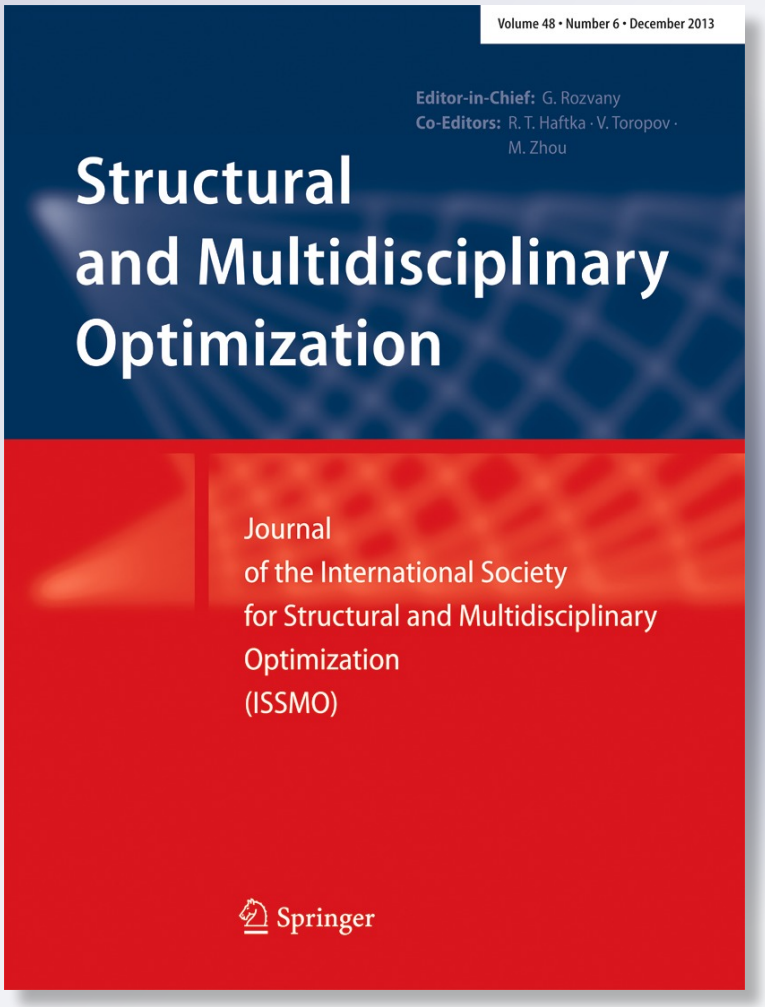
Investigating the commonality attributes for scaling product families using comprehensive product platform planning (CP³)

Souma Chowdhury, Achille Messac & Ritesh Khire

Structural and Multidisciplinary Optimization

ISSN 1615-147X
Volume 48
Number 6

Struct Multidisc Optim (2013)
48:1089-1107
DOI 10.1007/s00158-013-0953-2



Springer

Struct Multidisc Optim (2013) 48:1089–1107
DOI 10.1007/s00158-013-0953-2

RESEARCH PAPER

Investigating the commonality attributes for scaling product families using comprehensive product platform planning (CP^3)

Souma Chowdhury · Achille Messac · Ritesh Khire

Received: 14 December 2012 / Revised: 6 May 2013 / Accepted: 30 May 2013 / Published online: 4 July 2013

Abstract A product family with a platform paradigm is expected to increase the flexibility of the manufacturing process to market changes, and to take away market share from competitors that develop one product at a time. The recently developed Comprehensive Product Platform Planning (CP^3) method (i) presents a generalized model, (ii) allows the formation of product sub-families, and (iii) provides simultaneous identification and quantification of platform/scaling design variables. The CP^3 model is founded on a generalized commonality matrix that represents the platform planning process as a *mixed-binary nonlinear programming (MBNLP)* problem. This MBNLP problem is high-dimensional and multimodal, owing to the commonality constraint. In this paper, the complex MBNLP model is reduced to a tractable MINLP problem without resorting to limiting approximations; along the reduction process, redundancies in the original commonality matrix are also favorably addressed. To promote a better understanding of the importance of a reduced MINLP, this paper also provides a uniquely comprehensive formulation of the number of possible platform combinations (or commonality combinations). In addition, a new commonality index (CI) is developed to simultaneously account for the inter-product commonalities (based on design variable sharing) and the overlap between groups of products sharing different platform variables. To maximize the performance of the product family and the commonality index yielded by the new CP^3 model, we apply an advanced mixed-discrete Particle Swarm Optimization algorithm. The potential of the new CP^3 framework is illustrated through its application to design scalable families of electric motors. Maximizing the new CI produced families with more commonality among similar sets of motor variants (compared to maximizing the conventional CI), which can be a beneficial platform attribute for a wide range of product families.

Keywords Commonality matrix · Mixed integer nonlinear programming (MINLP) · Particle swarm optimization (PSO) · Product family · Product platform planning

S. Chowdhury
Multidisciplinary Design and Optimization Laboratory,
Department of Mechanical, Aerospace and Nuclear Engineering,
Rensselaer Polytechnic Institute, Troy, NY 12180, USA

A. Messac (✉)
Department of Mechanical and Aerospace Engineering,
Syracuse University, Syracuse, NY 13244, USA
e-mail: messac@syr.edu

R. Khire
United Technologies Research Center (UTRC),
East Hartford, CT 06118, USA

1 Introduction

The development of a family of products that satisfies different market niches introduces significant challenges to today's manufacturing industries—from development time to after-market services. A product family with a common platform paradigm offers a powerful solution to these daunting challenges (Simpson et al. 2006). Specifically, sharing of a common platform by different products is expected to result in: (i) reduced overhead, (ii) lower per-product cost, and (iii) increased profit. The key to a successful product family is the effectiveness of the product platform around which the product family is derived. This paper presents a product platform planning model that addresses the key aspects of scale-based product family design; particularly, we focus on how *inter-product commonality can be quantified and accounted for in the design process*. In this section,

a brief overview of the product family design methodologies is provided. A review of the existing measures of *inter-product commonalities* (in a family) and the unique attributes of these measures are presented in the later part of this section.

1.1 Product family design (PFD) and optimization

Depending on their design architecture, product families have traditionally been classified as (1) modular (module-based), or (2) scalable (scale-based). In a scale-based product family, each individual product is comprised of the same set of physical design variables. Different products in the family are developed by scaling the non-platform features (design variables) such that each product satisfies a unique set of requirements. In a module-based product family, distinct modules are added or substituted (on a common platform) to develop different products (Simpson et al. 1996; Simpson 2004). A popular example of a modular product family is the series of Sony Walkmans (Uzumeri and Sanderson 1995; Sanderson and Uzumeri 1997), whereas a standard example of a scalable product family is Boeing's 777 aircraft series (Sabbagh 1996).

Earlier scale-based product family design methodologies can be divided into two broad categories—(i) the *two-step approach*, and (ii) the *exhaustive approach*. A comprehensive list of different *two-step* methods can be found in the book chapter by Simpson (2006). Both of these approaches make limiting assumptions that restrict their applicability to the broad scope of product family design. A handful of new methods to design product families, which do not belong to the above two broad categories, have also been reported in the literature (Khire and Messac 2008; Khajavirad et al. 2009; Chen and Wang 2008); these methods address most of the limitations of the earlier methods, and also present other uniquely favorable characteristics. The application of innovative optimization approaches, such as physical programming (Messac 1996), has also been reported in the product family design literature (Messac et al. 2002a, b). Several well known methods exist in modular product family design, such as presented by Stone et al. (2000), Dahmus et al. (2001), Guo and Gershenson (2003), Fujita and Yoshida (2004), Jose and Tollenaere (2005), Kalligeros et al. (2006), Sharon et al. (2009), and Yu et al. (2007).

The Comprehensive Product Platform Planning (CP^3) framework, developed by Chowdhury et al. (2011), seeks to coherently address a wide range of problem scenarios. This framework presents a generalized mathematical model of the platform planning process, which yields a *mixed-binary nonlinear programming* (MBNLP) problem with a large number of binary variables. Owing to this high-dimensionality and likely multimodality (Chowdhury et al. 2011) of the product family model, this MBNLP problem presents unique numerical challenges. Chowdhury et al. (2011) developed and implemented a Platform Segregating Mapping Function (PSMF) method to convert the MBNLP problem into a less expensive continuous optimization problem; the approximated problem was solved using standard Particle Swarm Optimization (PSO). From a mathematical standpoint, the direct solution of the MBNLP problem, without making limiting approximations, is however more desirable. In this paper, the complex *mixed-binary nonlinear programming problem* defined by the original CP^3 model is hence reduced to a more tractable MINLP problem, without resorting to typical approximations. The MINLP problem is further simplified by eliminating the inherent redundancies in the original CP^3 model, attributed to the transitivity constraints. In the context of the complexity of the MINLP problem yielded by platform planning, we also provide a generalized quantification of the possible number of candidate platform combinations or commonality combinations. Subsequently, a solution strategy that uses a recently-developed powerful mixed-discrete Particle Swarm Optimization algorithm is formulated. This solution strategy is applied to design a family of universal electric motors.

1.2 Measures of inter-product commonality

Commonality metrics are measures of the extent of inter-product commonality in a family of products. One of the most popular product commonality metrics used in product family design is the Commonality Index (CI) developed by Martin and Ishii (1996). Khajavirad and Michalek (2008) illustrated the effectiveness of this commonality index (1996) in measuring the commonality-induced tooling cost savings. Kota et al. (2000) developed a a Product line Commonality Index (PCI) that provides an objective measure of product differentiation. In the PCI, only those product differences are penalized which correspond to features that can practically be shared between the product variants, thereby helpfully avoiding over-penalizing product families with greater feature mix.

