profit formulation that incorporates customer preference, process resource, supplier, and manufacturing cost.

Moreover, some studies apply the game theory model to deal with the joint optimization problems. Huang et al. (2007) proposed a three-move dynamic game theoretic approach to optimize the product configuration and supply chain. In the first move, the manufacturer takes its leading role to make decisions on product configuration and supplier selection. The concerned suppliers then make the second move to optimize their decisions including price discounts and their ordering policies. The manufacturer finishes the game by taking the last move to make their ordering decisions. Yang et al. (2015) formulated a leader-follower Stackelberg game model to jointly configure the modular product family design and distribution decisions. The upper-level optimization problem was to optimize the module selections and product variants while the lower-level one is to determine the distribution decisions. Du et al. (2014) applied a bi-level mixed nonlinear programming model to determine the module selection and parameter scaling in the product family design. The upper-level optimization seeks an optimal configuration of modules and module attributes by maximizing the customer-perceived utility and the lower-level one determines scaling design parameters by maximizing the module design parameters. Wang et al. (2016) formulated a nonlinear mixed integer bilevel programming model to deal with the product family architecture and supply chain configuration. The upper-level optimization problem aims to determine the optimal selection of base modules and compound modules in product family architecture, while the lower-level optimization problem deals with the relevant supply chain decisions, including supplier selection, manufacturer decision, assembly decision, and distribution center decision.

Recently, some studies have considered the relevant supply chain issues in PPC. Hanafy and ElMaraghy (2017) proposed a modular product platform configuration model to consider assembly line planning with the goals of minimizing assembly station quantity and cycle time. The model can efficiently design assembly line and platform configuration simultaneously. Miao et al. (2017) also formulated a bilevel nonlinear programming model to coordinate the platform configuration and product line planning. Xiong et al. (2018) proposed a Stackelberg game model to integrate modular

product platform and supply chain postponement decision. The upper-level optimization problem identifies the basic module and compound module by maximizing customer-perceived utilities and postponement utilities of product families, whereas the lower-level optimization problem selects the most appropriate postponement service providers with minimizing the total supply chain cost. Moussa and ElMaraghy (2022) formulated a non-linear model to design the optimal platform configurations while focusing on platform inventory management. The remaining inventory of platforms held in each production period could be utilized in the subsequent production periods in this model.

2.5 Optimization method

Various models and approaches have been developed to determine the optimal product platform configuration and product family design. Table 2.1 summarizes the main methods and approaches used in the prevailing studies. These relevant studies are categorized according to research domains, including product portfolio planning (PPP), product family design (PFD), product architecture design (PAD), product platform configuration (PPC) and product platform assignment (PPA). As shown in Table 2.1, combinatorial optimization is widely used to determine the optimal product portfolio. Stackelberg game theory is generally applied to deal with the joint optimization problem of PFD, PAD, and its related supply chain issues. Furthermore, most existing research assumes that the demand is deterministic so that the problems can be easily solved. Only a few studies consider demand uncertainty through stochastic programming or fuzzy optimization approach.

Searching for all possible combinations of modules, components, and design variables to configure multiple product platforms for a product family is a complicated combinational optimization problem. Heuristic algorithm has been frequently used to solve the problems. The genetic algorithm (GA) (e.g., Ben-Arieh et al., 2009; Yang et al., 2015) and simulated annealing algorithm (SA) (e.g., Sadeghi et al., 2011; Van den Broeke et al., 2015) are two major heuristics

algorithms. Some other solution methods are also applied sometime, e.g., branch and bound (B&B) (e.g., Hanafy and ElMaraghy, 2017; Van den Broeke et al., 2017), bilevel programming (B&P) (e.g., Du et al., 2014) or other commercial software CPLEX (e.g., Hanafy and ElMaraghy, 2015, 2017).

Finally, a variety of product family including electronic products such as notebook computer (Graves and Willems, 2005), modular phone (Hanafy and ElMaraghy, 2017), television receiver circuits (Fujita, 2004), printed circuit board of medical screen (Van den Broeke et al., 2015, 2017), touchscreen tablet (Hanafy and ElMaraghy, 2015); industrial products such as an electric motor (Du et al., 2014), cordless drill (Ben-Arieh et al., 2009), power transformer (Yang et al., 2015); complex system such as aircraft (Fujita, 2004), automotive climate control system (Nepal et al., 2012) are introduced to demonstrate the platform configuration and product family design.

Table 2.1 Summary of optimization methods for platform configuration, product portfolio and product family design

Research domain	Reference	Methodology	Consider supply chain decisions?	Demand uncertainty?	Solution algorithm			Case study
					GA	SA	Other	
PPP	Jiao and Zhang (2005)	Combinatorial optimization	N	N	X			Notebook computer
	Jiao et al. (2007)	Combinatorial optimization	N	N	X			Notebook computer
	Sadeghi et al. (2011)	Combinatorial optimization	N	N		X		Notebook computer
PFD	Fujita (2004)		N	N		X		Aircraft design and television receiver circuits
	Fujita et al. (2013)		Y	N	X			Numerical case
	Du et al. (2014)	Stackelberg game	N	N			BP	Electric motor
	Yang et al. (2015)	Stackelberg game	Y	N	X			Power transformer
	Wang et al. (2016)	Stackelberg game	Y	N	X			Power transformer
PAD	Zhang et al. (2008)		Y	N	X		EA	Numerical case
	Zhang et al. (2010)	Game theory	Y	N			IA	Numerical case
	Tan et al. (2022)	Mixed integer programming	Y	N	X			Bicycle
	Yang et al. (2018)	Stochastic programming	Y	Y			LRA; CPLEX	Computer
	Li et al. (2018)	Stochastic programming	Y	Y			BD; CPLEX	Ship engines and power generator
	Nepal et al. (2012)	Weighted goal programming	Y	Y			GP	Bulldozer and automotive climate control system
PPC	Qu et al. (2011)		Y	N	X			Motor product
	Ben-Arieh et al. (2009)	Non-linear programming	N	N	X			Cordless drill
	Hanafy and ElMaraghy (2015)	Non-linear programming	N	N			CPLEX	Touchscreen tablet
	Hanafy and ElMaraghy (2017)	Mixed integer programming	Y	N			CPLEX; B&B	Modular phone
	Moussa and ElMaraghy (2021)		N	N	X			Guiding bushing, gear shaft
	Miao et al. (2017)	Bilevel mixed 0-1 nonlinear	Y	N	X			Automotive
	ElMaraghy and Moussa (2019)	Mixed integer programming	N	N	X			Guiding bushing
	Moussa and ElMaraghy (2022)	Holistic non-linear	Y	N			GUROBI	Gear shaft
	Xiong et al. (2018)	Stackelberg game	Y	N	X			Laser printer

PPA	Van den Broeke et al. (2015)	N	Y		X		Printed circuit board
	Van den Broeke et al. (2017)	N	Y	X	X	B&B	Printed circuit board

Note: PPP, product portfolio planning; PFD, product family design; PAD, product architecture design; PPC, product platform configuration; PPA, product platform assignment.

GA, genetic algorithm; SA, simulated annealing; BP, bilevel programming; EA, enumerative algorithm; IA, iterative algorithm; LRA, lagrangian relaxation algorithm; BD, benders decomposition; B&B, branch and bound; GP, goal programming

2.6 Research gap

The research gap is stated in this section.

Firstly, the extant literature on PPD has identified many of the above-mentioned economic benefits of PPD from statistical analysis, conceptual models, and industry surveys. However, quantifying the economic benefits of PPD from a supply chain management perspective is important for strategic decision-making. Some cost models have been developed in the existing research including the single or several following cost items, e.g., development cost, purchasing cost, production cost, and inventory cost. A comprehensive cost model throughout the entire supply chain is still required. Using a more comprehensive cost model as the evaluation criterion can help to better evaluate the overall performance of PPD. A supply chain cost model including the development, purchasing, ordering, inventory, shortage, and salvage costs will be developed in chapter 3.

Secondly, modular design is recognized as the most important underlying architecture to support product family design and product platform design. Two types of module design approaches can be found, i.e., module selection and module integration. The module selection provides a broader possible of platform types and product variety while module integration facilitates the platform and module commonality to gain economic benefits. Excessive module types and quantities can incur uneconomical consequences and bring negative impacts on supply chain management. While over-pursuing the module integration can reap the benefits of commonality to some extent, it also accompanies some negative effects, such as reduced product differentiation and additional cost for excessive commonality. Traditional platform research either focuses on forming the product platform through module selections and combinations or clustering modules through the hierarchical decomposition of product functional requirements and manufacturing processes. To the best of our knowledge, little research considers the product platform configuration problems associated with the questions of module selection and integration. This problem will be targeted in chapter 3.

Thirdly, the platform configuration model is a combinatorial optimization problem. For a given set of modules or design variables, we may find an optimal combination of modules or design variables to satisfy customer preferences. However, when considering the platform design strategy, i.e., platform-product assignment decisions, the platform configuration will be changed. Moreover, many studies have considered SCM-related issues in product architecture and product family design. However, little attention has been paid to integrating SCM-related issues and PPC problems. Integrating the PPC and related supply chain issues is another research content in this study.

Finally, demand uncertainty is a huge challenge for manufacturers to manage the supply chain and platform configuration. Generally, the development of product platform is ahead of new product introduction (NPI), which is difficult to forecast the demand. Most of the existing research on platform configuration assumes that the demand is deterministic so that the problem can be easily dealt with. However, when considering the uncertain demand, the decision of platform configuration will be changed, and the optimization problem will become more complex. Formulating a flexible platform configuration model will be particularly important for platform-based product development. This research gap will be filled in chapter 5.

Chapter 3 Product platform configuration decision in NPD with module options

3.1 Introduction

Modular platform-based product development is a successful way to offer the required product variety while reducing the internal complexity. Various manufacturers, including automotive, consumer electronics, computers, and aircraft, have implemented the modular PPD approach to produce their products. Platform-based product development and modular design bless multiple benefits (Andersen et al., 2022), such as reduced development time and improved ability to upgrade products (Simpson, 2004), increased efficiency and reduced cost in manufacturing (Liu et al., 2010; Ben-Arieh et al., 2009), improved product quality and reduced waste (Pirmoradi et al., 2014). With modularity, it is possible to fulfill various functions and product variety through different combinations of modules.

Two types of modular design in product platform configuration can be found, i.e., module selection and module integration. The module selection allows product platform to be configured by choosing modules from a given module set. The module integration allows product platform to be configured with one common module to support a wider range of product requirements.

Introducing more modules in the platform configuration can provide more choices in the combinations between platforms and modules so as to satisfy the wider customer needs (Mikkola, 2007; Otto et al., 2016). However, too many modules may also increase complexity, development

*Partial content of this chapter has been published on the International Journal of Production Research.
Wang, T., Wang, J., Jin, G., & Matsukawa, H. (2022). Product platform configuration decision in NPD with uncertain demands and module options. *International Journal of Production Research*, 1-20.

costs, and manufacturing costs (Ripperda and Krause, 2017) and make demand forecast inaccurate (Wan et al., 2012). Increasing modules may also cause the proliferation of parts and associated costs such as parts procurement costs, module assembling costs, stocking costs, and aftercare costs (Fisher and Ittner, 1999; Salvador et al., 2002; AlGeddawy and ElMaraghy, 2013).

To mitigate these negative effects, the module integration approach is introduced by using one common module to satisfy the customer demand with the higher and lower specifications. Using module integration helps to improve the platform and module commonality, which enables the manufacturers to achieve the risk-pooling benefits in the process of development, procurement, manufacturing, and inventory (Huang et al., 2005). The fewer modules and platforms produce the lower product development cost, manufacturing cost, and inventory cost (Fisher and Ittner, 1999; Zhang et al., 2010; Agrawal et al., 2013). However, using higher specification modules for those customer needs with lower specifications will incur additional overdesign and procurement costs as well as offset the product variety (Krishnan and Gupta, 2001; Wan et al., 2012). Therefore, the important challenge in the product platform configuration problem is how to balance these two types of platform configuration. This challenge was targeted in this chapter.

We consider a modular platform configuration framework in which a set of module types with module options is offered to configure the product platform for an external product family. A module type is a unit to serve an identifiable product function, while multiple module options within a module type represent the differentiation of functions. The combination of modules and module options leads to product differentiation. This is similar to the assumption in other studies (Chakravarty and Balakrishnan, 2001; Zhang et al., 2008; Zhang et al., 2010). The concept of module type with module options is similar to the notion of selective module with module instances (Xiong et al., 2018) and that of replaceable component set with components (Gupta and Krishnan, 1999). Important of this assumption is that we pay a higher procurement cost for the higher specification module option, but the production and assembly cost is the same for the higher and lower module options. The higher

specification module option is allowed to satisfy the customer demand with the lower options in order to implement module integration.

The pertinent questions are:

(1) which product platforms should be developed, and how many quantities should be produced (the product platform configuration problem),

(2) which module options should be selected, and how many quantities should be procured (the module configuration problem).

To deal with the trade-off between the benefits of product platform commonality and the associated costs, we propose an optimization model for product platform configuration while including the cost items throughout the supply chain, i.e., development cost, procurement cost, setup cost for ordering, inventory cost, shortage cost, and salvage cost. A hybrid-search method (HSM) combining simulated annealing (SA) and variable neighbourhood search (VNS) is developed to solve the proposed model. The accuracy and efficiency of the proposed algorithm are evaluated by comparing it with an explicit enumeration algorithm. Moreover, a real case study on motherboard design of personal computer product families was presented to illustrate the proposed model and solution method. Results of the case study indicate that the proposed model and algorithm can effectively support the decision-making of product platform configuration for mass customization products.

3.2 Problem description and assumptions

3.2.1 Nomenclature

Table 3.1 presents the description of symbols used in the model.

Table 3.1 List of symbols

Symbol	Description
i	Index of product platform (or product variant), $i = 1,2,\cdots,I$
j	Index of module, $j = 1,2,\cdots,J$
k	Index of module option, $k = 1,2,\cdots,K_j$
N_{jk}	Design parameters depends on the specification of the module option m_{jk}
lt_{jk}	The procurement lead time of module option m_{jk}
pr_{jk}	The monthly production quantity of module option m_{jk}
d_i	Monthly demand of the product F_i
d_{jk}	Monthly demand of module option m_{jk}
u_{jk}	The annual demand of module option m_{jk}
σ_i	The monthly demand standard deviation of the product F_i
σ_{jk}	The monthly standard deviation of module option m_{jk}
σ_{jk}^{L}	The standard deviation of module option over a replenishment lead time
ρ_{i,i^*}	The correlation coefficient of the product F_i and product F_{i^*}
q_{jk}	Order quantity of module option m_{jk}
$G(z)$	Expected shortage per replenishment cycle
dc_{jk}	Fixed design and procurement cost of module option m_{jk}
pc_{jk}	Unit procurement price of module option m_{jk}
hc_{jk}	Unit inventory holding cost of module option m_{jk}
sc_{jk}	Unit shortage cost of module option m_{jk}
t_{ijk}	Availability parameter for selecting specification of the module option m_{jk} for platform P_i
A	Fixed setup cost for ordering one batch
α	Coefficient of fixed development cost
β	Coefficient of variable procurement cost
z	Safety factor
sl	Service level
b	Fixed delay factor of procurement lead time
$life$	Product lifetime
γ	Capital discount rate to calculate the Net Present Value of PPD cost
y_{jk}	Decision variable, takes value 1 if module option m_{jk} is used
x_{ijk}	Decision variable, indicate whether module option m_{jk} is used in product platform P_i

3.2.2 Problem description and assumptions

As shown in Figure 3.1, we assume that a product family has multiple products $F_i, (i = 1,2,\cdots,I)$ with different functional requirements to satisfy customer needs. Module $m_j, (j =$

$1,2,\cdots,J)$ is a unit with standard interface that provides an identifiable product function and is developed or designed by mechanical engineers and electronic engineers using various techniques, for example, group technology. Two types of modules are involved, i.e., common modules and variant modules. Each variant module has multiple options $m_{jk}, (k = 1,2,\cdots,K_j)$ with different cost-relevant design parameters N_{jk} to represent different function. For example, memory size of 8M, 16M, and 32M are the options for one specific memory module in cellular phone design. The common module only has one module option. In other words, a variant module with one option can be seen as a common module. For the sake of simplicity, we assume that a product F_i is derived from product platform P_i and that platforms with the same module and the associated options are merged into one platform later. We allow platform P_i to be configured as any possible platform according to the selection of modules and the associated options. The number of product platforms P_i and P_{i^*} $(i \neq i^*)$ are two different platforms, however, we set $P_i = P_{i^*}(i \neq i^*)$ later when two platforms have completely the same modules and the same associated options.

Different product platforms can select different module options to satisfy different product specifications. However, it may require a greater numbers of module options to be developed with higher development cost. In opposite, if we develop a single higher specification module option to meet the high-end and low-end customer demand associated with different product platforms, it will bring the higher procurement cost but the lower other costs. There may exist one balance point on the total PPD cost when determining the optimal product platform configuration. Furthermore, we simplify the description of functional requirements of products (number of product families or product brands) to directly map to the requirements for modules with module options. We also assume that all module options are given, and the maximum option number is K_j, for module m_j. The highest requirement of the option level for each module m_j in a set of product platforms is denoted by H_j which satisfies the inequality function $H_j \leq K_j$.

Assumption 1: We sort modules m_j by increasing value of N_{jk} $(N_{jk} \leq N_{j(k+1)})$. Value of

N_{jk} consists of development cost, procurement cost, and so on. This assumption is widely used in previous studies (e.g., Zhang et al., 2008; Van den Broeke et al., 2017). We further assume that the module development cost dc_{jk} of module m_j with option m_{jk} is proportional to the N_{jk}, i.e., $dc_{jk} = \alpha * N_{jk}$, and the procurement cost pc_{jk} of the module m_j with option m_{jk} equal to $\beta * N_{jk}$, where α and β are positive coefficients related to the development cost and procurement cost.

Assumption 2: A product platform P_i with higher specification module option can be used for product F_i with lower specification module option. The higher specification module option could be used to product platform without sacrificing product quality, however product platform cost increases. In the proposed optimization model, we define an availability parameter t_{ijk}. The t_{ijk} is binary variable and satisfies the inequality condition, $t_{ijl} \geq t_{ijk}$, which indicates that module option m_{jk} or higher specifications m_{jl} $(l > k)$ can be used to product platform P_i for deriving product F_i. For example, if we have two products F_1 and F_2 where the product F_1 requires module options m_{11}, m_{21}, m_{32} and the product F_2 requires module options m_{13}, m_{22}, m_{31}, then we have $t_{121} = 1, t_{222} = 1$ for module m_2, consequently $t_{122} = t_{123} = t_{223} = 1$ and $t_{221} = 0$.

Assumption 3: We can purchase all module options from outside. Economic order quantity (EOQ) is applied for procurement using average demand. Demand is uncertain and follows normal distribution. Procurement lead time of module m_j with option k, $lt_{jk}(q_{jk})$, includes production time in the suppliers and a fixed time delay b representing transportation time (Glock and Ries, 2013). In this study, we assume that one module option m_{jk} is purchased from one supplier. Dual sourcing or supply chain risk management is out of scope of this study.

$$lt_{jk}(q_{jk}) = \frac{q_{jk}}{pr_{jk}} + b \tag{3.1}$$

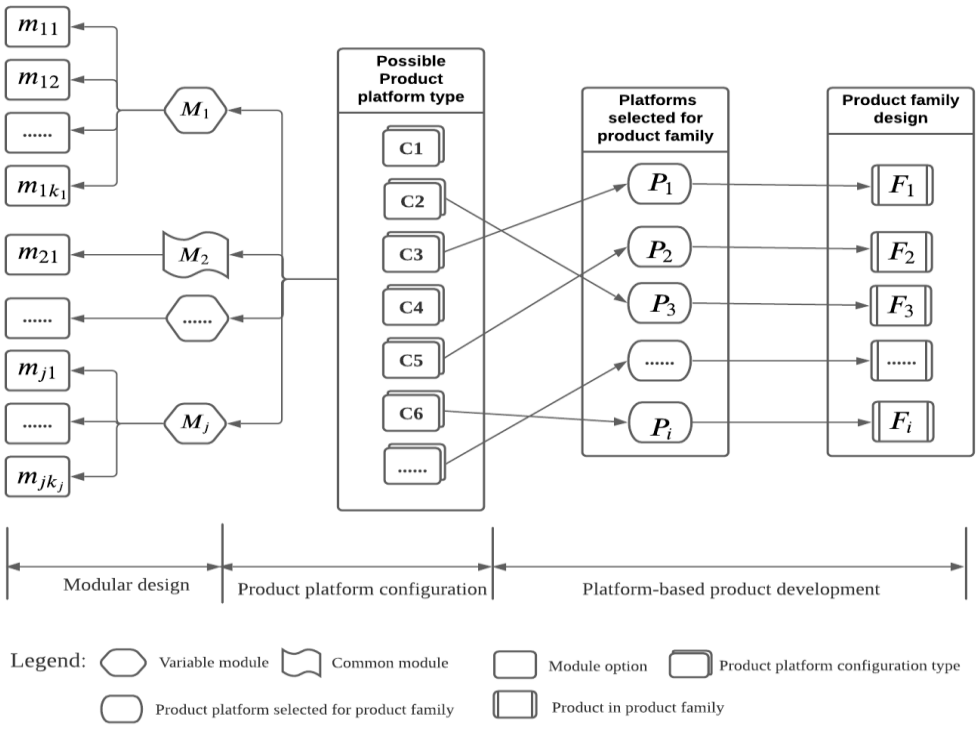

Figure 3.1 Product platform configuration problem with module options

3.3 Model formulation

Two decision variables are used in the proposed model.

(1) x_{ijk}: takes value 1, if module option m_{jk} is used to product platform P_i, otherwise 0.

(2) y_{jk}: takes value 1, if module option m_{jk} used to any product platforms, otherwise takes value 0.

Since each product platform must select one module option m_{jk} at most for any module m_j, it leads to the following constraint.

$$\sum_{k=1}^{K_j} x_{ijk} = 1 \quad (i = 1,2,\cdots,I; j = 1,2,\cdots,J) \tag{3.2}$$

The availability constraint can be written as follows, utilizing the availability parameter t_{ijk}. This constraint ensures one module option is selected at most, allowing a higher specification to replace a lower specification.

$$\sum_{k=1}^{K_j} t_{ijk} \cdot x_{ijk} = 1 \quad (i = 1,2,\cdots,I; j = 1,2,\cdots,J) \tag{3.3}$$

The selected number of module options will not exceed the number of available module options. Therefore, we have

$$H_j \leq K_j \quad (j = 1,2,\cdots,J) \tag{3.4}$$

In our model, we assume that demand quantity of products F_i and F_{i^*} have correlation, and the correlation coefficient is denoted using the symbol $\rho_{i,i^*}(i, i^* \in I; i \neq i^*)$. Assuming that the monthly demand for product follows a normal distribution, i.e., $\mu_i \sim N(d_i, \sigma_i^2)$ for $i = 1,2,\cdots,I$, the monthly demand of the module option $d_{jk} = \sum_{i=1}^{I} x_{ijk} \cdot d_i$ is normally distributed and the variance can be given as below function.

$$\sigma_{jk}^2 = \sum_{i=1}^{I} (x_{ijk} \cdot \sigma_i^2 + \sum_{i*=i+1}^{I} x_{ijk} \cdot x_{i*jk} \cdot 2 \cdot \rho_{i,i*} \cdot \sigma_i^2 \cdot \sigma_{i*}^2) \tag{3.5}$$

Further, we introduce t number of segments on ordering quantity pq_t and set multiple increasing base procurement costs pc_{jk}^t which represents linear quantity discount. This segment essentially represents a piecewise linear approximation of the nonlinear discount function.

$$pc_{jk}(u_{jk}) = \begin{cases} pc_{jk}^1 + r_1(u_{jk} - pq_1) & if\ 0 \leq u_{jk} \leq pq_1 \\ pc_{jk}^2 + r_2(u_{jk} - pq_2) & if\ pq_1 < u_{jk} \leq pq_2 \\ \vdots & \\ pc_{jk}^t + r_t(u_{jk} - pq_t) & if\ pq_{t-1} < u_{jk} \leq pq_t \end{cases} \tag{3.6}$$

In equation (3.6), r_t represents slope when the ordered quantity is located between pq_{t-1} and pq_t. Setting the different value of pc_{jk}^t and r_t, we may present concave discount function as well as convex discount function.