Thevenot and Simpson (2006) presented a comprehensive comparison of six commonality indices, including the CI (Martin and Ishii 1996) and the PCI (Kota et al. 2000); the other four indices compared were: (i) Degree of Commonality Index (DCI) (Collier 1981), (ii) Total Constant Commonality Index (TCCI) (Wacker and Trelevan 1986), (iii) Percent Commonality Index (%C) (Siddique et al. 1998), and (iv) Component Part Commonality ($CI^{(c)}$) (Jiao and Tseng 2000). In the context of product family design strategy, Thevenot and Simpson (2006) reported that the DCI, TCCI, and CI focus on the *number of common components*, the PCI focuses on the *non-differentiating components*, the %C focuses on the number of common components, common connections and

assembly, and the $CI^{(c)}$ focuses on the cost of the components. Partly similar to the $CI^{(c)}$, Chowdhury et al. (2011) also presented a measure of the cost reduction attributed to product commonality, which also accounts for the production volume.

Martin and Ishii (1996) points out that "to reduce the costs incurred to provide variety, the products variants in a family should be differentiated as late as possible". Consequently, delaying the branching and the sub-branching of the *product variation graph* (Martin and Ishii 1996) is preferable. *To this end, "the sharing of multiple parts in the same group of products" is thereby more helpful than "the sharing of one part in one group of products and another part in a different group of products"*. To further elucidate this concept, an example of a scalable family of glassware is considered. As shown in Fig. 1, three glassware variants are being designed: SMALL, MEDIUM and LARGE. Each glassware is comprised of two parts, the body and the base, which are further divided into a set three design variables: (i) *body diameter* (D), (ii) *body height* (H), and (iii) *base diameter* (B). It is helpful to note that practical manufacturing process involved in commercial glass making is not particularly considered in the use of this illustrative example—e.g. if each glass requires a separate mold, then dimensional commonality does not offer any significant manufacturing-overhead savings.

As shown in Fig. 1, the following commonalities are observed in this glassware family:

i. The *body diameter* is shared between the SMALL and MEDIUM glassware, denoted by $D2$.
ii. The *body height* is shared between the SMALL and LARGE glassware, denoted by $H1$.
iii. The *base diameter* is shared between the MEDIUM and LARGE glassware, denoted by $B1$.

However, along with $D2$, if the SMALL and MEDIUM glassware also shared the *base diameter B* (as shown by the dotted grey line), more manufacturing-cost savings is likely than that offered by the sharing of $B1$ between MEDIUM and LARGE glassware. This observation explains how "*the sharing of multiple parts in the same group of products*" is thereby more helpful. To its disadvantage, the standard *commonality index* (CI) values for the actual glassware platform plan (shown in Fig. 1) and the proposed (better) platform plan would be the same. To summarize, the standard *commonality index* (CI) makes the limiting assumption that the cost reductions attributed to the sharing of different parts (among products) are independent of each other, thereby not facilitating greater part sharing among similar groups of product variants. A majority of the other existing commonality metrics also do not facilitate more commonality among similar groups (subsets) of product variants.

A group of products that share a particular part is called a *inter-product commonality group* in the remainder of the paper. Two aspects need to be considered when addressing the overlap between *inter-product commonality groups* corresponding to different parts (or components or design variables):

i. The extent of overlap between a pair of *inter-product commonality groups*—how many and which products belong to both of the groups?
ii. When are the concerned distinct parts introduced in the *product variation chain*?

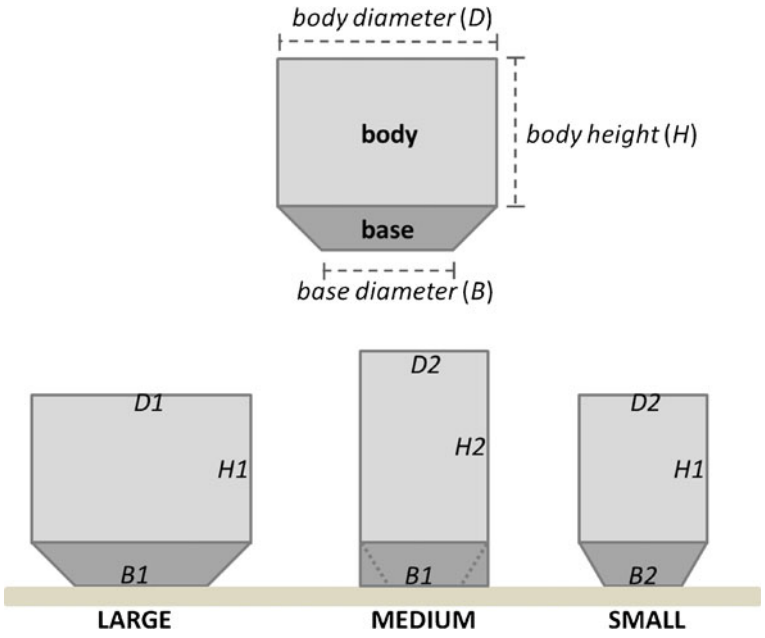


Fig. 1 A scalable product family of glassware

In this paper, we develop a new measure of commonality that seeks to address the first aspect of the overlap of *inter-product commonality groups*. This measure is an evolution from the standard CI (Martin and Ishii 1996), and therefore retains all the helpful properties of the standard CI. The second aspect of the *inter-product commonality group* overlap requires an understanding of the manufacturing process of the concerned products, which in turn requires an exploration of the *product variation chain* alternatives. The investigation of the product variation chain in this context is however not within the scope of this paper.

This paper therefore makes the following two major contributions:

I. Advances the CP^3 model to yield a more tractable MINLP problem, comprising a limited number of feasible commonality matrix options that can be prespecified during optimization; and
II. Develops a new and more comprehensive measure of inter-product commonality (in a family) that accounts for the membership-overlap among the variable-based product platforms.

Detailed description of these two contributions are presented in the following two sections (Sections 2 and 3). Section 4 discusses the solution of the advanced CP^3 model using mixed-discrete PSO, in the case of a scalable family of universal electric motors.

2 Comprehensive product platform planning (CP^3) model

In this section, we describe how the original CP^3 model, introduced by Chowdhury et al. (2011), can be further advanced to yield a more tractable product platform planning model. The section starts with a brief description of the original CP^3 model. Subsequently, we discuss how the inter-product transitivity constraints are addressed. The last part of this section presents the conversion of the high-dimensional *mixed-binary nonlinear programming problem* into a low-dimensional *mixed-integer nonlinear programming problem*.

2.1 An overview of the original CP^3 model

The CP^3 framework introduced a compact mathematical model of the product family design problem. Key features of this CP^3 model are:

i. This model presents a generalized and compact mathematical representation of the platform planning process, which is independent of any particular optimization strategy.
ii. This model avoids the "*all-common/all-distinct*" restriction, similar to that in the method by Fellini et al. (2005); thereby allowing the automatic formation of sub-families of products during the optimization process.
iii. This model facilitates simultaneous selection of platform/scaling design variables, and quantification of the optimal design variable values.

The CP^3 model defines a product platform as—"*A product platform is said to be created when more than one product variant in a family share a particular design variable*." In this case "sharing a design variable between two products" implies that the products are designed to have the same value of the concerned variable. Based on this concept, the commonality among products is concisely represented using a generalized matrix in the CP^3 model. This matrix is known as the *commonality matrix*, and is represented by λ. The *commonality matrix* for an N-product n-variable family is shown in (1). In this matrix definition, the generic parameter x_j^k denotes the j^{th} variable in product-k.

$$\lambda = \begin{bmatrix}
\lambda_1^{11} & \cdots & \lambda_1^{1N} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\lambda_1^{N1} & \cdots & \lambda_1^{NN} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \lambda_j^{11} & \cdots & \lambda_j^{1N} & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \lambda_j^{N1} & \cdots & \lambda_j^{NN} & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_n^{11} & \cdots & \lambda_n^{1N} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_n^{N1} & \cdots & \lambda_n^{NN}
\end{bmatrix}$$

$$\lambda_j^{kl} = \left\{ \begin{matrix} 1 \text{ , if } \lambda_j^{kk} = \lambda_j^{ll} = 1 \text{ and } x_j^l = x_j^k \\ 0 \text{ , otherwise} \end{matrix} \right\} \forall\, k \neq l$$

$$\lambda_j^{kk} = \begin{cases} 1 \text{ , if the } j^{\text{th}} \text{ variable is included in product} - k \\ 0 \text{ , if the } j^{\text{th}} \text{ variable is NOT included in product} - k \end{cases} \tag{1}$$

The *commonality matrix* is a symmetric block diagonal matrix, where the j^{th} block corresponds to the j^{th} design variable. The diagonal elements of the *commonality matrix* $\left(\lambda_j^{kk}\right)$ establish whether the j^{th} variable is included in product-k. In the case of scaling product families, all the diagonal elements, $\left(\lambda_j^{kk}\right)$, are fixed at one. The off-diagonal elements of the *commonality matrix*, $\left(\lambda_j^{kl}\right)$, determine whether the j^{th} design variable is shared by product-k and

product-l, i.e., $\lambda_j^{kl} = 1$, if shared, and $\lambda_j^{kl} = 0$, if not shared. These off-diagonal elements are treated as binary variables during the optimization process, and are called *commonality variables* (Chowdhury et al. 2011). Interestingly, for an actual commercial product-line development, the commonality matrix can be more meaningfully defined in terms of the manufacturing-process commonalities. In that case, $\lambda_j^{kl} = 1$ if and only if:

i. the *manufacturing process* that introduces/controls the j^{th} variable is the same in both products; and
ii. the value of the j^{th} variable is also designed to be the same in both products, or within a range that do not introduce additional tooling costs (or overhead costs).

The optimal platform planning problem should therefore be redefined to ensure that the design vector describing each product design represents the manufacturing processes involved along with the design variable values. However, further discussion regarding the role of manufacturing processes (and their order of occurrence) in introducing product differentiation is not within the scope of this paper.