The objective function of this paper is to minimize the total operation cost C_t consisting of six cost terms.

(1) Fixed development cost of module option, $C_{dc}(x)$, depending on the module options selection.

(2) Procurement cost considering quantity discount, $C_{pc}(x)$.

(3) Setup cost for ordering $C_{oc}(x)$.

(4) Inventory cost $C_{ic}(x)$ consisting of holding inventory cost and safety inventory cost.

(5) Shortage cost $C_{sc}(x)$ which depends on the anticipated number of replenishment cycles per year (u_{jk}/q_{jk}) and the Expected Shortage Per Cycle Replenishment (ESPCR). The ESPCR further depends on the standard deviation of demand over lead time σ_{jk}^{L} and the loss function $G(z) = \sigma_{jk}^{L} \cdot \{f_u(z) - z \cdot [1 - F_u(z)]\}$, where $f_u(z)$ is standard p.d.f and $F_u(z)$ is the standard c.d.f of demands (Van den Broeke et al., 2015; Silver et al., 2016, pp 262).

(6) Salvage cost $C_{bc}(x)$ which represents the leftover items that will not be used further at the end of the product lifetime, mainly the waste of safety stock in our model.

Since the PPD problem always has long time horizons (normally more than one year), the capital discount rate must be included when we add up the total cost from an inventory policy (Hillier and Lieberman, 2005, pp.837). We introduce the capital discount rate γ to calculate the Net Present Value (NPV) of total cost. The development cost in a PPD problem usually occurs only once and the other costs are yearly recurring over the expected product lifetime and are discounted using capital discount rate (γ).

We formulate the proposed model as below.

Minimize

$$C_t(x) = C_{dc}(x) + C_{pc}(x) + C_{oc}(x) + C_{ic}(x) + C_{sc}(x) + C_{bc}(x) \tag{3.7}$$

$$C_{dc}(x) = \sum_{j=1}^{J} \sum_{k=1}^{K_j} dc_{jk} \cdot y_{jk} \tag{3.8}$$

$$C_{pc}(x) = \sum_{t=1}^{life} \frac{1}{(1+\gamma)^t} [\sum_{j=1}^{J} \sum_{k=1}^{K_j} (pc_{jk}(u_{jk}) \cdot u_{jk}] \tag{3.9}$$

$$C_{oc}(x) = \sum_{t=1}^{life} \frac{1}{(1+\gamma)^t} (\sum_{j=1}^{J} \sum_{k=1}^{K_j} \frac{A \cdot u_{jk}}{q_{jk}}) \tag{3.10}$$

$$C_{ic}(x) = \sum_{t=1}^{life} \frac{1}{(1+\gamma)^t} [\sum_{j=1}^{J} \sum_{k=1}^{K_j} (\frac{hc_{jk} \cdot q_{jk}}{2} + hc_{jk} \cdot z \cdot \sigma_{jk}^{L}) \tag{3.11}$$

$$C_{sc}(x) = \sum_{t=1}^{life} \frac{1}{(1+\gamma)^t} \left[\sum_{j=1}^{J} \sum_{k=1}^{K_j} (\frac{G(z) \cdot sc_{jk} \cdot u_{jk}}{q_{jk}}) \right] \tag{3.12}$$

$$C_{bc}(x) = \sum_{j=1}^{J} \sum_{k=1}^{K_j} pc_{jk} \cdot z \cdot \sigma_{jk}^{L} \tag{3.13}$$

Subject to

$$lt_{jk}(q_{jk}) = \frac{q_{jk}}{pr_{jk}} + b \quad (j = 1,2,\cdots,J; k = 1,2,\cdots,K_j) \tag{3.1}$$

$$\sum_{k=1}^{K_j} x_{ijk} = 1 \quad (i = 1,2,\cdots,I; j = 1,2,\cdots,J) \tag{3.2}$$

$$\sum_{k=1}^{K_j} t_{ijk} \cdot x_{ijk} = 1 \quad (i = 1,2,\cdots,I; j = 1,2,\cdots,J) \tag{3.3}$$

$$H_j \leq K_j \quad (j = 1,2,\cdots,J) \tag{3.4}$$

$$\sigma_{jk}^2 = \sum_{i=1}^{I} (x_{ijk} \cdot \sigma_i^2 + \sum_{i^*=i+1}^{I} x_{ijk} \cdot x_{i^*jk} \cdot 2 \cdot \rho_{i,i^*} \cdot \sigma_i^2 \cdot \sigma_{i^*}^2)$$

$$(j = 1,2,\cdots,J; k = 1,2,\cdots,K_j) \tag{3.5}$$

$$\sum_{i=1}^{I} x_{ijk} \geq y_{jk} \quad (j = 1,2,\cdots,J; k = 1,2,\cdots,K_j) \tag{3.14}$$

$$\sum_{i=1}^{I} x_{ijk} \leq M \cdot y_{jk} \quad (j = 1,2,\cdots,J; k = 1,2,\cdots,K_j) \tag{3.15}$$

$$u_{jk} = 12 \cdot \left(\sum_{i=1}^{I} x_{ijk} \cdot d_i \right) \quad \left(j = 1,2,\cdots,J; k = 1,2,\cdots,K_j \right) \tag{3.16}$$

$$q_{jk} = \sqrt{\frac{2 \cdot A \cdot u_{jk}}{hc_{jk}}} \quad \left(j = 1,2,\cdots,J; k = 1,2,\cdots,K_j \right) \tag{3.17}$$

$$\sigma_{jk}^{L} = \sqrt{lt_{jk}} \cdot \sigma_{jk} \quad \left(j = 1,2,\cdots,J; k = 1,2,\cdots,K_j \right) \tag{3.18}$$

$$G(z) = \sigma_{jk}^{L} \cdot \{ f_u(z) - z \cdot [1 - F_u(z)] \} \quad \left(j = 1,2,\cdots,J; k = 1,2,\cdots,K_j \right) \tag{3.19}$$

$$y_{jk} \in \{0,1\} \quad \left(j = 1,2,\cdots,J; k = 1,2,\cdots,K_j \right) \tag{3.20}$$

$$x_{ijk} \in \{0,1\} \quad \left(i = 1,2,\cdots,I; j = 1,2,\cdots,J; k = 1,2,\cdots,K_j \right) \tag{3.21}$$

Constraint (3.2) specifies that only one module option m_{jk} for any module m_j can be selected by a product platform. Constraint (3.3) is an availability constraint that ensures assumption 2. Constraint (3.4) ensures that the number of module options used in the product platforms does not exceed the number of module options available at the design level. Constraints (3.14) and (3.15) confine the derivative decision variable y_{jk}, where M is a sufficiently large positive number. Equations (3.1), (3.16), and (3.17) refer to the value of parameters lt_{jk}, u_{jk}, and q_{jk}. The monthly demand variance σ_{jk}^2 and the standard deviation of module option over the replenishment lead time σ_{jk}^{L} are defined in equations (3.5) and (3.18). Equation (3.19) gives the loss function used to calculate the expected shortage. Constraints (3.20) and (3.21) restrict binary decision variables.

3.4 Algorithm development

The PPD problem can be understood as a large-scale combinatorial optimization problem. Since the problem is NP-hard, exact algorithms such as explicit enumeration can be very time-consuming for large-scale problems. To solve it within a reasonable time duration, heuristic algorithms like the simulated annealing algorithm (SA) are recognised as an efficient way (Sadeghi et al., 2011; Van den

Broeke et al., 2015, 2017). In this paper, we develop a new HSM based on simulated annealing and variable neighbourhood search. The core idea of the HSM is to find a local optimum quickly using the SA algorithm and find a global optimum by using variable neighbourhood, acceptance function, and global time varying parameters, which enables escaping from the local optimum. The proposed HSM algorithm includes three steps.

Step 1: Initialization. Set parameter values of the SA algorithm, such as the initial temperature, the number of iterations and steps, and so on. Set the initial product platform configuration decision with a binary variable and assign it to $x_{current}$. Define a set of neighbourhood structures $N^s(s = 1,2,\cdots,s_{max})$ and denote the set of solutions within the neighbourhood N^s with $N^s(x)$. The radius of the neighbourhood r_s is applied to control the neighbourhood size when searching for solutions around the current point, where $r_s = |x - x^s| = \sum_{\forall i,j,k}(|x_{ijk} - x_{ijk}^s|)$. We use a simple difference of the binary variable of current solution x and next feasible solution x^s to calculate the radius. We may use geometric distance here; however, it takes time for calculation.

Step 2: Local search. We apply the basic simulated annealing algorithm to find a local minimum solution x_{local} from $N^s(x)$. In each local search, we first start with an initial local minimum solution x_{local}, which equals the current global minimum solution $x_{current}$. Second, a new feasible solution x^s is created from $N^s(x)$ by generating random numbers corresponding to the solution matrix's element positions according to the different radius r_s and inverting the 0-1 values of these positions. Third, compare the objective value ($cost(x^s)$) associated with the new feasible solution x^s to the local minimum value ($cost(x_{local})$). Whenever the $cost(x^s)$ is less than $cost(x_{local})$ or the acceptable probability is over a random number between 0 and 1, the new feasible solution x^s replaces the local minimum solution x_{local}. Otherwise, the local minimum solution remains unchanged ($x_{local} = x_{local}$). The procedure stops whenever the SA algorithm condition either reaches the total iteration limit or improves saturated. Acceptance function and global time varying parameter will generate a new solution escaping from the local optimum.

Step 3: Neighbourhood change. Compare the global minimum ($cost(x_{current})$) with the local minimum ($cost(x_{local})$) obtained from $N^s(x)$ with local search SA algorithm. If the $cost(x_{local})$ is less than $cost(x_{current})$, then x_{local} replaces $x_{current}$ and neighbourhood N^s returns to N^1 for the next local search. Otherwise, the current solution remains unchanged ($x_{current} = x_{current}$) and the search explores the next neighbourhood N^{s+1} unless $s > s_{max}$. Note that we return to N^1 whenever $cost(x_{local})$ is less than $cost(x_{current})$.

The method repeats step 2 and step 3 until the stopping condition is met and the current solution $x_{current}$ is the global optimal solution to this problem. The pseudo-code of the proposed algorithm is briefly described in Figure 3.2.

Begin
Input: Set neighbourhood structures $N^s(s = 1,2,\cdots,s_{max})$
Set the parameters of SA algorithm, such as initial temperatures, temperature schedule, iteration number at each temperature interval, and so on.
Set the initial platform configuration decision and assign it to $x_{current}$
For $s = 1$ to s_{max} Do
Local search by SA:
$x_{local} = x_{current}$
Repeat:
Repeat:
Generate a new feasible solution x^s from $N^s(x)$ based on x_{local}
If $cost(x^s) \leq cost(x_{local})$
Then $x_{local} = x^s$
Else:
Choose a random probability uniformly in the range (0,1)
If $\exp\left(\frac{(-deltaC)}{T}\right) > p$
Then $x_{local} = x^s$
Else:
$x_{local} = x_{local}$
Until reach iteration number at each temperature interval
Decrease T according to temperature schedule
Until stopping condition is met
If $cost(x_{local}) \leq cost(x_{current})$
Then $x_{current} = x_{local}$
$s = 1$
Otherwise
$s = s + 1$
$x_{current} = x_{current}$
Until $s > s_{max}$
Output: best solution found $x_{current}$

Figure 3.2 Pseudo-code of the proposed algorithm

3.5 Numerical experiments

3.5.1 Parameter setting

A real case study on the motherboard design of personal computer product families was introduced to illustrate the proposed models and solution methods. The case study comes from a survey of two famous PC companies A and B. For the reason of confidentiality of company information, we assumed the parameter settings according to the case study. The motherboards are regarded as the product platforms of PC, and the components of PCs were understood as modules, e.g., processor, RAM, wireless network card, hotkeys, speakers. For example, there are three options for RAM module, i.e., 8GB, 16GB, and 32GB. Deciding on the number of motherboard types is a big challenge for case companies. It is difficult to forecast the demands of each type of motherboard during six months ahead of new product introduction (NPI). However, the development lead time of motherboard is usually longer than six months. The proposed model and method are going to support the problem solution in the case companies. Note that this study focuses on commercial goods for mass customization rather than on investment goods with high value and low volume.

A personal computer product family consists of eight products with different requirements, as enumerated in the right part of Table 3.2. The other parameters, such as the design parameters of module option (N_{jk}) and monthly production quantities, are also shown in Table 3.2. Four demand scenarios are prepared for the numerical examination as shown in Table 3.3. In scenario 1, we assume all expected demand and standard deviation are the same. In scenario 2, we divided products into 4 groups, and each group has a different expected demand and standard deviation. Scenario 3 is essentially the same as scenario 2, while product groups are sorted in the inversing order. In scenario 4, we set expected demand randomly using uniform distribution [120, 600] and set standard deviation

randomly using uniform distribution [10, 80] (see Table 3.3). The other parameters are presented in Table 3.4.

Table 3.2 The description of module and module option and the initial composition of products

Module	Module option (No.)	Module option description	N_{jk}	pr_{jk}	F1	F2	F3	F4	F5	F6	F7	F8
m_1	m_{11} (1)	Electrical mechanical component	4	4000	X	X	X	X	X	X	X	X
m_2	m_{21} (2)	Processor 1	10	3000	X				X			
	m_{22} (3)	Processor 2	16			X				X		
	m_{23} (4)	Processor 3	30				X				X	
	m_{24} (5)	Processor 4	38					X				X
m_3	m_{31} (6)	Controlled Rectifier	3	4000	X	X	X	X	X	X	X	X
m_4	m_{41} (7)	Hotkeys	2.5	4000	X	X	X	X	X	X	X	X
m_5	m_{51} (8)	Wireless network card 1	10	4000	X	X			X	X		
	m_{52} (9)	Wireless network card 2	13				X	X			X	X
m_6	m_{61} (10)	Speaker 1	20	3000	X	X	X	X				
	m_{62} (11)	Speaker 2	25						X	X	X	X
m_7	m_{71} (12)	RAM 1	64	2500	X	X						
	m_{72} (13)	RAM 2	80				X		X	X		
	m_{73} (14)	RAM 3	95					X			X	X
m_8	m_{81} (15)	HDD 1	38	3600	X	X			X	X		
	m_{82} (16)	HDD 2	47				X	X			X	X
m_9	m_{91} (17)	Battery	6	4000	X	X	X	X	X	X	X	X

Table 3.3 Product demand distribution

Demand scenarios	F1	F2	F3	F4	F5	F6	F7	F8
Scenario 1	$N(300,35^2)$	$N(300,35^2)$	$N(300,35^2)$	$N(300,35^2)$	$N(300,35^2)$	$N(300,35^2)$	$N(300,35^2)$	$N(300,35^2)$
Scenario 2	$N(120,20^2)$	$N(240,30^2)$	$N(360,40^2)$	$N(480,50^2)$	$N(120,20^2)$	$N(240,30^2)$	$N(360,40^2)$	$N(480,50^2)$
Scenario 3	$N(480,50^2)$	$N(360,40^2)$	$N(240,30^2)$	$N(120,20^2)$	$N(480,50^2)$	$N(360,40^2)$	$N(240,30^2)$	$N(120,20^2)$
Scenario 4	$N(148,22^2)$	$N(260,32^2)$	$N(302,32^2)$	$N(480,26^2)$	$N(159,12^2)$	$N(297,76^2)$	$N(530,38^2)$	$N(224,42^2)$

Table 3.4 Cost items and other input parameters

No.	Parameter	Symbol	Value
1	Capital discount rate	γ	10%
2	Product lifetime	$life$	2 years
3	Development coefficient	α	3000
4	Procurement cost coefficient	β	1
5	Setup cost for ordering one batch	A	200
6	Fixed delay factor	b	0.1
7	Unit inventory holding cost	hc_{jk}	30%-unit procurement cost
8	Unit shortage cost	sc_{jk}	Unit procurement cost
9	Services level	sl	95%
10	Unit procurement cost under t^{th} line segment	pc_{jk}^t	Common module $pc_{jk}^t = pc_{jk}$ Variable module $pc_{jk}^t = pc_{jk} - (t-1)$
11	Quantity discount slope	r_t	$0 < u_{jk} \leq 5000$: $r_1 = 0$ $5000 < u_{jk} \leq 10000$: $r_2 = -0.0002$ $10000 < u_{jk} \leq 20000$: $r_3 = -0.0001$ $20000 < u_{jk}$: $r_4 = -0.00001$

To manifest the benefits of PPD, we compare the results with independent product development. The independent approach is a particular case in which the products in a product family are developed independently without considering the combination of module selection and integration. In our case, all 9 modules and 17 module options are developed to satisfy 8 products. The PPD approach combines module selection and integration so as to choose cost-efficient product platforms satisfying all

specifications of the 8 products and their demands. The solution algorithm is coded using Python and runs on a PC with intel CPU, 1.80 GHz, and 8 GB of RAM DDR.

3.5.2 Experiments on the proposed algorithm

To demonstrate the accuracy and effectiveness of the proposed algorithm, we performed an explicit enumeration algorithm to compare it with the proposed Hybrid-Search Method (HSM). The explicit enumeration algorithm can provide an accuracy solution. However, it is very time-consuming when dealing with large-scale problems. Therefore, we use a small-scale problem in this research first. For the large-scale problem, we did not find an existing algorithm for benchmark, while we failed to get a solution using commercial software such as LINDO and etcetera. Furthermore, we are interested in the idea of a PPD model supported by module selection and integration, not the speed of the solution algorithm.

The comparative results of the two algorithms are reported in Table 3.5. We tested three different problem instances, varying the number of products, the number of modules, and the number of module options. The possible number of platforms is the number of product platforms configured without any constraints.

As shown in Table 3.5, we can see that the proposed HSM algorithm can provide a practically near-optimal solution and provide optimal solutions in most cases. For small-scale problems, the explicit enumeration algorithm finds the optimal solution faster, but it becomes time-consuming for large-scale problems.

Table 3.5 Comparative result of explicit enumeration algorithm and proposed algorithm

Problem instance (i,j,k)	Possible platform configuration for a product family	Enumeration algorithm		Proposed algorithm (Initial temperature = 30000; Temperature update factor = 0.98)					
				Iteration max at one temperature= 200 Neighborhood number=1		Iteration max at one temperature = 300 Neighborhood number = 2		Iteration max at one temperature = 400 Neighborhood number = 3	
		Time(s)	Optimal cost	Time(s)	Cost gap (%)	Time(s)	Cost gap (%)	Time(s)	Cost gap (%)
4-3-7	20736	2.68	3240405.05	4.48	0.00%	9.09	0.00%	14.04	0.00%
5-4-9	7962624	57.86	5687995.63	11.91	0.00%	18.86	0.00%	25.72	0.00%
6-5-11	12230590464	746.48	8169260.74	12.59	0.18%	32.15	0.00%	42.12	0.00%
8-5-11	2.81793E+13	3480.95	11684133.44	41.08	0.95%	104.21	0.00%	128.50	0.00%
10-4-9	6.34034E+13	6153.19	12279936.52	45.79	1.57%	167.26	0.00%	208.70	0.00%

3.5.3 Experimental results of case study

Table 3.6 shows the comparison results of operation cost contents for different product development approaches under different scenarios. The platform-based approach has a lower total operation cost than independent product development. More specifically, the platform-based approach leads to lower development cost, lower setup cost for ordering, lower inventory holding cost, lower safety inventory cost, and lower salvage cost while it increases total procurement cost and shortage cost.

The module integration may select higher specification module options in platform configuration so that to reduce the total number of module options, which leads to a lower development cost. Meanwhile, demand aggregation driven by module integration leads to a higher volume of module options, enabling economies of scale and volume discounts. In this paper we consider setup cost and inventory cost, therefore the high volume may reduce ordering times when purchasing modules from suppliers. Similarly, smaller number of modules and high volume will reduce demand deviation (sum of demand deviation is larger than the total demand deviation), and the reduction may lead to lower safety inventory costs and salvage costs. We sometimes call it the benefits of risk-pooling. On the other hand, module integration in the PPD approach accompanies

small additional costs due to over-design enabling product platforms sever for the products with the higher and lower specification module options, which increases the procurement cost and shortage cost.

Table 3.6 Results of operation cost contents

Scenario	Solution	Total cost	Development cost	Procurement cost	Setup cost for ordering	Holding inventory cost	Safety inventory cost	Shortage cost	Salvage cost
S1	Independent	11970696.26	1503000.00	10223594.18	94287.15	94287.15	16282.10	12390.56	26855.12
	PPD	11942714.17	1221000.00	10492455.27	87423.13	87423.13	15518.67	13448.23	25445.74
S2	Independent	12577223.20	1503000.00	10825253.62	94170.49	94170.49	17388.38	14517.11	28723.11
	PPD	12448096.39	1161000.00	11059300.22	84446.68	84446.68	16484.44	15417.33	27001.03
S3	Independent	11294443.67	1503000.00	9554463.26	91402.83	91402.83	16053.36	11828.03	26293.37
	PPD	11272765.28	1413000.00	9626599.40	89588.45	89588.45	15943.98	11978.24	26066.77
S4	Independent	12410826.86	1503000.00	10657588.23	94397.34	94397.34	17874.01	14017.94	29551.98
	PPD	12306928.84	1251000.00	10820793.50	87432.26	87432.26	17200.15	14861.65	28209.02

We then illustrate the result of product platform configurations for the externally given product family under each scenario in Table 3.7, in which the circle number represents the higher specification module option used to the platforms than product requirements. The optimal platform strategy configures 7 product platforms and selects 15 module options for a product family with 8 products in scenario 1, 5 product platforms with 13 module options in scenario 2, 7 product platforms with 16 module options in scenario 3, and 6 product platforms with 14 module options in scenario 4.

More specifically, under scenario 1, product platforms P_1 and P_2 use higher specification module option No.13 (i.e., m_{72}) instead of option No.12 (i.e., m_{71}) of products F_1 and F_2. Product platforms P_3 and P_7 choose higher specification module option No.5 (i.e., m_{24}) in place of option No.4 (i.e., m_{23}) while other selection of module options remains unchanged. Two module options No.4 and 12 (i.e., m_{23} and m_{71}) are not used to the platform configuration under scenario 1, which reduces the total number of module options to 15. Meanwhile, products F_7 and F_8 are derived from one product platform since platforms P_7 and P_8 have the same module options, which results in the total number of product platforms being 7.

Similarly, four module options No.2,4,8,12 (i.e., $m_{21}, m_{23}, m_{51}, m_{71}$) are not used to the product platform under scenario 2. Product F_1 and F_2 are derived from one platform ($P_1 = P_2$), product F_5 and F_6 are derived from one platform ($P_5 = P_6$), and product F_7 and F_8 are derived from the same platform ($P_7 = P_8$). The optimal product platform configurations developed 5 product platforms to serve the external product family. Under scenario 3, two product platforms (i.e., platforms P_7 and P_8) have the same selection of module options while only one module option No.4 (i.e., m_{23}) is not used to the platform configuration. Three module options No.2,8,12 (i.e., m_{21}, m_{51}, m_{71}) are replaced while 6 product platforms are configured to derive the product family under scenario 4. The specific platform configuration and the decision of module selection and integration under scenarios 2, 3, and 4 are shown in Table 3.7.

Table 3.7 Result of platform configurations for a product family

Platform	Initial product requirements	S1	S2	S3	S4
P1	[1,2,6,7,8,10,12,15,17]	[1,2,6,7,8,10,⑬,15,17]	[1,③,6,7,⑨,10,⑬,15,17]	[1,2,6,7,8,10,12,15,17]	[1,3,6,7,⑨,10,⑬,15,17]
P2	[1,3,6,7,8,10,12,15,17]	[1,3,6,7,8,10,⑬,15,17]	[1,3,6,7,⑨,10,⑬,15,17]	[1,3,6,7,8,10,12,15,17]	[1,3,6,7,⑨,10,⑬,15,17]
P3	[1,4,6,7,9,10,13,16,17]	[1,⑤,6,7,9,10,13,16,17]	[1,⑤,6,7,9,10,13,16,17]	[1,⑤,6,7,9,10,13,16,17]	[1,4,6,7,9,10,13,16,17]
P4	[1,5,6,7,9,10,14,16,17]	[1,5,6,7,9,10,14,16,17]	[1,5,6,7,9,10,14,16,17]	[1,5,6,7,9,10,14,16,17]	[1,5,6,7,9,10,14,16,17]
P5	[1,2,6,7,8,11,13,15,17]	[1,2,6,7,8,11,13,15,17]	[1,③,6,7,9,11,13,15,17]	[1,2,6,7,8,11,13,15,17]	[1,③,6,7,⑨,11,13,15,17]
P6	[1,3,6,7,8,11,13,15,17]	[1,3,6,7,8,11,13,15,17]	[1,3,6,7,⑨,11,13,15,17]	[1,3,6,7,8,11,13,15,17]	[1,3,6,7,⑨,11,13,15,17]
P7	[1,4,6,7,9,11,14,16,17]	[1,⑤,6,7,9,11,14,16,17]	[1,⑤,6,7,9,11,14,16,17]	[1,⑤,6,7,9,11,14,16,17]	[1,4,6,7,9,11,14,16,17]
P8	[1,5,6,7,9,11,14,16,17]	[1,5,6,7,9,11,14,16,17]	[1,5,6,7,9,11,14,16,17]	[1,5,6,7,9,11,14,16,17]	[1,5,6,7,9,11,14,16,17]
Number of module options	17	15	13	16	14
Number of platforms	8	7	5	7	6

Moreover, module integration is not always beneficial for a company. In Figure 3.3, we visualize how the total cost varies with the number of module options. The reduction in module option numbers increases the module integration level. We use the triangles in Figure 3.3 to mark the optimal platform decisions in each scenario and use the small numbers on the cost curves to represent the number of configured product platforms. We find that as the number of module options decreases, the total cost may decrease at the beginning, and then increase after reaching the minimum. Meanwhile, using only 9 or 10 module options to configure 1 or 2 product platforms contributes to the highest cost. The increased cost of using high specification modules in platforms to meet the low-end products may be

compensated by a reduced cost from module integration. The presented U-shaped cost curves reveal a trade-off relationship between module selection and integration in the PPD.

In addition, compared with using 11 module options, the total cost increases significantly when using 10 module options under scenarios 1, 2 and 4. While a larger cost increment is incurred when using 11 module options instead of 12 module options under scenario 3. There is a threshold in the change of cost increment when integrating modules. After the threshold, if reducing one module option, there may cause much more cost increment than the operations before the threshold. One reason may be that the reduced cost in development, setup, inventory, and salvage from the module integration is far less than the increased additional over-design cost. Meanwhile, the cost increment differs only slightly within a certain range. For example, there are approximate total costs when the number of module options used in the PPD is 14, 15, or 16 under scenario 1. Using more module selections does not significantly increase the total cost. Regarding the change in total cost, it is possible to provide more module selection and avoid significantly increased cost during module integration before thresholds.

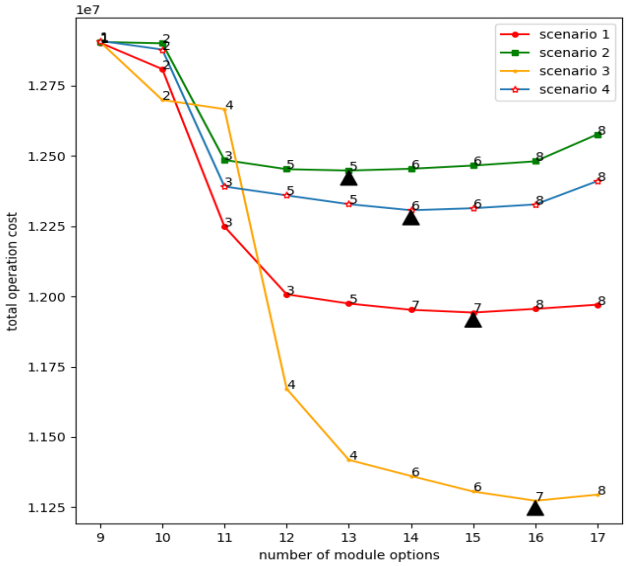

Figure 3.3 The operation cost varying with the number of module options

3.6 Sensitivity and robust result analysis

3.6.1 Sensitivity analysis

A sensitivity analysis of various input parameters was conducted to measure their effect on platform configuration decision. All input parameters were compared to those in scenario 4, shown in Table 3.3 and Table 3.4. The results of sensitivity analysis are plotted in Figures 3.4-3.8.

As shown in Figure 3.4, the higher unit development cost of modules tends to favour higher module integration and use fewer module selections. Compared to scenario 4 with a development cost parameter α equal to 3000, a lower development cost parameter, i.e., $\alpha = 2100$, results in more module options being used. The number of module options is 15 while 14 module options are used in scenario 4. A higher development cost parameter, $\alpha = 3900$, reduces the number of module options to 12, while 5 product platforms are configured instead of 6 in scenario 4. Increasing development cost leads to fewer module selections and higher module integration in order to achieve the benefit of platform commonality.

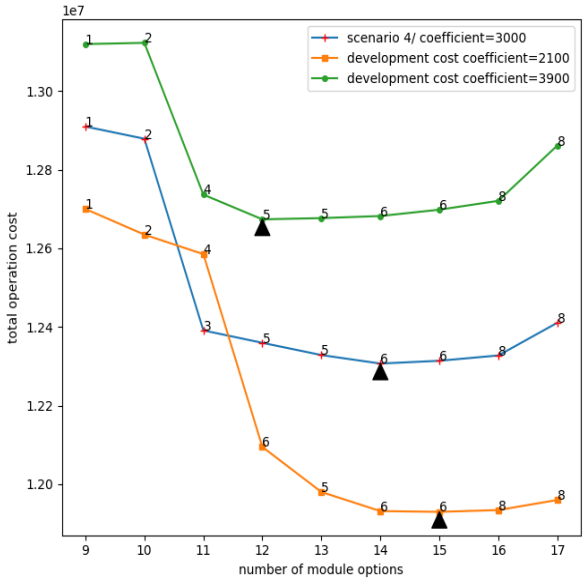

Figure 3.4 The impact of development cost

The effect of product lifetime on platform configuration is analysed. As shown in Figure 3.5, a longer product lifetime tends to provide more module selection while reducing module integration and increasing the number of product platforms. For example, when the product lifetime increases from 1 year to 2 years, 14 module options are used to configure 6 product platforms instead of 10 module options and 2 product platforms. When the product lifetime becomes longer, i.e., from 2 years to 3 years, the number of module options increases from 14 to 15 while the number of product platforms remains at 6. The reason may be that the increasing development cost incurred by introducing more module selection can be compensated by the recurring procurement cost, ordering cost, and inventory cost over a longer product lifetime.

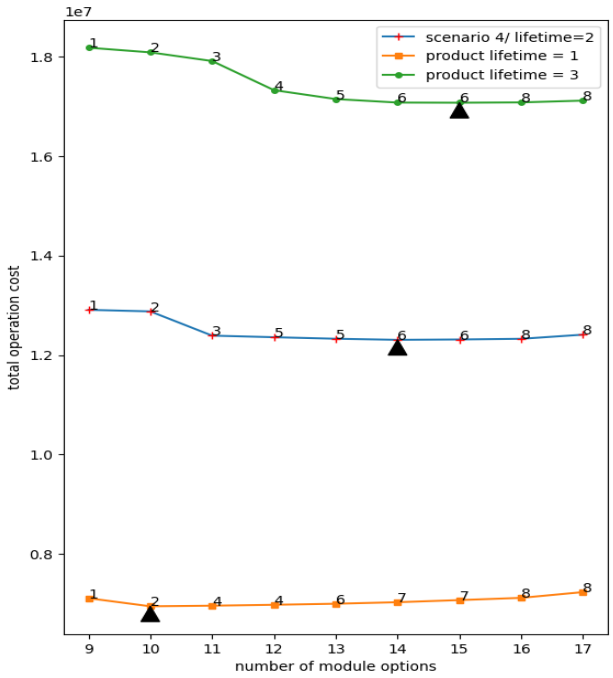

Figure 3.5 The impact of product lifetime

In the case of larger product demand, the number of product platforms and module options tends to be larger. As shown in Figure 3.6, compared to scenario 4 with 6 product platforms and 14 module options, 16 module options are used to configure 8 product platforms due to a 50% increment in demand mean value and standard deviation. With a 50% reduction in demand mean value and

standard deviation, the number of platforms reduces from 6 to 2, while the number of module options decreases from 14 to 10. The increasing demand brings more significant economies of scale, resulting in lower development costs and enabling more module selection.

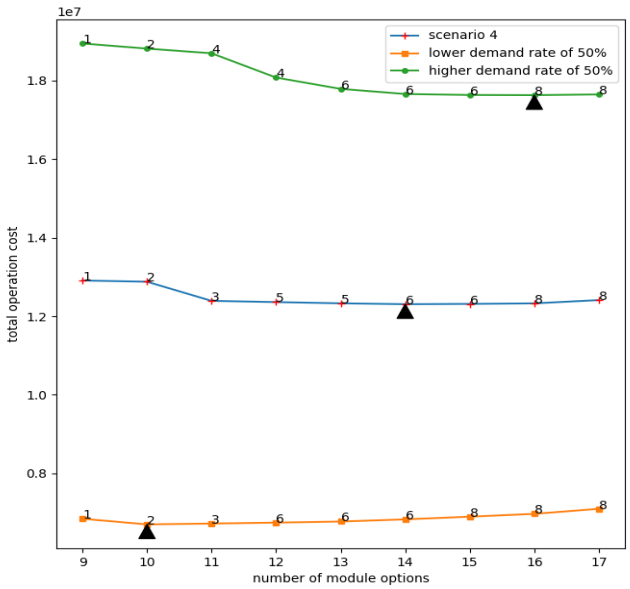

Figure 3.6 The impact of demand

The quantity discounts impacting on procurement cost was analysed. As shown in Figure 3.7, a higher quantity discount tends to more module selection. For example, if the quantity discount slope is increased by twice, the number of module options may increase from 14 to 15. In the case of a larger quantity discount slope (i.e., four times), the number of module options increases to 16. However, the larger quantity discount does not affect the platform configuration decision, where the number of platforms remains 6.

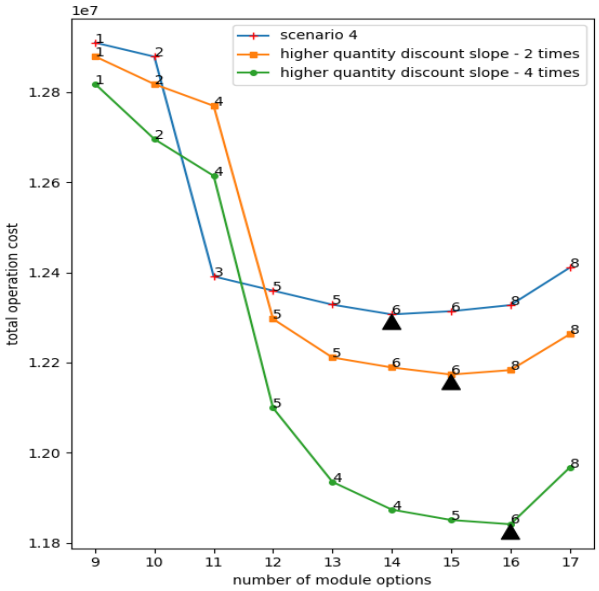

Figure 3.7 The impact of quantity discount

The results of other input parameters, such as service level, inventory holding rate, and unit ordering cost, are shown in Figure 3.8. In our case, we did not find a significant effect of these parameters on the platform configuration decision.

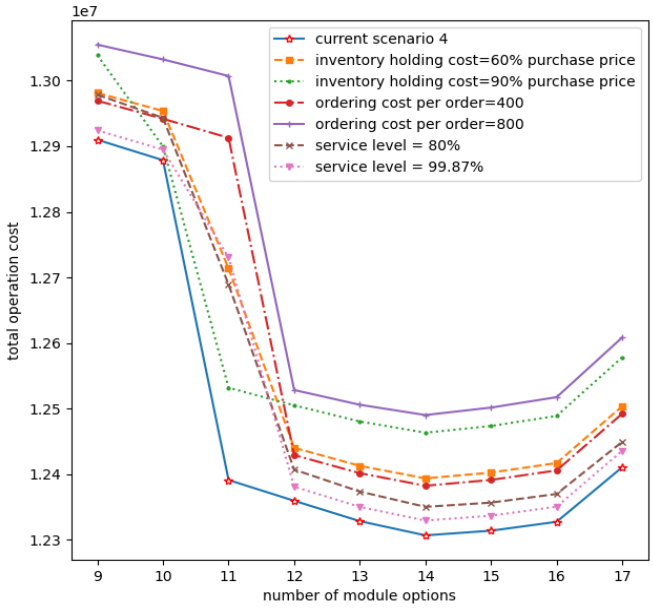

Figure 3.8 The impact of other input parameters

3.6.2 Robustness of results with respect to demand change

We test the robustness of our results with respect to varying demand. Figure 3.3 shows the different optimal decisions on module integration and platform configuration in four kinds of demand scenarios. Their respective optimal solutions were marked with triangles in the figures. Although there are different optimal decisions in different scenarios, these decisions may not change so greatly. It means that the module options to configure product platforms in different demand scenarios are slightly different in total cost. For example, the optimal solution is 15 module options in scenario 1. For scenario 2, the total cost of using 15 module options to configure 6 platforms is only 0.14% different compared to its optimal solution. Similarly, the cost gap between the solution with 15 module options and the optimal solution is 0.29% for scenario 3, and 0.06% for scenario 4. The cost gap is calculated by applying the total cost of the optimal solution under each demand scenario in Table 3.6 as a baseline. Since these four scenarios stand for four typical ones, it may suggest that the solutions from the proposed model may have robust performance in different scenarios.

We further examine the robustness of results on the module integration with the increasing and decreasing mean value and standard deviation under the same demand scenario, as shown in Figure 3.6. Compared to the respective optimal solutions, using 14 module options in the lower demand with half demand mean value and standard deviation has a 1.91% gap in the total cost, while the cost gap between it and the higher demand with 1.5 times of mean value and standard deviation is different by 0.16%. It suggests that developing 14 module options to configure platforms may provide a robust solution to cope with demand fluctuations.

3.7 Conclusions

In this chapter, a new model was proposed to determine the optimal platform configuration for the given product family while considering module selection and integration. More module selection

can provide the special module to meet specific customer demands, whereas module integration may use a single common module to satisfy multiple customer demands. When satisfying the high-end and low-end demands with one higher specification module, it would waste some functions of the module, but facilitate product platform commonality to gain scale economic benefits. By developing the model on the total operation cost of module acquisition, we obtained the optimal decision of platform configuration with module options while balancing module selection and integration. To solve our model, a HSM that combines SA algorithm and VNS was developed. The proposed model was evaluated through numerical examination, in which our algorithm can generate good solutions for different scenarios of parameter setting.

The economic performance of the PPD approach is examined. The numerical study shows that the total cost first decreases and then increases after reaching a minimum as the module integration increases. The changes in the total cost may be caused by the balance between different types of costs. More module integration can facilitate the platform commonality and reduce costs in development, setup, inventory, and salvage. However, integrating modules will bring higher over-design costs and higher procurement costs. Therefore, manufacturers must not over-pursue module integration in order to obtain the platform commonality benefits. Further analysis found that there is a threshold in the change of cost increment as more module integration occurs. One reduced module option can incur a larger cost increment after the threshold than before it. This finding provides insights into how to manage the trade-offs between module selection and integration in the PPD. Manufacturers need to identify the cost change threshold and avoid the module integration with a larger cost increment. In addition, more module selection can be offered to customers when the cost increase is acceptable. This finding can support the PPD approach to satisfy the diversity of customer needs.

The sensitive analysis shows that serval input parameters noticeably affect the platforms and module decisions. The increasing demand and the longer product lifetimes favour more product platforms and encourage module selections instead of module integration. The higher development

cost will reduce the number of product platforms and encourage module integration while restricting the module selection.

Finally, our study reveals the robustness of our results under the different demand scenarios and the same demand scenario with varying mean value and standard deviation. By using our model, we may find a robust solution for module integration and platform configuration to cope with demand fluctuations.

Chapter 4 Optimal platform configuration decision considering platform design strategy and supplier selection

4.1 Introduction

As one of the effective tools to implement mass customization, platform-based product development (PPD) has received increasing attention from academia and industry. Manufacturers implement the PPD approach to produce various products while obtaining benefits such as reduced development time and system complexity, reduced development cost and production costs, and improved ability to upgrade products (Simpson, 2004). The key in the PPD is product platform configuration (PPC). Two critical research problems in the PPC are: (1) how many and which type of product platform should be developed for a product family; (2) which product platform will be assigned to derive the product within a product family.

A product platform can be configured as a matching-designed platform that exactly complies with different product requirements. It may require more platforms to be developed for one product family, thus incurring a higher platform development cost. On the other hand, a platform can be configured as an under-designed or over-designed platform. When deriving a high-end product based on an under-designed platform, additional platform customization will be needed to update the

*Partial content of this chapter has been published on the Journal of Advanced Mechanical Design, Systems, and Manufacturing.
Wang, T., Wang, J., & Matsukawa, H. (2022). Integrating optimal configuration of product platform and supplier selection in mass customization. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, *16*(5), 1-15

required platform functions. When deriving a low-end product based on an over-designed platform, some functions on the platforms may be wasted. How to configure a set of proper product platforms for a product family is a crucial problem in the PPD.

After fixed the platform configuration, the modules are usually procured from suppliers. Different suppliers may provide different prices for modules. The relevant procurement decision can affect the design and development of product platform (Pirmoradi et al., 2014). Early involvement of suppliers at the design stage can improve the consistency between product design and manufacturing or supply process, so as to increase product profitability, reduce lead time, and improve quality (Zhang et al., 2009; Tan et al., 2022). Involving the supplier selection into the platform configuration will better facilitate the implementation of the PPD approach.

In this chapter, we develop a mathematical model to configure multiple platforms for a product family and the relevant supplier selection decision. A set of modules with multiple module options are offered to support different product functional requirements. The combination of module options from different modules supports the possible platform configurations.

The pertinent questions are targeted as follows.

(1) How many product platforms should be developed for one product family and what module options would be selected to constitute the product platforms (Platform configuration problem)?

(2) Which supplier should be selected for the procurement of module options (Supplier selection problem)?

To deal with the cost trade-off between platform development and customization in PPD, we quantify the total cost including the development cost of platforms, the sourcing cost including the procurement cost of module options, the ordering cost and inventory cost of module options based on classical Economic Order Quantity (EOQ) model, and the production customization cost to derive products from platforms. The proposed model is formulated as a nonlinear mixed-integer programming model. Several linearization methods are applied to linearize this model and a solution

method based on the commercial solver Gurobi is developed to solve this model.

4.2 Problem description and assumptions

4.2.1 Nomenclature

Table 4.1 presents the description of symbols used in the model.

Table 4.1 List of symbols

Notation	Definition
v	The index of product variants F_v $(v = 1,2,\dots,V)$ in a product family
i	The index of product platform P_i $(i = 1,2,\dots, I)$
j	The index of module m_j $(j = 1,2,\dots, J)$
k	The index of module option m_{jk} $(k = 1,2,\dots, K_j)$
s	The index of supplier S_s $(s = 1,2,\dots,S)$
N_{jk}	The design parameter of module option m_{jk}
dc_i	Variable development cost of product platform P_i
dc_{fix}	Fixed development cost of product platform
pc_{jks}	Unit purchasing cost of module option m_{jk} purchased from supplier S_s
h_{jks}	Unit inventory holding cost module option m_{jk} purchased from supplier S_s
q_{jks}	Order quantity of module option m_{jk} purchased from supplier S_s
u_v	Annual demand of product variant F_v
d_i	Annual demand of product platform P_i
d_{jk}	Annual demand of module option m_{jk}
α	Coefficient of variable development cost
β	Coefficient of purchasing cost
A	Fixed setup cost for ordering one batch
f	Unit customization cost related to assembly, disassembly, testing and so on
r_{vjk}	The product requirement for module option m_{jk}
δ_{vij}	Binary variable, takes value 1 if platform P_i is under-designed for module m_j when deriving product F_v
SSR	Set of selection rules $(m_{jk}, m_{j'k'})$ which represents that selection of module option m_{jk} requires module option $m_{j'k'}$ in the same configuration
SIR	Set of incompatible rules $(m_{jk}, m_{j^*k^*})$ which represents that module option m_{jk} and module option $m_{j^*k^*}$ cannot be used together in the same configuration
e_{vi}	Binary decision variable to indicate whether product F_v is derived from product platform P_i
w_i	Derivative binary variable to indicate whether product platform P_i is developed
x_{ijk}	Binary decision variable to indicate whether module option m_{jk} is used in product platform P_i
y_{jk}	Derivative binary variable to indicate whether module option m_{jk} is used
d_{jks}	Decision variable, the number of module option m_{jk} that is purchased from supplier S_s

4.2.2 Problem description

A product family has multiple products F_v $(v = 1,2,..,V)$ with different functional requirements. A module m_j $(j = 1,2,\dots,J)$ is a unit that serves an identifiable product function and is developed by engineers using various design methods. Two types of modules can be found, i.e., a variant module and a common module. Each variant module has multiple options m_{jk} $(k = 1,2,\dots,K_j)$ with different cost-relevant design parameters N_{jk} to represent different functional levels. A common module only has one module option and may be required by each product. For example, a personal computer product has a memory module to provide the storage function of computation data which has two module options, i.e., 8GB and 16GB memory sticks. The number of possible platform configurations depends on the combination of products, modules, and module options. For example, a product with four modules and three module options for each module may have $3^4 = 81$ possible platform configurations. A product family consisting of five products may produce $(3^4)^5 + (3^4)^5 + (3^4)^5 + (3^4)^5 + (3^4)^5 = 1.74339 \times 10^{10}$ possible platform configurations.