The CP^3 model formulates a generic equality constraint, called the *commonality constraint*, to test the compatibility between the *product platform plan* and the *physical design of each product*. The process of testing whether a product family (comprising N products and n design variables) satisfies this *commonality constraint* is explained in Fig. 2. In this figure, the black arrows represent the process direction and the grey arrows represent the flow of information (on as-needed basis). In Fig. 2, the parameter, m, is equal to $N(N-1)/2$; the tolerance parameter, ϵ, is used to relax the equality criterion into an inequality criterion – to allow manufacturing tolerances and/or to ease the constrained-optimization process.

$$X^T \Lambda X = 0$$

where

$$\Lambda = \begin{bmatrix}
\sum_{k\neq 1}\lambda_1^{1k} & \cdots & -\lambda_1^{1N} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\lambda_1^{N1} & \cdots & \sum_{k\neq N}\lambda_1^{Nk} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \sum_{k\neq 1}\lambda_j^{1k} & \cdots & -\lambda_j^{1N} & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & -\lambda_j^{N1} & \cdots & \sum_{k\neq N}\lambda_j^{Nk} & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sum_{k\neq 1}\lambda_n^{1k} & \cdots & -\lambda_n^{1N} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda_n^{N1} & \cdots & \sum_{k\neq N}\lambda_n^{Nk}
\end{bmatrix} \tag{2}$$

$$k = 1,\ 2,\ .....,\ N; \quad j = 1,\ 2,\ .....,\ n;$$

$$X = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^N & \cdots & x_j^1 & x_j^2 & \cdots & x_j^N & \cdots & x_n^1 & x_n^2 & \cdots & x_n^N \end{bmatrix}^T$$

Mathematically, the *commonality constraint* can be expressed in a matrix format as shown in (2). The matrix Λ is called the *commonality constraint matrix*. This matrix is also a symmetric block diagonal matrix, where the j^{th} block corresponds to the j^{th} design variable. Every block of the *commonality constraint matrix* can be expressed as a function of the corresponding *commonality matrix* block, which is

$$\Lambda_j = f_{con}(\lambda_j) \tag{3}$$

Further details of the *commonality matrix* formulation and illustration of the CP^3 model for a sample product family can be found in the paper by Chowdhury et al. (2011).

The objectives of product family optimization are: (i) the maximization of the product performances and (ii) the minimization of the overall cost of the family, while ensuring that the individual products satisfy their specified requirements. The specified requirements can be generally modeled as constraints. For a family of N products, each comprising a set of n design variables, the

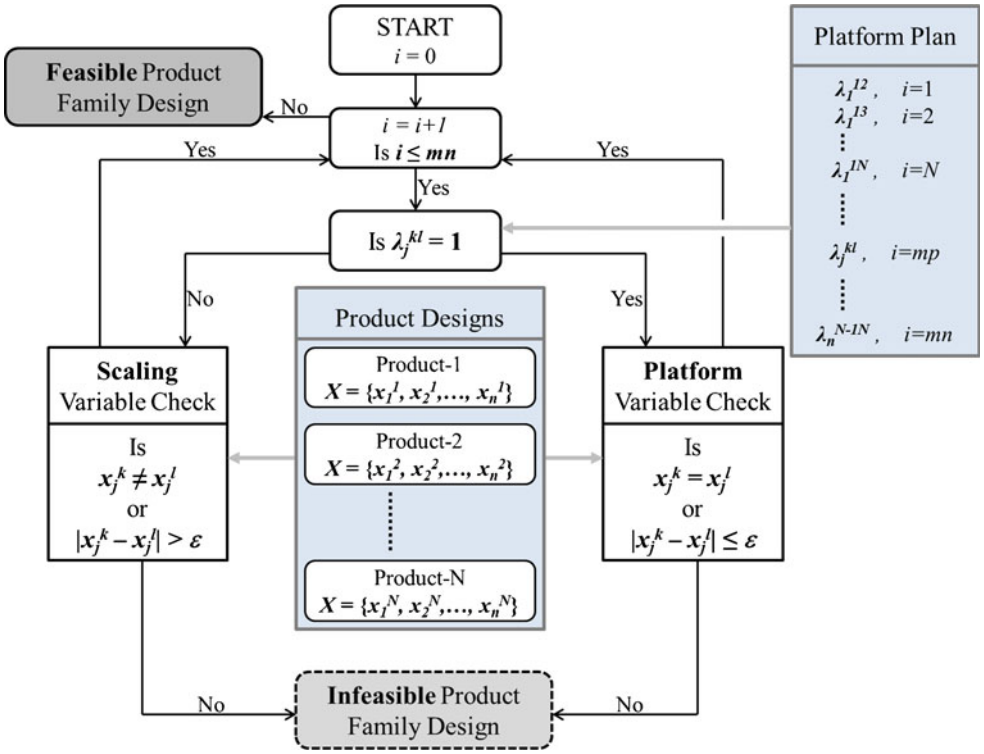


Fig. 2 The process of applying the *commonality constraint*

generalized MINLP problem presented by the CP[3] model can be expressed as

$$
\begin{aligned}
&\text{Max} \quad f_{\text{perf}}(X)\\
&\text{Min} \quad f_{cost}(X, \lambda)\\
&\text{subject to}\\
&\quad g_i(X) \le 0, \quad i = 1, 2, \ldots., p\\
&\quad h_i(X) = 0, \quad i = 1, 2, \ldots., q\\
&\quad X^T \Lambda X = 0\\
&\text{where}\\
&\quad \Lambda = f_{con}(\lambda)\\
&\quad X = \begin{bmatrix} x_1^1\ x_1^2 \cdots x_1^N \cdots x_j^1\ x_j^2 \cdots x_j^N \\ \cdots x_n^1\ x_n^2 \cdots x_n^N \end{bmatrix}^T\\
&\quad \lambda_j^{lk} \in B : B = \{0,\ 1\}\\
&\quad k, l = 1,\ 2, \ldots., N; \quad j = 1,\ 2, \ldots., n;
\end{aligned}
\tag{4}
$$

In (4), f_{perf} and f_{cost} are the objective functions that represent the performance and the cost of the product family, respectively; g_i and h_i represent the inequality and equality constraints related to the physical design of the products; and, the matrices Λ and λ are given by (2) and (1), respectively.

2.2 Ensuring feasible inter-product commonalities: transitivity constraints

Each block of the commonality matrix (λ), corresponding to one design variable, is comprised of at least $N(N-1)/2$ unknown *commonality variables* (all λ_j^{kl} with $k \neq l$). However, these λ_j^{kl}'s are not necessarily independent of each other, which leads to redundancy in the commonality matrix. More specifically, the knowledge of the commonalities between product-k and product-l and that between product-k and product-i can provide substantial (not necessarily complete) information regarding the commonality between product-l and product-i. The assumption that the commonality variables are independent (in the original CP[3] model) does not capture this flow of information. Resolving this redundancy can significantly reduce the dimensionality of the optimization problem. The interdependency of the *commonality variables*, pertinent to a particular design variable, x_j, is illustrated in the first three columns of Table 1. In this table, k, l, and i depict three product variants, and the term ID means "indeterminate".

Table 1 Interdependency of the *commonality variables*

If λ_j^{kl} is equal to	If λ_j^{ki} is equal to	Then λ_j^{li} is equal to	Then $\lambda_j^{kl} + \lambda_j^{ki} + \lambda_j^{li}$ is equal to
1	1	1	3
1	0	0	1
0	1	0	1
0	0	ID	0 or 1

We observe from Table 1 that the feasible combinations of λ_j^{kl}, λ_j^{ki}, and λ_j^{li} $(k \neq l \neq i)$ can be ensured by a new constraint defined as

$$\left(\lambda_j^{kl} + \lambda_j^{ki} + \lambda_j^{li} - 2\right)^2 > 0 \tag{5}$$

Hence, a particular *commonality matrix* would represent a feasible combination of platform-scaling design variables for a product family, only if the corresponding *commonality variables* abide by the constraint defined in (5). In order to determine this feasibility, the constraint in (5) has to be evaluated for each unique combination of products-i, j, and k, with respect to each system design variable; therefore, the number of necessary evaluations of (5) for a candidate product family design is given by

$$n \times {}^{N}C_3 = n \times N(N-1)(N-2)/6 \tag{6}$$

The application of this additional constraint during the course of optimization is likely to increase the computational expense. On the contrary, a judicious use of this constraint prior to the solution of the MINLP (given by CP^3) can decrease the overall computational expense significantly. To this end, we perform a "mixed-Binary Integer Problem to mixed-Integer Problem" conversion and further modification of the original MINLP problem presented by the CP^3 model, which is discussed in the next section.

Interestingly, for a modular product family, the predefined product architecture and the inter-module relationships (e.g. compatible, incompatible, mutually necessary) can be used to further limit the number of feasible commonality matrices for the family. In that case, each product may not be comprised of the same n design variables. Once the product-module architecture has been defined within the context of the commonality matrix, additional constraints can be formulated to identify infeasible module combinations. The simplification of the CP^3 model described in the subsequent sections will apply in that case as well, and provide an even more tractable platform planning model. The current paper is however focused on the advancement of the CP^3 model for scale-based product family applications.

2.3 Simplifying the CP^3 model

Each feasible platform plan, represented by a unique commonality matrix, is essentially a feasible combination of binary variables. A conversion of these known feasible combinations into a *set of feasible integers* that can be pre-specified for optimization is expected to alleviate the complexity of CP^3 optimization. Considering that the diagonal elements of the *commonality matrix* are known apriori, each block of this matrix presents $N(N-1)/2$ unknown binary variables. These variables are aggregated into a single binary string of length, $L = N(N-1)/2$, which is then converted into an integer variable, as given by

$$\begin{aligned} &z = s_1 \times 2^{L-1} + s_2 \times 2^{L-2} + \ldots + s_L \times 2^0 \\ &\text{where} \\ &s \in \{0, 1\} \text{ and } z \in \{0, 1, \ldots, 2^L - 1\} \end{aligned} \tag{7}$$

where s_i is the i^{th} element of the binary string. Therefore, for a n-variable scalable product family, the *commonality matrix* is replaced by a tractable set of n integer variables. Each of these *integer commonality variables* belong to the feasible set of integers: $Z = \left[0, 1, \ldots, 2^{N(N-1)/2} - 1\right]$.