In the platform-based product development, each product F_v in a product family is derived from one product platform. A matching platform has the same quantity of module options compared to the module option quantity required by products. In contrast, a non-matching platform can be configured as an under-designed or over-designed platform, which has the module options that respectively lower specification or higher specification than the product requirements. For example, a product family contains three products with the respective requirements. Product F_1 requires module option m_{11}, m_{21}, product F_2 requires module options m_{12}, m_{22} and product F_3 requires module options m_{11}, m_{22}. If we configure a platform P_1 with module options m_{12}, m_{21}, then platform P_1 is over-designed for module m_1 when deriving product F_1. This is because that product F_1 has a lower specification module option m_{11} than module option m_{12} used on platform P_1. Likewise, it

is an under-designed platform for module m_2 when deriving product F_2. Similarly, platform P_1 is over-designed for module m_1 and is under-designed for module m_2 when deriving product F_3, which we defined it as a hybrid platform.

Components of all modules and module options will be purchased from outside suppliers. A set of suppliers S_S $(s = 1,2,..,S)$ is offered to illustrate the supplier selection problem. We assume that each supplier S_S can only provide one kind of module, namely j^s, and it can produce serval module options for this module, depending on its capability. The supplier selection process during platform configuration decision for a product family is illustrated as the following Figure 4.1.

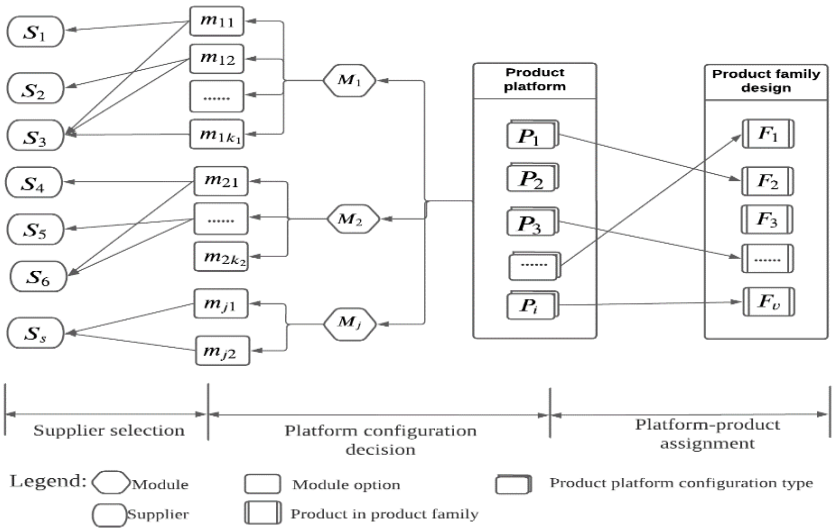

Figure 4.1 Module-based product platform configuration model considering supplier selection

4.2.3 Assumptions

Assumption 1: we sort different module options m_{jk} for module m_j in an increasing value of design parameter N_{jk} $(N_{jk} \leq N_{j(k+1)})$. The value of N_{jk} is corresponding to the different functional levels and is further assumed to be related to the costs of module options. This assumption is widely used in the previous studies (Chakravarty and Balakrishnan, 2001). The variable development cost dc_{jk} of module option m_{jk} used on the platform equal to $\alpha \cdot N_{jk}$ and the unit procurement cost

pc_{jk} of module option m_{jk} equal to $\beta \cdot N_{jk}$. A product platform with a higher specification of module option may have a higher development cost and higher procurement cost.

Assumption 2: We assume that the customization is only incurred when a product is derived from an under-designed platform. In the case of a matching or over-designed platform, there is no customization. In other words, we assume that the higher specification module option could be used for product platform without sacrificing product quality. However, the procurement cost of module option will increase if the low-end product is derived from an over-designed platform. We use the design parameter to calculate the performance gap for a particular module m_j between product F_v and platform P_i as follows.

$$\max\left(0, \sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk}\right) \qquad \forall v, i, j \tag{4.1}$$

Furthermore, we define a binary variable δ_{vij}, which takes value 1 if the platform P_i is under-designed for module m_j when deriving product F_v. The larger the performance gap, the more customization is required to derive the product from this platform.

For illustration, we present a simple example to show the calculation of the performance gap. Suppose a module m_1 has three module option m_{11}, m_{12}, m_{13} with respectively design parameters $N_{11} = 15, N_{12} = 21, N_{13} = 27$. Module m_2 contains two module options with respectively design parameters $N_{21} = 20, N_{22} = 28$. Product F_1 requires module options m_{13} and m_{21}. When product F_1 is derived from platform P_2 with module options m_{11}, m_{22}, then the performance gap for module m_1 equal to $\max(0, 27 - 15) = 12$ and the performance gap for module m_2 equal to $\max(0, 20 - 28) = 0$. In this case, platform P_2 is under-designed for module m_1 and is over-designed for module m_2 when deriving product F_1.

Assumption 3: We assume that we have module suppliers outside therefore we can purchase all module options we need. The economic order quantity (EOQ) model is applier for procurement by the manufacturer. It is widely acknowledged that the EOQ model is a good representation of a firm

action and is widely used in many firms for determining their inventory levels. Due to the complexity of our model, we do not consider a more complex inventory policy in our model.

4.3 Model formulation

In order to determine the optimal product platform configuration, a binary decision variable x_{ijk} is used to show whether to select module option m_{jk}. It equals to 1 if the module option m_{jk} is selected on the platform P_i, or 0 if not selected. The assignment decision between product and platform is denoted by a binary decision variable e_{vi}. It equals to 1 if product F_v is derived from platform P_i, or 0 otherwise. A derivative variable w_i is used to determine whether the product platform P_i is developed or not. Moreover, the supplier selection decision is denoted by the variable d_{jks}, which determines the number of module option m_{jk} purchased from supplier S_s.

The objective function of the proposed model is to minimize the total cost, including the development cost of product platforms C_d, the souring cost of module option C_s, and the production customization cost for deriving product from platform C_c. The development cost of product platforms contains two parts, the variable development cost associated to the selection of module options on the product platform and the fixed development cost depending on the number of product platforms developed. The sourcing cost of module option further includes the procurement cost, the ordering cost and inventory cost based on the EOQ model. The total customization cost to transform platform into product depends on the performance gap for all under-designed module options and the product demand. A unit customization cost $f \geq 0$ is included to represent the impact of the performance gap on the customization cost.

The proposed platform configuration model is thus formulated as follows.

Minimize $C_t = C_d + C_s + C_c$

$$C_d = \sum_{i=1}^{I} dc_i \cdot w_i + \sum_{i=1}^{I} dc_{fix} \cdot w_i \tag{4.2}$$

$$C_s = \sum_{s=1}^{S} \sum_{j^s=1}^{J^s} \sum_{k=1}^{K_j} (pc_{jks} \cdot d_{jks} + \frac{A \cdot d_{jks}}{q_{jks}} + \frac{hc_{jks} \cdot q_{jks}}{2}) \tag{4.3}$$

$$C_c = \sum_{v=1}^{V} \sum_{i=1}^{I} \sum_{j=1}^{J} [e_{vi} \cdot \left(\sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk} \right)^{+} \cdot u_v \cdot f] \tag{4.4}$$

Subject to

$$\sum_{i=1}^{I} e_{vi} = 1 \ \ (v = 1,2, \dots, V) \tag{4.5}$$

$$w_i = \begin{cases} 1, & \sum_{v=1}^{V} e_{vi} > 0 \\ 0, & \sum_{v=1}^{V} e_{vi} = 0 \end{cases} \tag{4.6}$$

$$\sum_{k=1}^{K_j} x_{ijk} = w_i \quad (i = 1,2, \dots, I; j = 1,2, \dots, J) \tag{4.7}$$

$$x_{ijk} \leq w_i \quad \left(i = 1,2, \dots, I; j = 1,2, \dots, J; k = 1,2, \dots, K_j\right) \tag{4.8}$$

$$y_{jk} = \begin{cases} 1, & \sum_{i=1}^{I} x_{ijk} > 0 \\ 0, & \sum_{i=1}^{I} x_{ijk} = 0 \end{cases} \tag{4.9}$$

$$d_i = \sum_{v=1}^{V} e_{vi} * u_v \quad (i = 1,2, \dots, I) \tag{4.10}$$

$$d_{jk} = \sum_{v=1}^{V} \sum_{i=1}^{I} x_{ijk} \cdot e_{vi} \cdot u_v + \sum_{v=1}^{V} \sum_{i=1}^{I} r_{vjk} \cdot \delta_{vij} \cdot e_{vi} \cdot u_v - \sum_{v=1}^{V} \sum_{i=1}^{I} \delta_{vij} \cdot e_{vi} \cdot u_v \cdot x_{ijk}$$

$$\left(j = 1,2, \dots, I; k = 1,2, \dots, K_j\right) \tag{4.11}$$

$$d_{jk} = \sum_{s=1}^{S} d_{jks} \ \left(j = 1,2,\dots,J; k = 1,2,\dots,K_j\right) \tag{4.12}$$

$$q_{jks} = \sqrt{\frac{2 * A * d_{jks}}{h_{jks}}} \quad (s = 1,2,\dots,S) \tag{4.13}$$

$$x_{ijk} \le x_{ij'k'} \ (i = 1,2,\dots,I; j,j'k,k' \in SSR) \tag{4.14}$$

$$x_{ijk} + x_{ij^*k^*} = 1 \ (i = 1,2,\dots,I; j,j^*k,k^* \in SIR) \tag{4.15}$$

Equation (4.5) assigns each product to one product platform. Equation (4.6) sets the value of w_i. Equation (4.7) specifies only one module option m_{jk} for any module m_j can be selected by each product platform when the platform P_i is existed. Equation (4.8) ensures that no module option m_{jk} will be selected on the platform P_i if this platform is not existed. Equation (4.9) determines value of variable y_{jk}. Equation (4.10) calculates the number of product platform P_i. Equation (4.11) calculates the number of module option m_{jk}. It has three parts, the number of module option used to the platform P_i, the number of module option used to upgrade the under-designed module option and the number of module options replaced on the under-designed platform due to the customization. Equation (4.12) ensures the number of module option d_{jk} should be satisfied by purchasing from all its available suppliers. Equation (4.13) gives out the optimal order quantity of module option m_{jk} purchased from supplier S_s. Equation (4.14) represents the selection rules between module options whereas Eq. (4.15) ensures the incompatible rules.

4.4 Model linearization and solution method

Since the proposed model is a nonlinear mixed-integer programming, we applied some linearization method to transform the nonlinear objective and nonlinear constraints into linear ones. Firstly, the objective function Eq.(4.4) contains a non-linear formulation of $(\sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk})^+$ and it can be transferred to a normal mixed-integer programming formulation by

defining a continuous variable z_{vij} and a binary variable θ_{vij} . Equations (4.16) - (4.20) are defined as follows to remove the form '$(.)^+$'

$$z_{vij} \geq \sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk} \tag{4.16}$$

$$z_{vij} \leq \sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk} + \theta_{vij} \cdot bigM \tag{4.17}$$

$$z_{vij} \leq \left(1 - \theta_{vij}\right) \cdot bigM \tag{4.18}$$

$$z_{vij} \geq 0 \tag{4.19}$$

$$\theta_{vij} \in \{0,1\} \tag{4.20}$$

Secondly, the Eq. (4.6) and (4.9) are replaced by Eq. (4.21) - (4.22) and Eq. (4.23) - (4.24), respectively.

$$\sum_{v=1}^{V} e_{vi} \geq w_i \tag{4.21}$$

$$\sum_{v=1}^{V} e_{vi} \leq w_i \cdot bigM \tag{4.22}$$

$$\sum_{i=1}^{I} x_{ijk} \geq y_{jk} \tag{4.23}$$

$$\sum_{i=1}^{I} x_{ijk} \leq y_{jk} \cdot bigM \tag{4.24}$$

Thirdly, the Eq. (4.11) is nonlinear constraint because it contains the form $x_{ijk} \cdot e_{vi}$ and $\delta_{vij} \cdot e_{vi}$. We define $g_{vijk} = x_{ijk} \cdot e_{vi}$ and Eq. (4.25) - (4.28) are employed as follows.

$$x_{ijk} \geq g_{vijk} \tag{4.25}$$

$$g_{vijk} \geq e_{vi} + x_{ijk} - 1 \tag{4.26}$$

$$e_{vi} \geq g_{vijk} \tag{4.27}$$

$$g_{vijk} \in \{0,1\} \tag{4.28}$$

Likewise, we define $h_{vij} = \delta_{vij} \cdot e_{vi}$ and the linearization result is as below.

$$\delta_{vij} \geq h_{vij} \tag{4.29}$$

$$h_{vij} \geq e_{vi} + \delta_{vij} - 1 \tag{4.30}$$

$$e_{vi} \geq h_{vij} \tag{4.31}$$

$$h_{vij} \in \{0,1\} \tag{4.32}$$

Similarly, we define $l_{vijk} = h_{vij} \cdot x_{ijk}$ and linearization result is as below.

$$h_{vij} \geq l_{vijk} \tag{4.33}$$

$$l_{vijk} \geq h_{vij} + x_{ijk} - 1 \tag{4.34}$$

$$x_{ijk} \geq l_{vijk} \tag{4.35}$$

$$l_{vijk} \in \{0,1\} \tag{4.36}$$

Then the proposed nonlinear mixed integer programming model in section 3.3.2 can be transferred to the following mixed-integer liner programming model.

Minimize $C_t = C_d + C_s + C_c$

$$C_d = \sum_{i=1}^{I} dc_i \cdot w_i + \sum_{i=1}^{I} dc_{fix} \cdot w_i \tag{4.2}$$

$$C_s = \sum_{s=1}^{S} \sum_{j^s=1}^{J^s} \sum_{k=1}^{K_j} (pc_{jks} \cdot d_{jks} + \frac{A \cdot d_{jks}}{q_{jks}} + \frac{hc_{jks} \cdot q_{jks}}{2}) \tag{4.3}$$

$$C_c = \sum_{v=1}^{V} \sum_{i=1}^{I} \sum_{j=1}^{J} e_{vi} \cdot z_{vij} \cdot u_v \cdot f \tag{4.37}$$

Subject to

Equations (4.5), (4.7), (4.8), (4.10) - (4.15), and (4.16) - (4.36).

The proposed model is implemented using python language and solved using Gurobi 9.5.0. A windows PC with intel CPU 1.8GHz and 8GB RAM is used. Numerical experiments will be conducted to illustrate the applicability of the proposed model and the effectiveness of the Gurobi-based solution method under different parameter settings.

4.5 Numerical experiments

4.5.1 Parameter setting

The proposed model and solution method are applied to a case study of motherboard design for the personal computer product family. The motherboards can be regarded as the product platforms of PC, and the components of PCs can be understood as modules, e.g., processor, RAM, wireless network card, hotkeys, speakers. For example, there are three options for RAM module, i.e., 8GB, 16GB, and 32GB. In this case study, we only present the variant modules and do not discuss the common modules since each product requires a common module. The case company wants to know how many and which type of motherboards should be developed for a given product family and which suppliers should be selected.

For the reason of confidentiality of company information, we assumed the parameter settings according to the case study. The case study considers 8 products, 4 modules with 11 module options, and 8 suppliers. The requirements of products are given, and the descriptions of functional requirements are simplified to the selection of modules and module options, as enumerated in the right part of Table 4.2. Three demand scenarios are prepared for the numerical examination, as shown in Table 4.3. In scenario 1, we set all product demand to be the same while we provide different demand for each product in scenario 2 and 3. Leadtime of the supplier is assumed to be the same for all suppliers, as well as other factors except cost. The objective in our model includes cost only, therefore we could not raise other issues for the supplier selection.

Table 4.2 The description of modules with module option and the initial product requirements

Module	Module option	No.	Module option description	Design parameters	V1	V2	V3	V4	V5	V6	V7	V8
m_1	m_{11}	1	Processor 1	10	X				X			
	m_{12}	2	Processor 2	14		X				X		
	m_{13}	3	Processor 3	18			X				X	
	m_{14}	4	Processor 4	22				X				X
m_2	m_{21}	5	RAM 1	15	X	X						
	m_{22}	6	RAM 2	21			X		X	X		
	m_{23}	7	RAM 3	27				X			X	X
m_3	m_{31}	8	SDD 1	20	X		X		X		X	
	m_{32}	9	SDD 2	28		X		X		X		X
m_4	m_{41}	10	Speaker 1	25	X	X	X	X				
	m_{42}	11	Speaker 2	35					X	X	X	X

Table 4.3 The different demand scenarios of products

Scenario	Total demand	V1	V2	V3	V4	V5	V6	V7	V8
Scenario 1	80000	10000	10000	10000	10000	10000	10000	10000	10000
Scenario 2	80000	5600	7200	15200	8000	13600	7200	19200	4000
Scenario 3	80000	15200	10400	18400	3200	4800	8800	7200	12000

All module options are available for purchase from 8 suppliers. The module options offered by each supplier are shown in Table 4.4. We assume that suppliers S_3 and S_6 have higher production capacity because they can offer all the module options for module m_1 and module m_2, respectively. Furthermore, we assume that the unit purchasing price for the same module option from supplier S_3 is 0.5% higher than the price charged by supplier S_1 and S_2. Similarity, supplier S_6 has a higher purchasing price compared to supplier S_4 and S_5. The higher purchasing price can be compensated by the reduced inventory cost and ordering cost due to the risk pooling incurred by centralized purchasing. Other input parameters are presented in Table 4.5.

Table 4.4 Module options offered by each supplier

Supplier	S1	S2	S3	S4	S5	S6	S7	S8
Module option	m_{11},m_{12}	m_{13},m_{14}	m_{11},m_{12}, m_{13},m_{14}	m_{21},m_{22}	m_{22},m_{23}	m_{21},m_{22}, m_{23}	m_{31},m_{32}	m_{41},m_{42}

Table 4.5 Other input parameters used for the model

Parameters	Description	Value
α	Coefficient of variable development cost	4000
β	Coefficient of procurement cost	1
A	Ordering cost	500
f	Unit customization cost	1
dc_{fix}	Fixed cost of platform development	20000
hc_{jk}	Inventory holding cost	30%*pc_{jk}
$life$	Product lifetime	2 years
SSR	Set of selection rules	m_{14}, m_{32}
SIR	Set of incompatible rules	m_{11}, m_{23}

4.5.2 Experimental results

Table 4.6 presents the optimal total cost and platform configuration for different demand scenarios. As shown in Table 4.6, the optimal platform configuration is to develop 3 platforms for scenario 1 while developing 4 product platforms for scenario 2 and 3. In the first demand scenario, products F_1 and F_2 are derived from platform P_8 with the module option No.2,6,8,10 (i.e., m_{12}, m_{22}, m_{31}, m_{41}), products F_3, F_5, F_6 and F_7 are derived from platform P_7 with the module option No.3,6,8,11 (i.e., m_{13}, m_{22}, m_{31}, m_{41}), and products F_4 and F_8 are derived from platform P_1 with No.4,7,9,10 (i.e., m_{14}, m_{23}, m_{32}, m_{41}).

Specifically, platform P_8 is over-designed for module m_1 when deriving product F_1 while it is under-designed when deriving product F_2. Compared to the product requirements, platform P_8 has a higher specification module option No.2 (m_{12}) than the module option No.1 (m_{11}) needed by product F_1, which enables platform P_8 to be over-designed for product F_1. When deriving product F_2, platform P_8 is under-designed for module m_3. The platform P_8 configure a module option No.8 (m_{31}), however, the product F_2 requires a higher specification module option No.9 (m_{32}). Some additional customizations will be incurred to upgrade the under-designed module option No.8 to No.9, which will bring the additional customization cost. Similarly, platform P_7 is over-designed for product F_3 and F_5 while it is under-designed for product F_7. In particular, when deriving

product F_6, the platform P_7 becomes a hybrid-designed platform since it is over-designed for module m_2 and under-designed for module m_3. Platform P_1 with module option No.4,7,9,10 is matching-designed for product F_4 while it is under-designed for module m_4 when deriving product F_8 from it. Specific platform configurations and the assignment decision between products and platforms for scenario 2 and 3 can be found in the Table 4.6.

Moreover, the demand variation will affect the platform configuration and its design decision when deriving products from platforms. The optimal number of platforms developed in scenario 2 and scenario 3 is 4. However, the platform configuration for each platform in scenario 2 and 3 are totally different. For example, scenario 2 has a high-end platform P_1 with module option No. 4,7,9,10, however, scenario 3 configure a similar high-end platform P_7 with module option No. 4,7,9,11.

Meanwhile, the design decision that determines whether the platform is matching or nonmatching also varies with the demand variation. For instance, in scenario 2, product F_1 with the module option No.1,5,8,10 (i.e., m_{11}, m_{21}, m_{31}, m_{41}) is derived from an over-designed platform P_2 with module option No.2,6,8,10 (i.e., m_{12}, m_{22}, m_{31}, m_{41}). However, product F_1 is derived from its matching designed platform P_8 with module option No.1,5,8,10 in scenario 3. Similarly, product F_2 is derived from a hybrid platform in scenario 2, however, it is derived from matching designed platform in scenario 3. This results in 8 products being derived from 1 over-designed, 1 hybrid-designed, 3 matching-designed and 3 under-designed platforms in scenario 2, indicated as "1O,1H,3M and 3U". In scenario 3, 8 products are derived from 1O, 4M and 3U platforms.

In addition, maximizing the platform commonality does not produce a cost-efficient solution. Figure 4.2 shows how the total cost changed as the number of platforms is developed. As shown in Figure 4.2, developing one platform for three demand scenarios accompanies a higher total cost. The platform commonality can be achieved in two ways in our paper, including configuring an under-designed platform to customize products or configuring an over-designed platform. Our results

examine that both ways are not economical because deriving products from the under-designed platform incurs additional customization costs while using over-designed platform will bring additional material costs.

The supplier decisions under different demand scenarios are shown in Figure 4.3-4.5. In the scenario 2 and 3, all four module options (i.e., m_{11}, m_{12}, m_{13}, m_{14}) of module m_1 are purchased from supplier S_3 even purchasing from supplier s_3 has a 0.5% higher purchasing price than the price purchasing from supplier S_1 and S_2. This is because the risk pooling incurred by centralized purchasing can reduce the ordering cost and inventory cost. However, module option m_{12} is purchased from supplier S_1 and module options m_{13} and m_{14} are purchased from supplier S_2 in scenario 1. The reduced number of module options in scenario 1 alleviates the benefits of the centralized procurement, and the increased quantity of higher specification module options makes a greater incremental purchasing cost when purchasing from supplier S_3. For example, the number of module option m_{13} is 40000 in scenario 1 compared to 34400 in scenario 2 and 25600 in scenario 3.

Table 4.6 Results of platform configuration for different demand scenarios

Scenario		Scenario 1		Scenario 2		Scenario 3	
Cost items	C_t	17198836.22		17040431.75		16588925.92	
	C_d	1139999.994		1543999.995		1480000	
	C_s	15378836.22		14965231.76		14526525.92	
	C_c	680000		531200		582400	
CPU time(s)		225s		178s		185s	
Product requirement		Assignment and design decision	Platform configuration	Assignment and design decision	Platform configuration	Assignment and design decision	Platform configuration
F_1[1-5-8-10]		e(1,8)-Over	P_1[4-7-9-10]	e(1,2)-Over	P_1[4-7-9-10]	e(1,8)-Match	P_1[2-5-9-10]
F_2[2-5-9-10]		e(2,8)-Under		e(2,2)-Hybrid	P_2[2-6-8-10]	e(2,1)-Match	P_2[3-6-8-10]
F_3[3-6-8-10]		e(3,7) -Over		e(3,2)-Under		e(3,2)-Match	
F_4[4-7-9-10]		e(4,1)-Match	·	e(4,1)-March		e(4,7)-Over	
F_5[1-6-8-11]		e(5,7)-Over		e(5,7)-Match		e(5,8)-Under	
F_6[2-6-9-11]		e(6,7)-Hybrid		e(6,7)-Under	P_6[3-7-8-11]	e(6,1)-Under	
F_7[3-7-8-11]		e(7,7)-Under	P_7[3-6-8-11]	e(7,6)-Match	P_7[1-6-8-11]	e(7,2)-Under	P_7[4-7-9-11]
F_8[4-7-9-11]		e(8,1)-Under	P_8[2-5-8-10]	e(8,1)-Under		e(8,7)-Match	P_8[1-5-8-10]

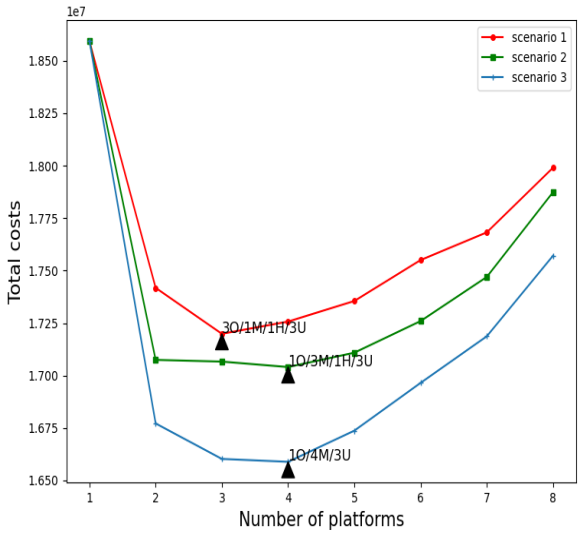

Figure 4.2 The total cost varies with the number of platforms under three demand scenarios

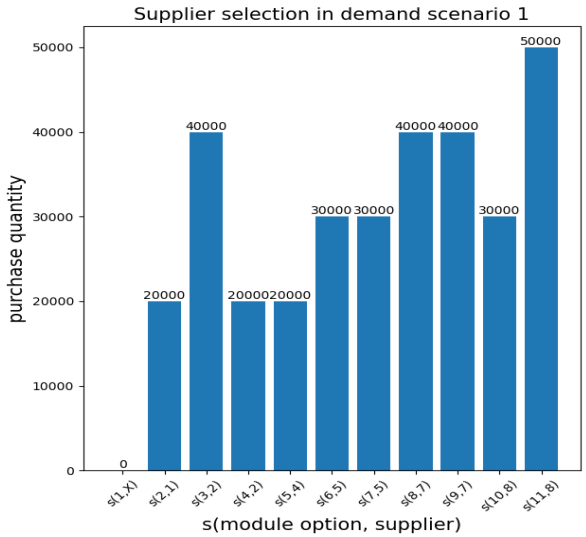

Figure 4.3 Supplier selection decision and purchase quantity of module option in demand scenario 1

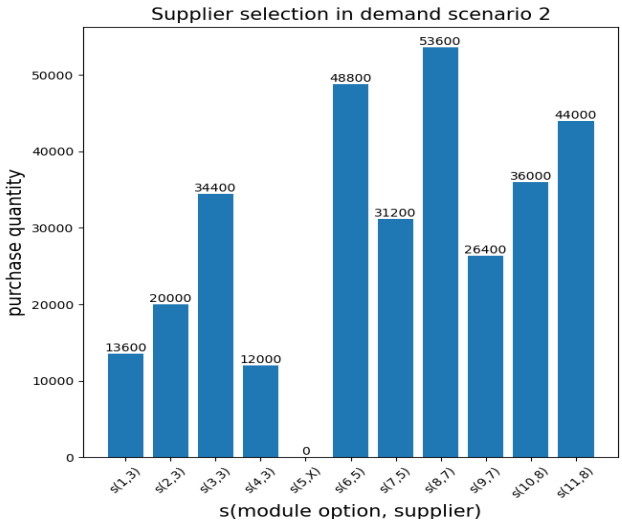

Figure 4.4 Supplier selection decision and purchase quantity of module option in demand scenario 2

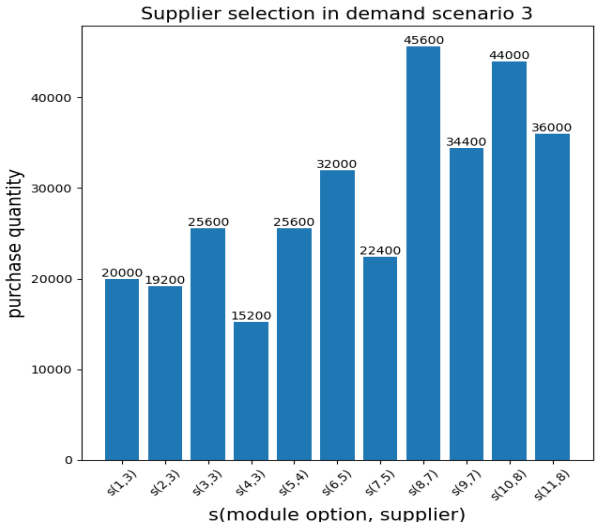

Figure 4.5 Supplier selection decision and purchase quantity of module option in demand scenario 3

4.6 Sensitivity analysis

To measure the impact of various parameters on the optimal platform configuration, sensitivity analyses of the cost parameters, product demand and product lifetime are conducted.

4.6.1 Cost sensitivity analysis

Firstly, we investigate the impact of cost parameters including coefficient of variable development cost α and the unit customization cost f. Figure 4.6 illustrates how the optimal platform configuration varies with different cost parameter α and f. In the Figure 4.6, 15 cost scenarios were tested by combining 5 parameters α from 1000 to 5000 and 3 parameters f from 1 to 3, indicated by the symbols $cs_{\alpha f}$. For example, the cost scenario cs_{21} represents a cost scenario with $\alpha = 2000$ and $f = 1$. Other parameter settings are the same as demand scenario 2. In Figure 4.6, the current situation cs_{41} is the optimal platform configuration discussed in section 4.5.2, in which 8 products are derived from 4 platforms with the design decisions represented by 1O,1H,3M, and 3U.

As we can see in Figure 4.6, the number of product platforms will decrease with an increased variable development cost α. For example, the number of platforms is 7 in scenario cs_{11} while it decreases to 2 in scenario cs_{51} as the variable development parameter α increases from 1000 to 5000. Likewise, as α increases, the number of platforms decreases from 7 in scenario cs_{12} to 3 in scenario cs_{52} and from 8 in scenario cs_{13} to 3 in scenario cs_{53}. In the case of the higher development cost, the company tends to develop fewer product platforms.

Moreover, the optimal number of platforms is likely to increase when the customization cost is higher. For example, the number of platforms increases from 7 in scenario cs_{11} to 8 in scenario cs_{13} as the unit customization cost f increase from 1 to 3. Similarly, the number of platforms increases from 4 in scenario cs_{41} to 5 in scenario cs_{43} and from 2 in scenario cs_{51} to 3 in scenario cs_{53}. The additional customization is to be avoided by deriving products from their own matching or over-designed platforms when the customization cost is higher.

In addition, the number of products derived from their matching designed platform will decrease as the development cost increase. For example, 8 products are derived from 7 matching designed platforms in scenario cs_{11}. The number of products derived from matching designed platforms is 5 in scenario cs_{21} and cs_{31} while it is 3 in scenario cs_{41} and 1 in scenario cs_{51}. The same trend can be found in the other scenarios. For example, in the case of customization cost equal to 2, the number of products deriving from matching platform decrease from 7 in scenario cs_{12} to 2 in scenario cs_{52}.

On the other hand, deriving products from the over-designed platforms become more frequent as the development cost increases. For example, no product is derived from the over-designed platform in the case of low development cost (i.e., scenario cs_{11}, cs_{12} and cs_{13}), while 4 products are derived from the over-designed platforms in scenario cs_{51} and 6 products are derived from the over-designed platforms in scenario cs_{52} and cs_{53}. Our analysis of the results indicates that over-design is more prevalent in the presence of high development cost and high customization cost. In

contrast, the matching design of platforms is more suitable for low development cost and high customization cost, such as scenarios cs_{13} and cs_{12}.

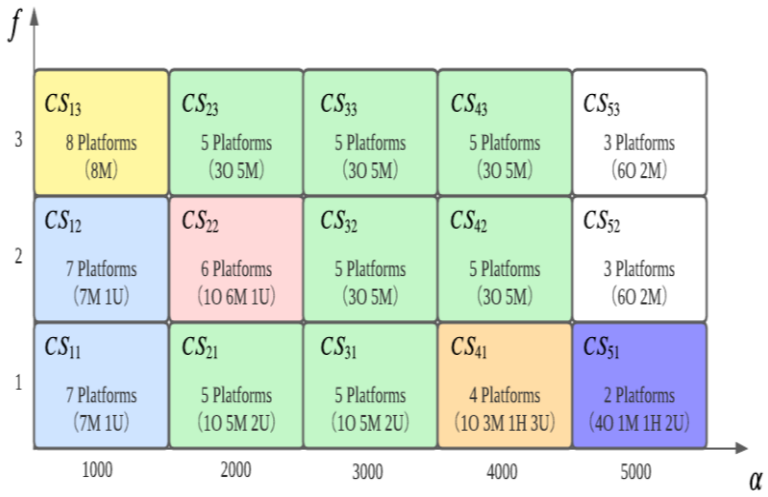

Figure 4.6 Optimal platform configuration decision with varying development cost and customization cost

4.6.2 Demand sensitivity analysis

A sensitivity analysis on demand is conducted to illustrate the impact of demand. The demand rates for different products and unit customization cost are set to be the same as those in scenario 2. For example, the demand of product F_1 is 5600 in scenario 2, which contributes a demand rate equal to 7%. Thus, the demand of product F_1 is 4200 when the total demand is 60000 and it is 8400 when the total demand is 10000. As shown in Figure 4.7, an increased total demand will configure more product platforms for a product family. For example, 2 platforms are configured to derive 8 products when the total demand is 60000, while 4 platforms are configured in the case of total demand equal to 80000 and 5 platforms are offered when the total demand is 10000. Meanwhile, when demand is high, manufacturers tend to derive products from their matching designed platforms. For instance, 8 products are derived from 1O, 5M, and 2U platforms when the total demand is 100000. In contrast, 8 products are derived from 1O, 3M, 1H, and 3U platforms in the case of low total demand equal to

80000 and from 4O, 1H, 1M, and 2U platforms in the case of lower total demand equal to 60000.

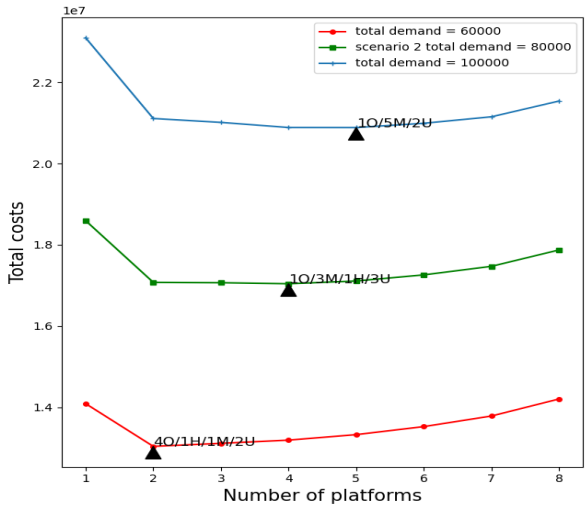

Figure 4.7 The impact of varying product demand on platform configuration decision

4.6.3 Product lifetime sensitivity analysis

The impact of product lifetime on platform configuration is analyzed. As shown in Figure 4.8, with a longer product lifetime, the number of platforms increases. For example, compared to 4 platforms configured with 2 years product lifetime, the number of platforms is 5 in the case of 3 years and 6 in the case of 4 years. Moreover, as the product lifetime increases, the number of products derived from the matching designed platforms also increases. When the product lifetime is 4 years, 8 products are derived from 6M and 2U platforms while it is 1O, 5M, and 2U in the case of 3 years and 1O, 3M, 1H, and 3U in the case of 2 years. The longer product lifetime enables the high development cost associated with developing more matching designed platform relatively low in terms of total costs.

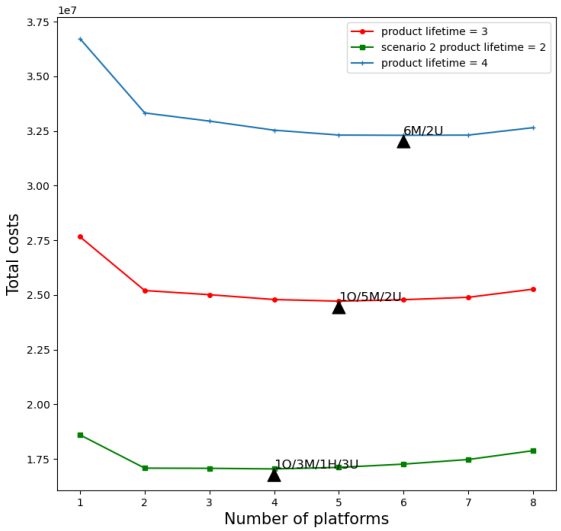

Figure 4.8 The impact of varying product lifetime on platform configuration decision

4.7. Conclusions

In this chapter, we studied a platform configuration problem while considering platform design strategy and supplier selection. The proposed model was formulated as a non-linear mixed-integer programming model. A cost model including the development cost, sourcing cost and customization cost was developed to illustrate the cost trade-off between platform development and customization. The proposed model was linearized and solved by commercial solver Gurobi. By minimizing the total cost, the optimal number and platform configuration decision for a given product family was obtained, as well as its relevant supplier selection. Numerical experiments show that the proposed model can be effectively applied to a joint optimization problem of platform configuration and supplier selection. Moreover, the proposed solving algorithm applying the linearization method and Gurobi solver can effectively generate optimal solutions for different parameter settings.

The results of numerical experiments show that the optimal combination of supplier and optimal number of platforms are depending on the given parameters, such as various cost parameters included in the model, product demand, and product lifetime. The cost sensitivity analyses show that the optimal number of product platforms will decrease as the variable development cost increases, while

the number of platforms is likely to increase as the customization cost increases. Moreover, the model can give guidance to what extent the platform should be matching-designed, under-designed or over-designed with regard to the products derived from them. As the development cost increases, few products are derived from their matching-designed platforms, while more products are derived from the over-designed platforms. The over-designed platform is more prevalent in the presence of high development cost and high customization cost. In contrast, the matching designed platforms is more suitable for low development cost and high customization cost.

In addition, the increasing total demand drives the model to develop more platforms and tends to derive products from their matching designed platforms. Likewise, the more platforms will be developed for a product family and the more matching-designed platforms will be configured with a longer the product lifetime.

Chapter 5 A stochastic programming approach for product platform configuration under demand uncertainty

5.1 Introduction

Platform-based product development (PPD) is an effective way to achieve mass customization. Through PPD approach, various products within a product family can be effectively developed based on the product platform. One important problem in PPD is product platform configuration (PPC). Two critical research problems in PPC are: (1) how many and which type of product platform should be developed for a product family; (2) which product platform will be assigned to derive the product within a product family.

Different types of platform design strategies can be found, i.e., matching-designed, under-designed, and over-designed. A matching-designed platform configures the same module options that exactly match the product requirements while the under-designed platform or the over-designed platform has module options with lower specification or higher specification than the product requirements. Forming the different types of platforms contributes a different impact on the platform-related costs. For example, developing a matching-designed platform for each product within a product family requires more product platforms, which results in a higher total development cost. However, the platform customization is not required. On the other hand, developing an under-designed platform contributes a lower development cost and a higher customization cost while

developing an over-designed platform brings a higher development cost and a lower customization cost. How to weigh the different platform design strategies is a crucial problem when configuring the product platforms.

In addition, demand uncertainty is a huge challenge in supply chain management. Customer demand is always uncertain and unpredictable. Due to demand uncertainty, all the sectors of supply chain may experience negative effects, which makes an inefficient supply chain and reduces revenue. For example, a larger backlog of products may be incurred if overproduction. Conversely, it may also cause a loss of market share due to stock-outs.

The PPC decision is endogenously linked to supply chain-related activities, e.g., procurement, manufacturing, inventory, and transportation. The risks and uncertainty associated with supply chain have a significant impact on PPC decision. Generally, the development of product platform is ahead of the new product introduction (NPI), which is difficult to forecast the demand. How to determine the optimal PPC decision under demand uncertainty is highly important.

In this chapter, a new platform configuration model is proposed to handle demand uncertainty. The proposed model is formulated as a two-stage stochastic programming model while every possible random demand is represented by a scenario with an associated probability. A linear programming embedded genetic algorithm is developed to solve the model. The proposed algorithm deals with the binary variables for platform configuration by using a genetic algorithm (GA) and determines the integer variables by solving a linear programing subproblem through Gurobi solver. Numerical experiments are conducted to illustrate the proposed model and algorithm.

5.2 Problem description and assumptions

5.2.1 Nomenclature

Table 5.1 illustrates the description of symbols used in the model.

Table 5.1 List of symbols

Notation	Definition
v	The index of product variants F_v $(v = 1,2, \dots, V)$ in a product family
i	The index of product platform P_i $(i = 1,2, \dots, I)$
j	The index of module m_j $(j = 1,2, \dots, J)$
k	The index of module option m_{jk} $(k = 1,2, \dots, K_j)$
s	The index of scenario s $(s = 1,2, \dots, S)$
N_{jk}	The design parameter of module option m_{jk}
dc_i	Variable development cost of product platform P_i
dc_{fix}	Fixed development cost of product platform
pc_i^{mp}	Material cost of product platform through pre-production in first stage
pp_i^{mp}	Production cost of product platform through pre-production in first stage
pc_i^{cu}	Material cost of product platform through post-production in second stage
pp_i^{mp}	Production cost of product platform through post-production in second stage
α	Coefficient of variable development cost
β	Coefficient of material cost of module option for first stage
γ	Coefficient of material cost of module option for second stage
τ	Coefficient of penalty cost
f	Unit customization cost related to derive unsatisfied platform to product
b_i	Penalty cost of excessive product platform P_i
r_{vjk}	The product requirement for module option m_{jk}
δ_{vij}	Binary variable, takes value 1 if platform P_i is under-designed for module m_j when deriving product F_v
SSR	Set of selection rules $(m_{jk}, m_{j'k'})$ which represents that selection of module option m_{jk} requires module option $m_{j'k'}$ in the same configuration
SIR	Set of incompatible rules $(m_{jk}, m_{j^*k^*})$ which represents that module option m_{jk} and module option $m_{j^*k^*}$ cannot be used together in the same configuration
$prob^s$	Probability of scenario s
e_{vi}	Binary decision variable to indicate whether product F_v is derived from product platform P_i
w_i	Derivative binary variable to indicate whether product platform P_i is developed
x_{ijk}	Binary decision variable to indicate whether module option m_{jk} is used in product platform P_i
$\emptyset_i$	The quantity of pre-production product platform P_i before the confirmation of product demand
φ_i^s	The quantity of post-production platform P_i under demand scenario s after the confirmation of product demand

5.2.2 Problem description

As shown in Figure 5.1, the product platform is configured by selecting a set of modules and module options. A module m_j $(j = 1,2, \dots, J)$ is a unit that serves an identifiable product function and is developed in prior. Two types of modules can be found, i.e., a variant module and a common

module. Each variant module has multiple options m_{jk} $(k = 1,2, \dots, K_j)$ with different cost-relevant design parameters N_{jk} to represent different functional levels. A common module only has one module option and is required by each product. The combinations of modules and module options enable the possible product platform configurations.

A product family has multiple products F_v $(v = 1,2,..,V)$ with different functional requirements, which represented by the selection of modules and module options. Through PPD, each product within a product family can be derived from one product platform. Four scenarios of platform design strategies can be found, i.e., matching-design, over-design, under-design, and hybrid-design. A matching-designed platform has the same selection of module options compared to the dedicated product requirements. In contrast, a non-matching designed platform can be configured as an under-designed or over-designed platform, which has lower specification or higher specification module options than the product requirements. A hybrid-designed platform contains both lower and higher specification module options compared to the product requirements. For example, a product family contains three products with the respective requirements. Product F_1 requires module option m_{11}, m_{21}, product F_2 requires module options m_{12}, m_{22} and product F_3 requires module options m_{11}, m_{22}. If we configure a platform P_1 with module options m_{12}, m_{21}, then platform P_1 is over-designed for module m_1 when deriving product F_1. This is because product F_1 has a lower specification module option m_{11} than module option m_{12} used on platform P_1. Likewise, it is an under-designed platform for module m_2 when deriving product F_2. Similarly, platform P_1 is over-designed for module m_1 and is under-designed for module m_2 when deriving product F_3, which we defined it as a hybrid-designed platform.

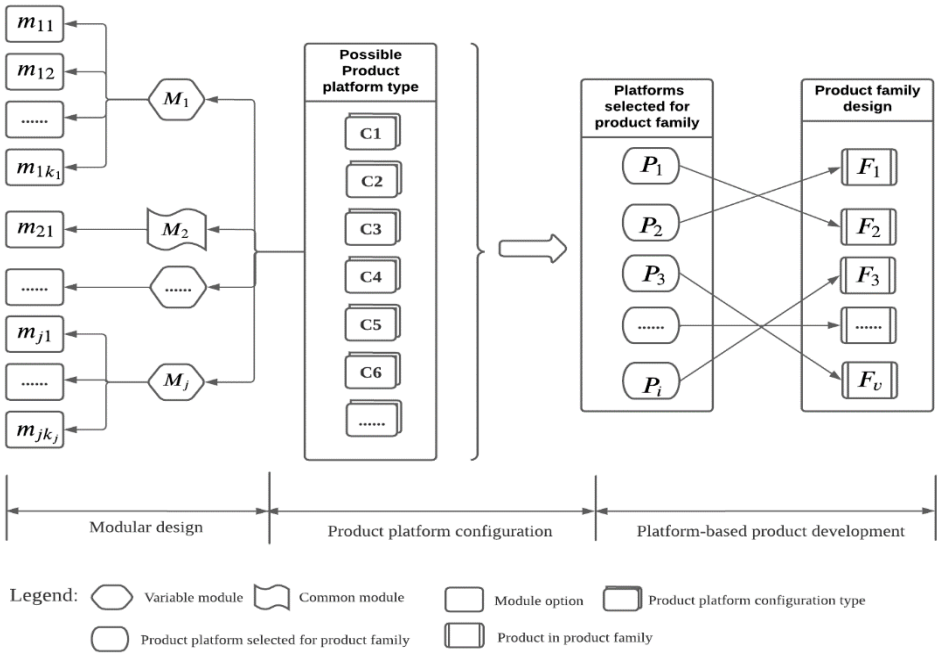

Figure 5.1 Product platform configuration with modules and module options for product family

One of the most important benefits of PPD is the economies of scale gained through the mass production of shared modules, components, and parts. Thus, two stages can be identified in the PPD, i.e., the platform configuration stage and the platform customization stage. The platform configuration stage determines the number of platforms and the module selection on the platforms as well as product-platform assignment decision. The platform customization stage enables unsatisfied platforms to be upgraded to meet the requirements of dedicated products when receiving the customer order.