The constraint in (5) allows only particular combinations of the binary *commonality variables*. Therefore, we can eliminate/neglect the integer values that correspond to infeasible combinations of the binary *commonality variables*. This elimination reduces the set of feasible values for the *integer commonality variables* (to be specified prior to optimization), which in turn results in a computationally less expensive CP^3 model. An efficient and generic pseudocode is developed to determine the appropriate reduced set (Z) of the *integer commonality variables* that correspond to feasible combinations of the binary *commonality variables*. This pseudocode can be represented as:

```
mat = unit matrix
M = N(N − 1)/2
z_max = 2^M − 1
l = 1
for z = 0 : z_max
  **converting z into commonality matrix block**
  convert z into a binary string s of length M
  k = 1
  for i = 1 : N − 1
    for j = i + 1 : N
      mat(i, j) = s(k) and mat(j, i) = s(k)
      k = k + 1
    end
  end
  **determining feasibility of the commonality matrix block**
  for i = 3 : N
    for j = 1 : i − 2
      for k = j + 1 : i − 1
        if mat(j, k) + mat(j, i) + mat(k, i) = 2
          discard z
          continue with the outermost z loop
        end
      end
    end
  end
  Z(l) = z
  l = l + 1
end
```

(8)

In the above pseudocode (8), *mat* is a *commonality matrix* block corresponding to a particular physical design variable; and z_{max} represents the domain size of the integer variable, which is also equal to the total number of possible unique *commonality matrix* blocks for a family of a given number of product variants (given N). To evaluate a candidate design during product family optimization, the *integer commonality variable* vector (for that design) should be converted to the corresponding commonality matrix. This conversion can be accomplished by implementing the part of the pseudocode (in (8)) called "*converting z into a commonality block*", which can be concisely expressed as

$$\lambda = f_{com}(Y)$$
$$Y = [z_1\ z_2\ \cdots\ z_n] \tag{9}$$

The new tractable MINLP problem, resulting from simplifying the CP^3 model, can be expressed as

$$\text{Max}\quad f_{\text{perf}}(X)$$
$$\text{Min}\quad f_{cost}(X, Y)$$

subject to

$$\begin{aligned} &g_i(X) \le 0, \qquad i = 1, 2, ..., p \\ &h_i(X) = 0, \qquad i = 1, 2, ..., q \\ &X^T \Lambda X = 0 \end{aligned} \tag{10}$$

where

$$\begin{aligned} &\Lambda = f_{con}(\lambda), \quad \text{and} \\ &\lambda = f_{com}(Y) \\ &X = \left[x_1^1\ x_1^2 \cdots x_1^N \cdots x_j^1\ x_j^2 \cdots x_j^N \cdots x_n^1\ x_n^2\ \cdots\ x_n^N\right]^T \\ &Y = [z_1\ z_2\ \cdots\ z_n] \\ &k = 1, 2, \ldots, N; \qquad j = 1, 2, \ldots, n; \end{aligned}$$

We apply the pseudocode in (8) to analyze the CP^3 model simplification for scaling product families with 3 to 7 product variants. The total range of an *integer commonality variable*, $0 - z_{max}$, increases exponentially with the number of product variants, where the exponent itself grows quadratically; this exponential variation is evident from (7). We found that, although the number of feasible values (due to the constraint in (5)) for the *integer commonality variable* also increases exponentially, the exponent grows linearly with the number of product variants. This difference in the variation of domain size of the integer variable is illustrated in Fig. 3. The Y-axis in Fig. 3 represents the logarithm of the domain size of each integer variable (i.e., the number of unique *commonality matrix* blocks). Figure 3 shows that the size of the feasible set Z for the integer variables, representing feasible commonality matrix blocks, is significantly smaller than the total range for the integer variable $(0 - z_{max})$; for example, in the case of a 7-product family, $z_{max} = 2,097,152$, whereas the size of the set Z is only 877, i.e., an *integer commonality variable* can take

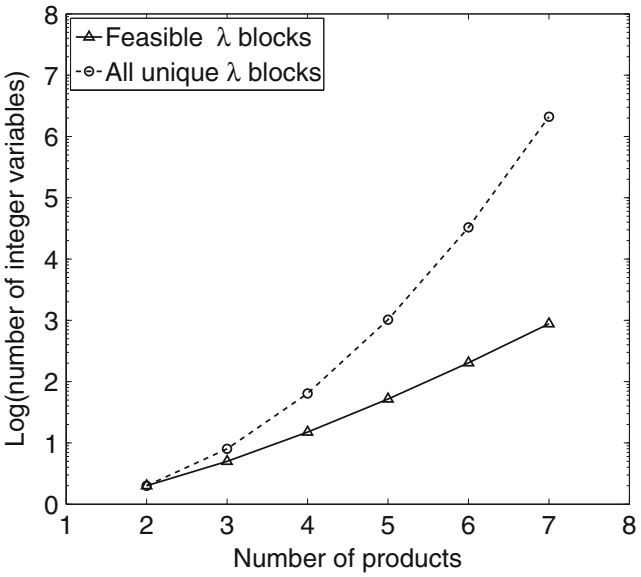


Fig. 3 Domain size of the integer variable (number of unique *commonality matrix* blocks)

877 different values for a 7-product family. In the case of commercial products, certain platforms might be known to be undesirable (or practically infeasible) even prior to optimal platform planning, where such knowledge can come from previous user-experience or market-based decisions (e.g., a car manufacturer decides to have different types of seats in their full-size and luxury sedans). In that case, the *commonality integer* values corresponding to the *practically undesirable* platforms can also be eliminated from the feasible set (Z) prior to optimization, hence further reducing the computational burden of the typically expensive product family optimization process.

We observe from Fig. 3 that M_Z varies with the number of product variants at an increasing rate. For example, in the case of a family of 7 product variants (comprising n design variables each), the number of possible combinations is 877^n; this number is substantial for products with more than a handful (4 or greater) of parts or design variables. Hence, any form of exhaustive approach to product family design is likely to be practically unrealistic. Figure 4 shows the distribution of the feasible values of the *integer commonality variables*, where the black circle symbols represent feasible integer values. We observe that the frequency of feasible integer values is noticeably higher towards the lower end of the range, evident from the higher histogram bars in the range, $0 - 500,000$. The frequency of feasible integer values also seem to follow a periodic trend.

A preliminary analysis showed that for a family of 8 product kinds, the determination of the feasible set would take 75 hours (approximately), running on a 2.83 Ghz Intel Core 2 Quad, 8Gb system (using MATLAB). The expected high computational time can be attributed to the very large

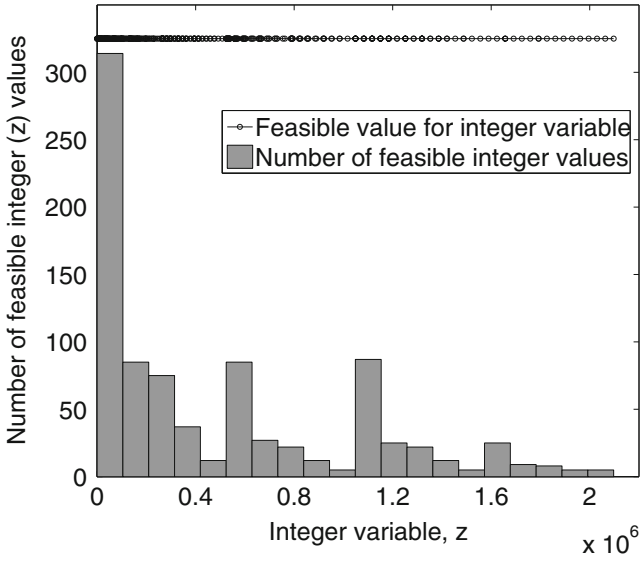


Fig. 4 Frequency of feasible values of the integer variable

domain size of the integer variable (total number of unique *commonality matrix* blocks), which is $z_{max} = 268,435,456$ in the case of the 8-product family. However, the determination of the feasible set for any given number of product variants is a one-time process—once determined, that set can be used for any scaling families of similar number of product variants. For product families with up to 8 product variants, the feasible set (Z) can be downloaded from Ref Chowdhury et al. (2012a). For families with less than 8 products, the list should be used only up to the integer that corresponds to a binary string of $n(n-1)/2$ ones (e.g., up to [1 1 1 1] or 63 for a 4-product family).

2.4 Quantifying the number of possible platform combinations

Each combination of feasible values of the integer variables (that represent the *commonality matrix* blocks) represents a candidate combination of "*sub-platform - platform - scaling*" variables for the product family, i.e., a candidate product platform plan. Interestingly, when the possibility of sub-families is accounted for, the number of such feasible candidate combinations is significantly higher than 2^n. The total number of such possible combinations is instead equal to $(M_Z)^n$, where M_Z represents the number of elements in the set Z, and n is the number of physical design variables. The popularly accepted estimation of "*the number of possible platform combinations*" is that given by Fellini et al. (2006), which is expressed as:

$$M_{PPC} = \left(1 + \sum_{i=2}^{N} \frac{N!}{i!(N-i)!}\right)^n \tag{11}$$

In the above expression, N represents the total number of product variants and n represents the number of components/variables in each product variant, assuming each variant has the same shareable physical components/variables.

However, this expression does not consider the possibility of the simultaneous existence of multiple platforms, which comprise different groups of product variants, with respect to a single shareable element/component. For example, in the case of a 4-product family, (11) does not consider the following possible platform combinations: (i) [A,A,B,B], (ii) [A,B,A,B], and (iii) [A,B,B,A]. As a result, *the number of possible platform combinations* for a single shareable component yielded by (11) is 12, whereas the actual number is 15. A comprehensive estimation of M_{PPC} is practically not straightforward.