The platform configuration stage is ahead of the realization of real demand. In order to respond quickly to the market, a pre-production strategy of product platform is applied in this study to pre-produce the platforms and purchase all the related modules, components, and parts in advance. We assume that the pre-production strategy of platforms allows the manufacturers to produce the platforms at a lower cost and receive purchase discount from the suppliers due to pre-procuring. The two stages under demand uncertainty are illustrated in Figure 5.2.

Step 1. Configure product platforms by selecting modules and module options according to the

given product requirements.

Step 2. Determine the quantity of pre-production platforms and purchasing relevant modules and components.

Step 3. If the product demand cannot be satisfied by the pre-production platforms, the manufacturer needs to produce additional platforms through post-production mode.

Step 4. The under-satisfied function should be customized to satisfy the dedicated product requirements.

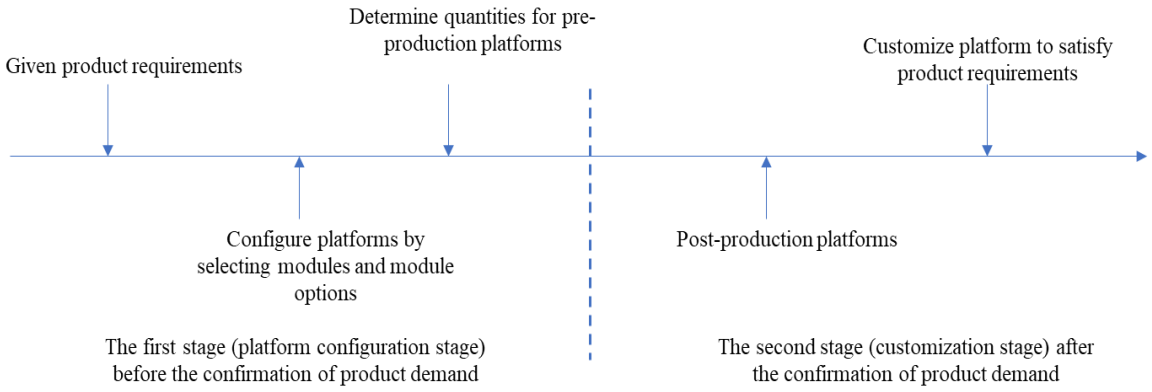

Figure 5.2 Two-stage platform configuration process under demand uncertainty

5.2.3 Assumptions

For formulating the mathematical model, the following assumptions will be introduced.

Assumption 1: the different module options m_{jk} for module m_j are sorted in an increasing value of design parameter N_{jk} ($N_{jk} \leq N_{j(k+1)}$). The value of N_{jk} is corresponding to the different functional levels and is further assumed to be related to the costs of module options. Therefore, the variable development cost of module option m_{jk} used on the platform equal to $\alpha \cdot N_{jk}$. The unit material cost of module option m_{jk} for first stage equals to $\beta \cdot N_{jk}$ while the unit material cost of module option for second stage is $\gamma \cdot N_{jk}$. The penalty cost of module option m_{jk} equals to $\tau \cdot N_{jk}$. A product platform with a higher specification of module option may have a higher development cost,

higher material cost and higher penalty cost of excessive production.

Assumption 2: we assume that the customization is only incurred when a product is derived from an under-designed platform. In the case of a matching-designed or over-designed platform, there is no customization. In other words, we assume that the higher specification module option could be used for product platform without sacrificing product quality. However, the procurement cost of module option will increase if the low-end product is derived from an over-designed platform. The design parameter is used to calculate the performance gap for a particular module m_j between product F_v and platform P_i as follows.

$$\max\left(0, \sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk}\right) \qquad \forall v, i, j \tag{5.1}$$

Furthermore, we define a binary variable δ_{vij}, which takes value 1 if the platform P_i is under-designed for module m_j when deriving product F_v. The larger the performance gap, the more customization is required to derive the product from this platform.

Assumption 3: the unit production cost and material cost of platforms in the platform configuration stage is lower than those in the customization stage because of the mass production mode and purchase discount obtained from suppliers.

Assumption 4: a penalty cost of excessive platforms is introduced to avoid producing excessive platforms and incurring excessive inventory.

Assumption 5: all the product demand should be satisfied through pre-production and post-production of platforms and no shortage is allowed.

5.3 Model formulation

The PPC problem under demand uncertainty can be formulated as a two-stage stochastic programming model. In the first stage, the product platform configuration and the quantity of pre-

production platforms will be determined before the confirmation of product demand. A binary decision variable x_{ijk} is used to indicate whether the module option m_{jk} is selected on the product platform P_i. It equals to 1 if the module option m_{jk} is selected on the platform P_i. The assignment decision between product and platform is denoted by a binary decision variable e_{vi}. It equals to 1 if product F_v is derived from platform P_i, or 0 otherwise. A derivative variable w_i is used to determine whether the product platform P_i is developed or not. A continuous variable $\emptyset_i$ is proposed to determine the quantity of platforms through pre-production in the first stage.

In the second stage, when the demands for all products are realized, the model determines the quantity of post-production platforms and excessive platforms under demand scenario s. If the product demand cannot be satisfied by the quantity of pre-production platforms, then additional platforms are required through post-production. A continuous variable φ_i^s is proposed to determine the quantity of post-production platforms under scenario s and π_i^s is the quantity of excessive platforms under scenario s.

The uncertainty of product demand is represented by a finite set of scenarios with a possible probability associated with each scenario. Each scenario contains a set of product demand data indicated by $u^s = [u_1^s, u_2^s, \dots, u_v^s]$. The values of the second-stage variables depend on the value of each set of stochastic product demand.

The objective function of this model is to minimize the total cost, including the development cost of product platform C_d, the material cost of pre-production platforms C_m^{mp} and production cost C_p^{mp} of pre-production platform in the first stage, the material cost of post-production C_m^{cu} and production cost C_p^{cu} of post-production platform in second stage, the customization cost C_c and the penalty cost of excessive platform C_s. The development cost of platform contains two parts, the variable development cost associated with the selection of module options on the platforms and the fixed development cost depending on the number of platforms. The material cost and production cost of platform mainly depend on the selection of module options on the platform. The total customization

cost depends on the performance gap for all under-designed module options compared with product requirements. A unit customization cost f is included to represent the impact of the performance gap on the customization cost.

The proposed two-stage stochastic model for platform configuration is formulated as follows.

Minimize

$$C_t = C_d + C_m^{mp} + C_p^{mp} + C_m^{cu} + C_p^{cu} + C_c + C_s \tag{5.2}$$

$$C_d = \sum_{i=1}^{I} dc_i \cdot w_i + \sum_{i=1}^{I} dc_{fix} \cdot w_i \tag{5.3}$$

$$C_m^{mp} = \sum_{i=1}^{I} pc_i^{mp} \cdot \emptyset_i = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} x_{ijk} \cdot \beta \cdot N_{jk} \cdot \emptyset_i \tag{5.4}$$

$$C_p^{mp} = \sum_{i=1}^{I} pp_i^{mp} \cdot \emptyset_i \tag{5.5}$$

$$C_m^{cu} = \sum_{s=1}^{S} prob^s \left(\sum_{i=1}^{I} pc_i^{cu} \cdot \varphi_i^s \right) = \sum_{s=1}^{S} prob^s \left(\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} x_{ijk} \cdot \gamma \cdot N_{jk} \cdot \varphi_i^s \right) \tag{5.6}$$

$$C_p^{cu} = \sum_{s=1}^{S} prob^s \left(\sum_{i=1}^{I} pp_i^{cu} \cdot \varphi_i^s \right) \tag{5.7}$$

$$C_c = \sum_{s=1}^{S} prob^s \{ \sum_{v=1}^{V}\sum_{i=1}^{I}\sum_{J=1}^{J} e_{vi} \left(\sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk} \right)^{+} \cdot u_v^s \cdot f \} \tag{5.8}$$

$$C_s = \sum_{s=1}^{S} prob^s (b_i * \pi_i^s) = \sum_{s=1}^{S} prob^s \left(\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} x_{ijk} \cdot \tau \cdot N_{jk} \cdot \pi_i^s \right) \tag{5.9}$$

Subject to

$$\sum_{i=1}^{I} e_{vi} = 1 \ \ (v = 1,2, \dots, V) \tag{5.10}$$

$$w_i = \begin{cases} 1, & \sum_{v=1}^{V} e_{vi} > 0 \\ 0, & \sum_{v=1}^{V} e_{vi} = 0 \end{cases} \quad (5.11)$$

$$\sum_{k=1}^{K_j} x_{ijk} = w_i \quad (i = 1,2,\dots,I; j = 1,2,\dots,J) \quad (5.12)$$

$$x_{ijk} \leq w_i \quad \left(i = 1,2,\dots,I; j = 1,2,\dots,J; k = 1,2,\dots,K_j\right) \quad (5.13)$$

$$x_{ijk} \leq x_{ij'k'} \ (i = 1,2,\dots,I; j, j'k, k' \in SSR) \quad (5.14)$$

$$x_{ijk} + x_{ij^*k^*} = 1 \ (i = 1,2,\dots,I; j, j^*k, k^* \in SIR) \quad (5.15)$$

$$\emptyset_i \leq w_i \cdot bigM \quad (5.16)$$

$$\varphi_i^s \leq w_i \cdot bigM \quad (5.17)$$

$$\pi_i^s \geq \phi_i + \varphi_i^s - \sum_{v=1}^{V} e_{vi} \cdot u_v^s \quad (i = 1,2,\dots,I,\ s = 1,2,\dots S) \quad (5.18)$$

$$\emptyset_i + \varphi_i^s \geq \sum_{v=1}^{V} e_{vi} \cdot u_v^s . \quad (i = 1,2,\dots,I; s = 1,2,..,S) \quad (5.19)$$

Equation (5.10) assigns each product to one product platform. Equation (5.11) sets the value of w_i. Constraint (5.12) specifies only one module option m_{jk} for any module m_j can be selected by each product platform when the platform P_i is developed. Equation (5.13) ensures that no module option m_{jk} will be selected on the platform P_i if this platform is not developed. Constraint (5.14) represents the selection rules between module options whereas constraint (5.15) ensures the incompatible rules. Equation (5.16) states that no pre-production platform will be produced if this platform is not developed while Equation (5.17) states that no post-production platform will be produced if this platform is not developed. Equation (5.18) gives out the quantity of excessive platforms. Constraint (5.19) ensures that the product demand under each scenario should be satisfied by the sum of pre-production platforms and post-production platforms.

5.4 Algorithm development

The product platform configuration problem is a larger-scale combinatorial optimization problem. The exact algorithm can be very time-consuming for large-scale problems while the meta-heuristic algorithm is more effective than the exact algorithm in solving such problem. Among the most popular heuristic algorithm, genetic algorithm (GA) is widely used. Meanwhile, the proposed two-stage stochastic model is a nonlinear mixed-integer programming model. With nonlinear terms and integer variables, it is difficult to solve it by using GA alone. Therefore, a linear programming embedded GA is developed. The proposed algorithm searches the binary variables for platform configuration by using the GA and determines the integer variables by solving a linear programming subproblem using the Gurobi solver.

In the following subsection, the genetic algorithm for platform configuration is introduced including coding and decoding the chromosome to represent a feasible solution, fitness function, new mutation and crossover method as well as solving a subproblem by using linearization methods and Gurobi solver. Figure 5.3 illustrates the main procedure of the proposed algorithm.

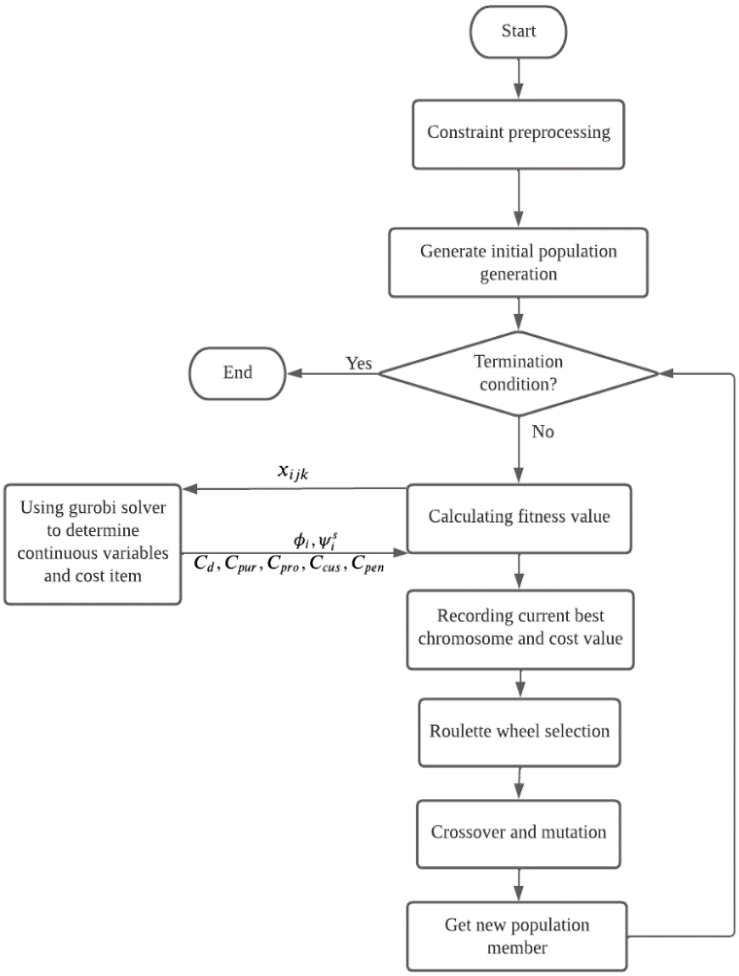

Figure 5.3 The main procedure of proposed algorithm

5.4.1 Chromosomal encoding of a solution

The key point of GA is to encode the decision variable by using the chromosome, substring, and gene. As shown in Figure 5.4, a platform configuration solution is encoded in a finite length string, namely chromosome. Each chromosome consists of substrings that represent a single platform configuration decision. Each substring comprises all the module options whilst each module option is represented by a gene. Each gene has a binary value (0 or 1) to indicate if the module option is used on the corresponding product platform. The gene takes value 1 if the module option represented by this gene is used on the corresponding platform configuration. Otherwise, it takes the value 0.

The length of chromosome is equal to the number of products within a product family. For forming the chromosome, we deal with the decision variable e_{vi} by assuming that a product F_v is derived from the platform P_i where $v = i$. However, the platform with the same module and module options are merged into one platform when we calculate the objective function. The length of the substring is equal to the number of total module options. For example, a product family contains 5 products with 4 modules and 11 module options. The length of substring (single platform configuration) equals 11 and the chromosome would contain 55 genes (5 platforms multiply 11 module options).

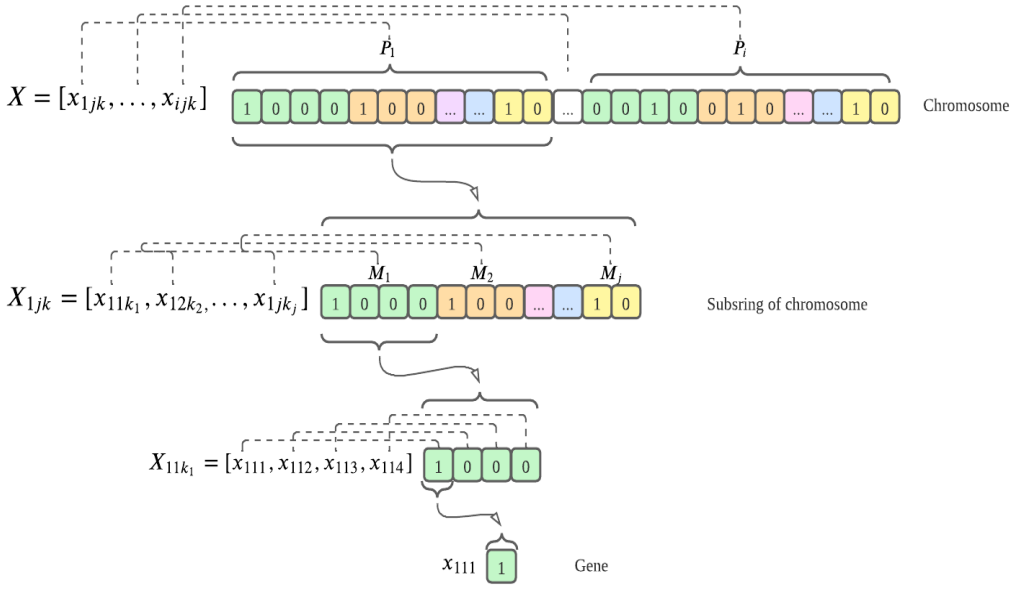

Figure 5.4 Encoding scheme illustration

5.4.2 Initial population

The initial population contain a certain number of chromosomes which equals the predetermined population size. Each chromosome is generated by randomly assigning the value (0 or 1) to each gene in this chromosome. To ensure a feasible initial solution, a feasibility check mechanism is applied in this process. The feasibility checks mainly deal with the platform configuration constraints, such as constraint (5.10), constraint (5.14) and constraint (5.15).

5.4.3 Fitness function

The fitness function is used to evaluate the fitness of each chromosome. The fitness function is the total cost of platform configuration as described in section 5.3. To calculate the value of the fitness function and the value of continuous variables, a linear programming subproblem is solved by using some linearization method and Gurobi solver.

Firstly, the objective function Equation (5.8) contains a non-linear formulation of $(\sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk})^+$ and it can be transferred to a normal mixed-integer programming formulation by defining a continuous variable z_{vij} and a binary variable θ_{vij}. Equations (5.20) - (5.24) are defined as follows to remove the form '$(\ )^+$'.

$$z_{vij} \geq \sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk} \tag{5.20}$$

$$z_{vij} \leq \sum_{k=1}^{K_j} r_{vjk} \cdot N_{jk} - \sum_{k=1}^{K_j} x_{ijk} \cdot N_{jk} + \theta_{vij} \cdot bigM \tag{5.21}$$

$$z_{vij} \leq \left(1 - \theta_{vij}\right) \cdot bigM \tag{5.22}$$

$$z_{vij} \geq 0 \tag{5.23}$$

$$\theta_{vij} \in \{0,1\} \tag{5.24}$$

The objective function Equation (5.7) then can be formulated as follows.

$$C_c = \sum_{s=1}^{S} prob^s \{ \sum_{v=1}^{V} \sum_{i=1}^{I} \sum_{J=1}^{J} e_{vi} \cdot z_{vij} \cdot u_v^s \cdot f \} \tag{5.25}$$

Therefore, the fitness function is formulated as follows.

Minimize

$$C_t = C_d + C_m^{mp} + C_p^{mp} + C_m^{cu} + C_p^{cu} + C_c + C_s$$

Equations (5.3) -(5.7), Eq. (5.25), and Eq. (5.9)

The fitness value of each chromosome will be calculated within the population size for the entire

generation until the genetic algorithm search stops, and an optimal or near-optimal solution is reached.

5.4.4 Selection

The selection in GA is to generate the next generation population (offspring) by selecting the parent chromosome from the current population. In this model, the parent chromosome will be selected by roulette wheel selection where all the chromosomes in the current population are placed according to their fitness value on a roulette wheel. Since the objective function is to minimize the total cost, the chromosome has a better performance if the total cost is lower. Thus, the selection area of the wheel corresponding to each chromosome is the reciprocal of its fitness value. The chromosome with a lower total cost (higher fitness) has a higher probability of being selected more times. Then, a random number is generated to select one of the chromosomes to be the parent chromosome for crossover and mutation.

5.4.5 Crossover

The crossover will be applied to generate the offspring by inheriting the gene information of the two parent chromosomes. A specific crossover procedure is developed to deal with this specific problem. As mentioned before, each chromosome consists of a set of substrings and each substring contains genes that represent all the module options. Since the different modules have a different number of module options, the crossover position can be specified according to the specific number of module options for a module. For illustration, we present an example that considers 3 products within a product family and 3 modules with 9 module options. Module m_1 has 4 module options, module m_2 has 3 module options and module m_3 has 2 module options. Then a set of crossover points is 4, 7, 9, 13, 16, 18, 22, 25, and 27. A random number can be generated from this specific set of possible crossover points. The offspring chromosome inherits the gene information of the father

chromosome before the random number and the gene information of the mother chromosome after the random number. Another offspring chromosome inherits the rest of gene information of the parent chromosome. This specific crossover procedure ensures that the offspring platform configuration is feasible for the module selection.

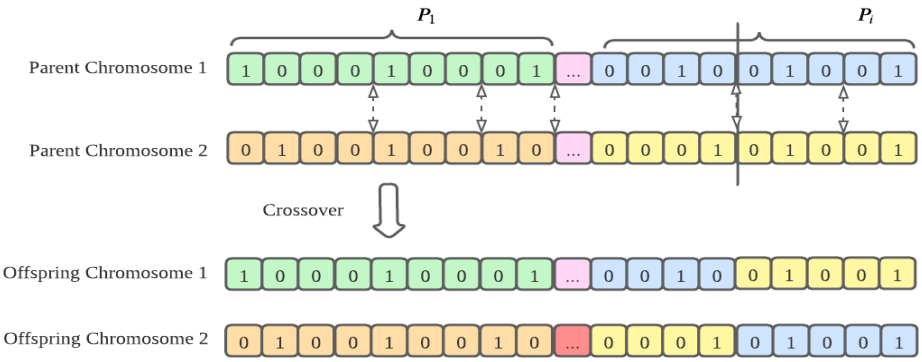

Figure 5.5 Crossover

5.4.6 Mutation

The mutation plays a crucial role in searching the possible feasible solutions. It is crucial to the convergence of the genetic algorithm. We use a 0-1 flip-flop mutation for gene mutation due to the binary encoding gene in this model. The mutation procedure is achieved by generating three random number a, b, and c where $a \in [0, i]$, $b \in [0, j]$, and $c \in [0, k_j]$. The random number a is generated from 0 to i to choose the mutation platform and the random number b is generated from 0 to j to choose the mutation module. The number c will be generated from 0 to k_j and inverting the 0-1 value of this position. The feasibility check will be also applied to ensure only one module option for a specific module is used in a platform configuration. For example, Figure 5.6 presents a case of 3 products and three modules with 9 module options. Generating a random number $a = 2$, $b = 1$ and $c = 4$, then the offspring chromosome changes the value of the red marked position from 0 to 1 and other gene values of this module are changed from 1 to 0.

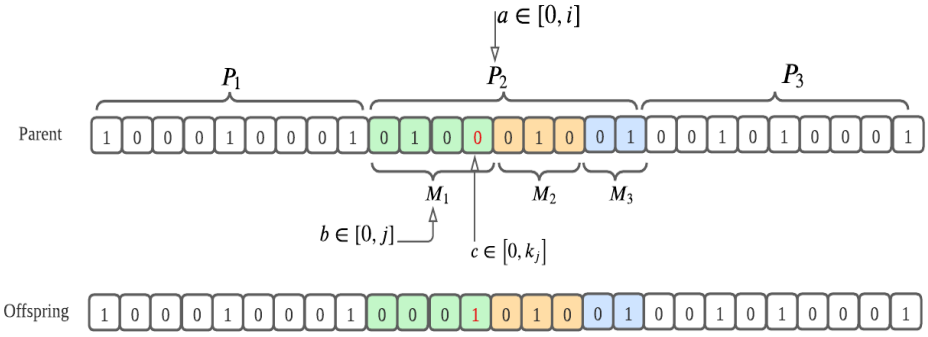

Figure 5.6 Mutation

5.4.7. Stopping criteria

The genetic search is repeated until pre-defined stopping conditions are reached. The stopping condition considered in this proposed algorithm is reaching a pre-selected number of generations, or the best solution remains unchanged for a certain number of successive generations.