The number of ways a platform comprising k product variants can be formed out of the total set of N product variants is equal to ${}^{N}C_k$, which is similar to each term in (11). To comprehensively estimate *the number of possible platform combinations*, it should be perceived as a *grouping/partitioning* problem, which is a class of combinatorics. Lets consider that K products variants (with $K < N$) do not belong to any group, or are unique with respect to the concerned component, and the remaining $N-K$ product variants are divided into G groups. The generic P_i (for $i <= G$) is assumed to represent the number of variants in the i^{th} group. In this case, $\sum_{i=1}^{G} P_i = N-K$, and G can take values between 1 and $(N-K)/2$ if $N-K$ is even, and between 1 and $(N-K-1)/2$ if N is odd. With this definition, there are primarily two different grouping scenarios that can occur with respect to a single shareable element:

Scenario 1 If the G groups are of unequal sizes, the number of possible combinations (M_G) of such groups is given by

$$M_K = \frac{N!}{P_1!P_2!\cdots P_G!K!} \tag{12}$$

Scenario 2 If the G groups are of equal sizes (P_1), the number of possible combinations (M_G) of such groups is given by

$$M_K = \frac{N!}{(P_1!)^G G!K!} \tag{13}$$

A combination of the above two scenarios is also possible; the expression of M_G in that case can be readily derived from (12) and (13). Therefore, *the number of*

possible platform combinations can be comprehensively expressed as

$$M_{PPC} = \left(2 + \sum_{K=1}^{N-2} \left(\sum_{\forall \sum_{i=1}^{G} P_i = N-K} M_K \right) \right)^n \tag{14}$$

where

$2 \leq P_i \leq N - K$ and $G \in [1, \lfloor N - K \rfloor]$

In (14), the number 2 precedes the summation term in order to include the two nominal platform plans: (i) all product variants are unique with respect to the concerned part/variable, and (ii) all product variants share the concerned part/variable. The number of unique ways of having $\sum_{i=1}^{G} P_i = N - K$ in this case is equivalent to *the number of possible partitions of the natural number* $(N - K)$ *into other natural numbers greater than 1*. This number could be obtained from the general theory of partition (Andrews 1998). To the best of our knowledge, such a comprehensive quantification of the "*the number of possible platform combinations*" does not exist in the literature.

3 Formulating the measure of commonality

The commonality objective in product family design should account for the degree of component/part/design-attribute sharing among the different products in the family. Several metrics for measuring the commonality in product families have been proposed, as discussed in Section 1.2. In this section, we show how one of the popular measures of commonality can be directly derived from the CP^3 commonality matrix. Subsequently we formulate a more advanced measure of commonality that can account for *the product-set intersection between different variable-based platforms*.

3.1 Standard commonality index

Among the proposed metrics that provide a measure of tooling cost savings attributed to component sharing, the *commonality index* developed by Martin and Ishii (1996) has been reported to be one of the most appropriate choice (Khajavirad and Michalek 2008). This *commonality index* is essentially based on the ratio of "the number of unique parts" to "the total number of parts" in the product family. In this context, *parts* can be either *components* or *design variables* of a product. For a family of N product variants, the *commonality index* (CI) can be mathematically defined as

$$CI = 1 - \frac{u - max(n_k)}{\sum_{k=1}^{N} n_k - max(n_k)} \tag{15}$$

where u represents the actual number of unique parts in the whole product family; n_k represents the number of parts in the k^{th} product. The "$-max(n_k)$" term is included in the definition to ensure that the CI varies between 0 and 1. According to this definition,

i. when the product variants in a family are identical (i.e., all parts are shared among all products), the value of CI is a maximum of one, and
ii. when the product variants are completely unique (i.e., no parts are shared among the product variants), the value of CI is a minimum of zero.

In the CP^3 *commonality matrix*, each $N \times N$ block represents the *product platform plan* with respect to one physical design variable (or one part). If the j^{th} part is shared between product-k and product-l, the k^{th} and the l^{th} rows in the corresponding *commonality matrix* block are identical. Owing to this matrix property, the total number of unique parts in the product family is readily given by the rank of the *commonality matrix*. The commonality index can then be defined in terms of the *commonality matrix* as

$$CI = 1 - \frac{R_\lambda - max(n_k)}{\sum_{k=1}^{N} n_k - max(n_k)} \tag{16}$$

where R_λ is the rank of the *commonality matrix* λ. In the case of a scaling product family, where each product comprises n parts, the commonality index can be simplified to

$$CI = 1 - \frac{R_\lambda - n}{n(N-1)} \tag{17}$$

3.2 Advancing the commonality index

In order to promote "*more commonality among similar product groups*" instead of "*distributed commonality among different product groups*", a new measure of commonality is developed. The new measure is called the *Cross-Commonality Index (CCI)*. From a practical standpoint, the actual cost benefit of increasing the overlap between *inter-product commonality groups* corresponding to different parts depends on the manufacturing process and the overall product architecture. The standard *commonality index* provides a generic artificial quantification of the cost reduction attributed to inter-product commonalities. Additionally, the proposed *Cross-Commonality Index* is intended to provide a generic artificial quantification of the cost reduction attributed to the overlap between *inter-product commonality groups*.

The overlap between two *inter-product commonality groups* should essentially indicate the number of products that belong to both groups, where each group is comprised of products that share a particular part. At the same time,

there can be multiple *inter-product commonality groups* with respect to a single part, unless an "*all-common or all distinct*" assumption is made. Both of these factors are coherently captured by the *commonality matrix* formulation. Therefore, with respect to two distinct parts, i and j, the degree of similarity between the corresponding two *commonality matrix* blocks, λ_i and λ_j, provides an effective representation of the overlap. The *Cross-Commonality Index* (CCI) is then formulated based on the hypothesis that, the more the similarity, the more is the cross-commonality.

The degree of similarity S_{ij} between two *commonality matrix* blocks, λ_i and λ_j, can be expressed as

$$S_{ij} = 1 - \frac{2R_{ij} - R_i - R_j}{2N - R_i - R_j} \tag{18}$$

where R_i and R_j are the ranks of the matrix blocks λ_i and λ_j, which correspond to the i^{th} and the j^{th} parts, respectively; and R_{ij} is the rank of the matrix given by the element by element multiplication (Hadamard product) of λ_i and λ_j. By virtue of its definition, $R_{ij} \leq min(R_i, R_j)$. The degree of similarity between the two *commonality matrix* blocks can be further simplified to

$$S_{ij} = 1 - \frac{R_{ij} - 1}{N - 1} \tag{19}$$

The definitions in (18) and (19) allows S_{ij} to vary between 0 and 1.

For a family of N product variants and n parts/variables, the overall degree of inter-platform overlap can be represented by the aggregate degree of similarity (S) within all possible pairs of commonality blocks, which can be approximated to

$$S = 1 - \frac{\sum\limits_{\forall i \neq j} (R_{ij} - 1)}{\sum\limits_{\forall i \neq j} (N - 1)}, \quad \text{where } i, j = 1, 2, \ldots, n \tag{20}$$

The relative importance of inter-product commonality and inter-platform overlap is strongly subjective—This relative importance depends on the manufacturer standpoint and the actual economics of the manufacturing processes involved. Hence, in this paper, we provide a weight parameter α to represent their relative importance in the new *Cross-Commonality Index* (CCI). CCI can essentially be quantified as a combination of the inter-product commonality (standard CI, (17)) and the inter-platform overlap (S, (20)). Hence, we expresses CCI as

$$CCI = 1 - \alpha \frac{R_\lambda - max(n_k)}{\sum\limits_{k=1}^{N} n_k - max(n_k)} - (1 - \alpha) \frac{\sum\limits_{\forall i \neq j} (R_{ij} - 1)}{\sum\limits_{\forall i \neq j} (N - 1)}$$

$$\text{where } i, j = 1, 2, \ldots, n \tag{21}$$

where the weight parameter α can take a value in the range 0 to 1.

For a scaling product family, where each product variant is comprised of n variables/parts, the CCI can be simplified to

$$CCI = 1 - \alpha \frac{R_\lambda - n}{n(N-1)} - (1 - \alpha) \frac{2 \sum\limits_{\forall i \neq j} (R_{ij} - 1)}{n(n-1)(N-1)} \tag{22}$$

$$\text{where } i, j = 1, 2, \ldots, n$$

In (21) and (23), the second term represents *the inter-product variation with respect to the product-parts*, and the third term represents *the variation among the platform plans for the product-parts*. Both of these variations cause the product differentiation to occur earlier at the design/conceptualization stage and the manufacturing stage, which leads to higher overhead costs.

If the platform configurations with respect to each part are similar to each other, i.e., each part is shared in a similar fashion among the product variants, then $R_{ij} = R_i = R_j = R_\lambda / n, \quad \forall\, i \neq j$. In that case, the third CCI term in (23) becomes equal to the second CCI term (without the weights), i.e.,

$$\frac{R_\lambda - n}{n(N-1)} = \frac{2 \sum\limits_{\forall i \neq j} (R_{ij} - 1)}{n(n-1)(N-1)} \tag{23}$$

Consequently, the CCI expression becomes equivalent to the standard CI expression, when there is no variation among the platform plans for the product-parts. On the other hand, if there is no similarity between the platform configurations for each product-part, then $R_{ij} = N, \quad \forall\, i \neq j$. In that case, the third CCI term in (23) becomes equal to one (without the weight), which leads to $CCI = \alpha \times CI$. Therefore , the *cross-commonality index* (CCI) for any product family is bounded by "CI" and "$\alpha \times CI$". Such mathematical properties are desirable in the context of commonality quantification.

A careful mathematical investigation of the *inter-product variation* and the *inter-platform variation* terms (proposed in this paper) indicated that the latter is always less than or equal to the former. For a family of 5 products comprising 2 physical parts each, this observation is visually illustrated in Fig. 5; this figure shows all the possible values of the *inter-product variation* (X-axis) and the *inter-platform variation* (Y axis). For a scale-based family of 5 products, there are 52 different platform plans possible (i.e., $M_Z = 52$)) with respect to each product-part; 52 different values of the *integer commonality variable* (z) are thus allowed for each part. Therefore, for 2 physical parts, a total of 52×52 unique platform combinations are possible (with many coinciding values) as displayed in Fig. 5. The dashed 45 degree line in

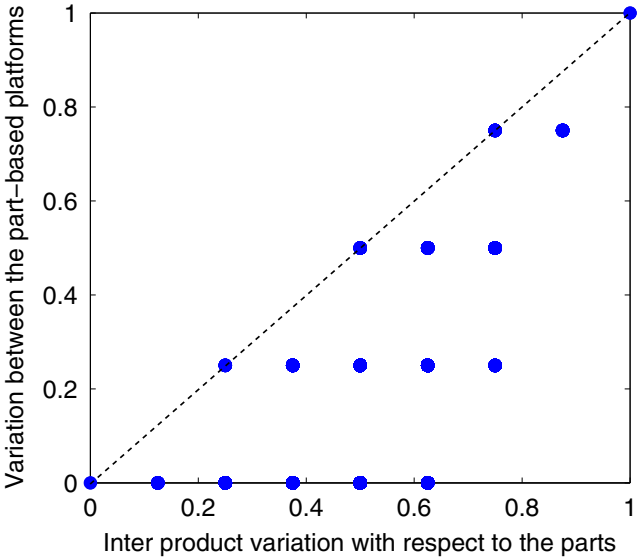


Fig. 5 The inter-product variation and the inter-platform variation for all possible platform combinations in a 5-product 2-part family

Fig. 5 denotes equality among the X and Y axis values in this figure, to help interpretation of the index values.

To illustrate the difference between the new CCI and the conventional CI, we consider a simple example of "*a family of 3 product variants, each comprising 2 physical parts*". Five unique platform configurations are possible with respect to each part. The values of CI and CCI are estimated for all the possible 5×5 product platform combinations, and are respectively displayed in Fig. 6a and b. The value of α is set to 0.5 in the CCI estimation. In these figures, each *integer commonality variable* combination, (z_1, z_2), represents a unique product platform plan. The shades of the circles, corresponding to each platform combination, depict the values of CI and CCI in the respective figures.

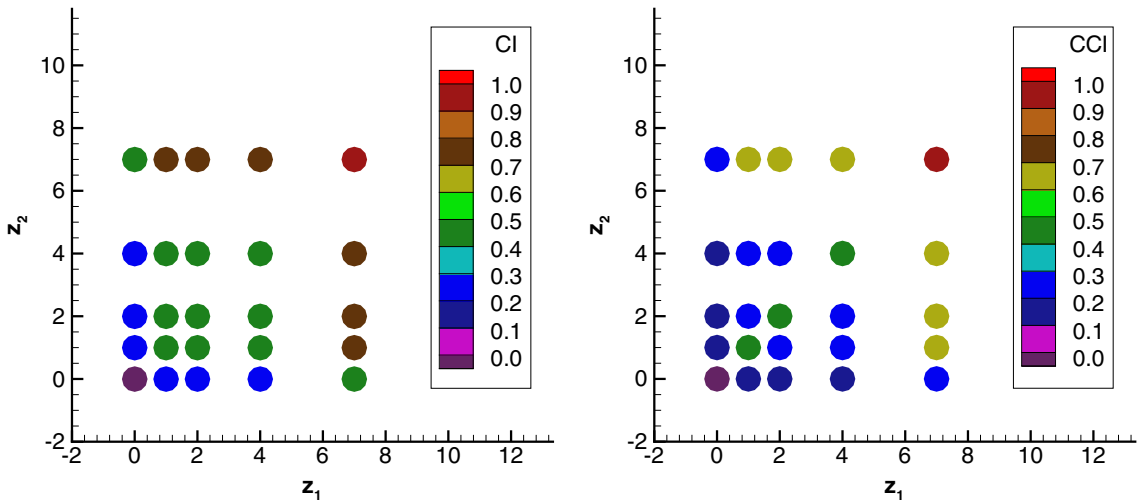
It is observed from Fig. 6a and b that the values of CCI and CI are the same for the diagonal circles. These diagonal circles correspond to platform plans where the platform configuration is the same with respect to both parts. However, for the off-diagonal circles, the CCI values are expectedly observed to be less than the CI values. This difference can be attributed to the lower cross-commonality among the platforms. For example, let us consider the platform combination defined by: $z_1 = 4$ and $z_2 = 2$. The commonality matrix-blocks for the two parts for this platform combination are given by:

$$\lambda_1 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \lambda_2 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{24}$$

The above matrices show that part 1 is shared by products 1 and 2, whereas part 2 is shared by products 1 and 3. Hence, in this case, the platform plans with respect to parts 1 and 2 are different. As a result, the CCI value is lower than the CI value for this platform combination. In this case, $CI = 0.5$, and $CCI = 0.25$. This example shows that the new commonality measure (CCI) can effectively capture the impact of having cross-commonality among platforms or its lack thereof.

4 CP^3 optimization

To explore and validate the helpful attributes of the advanced CP^3 model proposed in this paper, the model is applied to design scale-based family of universal electric motors. Both standard gradient-based MINLP solvers



(a) *Commonality index (CI)* (b) *Cross-commonality index (CCI)*

Fig. 6 The *integer commonality variable* combinations (platform plans) for the "3-product 2-part" family

and heuristic algorithms suitable for mixed-discrete optimization can be leveraged to solve the CP^3 optimization problem. The former class of algorithms however might encounter suboptimal stagnation issues owing to the characteristic multimodality of the commonality constraint. In this paper, the recently developed Mixed-Discrete Particle Swarm Optimization (MDPSO) algorithm is implemented to obtain the optimized platform plans for the motor family. A summary of the key features of the MDPSO algorithm are provided in this section, followed by a brief description of the test problem (motor family). The later part of this section presents and discusses the optimized product platform plans obtained for the motor family through CP^3.

4.1 Mixed-discrete particle swarm optimization (MDPSO)

PSO is one of the most well known stochastic optimization algorithms, originally developed by Kennedy and Eberhart (1995). Several improved variations of the algorithm have later appeared in the literature and have been used in popular commercial optimization packages. In this paper, we use a Mixed-Discrete PSO (MDPSO) algorithm developed by Chowdhury et al. (2012b, 2013). This algorithm has been successfully tested on a wide range of *mixed-discrete constrained optimization* test problems (Chowdhury et al. 2013). Prominent advantages of the MDPSO algorithm over a conventional PSO algorithm include:

i. an ability to deal with both discrete and continuous design variables, and
ii. an explicit diversity preservation capability to minimize the possibility of premature stagnation of particles.

These features are uniquely helpful in addressing the product platform planning problem, which is likely to be multimodal and involve a combination of integer and continuous variables.

The basic steps of the advanced algorithm are summarized as

$$\begin{aligned} x_i^{t+1} &= x_i^t + v_i^{t+1} \\ v_i^{t+1} &= \alpha v_i^t + \beta_l r_1 \left(p_i - x_i^t\right) + \beta_g r_2 \left(p_g - x_i^t\right) \\ &\quad + \gamma r_3 \left(x_i^t - p_g\right) \end{aligned} \tag{25}$$

where, x_i^t is the location of the i^{th} particle in the population (swarm) at the t^{th} iteration; r_1, r_2, and r_3 are random numbers between 0 and 1; p_i is the best candidate solution found for the i^{th} particle; p_g is the best candidate solution for the entire population; α, β_l and β_g are user defined constants that are respectively associated with exploration, exploitation, and diversity preservation in the particle population; γ is the diversity preservation coefficient that is evaluated adaptively as a function of the prevailing diversity in the population at the concerned iteration. This diversity preservation coefficient is scaled using a user-defined parameter (constant) γ_0. The last term in (25) decelerates the motion of particles towards the (then current) global best, thereby maintaining diversity and mitigating premature convergence. Such an explicit diversity preservation operator is often necessary for solving complex optimization problems that involve multimodal criterion functions and a large number of constraints, as is the case with the product family design application presented in this paper. Further discussion of the estimation of the population diversity and the formulation of the γ parameter can be found in the papers on MDPSO (Chowdhury et al. 2012b, 2013).

In this algorithm, particles are compared using the principles of constrained non-dominance, introduced by Deb et al. (2002). In this technique, solution-i is said to dominate solution-j if:

- solution-i is feasible and solution-j is infeasible or,
- both solutions are infeasible and solution-i has a net smaller constraint violation than solution-j or,
- both solutions are feasible and solution-i weakly dominates solution-j.

If none of the above conditions apply, then both of the solutions are considered non-dominated with respect to each other.

In order to address discrete design variables, we implement the vertex approximation technique (Chowdhury et al. 2010). In this technique, continuous optimization is used as the primary search strategy; subsequently, each candidate solution is approximated to a nearby feasible *discrete domain location*.

4.2 Test problem description: universal electric motor

Universal electric motors are capable of delivering more torque than other single phase motors, and can operate using both direct current (DC) and alternating current (AC) (Chapman 2002). As a result of such high performance characteristics, universal motors have been frequently used in a variety of applications, e.g., electric drills and saws, blenders, vacuum cleaners, and sewing machines (Khire 2006). Extensive analysis and detailed equations related to the design of the universal electric motor can be found in the paper by Simpson et al. (2001).

In this paper, we apply the CP^3 framework to develop scale-based product families of two, four, and six universal electric motors, where each motor is required to satisfy unique torque specifications (T_{rq}), as given in Table 2. The overall objective is to design a family of motors that have

Table 2 Torque requirements of the electric motors

Motor number	1	2	3	4	5	6
Torque (N/m)	0.05	0.1	0.125	0.15	0.2	0.25

high efficiencies and low masses and also have *a high degree of commonality* among them. Each motor is subjected to additional design constraints, regarding (i) the output power (P_{out}), (ii) the total mass (M_{total}), (iii) the efficiency (η), (iv) the magnetization intensity (H), (v) the ratio of the outer radius (r_o) to the thickness (t) of the stator, and (vi) the current (I). Each motor comprises seven physical design variables; the corresponding variable limits are given in Table 3.

The design optimization of the family of universal electric motors in this paper involves the simultaneous (i) maximization of the *aggregate performance* of the motor variants (in the family), and (ii) maximization of the inter-product commonalities. In this context, the following two case studies are performed:

Case 1 Design families of 2, 4, and 6 motors by simultaneously maximizing the aggregate performance of the family and the **CI**.

Case 2 Design families of 4 and 6 motors by simultaneously maximizing the aggregate performance of the family and the **CCI**.

It is helpful to note that in the case of 2 products, maximizing CI is the same as maximizing CCI. A value of 0.5 is used for the user-defined weight factor α in CCI.

The aggregate performance for the electric motor family, f_{perf}, is represented as a combination of the efficiencies and the masses of the motors in the family (Khajavirad et al. 2009), and is expressed as

$$f_{\text{perf}} = \frac{1}{N}\sum_{k=1}^{10}\left(0.5\eta^k + 0.5(1-\mu^k)\right) \tag{26}$$

where η^k and μ^k respectively represent the efficiency and the mass of motor-k, and N is the number of motor variants in the family.