5.5 Numerical experiments

5.5.1 Parameter setting

The proposed model and solution method are applied to a case study of motherboard design for personal computer product family. The motherboards can be regarded as the product platforms of PC, and the components of PC can be understood as modules, e.g., processor, RAM, wireless network card, hotkeys, speakers. For example, there are three options for RAM module, i.e., 8 GB, 16 GB, and 32 GB. In this case study, we only present the variant modules and do not discuss the common modules since each product requires a common module.

A product family contains 5 products, 4 modules with 11 module options. The product requirements are given, and the descriptions of functional requirements are simplified to the selection of modules and module options, as enumerated in the right part of Table 5.2. The product demand

follows the normal distribution. Three demand distribution cases with different mean values and variances are prepared for the numerical examination, as shown in Table 5.3. The solution algorithm is coded using Python and runs on a PC with Apple M1 and 16 GB RAM.

Table 5.2 The description of modules with module option and the initial product requirements

Module	Module option	No.	Module option description	Design parameters	F_1	F_2	F_3	F_4	F_5
m_1	m_{11}	1	Processor 1	2	X				
	m_{12}	2	Processor 2	4		X	X		
	m_{13}	3	Processor 3	7				X	
	m_{14}	4	Processor 4	9					X
m_2	m_{21}	5	RAM 1	6	X	X			
	m_{22}	6	RAM 2	11			X	X	
	m_{23}	7	RAM 3	16					X
m_3	m_{31}	8	SDD 1	8	X	X		X	
	m_{32}	9	SDD 2	15			X		X
m_4	m_{41}	10	Speaker 1	12	X		X		
	m_{42}	11	Speaker 2	17		X		X	X

Table 5.3 The different demand scenarios of products

Demand distribution	Mean value	F_1	F_2	F_3	F_4	F_5
Demand case 1	10000	(9000,11000)	(9200,10800)	(9500,10500)	(9300,10700)	(9400,10600)
Demand case 2	15000	(14000,16000)	(14200,15800)	(14500,15500)	(14300,15700)	(14000,16000)
Demand case 3	20000	(19000,21000)	(19200,20800)	(19500,20500)	(19400,20600)	(19300,20700)

Table 5.4 Other input parameters used for the model

Parameters	Description	Value
α	Coefficient of variable development cost	1000
β	Coefficient of material cost of module option for first stage	1
γ	Coefficient of material cost of module option for second stage	1.05
τ	Coefficient of penalty cost	0.2
f	Unit customization cost	1.5
dc_{fix}	Fixed cost of platform development	20000
SSR	Set of selection rules	m_{14}, m_{32}
SIR	Set of incompatible rules	m_{11}, m_{23}

5.5.2 Experiments on the efficiency of the proposed algorithm

To demonstrate the accuracy and efficiency of the proposed algorithm, we use 13 different problem instances with different numbers of products, modules, module options and scenarios to compare the result of the proposed algorithm and the optimal results solved by Gurobi solver directly. The number of products within a product family, modules and module options and the number of scenarios determine the problem size, which further affects the computation complexity.

The comparative results of the two solution methods are presented in Table 5.5. The possible number of platform configurations is the number of product platforms configured without any constraints. For ensuring the Gurobi solver can solve the problem instances directly, the experiments just consider a problem instance with 6 products, 5 modules, 13 module options and 20 scenarios. Important point is that the computation time of the Gurobi algorithm increases rapidly along with the problem size, while the proposed algorithm increases linearly and slowly. Figure 5.7 shows the trend of the rapid increase of Gurobi algorithm's computation time. We argue that the proposed algorithm can feedback feasible solution for practical big size problem, while the Gurobi algorithm cannot.

As shown in Table 5.5, the proposed linear programming embedded GA algorithm has a good performance for obtaining the near-optimal solution in most cases. In addition, the computation time has a substantial reduction. For a small-scale problem, the Gurobi solver can find the optimal solution faster, but it becomes very time-consuming for large-scale problems.

Table 5.5 Problem size and comparison result between proposed algorithm and Gurobi solver

Case id (I, J,K,S)	Number of possible platform configurations	Gurobi		Proposed algorithm					
				Population size=100			Population size=200		
		Obj.	CPU times (s)	Obj.	CPU times (s)	Gap (%)	Obj.	CPU times (s)	Gap (%)
4-3-9-20	331776	2036854.39	20.00	2037516.96	53.74	0.03%	2037516.96	101.94	0.03%
4-3-9-30	331776	2032088.48	25.00	2032348.08	54.65	0.01%	2032348.08	112.63	0.01%
4-4-11-20	160000	2996042.33	55.00	2996654.06	58.13	0.02%	2996654.06	113.52	0.02%
4-4-11-30	810000	2986444.82	65.00	2987250.67	67.58	0.03%	2987250.67	139.15	0.03%
5-3-9-20	7962624	2322782.15	220.00	2323928.91	70.93	0.05%	2323928.91	140.68	0.05%
5-3-9-30	7962624	2314864.03	340.00	2315529.99	81.41	0.03%	2315529.99	164.84	0.03%
5-4-11-20	2.55E+08	3554502.42	1320.00	3557456.40	77.16	0.08%	3555784.10	158.14	0.04%
5-4-11-30	2.55E+08	3538885.34	1690.00	3539767.37	111.80	0.02%	3539525.70	193.65	0.02%
5-5-13-20	8.15E+09	4489114.01	8650.00	4489114.01	117.12	0.00%	4489114.01	191.11	0.00%
6-3-9-20	1.91E+08	2964406.98	2325.00	2965496.31	103.41	0.04%	2965496.31	227.93	0.04%
6-4-11-20	1.22E+10	4318805.51	3840.00	4322572.98	155.35	0.09%	4320161.87	243.46	0.03%
6-4-11-30	1.22E+10	4302602.98	5255.00	4303958.95	203.73	0.03%	4303958.95	317.22	0.03%
6-5-13-20	7.83E+11	5429645.75	12445.00	5429645.75	166.81	0.00%	5429645.75	293.71	0.00%

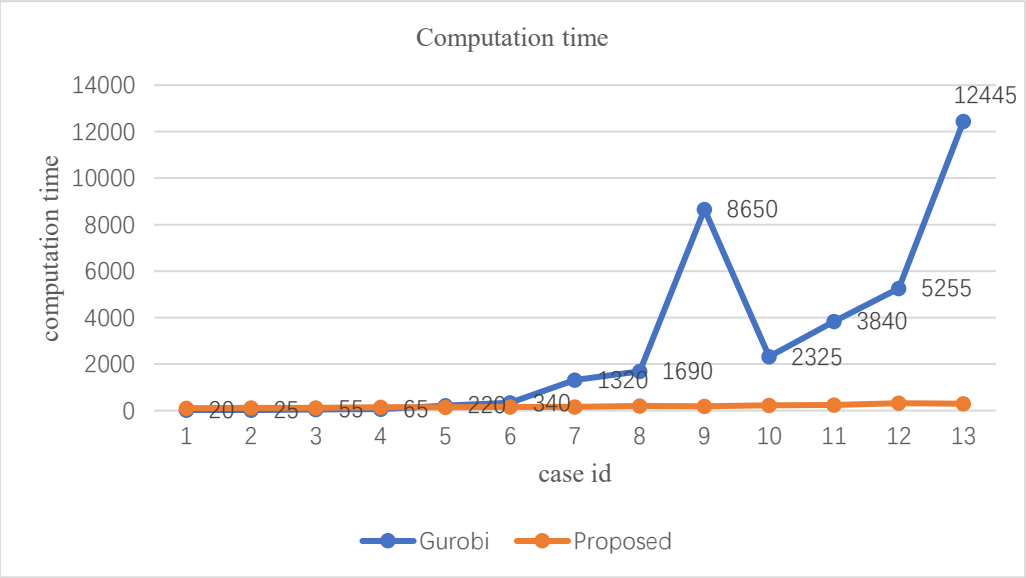

Figure 5.7 Rapid increase of computation time of Gurobi algorithm for different cases

The number of scenarios has a significant impact on the result of the stochastic platform configuration model. To determine the appropriate number of scenarios for case study and sensitivity analysis, we perform the experiments for different numbers of scenarios. The experiments will use a case with 5 products, 4 modules and 11 module options because we need the result obtained by the

Gurobi solver as a benchmark. 7 different scenarios (i.e., 10, 15, 20, 30, 40, 50, and 80) are used in the comparative experiments. As shown in Table 5.6 and Figure 5.8, the Gurobi algorithm solves the case with 20 scenarios in around 1400 seconds while it requires more than 6900 seconds to solve the case with 80 scenarios. However, the total cost obtained by solving the case with 80 scenarios improves by only 0.46% compared with the case with 20 scenarios. In the case of 10 scenarios, the total cost has a 0.92% gap compared with the case with 80 scenarios. Therefore, the following numerical experiments are based on a dataset of 20 scenarios.

Table 5.6 Comparative result for different number of scenarios

Case id (I,J,K)	number of scenarios	Gurobi			Proposed algorithm (population size = 1000)			
		Obj.	CPU times (s)	Gap % (compare with 80 scenarios)	Obj.	CPU times (s)	Gap % (compare with Gurobi)	Gap % (compare with 80 scenarios)
5-4-11	10	3562155.30	730	0.92%	3562197.02	498.11	0.00%	0.89%
5-4-11	15	3553168.02	965	0.66%	3553889.66	646.23	0.02%	0.65%
5-4-11	20	3545975.00	1470	0.46%	3546255.56	794.47	0.01%	0.44%
5-4-11	30	3538885.34	1690	0.26%	3539525.70	854.49	0.02%	0.25%
5-4-11	40	3537433.90	2495	0.22%	3537743.99	1125.76	0.01%	0.20%
5-4-11	50	3532291.38	3825	0.07%	3532804.41	1526.05	0.01%	0.06%
5-4-11	80	3529812.59	6970	0.00%	3530796.22	1836.00	0.03%	0.00%

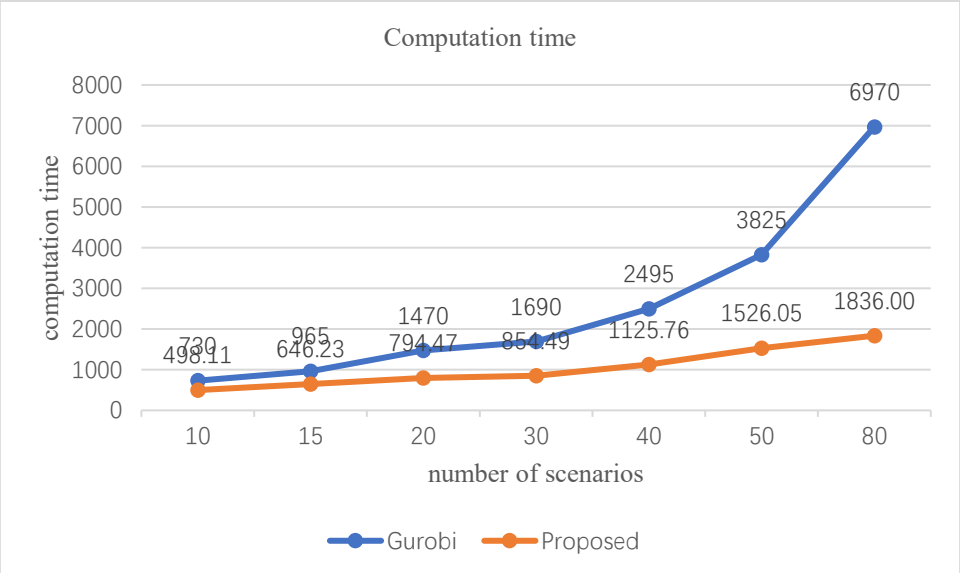

Figure 5.8 Rapid increase of computation time of Gurobi algorithm for different scenarios

5.5.3 Experiments on the case study

The optimal platform configuration decision for different demand case is presented in Table 5.7. As shown in Table 5.7, the optimal platform configuration is to develop 3 platforms for demand case 1 while developing 4 platforms for case 2 and 5 platforms for case 3. Specifically, in the demand case 1, product F_1 is developed from platform P_2 with the module option No.1,5,8,10 (i.e., m_{11}, m_{21}, m_{31}, m_{41}), products F_2 and F_5 are developed from platform P_5 with module option No.2,5,8,11 (i.e., m_{12}, m_{21}, m_{31}, m_{42}), products F_3 and F_5 are developed from platform P_4 with No.2,6,9,10 (i.e., m_{12}, m_{22}, m_{32}, m_{41}).

Another important decision is the platform design decision, i.e., what type of platforms will be developed. As shown in Table 5.7, product F_1, F_2, F_3 are derived from their respective matching-designed platform. The platforms have the same module option compared with the product requirements. Product F_4 is developed from the under-designed platform P_5 and product F_5 is developed from the under-designed platform P_4. Compared to the product requirements, platform P_5 has a lower specification module option No.2 (m_{12}) and No.5 (m_{21}) than the module option No.3 (m_{13}) and No.6 (m_{22}) needed by product F_4, which enables platform P_5 to be under-designed for product F_4. Some additional customizations will be incurred to upgrade the under-designed module option No.2 and No.5 to option No.3 and No.6, which will bring the additional customization cost. Similarly, platform P_4 is under-designed for module m_1, m_2, m_4 when deriving product F_5. The platform P_4 has a module option No.2 (m_{12}), No.6 (m_{22}), and No.10 (m_{41}), however, the product F_5 requires a higher specification module option No.4 (m_{14}), No.7 (m_{23}), and No.11 (m_{42}).

Moreover, as the demand quantity increases (i.e., a larger mean value), the number of platforms increases. As shown in Table 5.7, the optimal platform strategy configures 3 platforms in demand case 1 with mean value of 10000. However, 4 platforms will be developed in demand case 2 with mean value of 15000 and 5 platforms are developed in demand case 3 with mean value of 20000.

Table 5.7 Results of platform configuration for different demand cases

Demand case	Case 1 (Mean value=10000)			Case 2 (Mean value = 15000)			Case 3 (Mean value = 20000)		
C_t	2439739.07			3547328.52			4622176.52		
C_d	165000			250000			305000		
C_m^{mp}	1743138.48			2846247			3998173.59		
C_p^{mp}	95467.73			140845.95			194904.73		
C_m^{cu}	84575.24			143844.72			115494.69		
C_p^{cu}	7046.99			9600.76			8524.48		
C_c	344399.85			156699.32			0		
C_s	110.78			90.77			79.03		
Product requirement	Assignment and design decision	Platform configuration	quantity of pre-assembly platform	Assignment and design decision	Platform configuration	quantity of pre-assembly platform	Assignment and design decision	Platform configuration	quantity of pre-assembly platform
F1[1,5,8,10]	e(1,2)-match			e(1,2)-match	P1[2,6,9,10]	14514.48	e(1,1)-match	P1[1,5,8,10]	19604.99
F2[2,5,8,11]	e(2,5)-match	P2[1,5,8,10]	9105.19	e(2,2)-under	P2[1,5,8,10]	28766.64	e(2,2)-match	P2[2,5,8,11]	19218.04
F3[2,6,9,10]	e(3,4)-match			e(3,1)-match	P3[4,7,9,11]	14257.11	e(3,5)-match	P3[2,6,9,10]	19650.84
F4[3,6,8,11]	e(4,5)-under	P4[2,6,9,10]	19455.65	e(4,5)-match			e(4,4)-match	P4[3,6,8,11]	19407.68
F5[4,7,9,11]	e(5,4)-under	P5[2,5,8,11]	19173.02	e(5,3)-match	P4[3,6,8,11]	14384.11	e(5,3)-match	P5[4,7,9,11]	19570.81

5.5.4 Evaluation on the effectiveness of platform configuration stochastic model

To compare the effectiveness of the formulated stochastic model with the deterministic platform configuration model, the value of stochastic solution (VSS) and the expected value of perfect information (EVPI) are calculated. For calculating the VSS and EVPI, three methods are required, i.e., proposed stochastic model, deterministic model, and wait-and-see model. The deterministic model is formulated by using the average value of demand generated by a stochastic model. The VSS is the difference between the objective function value of the stochastic model and the expected value solution (EEV), whereas the EVPI is calculated as the difference between the wait-and-see solution (WSS) and the objective function value of stochastic programming (SP). Other parameter settings are the same as demand case 2 with a mean value of 15000.

To calculate the VSS, we first calculate the EEV. The EEV will be calculated through two steps.

In the first step, we use the average value of demand scenarios generated by the stochastic model to obtain the first stage variables. The obtained first stage solutions are then used to calculate the second stage variables and obtain the objective function value. The corresponding objective value is the EEV. The value of EEV is not better than the objective value of stochastic model because the solution obtained by the deterministic model using average data is not the optimal solution for the original stochastic model. The gap between the EEV and SP is the VSS, equal to 34259.6, which implies the cost of ignoring uncertainty when making decisions.

To calculate the EVPI, the value of WSS is calculated first. The WSS model is to solve the stochastic model using single demand scenario for all the demand scenarios generated previously. After solving 20 scenarios, the average value of their objective values is denoted by WSS. The gap between the SP and WSS is the EVPI, which is equal to 31981.13 in this experiment. The EVPI indicates how much the decision marker is willing to pay in order to know the exact information about uncertainty.

Table 5.8 Comparative result of stochastic and deterministic solutions under demand uncertainty

Models		Total cost			
Stochastic programming (SP)		3547328.53			
Expected value solution (EEV)		3581588.13			
Wait and see solutions (WSS)		3579309.66			
Scenario	Probability	Total cost	Scenario	Probability	Total cost
1	0.05	3590440.68	11	0.02	3606752.74
2	0.01	3557659.09	12	0.03	3586547.56
3	0.08	3632868.39	13	0.05	3552657.33
4	0.08	3566738.09	14	0.04	3577035.71
5	0.05	3557765.44	15	0.06	3624308.03
6	0.03	3570648.94	16	0.11	3534909.11
7	0.05	3590303.79	17	0.1	3579299.91
8	0.06	3637916.21	18	0.04	3560934.11
9	0.02	3570119.36	19	0.05	3530489.26
10	0.01	3581850.56	20	0.06	3576460.84

5.6 Sensitivity analysis

5.6.1 Cost sensitivity analysis

The impact of cost parameters including coefficient of variable development cost α and the unit customization cost f is analyzed in this section. Meanwhile, the demand fluctuation is also analyzed by setting different variances of demand distribution. Two different variances of demand distribution are provided. Figure 5.9 illustrates how the optimal platform configuration varies with different cost parameter α and f with a demand distribution U (a, b) while Figure 5.10 shows the results with a lager variance U (0.25a, b+0.75a). 12 cost scenarios were tested by combining 4 parameters α from 1000 to 4000 and 3 parameters f from 1.5 to 2.5, indicated by the symbols $cs_{\alpha f}$. For example, the cost scenario cs_{21} represents a cost scenario with $\alpha = 2000$ and $f = 1$. Other parameter settings are the same as demand case 2. The current situation cs_{11} in Figure 5.9 is the optimal platform configuration discussed in section 5.5.3, in which 5 products are derived from 4 platforms with the design decisions represented by 4M and 1U.

The increased variable development cost α enables the model to configure less product platforms. For example, as shown in Figure 5.9, the number of platforms is 4 in scenario cs_{11} while it is 2 in scenario cs_{41} as the variable development parameter α increases from 1000 to 4000. Likewise, as α increases, the number of platforms decreases from 5 in scenario cs_{12} to 2 in scenario cs_{42} and from 5 in scenario cs_{13} to 2 in scenario cs_{43}. In the case of the larger demand fluctuation, the same trend can be found. For example, as shown in Figure 5.10, the number of platforms is 3 in scenario cs_{11}, however, it is 2 in scenario cs_{21} and cs_{31} while it is 1 in scenario cs_{41}.

Moreover, the optimal number of platforms is likely to increase as the customization cost increases. For example, in the case of lower demand fluctuation shown in Figure 5.9, the number of

platforms increases from 4 in scenario cs_{11} to 5 in scenario cs_{13} as the unit customization cost f increase from 1.5 to 2.5. Similarly, the number of platforms increases from 2 in scenario cs_{21} to 5 in scenario cs_{23} and from 2 in scenario cs_{31} to 4 in scenario cs_{33}. In the case of larger demand fluctuation, the number of platforms also increases as the customization cost increases. As shown in Figure 5.10, 3 platforms are developed in the scenario cs_{11} while 5 platforms are developed in scenario cs_{12} and cs_{13}.

In addition, the number of products derived from their matching designed platform will decrease as the development cost increase. For example, as shown in Figure 5.9, 5 products are derived from 4 matching-designed platforms in scenario cs_{11}. The number of products derived from matching-designed platforms is 2 in scenario cs_{21}, cs_{31}, and cs_{41}. The same trend can be found in a larger demand fluctuation. As shown in Figure 5.10, the number of products derived from matching platform decrease from 3 in scenario cs_{11} to 2 in scenario cs_{21} and cs_{31} while no matching-designed platform is developed in scenario cs_{41}. On the other hand, deriving products from the over-designed platforms become more frequent in the presence of high development cost and customization cost. For example, as shown in Figure 5.9, no product is derived from the over-designed platform in the case of low development cost (e.g., scenario cs_{11}, cs_{12} and cs_{13}), while 1 product is derived from the over-designed platforms in scenario cs_{42} and 3 products are derived from the over-designed platforms in scenario cs_{43}.

The under-designed platforms may be more prevalent under the larger demand fluctuation. For example, in the scenario cs_{11}, 5 products are developed from 4M,1U platforms in Figure 5.9 while 5 products are developed from 3M,2U platforms in Figure 5.10. In scenario cs_{41}, the number of under-designed platform is 3 in Figure 5.9 while it is 4 in Figure 5.10. Less platforms and more under-designed platforms can be subsequently customized to cope with the greater fluctuation in demand.

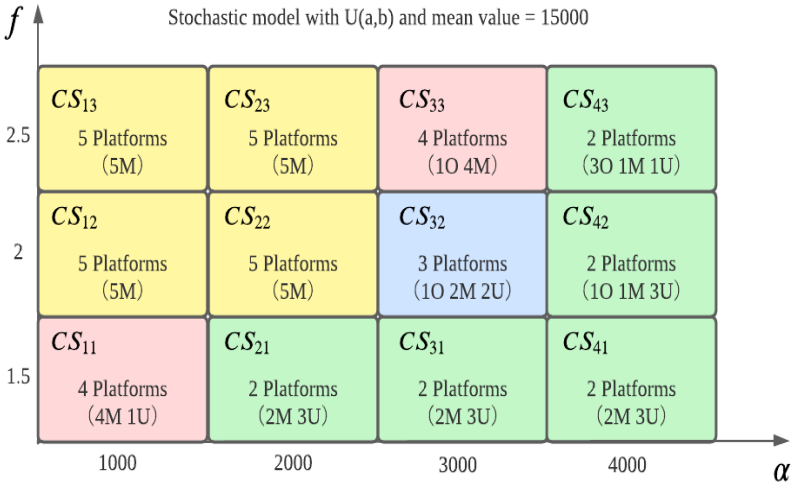

Figure 5.9 Optimal platform design decision with varying development and customization cost under demand distribution U (a, b)

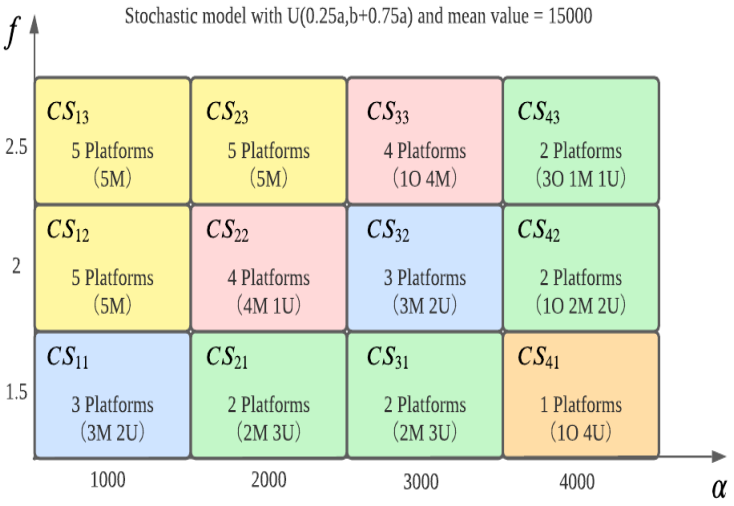

Figure 5.10 Optimal platform design decision with varying development and customization cost under demand distribution U (0.25a, b+0.75a)

5.6.2 Demand sensitivity analysis

Three demand cases with different mean values and variances are provided to illustrate the impact of demand on the optimal platform configuration. As shown in Figure 5.11 – 5.13, we found that the

number of platforms increases as the demand quantity represented by the mean value increases. For example, 3 platforms are configured to derive 5 products in Figure 5.11, while 4 platforms are configured in Figure 5.12 and 5 platforms are offered in Figure 5.13.

Compared with the results under different demand variances, we found that the number of platforms may decrease as demand uncertainty increases. As shown in Figure 5.11, 3 platforms are developed in the case of a demand distribution U (a, b) while 2 platforms are developed when the demand distribution follows U (0.25a, b+0.75a). The same trends can be found in Figure 5.12 and Figure 5.13.

Furthermore, the under-designed platforms may be more prevalent when the demand uncertainty is higher. As shown in Figure 5.11, 5 products are developed from 3M,2U platforms under a demand distribution U (a, b) while 5 products are developed from 1O,1M,3U platforms under a demand distribution U (0.25a, b+0.75a). The same trends can be found in Figure 5.12 and 5.13. As shown in Figure 5.12, the number of under-designed platform increase from 1 to 2 as the demand uncertainty increases. Less platforms and more under-designed platforms are used to hedge the risk of demand uncertainty.

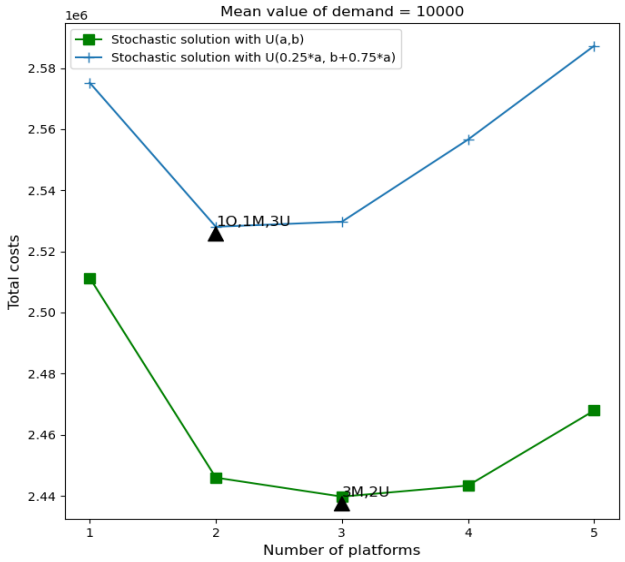

Figure 5.11 The impact of demand fluctuation on platform configuration decision with demand mean value 10000

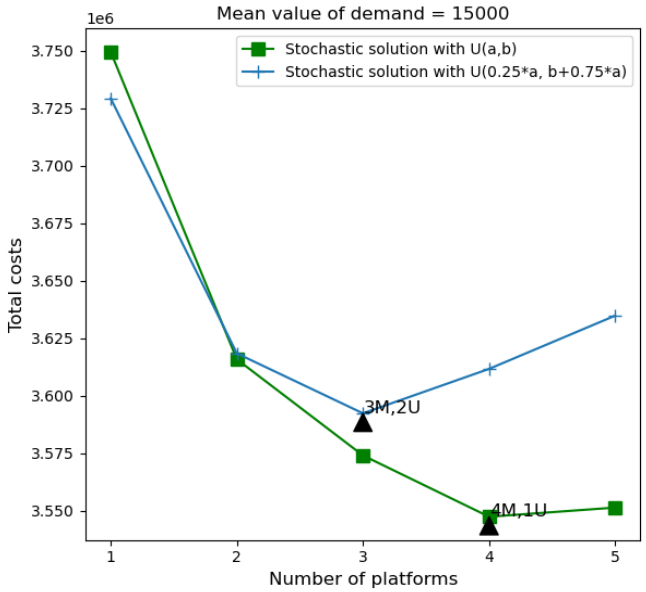

Figure 5.12 The impact of demand fluctuation on platform configuration decision with demand mean value 15000

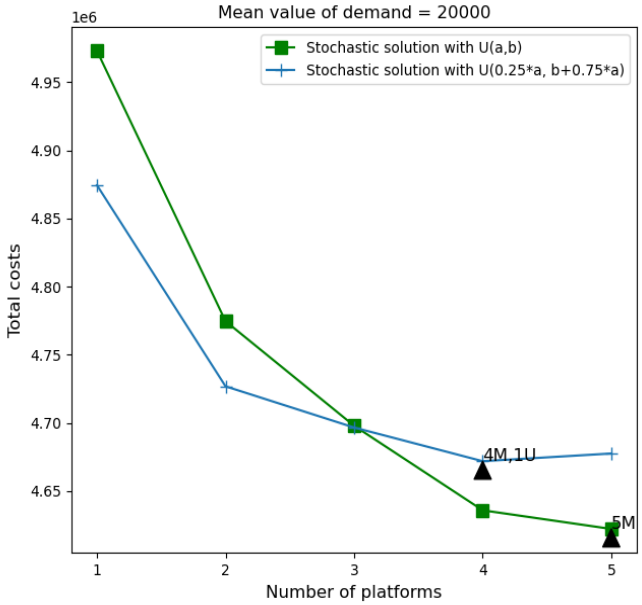

Figure 5.13 The impact of demand fluctuation on platform configuration decision with demand mean value 20000

5.7 Conclusions

In this section, we studied the platform configuration problem considering demand uncertainty. The proposed model was formulated as a two-stage stochastic programming model. The demand uncertainty was modelled using scenario-based stochastic programming where every possible random situation was represented by a scenario with an associated probability. Two stages of the PPC problem with uncertain demand were identified, i.e., the platform configuration stage and customization stage. A cost model including the development cost of platform, production and material cost for two different stages, customization cost, penalty cost of excessive platforms was developed. A linear programming embedded genetic algorithm was developed to solve the proposed model.

Numerical experiments show that the optimal number of platforms increases as the demand quantity increases while the optimal number of platforms decreases as the demand uncertainty increases. Under-designed platforms may be more prevalent when the demand uncertainty is higher.

Moreover, a cost sensitivity analysis is conducted to illustrate the impact of development cost and customization cost on the optimal platform configuration. The number of platforms will decrease as the development cost increases while the number of platforms is likely increases as the customization cost increases. Meanwhile, the number of products derived from their matching designed platform will decrease as the development cost increase. However, deriving products from over-designed platforms become more frequent in the presence of high development cost and customization cost.

Chapter 6 Conclusions and future work

6.1 Conclusions

Platform-based product development (PPD) is a cost-efficient approach to achieve mass customization. In this study, the product platform configuration (PPC) problem is examined from a supply chain management perspective. A modular platform configuration model is targeted and various optimization methods are applied to obtain the optimal platform configuration decision. The main conclusions are summarized as follows.

In chapter 3, a platform configuration model considering module selection and integration was examined. The results show that PPD approach can reduce supply chain costs by applying module integration in the platform configuration. However, over-pursuing module integration is not always beneficial for a company, which will bring a higher cost sometimes. Manufacturers need to balance module integration and module selection when designing product platforms. Moreover, sensitivity analysis shows that several parameters noticeably affect the platform configuration decision. The larger demand and the longer product lifetimes favor more product platforms and encourage module selection instead of module integration. The higher development cost will reduce the number of product platforms and encourage module integration while restricting the module selection.

In chapter 4, a platform configuration model considering platform design strategy and supplier selection was proposed. The results show that the optimal number of product platforms will decrease as the development cost increases, while the number of platforms increases as the customization cost increases. Moreover, as the development cost increases, few products are derived from their matching-designed platforms, while more products are derived from the over-designed platforms.

The over-designed platform is more prevalent in the presence of high development cost and high customization cost. In contrast, the matching-designed platform is more suitable for the case of low development cost and high customization cost. In addition, the larger total demand and longer product lifetime drive the model to develop more platforms and tend to derive products from their matching-designed platforms.

In chapter 5, a stochastic programming model was proposed to handle the demand uncertainty while considering platform customization. The results show that the number of platforms increases as the demand quantity increases while the number of platforms decreases as the demand uncertainty increases. Matching-designed platforms are less used in the case of greater demand fluctuation. In contrast, under-designed platforms are more frequently used to hedge the risk of demand uncertainty. In addition, the impact of development cost and customization cost on platform configuration decision is similar under different demand uncertainty. The number of platforms decreases as the development cost increases while the number of platforms increases as the customization cost increases. Less products will be derived from their matching-designed platforms as the development cost increases. However, deriving products from over-designed platforms becomes more frequent in the presence of high development cost and customization cost.

6.2 Limitations and future work

The limitations and future works are presented as follows.

Firstly, the product requirements corresponding to the customer preferences are assumed to be given. However, the customer needs for the product requirements, features, and functional levels are changing, and we do not know once we receive orders. A flexible platform configuration model considering the changing customer requirements is required.

Secondly, our proposed model focuses more on the internal activities within a manufacturer, such as procurement, manufacturing, production, and inventory control. The suppliers and factories are

located all over the world in a global supply chain environment. Investigating the platform configuration problem from a global supply chain perspective is another interesting problem.

Thirdly, this study does not address the interactive mechanism of multiple departments. The coordination mechanism of multiple departments is regarded to be the key point to supply chain management. It will be meaningful to consider the impact of coordination mechanisms with external suppliers or internal departments on the platform configuration problem.

Finally, our research considers only cost, all those other parameters that affect decision-making on supplier selection does not be considered. Our model includes two types of costs, i.e., the engineering cost and the SCM-related cost. However, the detail of the cost in practice may be different from the proposed model. Detail costs and factors should be treated carefully to satisfy the purpose of the decision-making on platform configuration.

References

Agrawal, T., Sao, A., Fernandes, K. J., Tiwari, M. K., Kim, D. Y., 2013. A hybrid model of component sharing and platform modularity for optimal product family design. International Journal of Production Research 51 (2), 614-625.

AlGeddawy, T., ElMaraghy, H., 2013. Reactive design methodology for product family platforms, modularity and parts integration. CIRP Journal of Manufacturing Science and Technology 6 (1), 34-43.

Andersen, R., Brunoe, T. D., Nielsen, K., 2022. Platform-based product development in the process industry: a systematic literature review. International Journal of Production Research, 1-24.

Askhøj, C., Mortensen, N. H., 2020. Deciding on the total number of product architectures. Concurrent Engineering 28 (1), 20-31.

Ben-Arieh, D., Easton, T., Choubey, A. M., 2009. Solving the multiple platforms configuration problem. International Journal of Production Research 47 (7), 1969-1988.

Bonvoisin, J., Halstenberg, F., Buchert, T., Stark, R., 2016. A systematic literature review on modular product design. Journal of Engineering Design 27 (7), 488-514.

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

Chen, C., Wang, L., 2008. Product platform design through clustering analysis and information theoretical approach. International Journal of Production Research 46 (15), 4259-4284.

Colombo, E. F., Shougarian, N., Sinha, K., Cascini, G., de Weck, O. L., 2020. Value analysis for customizable modular product platforms: theory and case study. Research in Engineering Design 31, 123-140.

Corporation., T., 2021. TNGA. Mobility. Toyota Motor Corporation Official Global Website. Accessed 5 November 2021. Available at: https://global.toyota/en/mobility/tnga/.

Du, G., Jiao, R. J., Chen, M., 2014. Joint optimization of product family configuration and scaling design by Stackelberg game. European Journal of Operational Research 232 (2), 330-341.

ElMaraghy, H., Schuh, G., ElMaraghy, W., Piller, F., Schönsleben, P., Tseng, M., Bernard, A., 2013. Product variety management. CIRP Annals 62 (2), 629-652.

ElMaraghy, H., Moussa, M., 2019. Optimal platform design and process plan for managing variety using hybrid manufacturing. CIRP Annals 68 (1), 443-446.

Ericsson, A., Erixon, G., 1999. Controlling design variants: modular product platforms. Society of Manufacturing Engineers.

Fisher, M. L., Ittner, C. D., 1999. The impact of product variety on automobile assembly operations: Empirical evidence and simulation analysis. Management Science 45 (6), 771-786.

Fogliatto, F. S., Da Silveira, G. J., Borenstein, D., 2012. The mass customization decade: An updated review of the literature. International Journal of Production Economics 138 (1), 14-25.

Fujita, K., Amaya, H., Akai, R., 2013. Mathematical model for simultaneous design of module commonalization and supply chain configuration toward global product family. Journal of Intelligent Manufacturing 24, 991-1004.

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

Galizia, F. G., ElMaraghy, H., Bortolini, M., Mora, C., 2020. Product platforms design, selection and customisation in high-variety manufacturing. International Journal of Production Research 58 (3), 893-911.

Glock, C. H., Ries, J. M., 2013. Reducing lead time risk through multiple sourcing: the case of stochastic demand and variable lead time. International Journal of Production Research 51 (1), 43-56.

Goswami, M., Pratap, S., Kumar, S. K., 2016. An integrated Bayesian-Game theoretic approach for product portfolio planning of a multi-attributed product in a duopolistic market. International Journal of Production Research 54 (23), 6997-7013.

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

Hanafy, M., ElMaraghy, H., 2015. A modular product multi-platform configuration model. International Journal of Computer Integrated Manufacturing 28 (9), 999-1014.

Hanafy, M., ElMaraghy, H., 2017. Modular product platform configuration and co-planning of assembly lines using assembly and disassembly. Journal of Manufacturing Systems 42, 289-305.

Hillier, F., Lieberman, G., 2005. Introduction to operation research (Eighth edition). New York: McGraw-Hill.

Huang, G. Q., Zhang, X. Y., Liang, L., 2005. Towards integrated optimal configuration of platform products, manufacturing processes, and supply chains. Journal of Operations Management 23 (3-4), 267-290.

Huang, G. Q., Zhang, X. Y., Lo, V. H., 2007. Integrated configuration of platform products and supply chains for mass customization: a game-theoretic approach. IEEE Transactions on Engineering Management 54 (1), 156-171.

Jacobs, F. R., Chase, R. B., 2018. Operations and supply chain management (p.40). New York: McGraw-Hill Education.

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

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

Jiao, J. R., Zhang, Y., Wang, Y., 2007. A heuristic genetic algorithm for product portfolio planning. Computers & Operations Research 34 (6), 1777-1799.

Krishnan, V., Gupta, S., 2001. Appropriateness and impact of platform-based product development. Management Science 47 (1), 52-68.

Li, X., Yang, D., Hu, M., 2018. A scenario-based stochastic programming approach for the product configuration problem under uncertainties and carbon emission regulations. Transportation Research Part E: Logistics and Transportation Review 115, 126-146.

Liu, Z., Wong, Y. S., Lee, K. S., 2010. Modularity analysis and commonality design: a framework for the top-down platform and product family design. International Journal of Production Research 48 (12), 3657-3680.

Luo, X. G., Kwong, C. K., Tang, J. F., Deng, S. F., Gong, J., 2011. Integrating supplier selection in optimal product family design. International Journal of Production Research 49 (14), 4195-4222.

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

Miao, C., Du, G., Jiao, R. J., Zhang, T., 2017. Coordinated optimisation of platform-driven product line planning by bilevel programming. International Journal of Production Research 55 (13), 3808-3831.

Mikkola, J. H., 2007. Management of product architecture modularity for mass customization: modeling and theoretical considerations. IEEE Transactions on Engineering Management 54 (1), 57-69.

Moussa, M., ElMaraghy, H., 2020. A genetic algorithm-based model for product platform design for hybrid manufacturing. Procedia CIRP 93, 389-394.

Moussa, M., ElMaraghy, H., 2021. Multiple platforms design and product family process planning for combined additive and subtractive manufacturing. Journal of Manufacturing Systems 61, 509-529.

Moussa, M., ElMaraghy, H., 2022. Multi-period additive/subtractive product platform design and inventory management. International Journal of Production Research 60 (24), 7262-7280.

Nepal, B., Monplaisir, L., Famuyiwa, O., 2012. Matching product architecture with supply chain design. European Journal of Operational Research 216 (2), 312-325.

Okpoti, E. S., Jeong, I. J., Moon, S. K., 2019. Decentralized determination of design variables among cooperative designers for product platform design in a product family. Computers & Industrial Engineering 135, 601-614.

Otto, K., Hölttä-Otto, K., Simpson, T. W., Krause, D., Ripperda, S., Ki Moon, S., 2016. Global views on modular design research: linking alternative methods to support modular product family concept development. Journal of Mechanical Design 138 (7), 1101-1116.

Pine, B. J., Victor, B., Boynton, A. C., 1993. Making mass customization work. Harvard business review 71 (5), 108-116.

Pirmoradi, Z., Wang, G. G., Simpson, T. W., 2014. A review of recent literature in product family design and platform-based product development. Advances in product family and product platform design: methods & applications, 1-46.

Qu, T., Bin, S., Huang, G. Q., Yang, H. D., 2011. Two-stage product platform development for mass customisation. International Journal of Production Research 49 (8), 2197-2219.

Ripperda, S., Krause, D., 2017. Cost effects of modular product family structures: Methods and quantification of impacts to support decision making. Journal of Mechanical Design 139 (2), 1-12.

Sadeghi, A., Alem-Tabriz, A., Zandieh, M., 2011. Product portfolio planning: a metaheuristic-based simulated annealing algorithm. International Journal of Production Research 49 (8), 2327-2350.

Salvador, F., Forza, C., Rungtusanatham, M., 2002. Modularity, product variety, production volume, and component sourcing: theorizing beyond generic prescriptions. Journal of Operations Management 20 (5), 549-575.

Silver, E. A., Pyke, D. F., Thomas, D. J., 2016. Inventory management and production management in supply chains (Fourth edition). Boca Raton: CRC Press.

Simpson, T. W., 2004. Product platform design and customization: Status and promise. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 18 (1), 3-20.

Simpson, T. W., Jiao, J., Siddique, Z., Hölttä-Otto, K., 2014. Advances in product family and product platform design. New York: Springer.

Song, Q., Ni, Y., 2019. Optimal platform design with modularity strategy under fuzzy environment. Soft Computing 23, 1059-1070.

Tan, C., Barton, K., Hu, S. J., Freiheit, T., 2022. Integrating optimal process and supplier selection in personalised product architecture design. International Journal of Production Research 60 (8), 2461-2480.

Tseng, M. M., Jiao, J., 2001. Mass customization. Handbook of Industrial Engineering 3, 684-709.

Van den Broeke, M., Boute, R., Samii, B., 2015. Evaluation of product-platform decisions based on total supply chain costs. International Journal of Production Research 53 (18), 5545-5563.

Van den Broeke, M., Boute, R., Cardoen, B., Samii, B., 2017. An efficient solution method to design the cost-minimizing platform portfolio. European Journal of Operational Research 259 (1), 236-250.

Wang, J., Chen, Z., Wu, Z., 2018. A configuration study on supply chain strategy with focus on product variety management. Journal of Advanced Mechanical Design, Systems, and Manufacturing 12 (3),1-16.

Wang, D., Du, G., Jiao, R. J., Wu, R., Yu, J., Yang, D., 2016. A Stackelberg game theoretic model for optimizing product family architecting with supply chain consideration. International Journal of Production Economics 172, 1-18.

Wang, T., Wang, J., Matsukawa, H., 2022. Integrating optimal configuration of product platform and supplier selection in mass customization. Journal of Advanced Mechanical Design, Systems, and Manufacturing 16 (5), 1-15.

Wang, T., Wang, J., Jin, G., Matsukawa, H., 2022. Product platform configuration decision in NPD with uncertain demands and module options. International Journal of Production Research, 1-20.

Xiong, Y., Du, G., Jiao, R. J., 2018. Modular product platforming with supply chain postponement decisions by leader-follower interactive optimization. International Journal of Production Economics 205, 272-286.

Yang, D., Li, X., Jiao, R. J., Wang, B., 2018. Decision support to product configuration considering component replenishment uncertainty: A stochastic programming approach. Decision Support Systems 105, 108-118.

Yang, D., Jiao, J. R., Ji, Y., Du, G., Helo, P., Valente, A., 2015. Joint optimization for coordinated configuration of product families and supply chains by a leader-follower Stackelberg game. European Journal of Operational Research 246 (1), 263-280.

Zhang, X., Huang, G. Q., Rungtusanatham, M. J., 2008. Simultaneous configuration of platform products and manufacturing supply chains. International Journal of Production Research 46 (21), 6137-6162.

Zhang, X., Huang, G. Q., Humphreys, P. K., Botta-Genoulaz, V., 2010. Simultaneous configuration of platform products and manufacturing supply chains: comparative investigation into impacts of different supply chain coordination schemes. Production Planning & Control 21 (6), 609-627.

Zhang, H., Qin, S., Li, R., Zou, Y., Ding, G., 2020. Progressive modelling of feature-centred product family development. International Journal of Production Research 58 (12), 3701-3723.

Zhao, S., Zhang, Q., Peng, Z., Lu, X., 2022. Product platform configuration for product families: Module clustering based on product architecture and manufacturing process. Advanced Engineering Informatics 52, 101622.

Acknowledgements

First of all, I would like to thank my supervisor, Prof. Hiroaki Matsukawa, for his patient guidance and valuable advice. He has devoted much time and effort to instruct my research and academic writing. I have benefited greatly from his profound knowledge and plentiful experience. I am deeply grateful for his continuous assistance and support in all the time of my research and my life.

I would also like to express my appreciation to my master supervisor, Prof. Jian Wang at Shanghai University. Prof. Wang introduces me to the areas of product variety management, modular design, and product platform. I sincerely thank him for his valuable suggestions and encouragement during the past eight years.

I gratefully acknowledge the scholarship provided by the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan. I also thank the Keio leading-edge laboratory of Science and Technology for providing research funding.

Finally, I am grateful to all my friends and laboratory members for their discussions and help. A special thanks to my family. Thanks to their unconditional support and encouragement during my Ph.D. period.