To obtain a single optimized product platform plan for each case, we construct aggregate objective functions (to be **maximized**). The aggregate objective functions for the motor family, f_{MF}, for the two cases are expressed as

$$\begin{aligned}&\text{Case 1}: \quad f_{MF} = 0.5 \times f_{\text{perf}}^2 + 0.5 \times CI^2\\ &\text{Case 2}: \quad f_{MF} = 0.5 \times f_{\text{perf}}^2 + 0.5 \times CCI^2\end{aligned} \tag{27}$$

The overall optimization problem can then be defined as

$$\begin{aligned}&\text{Max} \quad f_{MF}(X, Y)\\ &\text{subject to}\\ &\left.\begin{aligned}&T^k = T^k_{rq}\\ &P_{out} = 300\ \text{W}\\ &M^k_{total} \le 2\ \text{kg}\\ &H^k \le 5000\ \text{A.turns/m}\\ &\eta^k \ge 0.15\\ &\frac{r^k_0}{t^k} \ge 1\\ &0.1\ \text{A} \le I \le 6.0\ \text{A}\end{aligned}\right\}\ \text{Physical Design Constraints}\\ &\left.\begin{aligned}&X^T \Lambda X = 0\\ &\text{where}\\ &\Lambda = f_{con}(\lambda), \quad \text{and}\\ &\lambda = f_{com}(Y)\end{aligned}\right\}\ \text{Commonality Constraint}\end{aligned} \tag{28}$$

and where

$$\begin{aligned}&X = \left[N_C\ N_S\ A_{wa}\ A_{wf}\ r_0\ t\ L\right]^T\\ &Y = [z_1\ z_2\ \cdots\ z_7]\\ &k = 1, 2, \ldots, N\end{aligned}$$

4.3 Results and discussion

A total number of 1,000,000, 1,500,000, and 2,000,000 function evaluations are allowed to design the families of two, four, and six electric motors, respectively, for both case studies. The values of the prescribed PSO parameters used for the case studies are provided in Table 4.

Feasible solutions were successfully obtained for each case. The *optimized platform plans* obtained for the three

Table 3 Design variable limits for the electric motor

Design variable	Lower limit	Upper limit
Number of turns on the armature (N_a)	100	1500
Number of turns on each field pole (N_f)	1	500
Cross-sectional area of the armature wire (A_{wa})	0.01 mm^2	1.00 mm^2
Cross-sectional area of the field pole wire (A_{wf})	0.01 mm^2	1.00 mm^2
Radius of the motor (r_o)	10.00 mm	100.00 mm
Thickness of the stator (t)	0.50 mm	10.00 mm
Stack length of the motor (L)	1.00 mm	100.00 mm

Table 4 User-defined constants in PSO

Constant	Value
α	0.5
β_g	1.4
β_l	1.4
γ_0	10.0
Population size	$50 \times n(N+1)$

motor families in Case 1 are illustrated in Tables 5, 6 and 8. The *optimized platform plans* obtained for the two motor families in Case 2 are illustrated in Tables 7 and 9. Each uppercase alphabet in these tables represents a platform. In each of these tables, blocks displaying similar alphabets imply that the corresponding products are members of a particular platform, i.e., a set of products sharing a particular design variable. A block, displaying the "-" symbol, represents a non-platform (scaling) design variable value, thereby implying that the corresponding design variable value is not shared by more than one product.

It is observed (from the platform plan tables) that, both in the case 4 motors and 6 motors, maximizing the CCI produced a product platform plan that is significantly different from that produced by maximizing CI. Expectedly, it is also observed that the tendency to share variables among similar sets of products is more pronounced in Case 2 than in Case 1. For example, the following is observed in the case of the 4-motor families:

i. For the 4-motor family in Case 1, motors 1 and 3 share the *number of turns on the armature* (N_a), the *radius of the motor* (r_o) and the *stack length of the motor* (L).
ii. For the 4-motor family in Case 1, motors 2 and 4 share the *radius of the motor* (r_o) and the *stack length of the motor* (L).
iii. For the 4-motor family in Case 2, motors 1 and 3 share the *number of turns on the armature* (N_a), the *cross-sectional area of the armature wire* (A_{wa}) and the *stack length of the motor* (L).

Table 5 Optimized product-platform-plan for the family of 2 motors in Case 1 (maximizing CI)

Variable	Motor-1	Motor-2
N_a	–	–
N_f	–	–
A_{wa}	–	–
A_{wf}	–	–
r_o	–	–
t	A	A
L	–	–

Table 6 Optimized product-platform-plan for the family of 4 motors in Case 1 (maximizing CI)

Variable	Motor-1	Motor-2	Motor-3	Motor-4
N_a	A	–	A	A
N_f	–	–	–	–
A_{wa}	–	–	–	–
A_{wf}	–	–	–	–
r_o	B	C	B	C
t	D	D	–	–
L	E	F	E	F

iv. For the 4-motor family in Case 2, motors 2 and 3 share the *number of turns on the armature* (N_a), the *number of turns on each field pole* (N_f), the *cross-sectional area of the field pole wire* (A_{wf}), and the *radius of the motor* r_o).

Although maximizing the CCI allowed more commonality among similar sets of products, interrelation among variables with respect to the process of manufacturing is not accounted for by the current model. For example, it may be more helpful from the manufacturing perspective, if motors 2 and 3 shared the *cross-sectional area of the armature wire* (A_{wa}) instead of the *radius of the motor* (r_o) (observation-iv). As mentioned in Section 1.2, characterizing and incorporating the product variation chain options (from the manufacturing perspective) into the product family model should address this issue. Exploring and accounting for product variation chain options is therefore an important topic for future research in product family design.

Further insights into the differing results obtained, when maximizing CCI instead of CI, are provided by Figs. 7 and 8. These figures illustrate the performance and commonality metrics yielded by the optimized product platform plans in Cases 1 and 2; in this case, the performance of the family is a combination of the weights

Table 7 Optimized product-platform-plan for the family of 4 motors in Case 2 (maximizing CCI)

Variable	Motor-1	Motor-2	Motor-3	Motor-4
N_a	A	A	A	–
N_f	–	B	B	–
A_{wa}	C	D	C	D
A_{wf}	–	E	E	–
r_o	–	F	F	–
t	–	–	–	–
L	G	–	G	G

Table 8 Optimized product-platform-plan for the family of 6 motors in Case 1 (maximizing CI)

Variable	Motor-1	Motor-2	Motor-3	Motor-4	Motor-5	Motor-6
N_a	A	A	A	A	–	–
N_f	B	B	C	B	B	C
A_{wa}	D	E	E	–	D	–
A_{wf}	–	–	–	–	–	–
r_o	–	–	–	–	–	–
t	F	F	–	–	–	F
L	G	G	G	–	–	–

and the efficiencies of the motor variants. The values of the performance and the commonality objectives for the optimized platform plans are also summarized in Table 10.

It is observed from Table 10 and Figs. 7 and 8 that maximizing CCI allowed more commonality—the commonality metric values for the 4-motor family and the 6-motor family in Case 2 are higher than those obtained when CI was maximized (Case 1). Interestingly, the higher commonality (obtained in Case 2) did not demand a greater compromise in performance as illustrated by the similarity in the performance variation plots in Figs. 7 and 8. This observation shows that the optimized platform plans obtained by MDPSO in the case of 4-motor and 6-motor families may not be the global optimum. The employment of local refinement techniques (as a post process to MDPSO application) that use gradient-based optimization methodologies may be helpful to accomplish further convergence. Alternatively, future work can also explore how to advance conventional MINLP solvers to solve the complex CP^3 optimization problem.

To allow a more insightful appreciation of the performance (and limitations) of the CP^3 method, we compare the results with that obtained by applying a genetic algorithm-based single-stage a posteriori method (Simpson 2006). It is however important to note that a direct comparison of CP^3 with existing/reported platform planning methods (in the context of electric motors) is challenging, since these methods generally address different objectives, make different assumptions, and consider different numbers of product variants. We choose the GA-based method, since it solves relatively similar objectives and also adheres to similar assumptions (e.g., platform can be formed by subsets of product variants). However, the GA-based method solves the problem for 10 motor variants, while in this paper we consider 2, 4, and 6 motor variants. Hence, a comparison of the CP^3 results with that of the first two, first four, and first six motors given by the GA-based single-stage a posteriori method is performed; Table 11 illustrates this comparison for the case when CCI is maximized in CP^3.

Table 11 shows that, compared to CP^3, the GA-based method accomplishes a greater degree of commonality (higher CI), while producing motors that are lighter but less efficient. The difference in the average mass of the optimized motor family obtained by CP^3 and that by the GA-based method increases as a greater number of motor variants is considered. On the other hand, the difference in commonality between the two methods decreases as a

Table 9 Optimized product-platform-plan for the family of 6 motors in Case 2 (maximizing CCI)

Variable	Motor-1	Motor-2	Motor-3	Motor-4	Motor-5	Motor-6
N_a	A	A	–	A	–	–
N_f	B	B	B	–	–	–
A_{wa}	C	D	D	C	–	D
A_{wf}	E	–	–	–	E	–
r_o	F	–	F	–	F	–
t	G	G	G	H	G	H
L	I	I	J	K	J	K

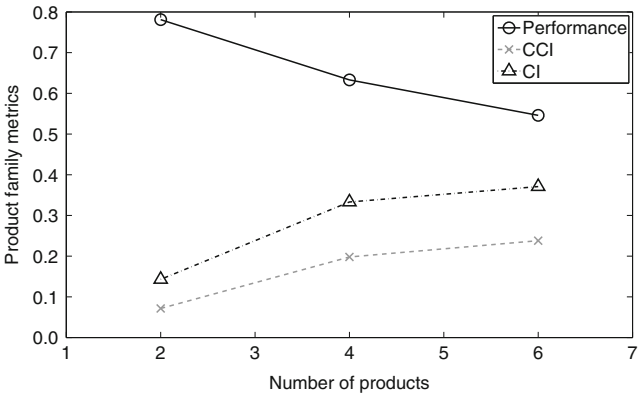


Fig. 7 Variation of performance and commonality indices with the number of product variants in the family (when maximizing CI)

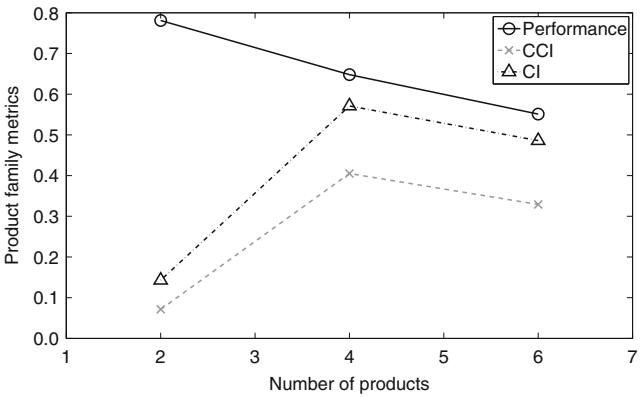


Fig. 8 Variation of performance and commonality indices with the number of product variants in the family (when maximizing CCI)

Table 10 Values of the performance and commonality objectives for the optimized motor platform plans

Number of motors	Performance (f_{perf})		CI		CCI	
	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2
2	0.78	0.78	0.14	0.14	0.07	0.07
4	0.63	0.65	0.33	0.57	0.20	0.41
6	0.55	0.55	0.37	0.49	0.24	0.33

greater number of motor variants is considered. The current optimization method in CP^3 is most likely causing the optimal solutions to be more biased towards greater motor efficiencies. The degree of commonality accomplished while maintaining attractive efficiency levels, especially for higher number of motor variants, is promising. However, it is challenging to distinctly attribute the results to the features of the CP^3 model and that of the optimization method employed. To its advantage the CP^3 model can be solved by any powerful MINLP solver that could handle multimodal functions.

5 Conclusion

This paper presents an advancement of the Comprehensive Product Platform Planning (CP^3) framework and an insightful investigation of the commonality representation provided by this framework. The original CP^3 framework presented an encompassing mathematical model of the platform planning process (CP^3 model), which yields a complex *mixed-binary nonlinear programming (MBNLP)* problem. The primary contribution of this platform planning model was the *commonality matrix* (composed of binary elements) that is to be treated as a variable in the optimization of the product family. For a "N-product n-variable" family, the total number of possible unique *commonality matrices* is $\left(2^{N(N-1)}\right)^n$, which is computationally expensive for optimization. In this paper, we developed a methodology to reduce the size of the *commonality matrix* search space by several orders of magnitude, without making any limiting approximations—for example, the search space size is reduced from 2097152^n to 877^n in the case of a 7-product family. This helpful simplification of the CP^3 model is accomplished through (i) appropriate aggregation of the *commonality variables* into binary strings, (ii) conversion of the binary strings into integers, and (iii) elimination of infeasible candidate *commonality matrices* prior to optimization (based on the transitivity constraint). To optimize the product platform plan yielded by the advanced CP^3 model, a mixed-discrete PSO algorithm is employed. The new CP^3 framework is applied to design scalable families of two, four and six electric motors. Compared to a GA-based single-stage product family design method, CP^3 provided significantly greater motor efficiencies, at the cost of greater motor mass and lower commonality. However, the degree of commonality became favorably comparable, while CP^3 retained higher motor efficiencies, when greater numbers of motor variants were considered.

It is important to note that the number of candidate platform configurations increases exponentially with the number of product variants in the family. The direct solution of the platform planning problem using CP^3 (with MDPSO) can thus be very effective for applications with smaller product lines (e.g., Apple laptops), and become increasingly computationally-expensive for applications with large product lines (e.g., Dell or HP Laptops). At the same time, products with greater design-variable dimensionality and higher number of system constraints (e.g., automobile or aircraft) can pose appreciable computational or convergence challenges to direct MINLP solution using the CP^3 framework. For the platform planning of such complex systems, it might be beneficial to adopt decomposition-based solution methods such as analytical target cascading along with the CP^3 model.

A new measure of inter-product commonalities called Cross Commonality Index (CCI) is also developed, which is an evolution from the state-of-the-art in commonality representation. Unlike its predecessors, CCI helpfully considers the overlap between product platforms in addition to the number of products in each variable-based platform. Maximization of CCI pushes product differentiation steps further downstream along the product variation chain (during the manufacturing process); hence, it provides a more meaningful means of introducing cost-saving commonalities into the product family. Case studies are performed to separately maximize the conventional CI and the new CCI for the same family of motors. Expectedly, maximizing CCI allowed more commonality among similar sets of

Table 11 Comparing the performance and commonality in the optimal platform plans obtained by CP^3 and a GA-based single-stage method

Number of motors	Avg. efficiency (CP^3)	Avg. efficiency (GA)	Avg. mass (kg) (CP^3)	Avg. mass (kg) (GA)	CI (CP^3)	CI (GA)
2	82.7 %	74.5 %	0.53	0.43	0.14	0.43
4	89.6 %	71.8 %	1.20	0.50	0.57	0.62
6	85.9 %	68.8 %	1.51	0.57	0.49	0.54

motors. The results show that it is also important to consider the relationship between *the design variables* and *the actual product differentiation processes required to be performed during manufacturing*. Modeling and incorporating this relationship-information is therefore a key direction for future research.

Acknowledgments Support from the National Science Foundation Awards CMMI-1100948, and CMMI-0946765 is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the NSF.

References

Andrews GE (1998) The theory of partitions. Cambridge University Press, Cambridge

Chapman SJ (2002) Electric machinery and power system fundamentals. McGraw-Hill, New York

Chen C, Wang LA (2008) Modified genetic algorithm for product family optimization with platform specified by information theoretical approach. J Shanghai Jiaotong Univ (Science) 13(3):304–311

Chowdhury S, Messac A, Khire R (2010) Developing a non-gradient based mixed-discrete optimization approach for comprehensive product platform planning (cp3). In: 13th AIAA/ISSMO multidisciplinary analysis optimization conference. AIAA, Fort Worth

Chowdhury S, Messac A, Khire R (2011) Comprehensive product platform planning (cp^3) framework. ASME J Mech Des (Special Issue on Designing Complex Engineered Systems) 133(10):101004

Chowdhury S, Messac A, Khire R (2012a) Set of feasible integers (z) defining product platform plans (for scaling families). https://messac.expressions.syr.edu/wp-content/uploads/2012/10/Feasible-Integers-8-Product-Family.xlsx

Chowdhury S, Zhang J, Messac A (2012b) Avoiding premature convergence in a mixed-discrete particle swarm optimization (mdpso) algorithm. In: 53rd AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference, AIAA. Honolulu, AIAA, pp 2012–1678

Chowdhury S, Tong W, Messac A, Zhang J (2013) A mixed-discrete particle swarm optimization with explicit diversity-preservation. Struct Multidiscip Optim 47(3):367–388

Collier DA (1981) The measurement and operating benefits of component part commonality. Decis Sci 12:85–96

Dahmus JB, Gonzalez-Zugasti JP, Otto KN (2001) Modular product architecture. Des Stud 22:409–424

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multi-objective genetic algorithm: Nsga-ii. IEEE Trans Evol Comput 6(2):182–197

Fellini R, Kokkolaras M, Papalambros P, Perez-Duarte A (2005) Platform selection under performance bounds in optimal design of product families. ASME J Mech Des 127:524–535

Fellini R, Kokkolaros M, Papalambros PY (2006) Product platform and product family design: methods and applications. Springer, chap 9 Commonality Decisions in Product Family Design, pp 157–185

Fujita K, Yoshida H (2004) Product variety optimization simultaneously designing module combination and module attributes. Concurr Eng 12(2):105–118

Guo F, Gershenson JK (2003) Comparison of modular measurement methods based on consistency analysis and sensitivity analysis. In: ASME 2003 international design engineering technical conferences (IDETC), ASME. Chicago, ETC2003/DTM-48634

Jiao J, Tseng MM (2000) Understanding product family for mass customization by developing commonality indices. J Eng Des 11:225–243

Jose A, Tollenaere M (2005) Modular and platform methods for product family design: literature analysis. J Intell Manuf 16:371–390

Kalligeros K, de Weck O, de Neufville R (2006) Platform identification using design structure matrices. In: Sixteenth annual international symposium of the international council on systems engineering (INCOSE). INCOSE, Orlando

Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: IEEE international conference on neural networks. IEEE, Piscataway, IV, pp 1942–1948

Khajavirad A, Michalek JJ (2008) A decomposed gradient-based approach for generalized platform selection and variant design in product family optimization. ASME J Mech Des 130: 071101:1–8

Khajavirad A, Michalek JJ, Simpson TW (2009) An efficient decomposed multiobjective genetic algorithm for solving the joint product platform selection and product family design problem with generalized commonality. Struct Multidiscip Optim 39(2):187–201

Khire RA (2006) Selection-integrated optimization (sio) methodology for adaptive systems and product family optimization. PhD thesis, Rensselaer Polytechnic Institute, Troy

Khire R, Messac A (2008) Selection-integrated optimization (sio) methodology for optimal design of adaptive systems. ASME J Mech Des 130(10):101401:1–13

Kota S, Sethuraman K, Miller R (2000) A metric for evaluating design commonality in product families. ASME J Mech Des 122:403–410

Martin M, Ishii K (1996) Design for variety: a methodology for understanding the costs of product proliferation. In: ASME design engineering technical conferences and computers in engineering conference. ASME, Irvine, 96-DETC/DTM-1610

Messac A (1996) Physical programming: effective optimization for computational design. AIAA J 34(1):149–158

Messac A, Martinez MP, Simpson TW (2002a) Effective product family design using physical programming. Eng Optim 124(3):245–261

Messac A, Martinez MP, Simpson TW (2002b) Introduction of a product family penalty function using physical programming. ASME J Mech Des 124(2):164–172

Sabbagh K (1996) Twenty-first century jet: the making and marketing of Boeing 777. Scribner, USA

Sanderson SW, Uzumeri M (1997) Managing product families. IRWIN, USA

Sharon A, Dori D, de Weck O (2009) Model-based design structure matrix: deriving a dsm from an object-process model. In: Second international symposium on engineering systems. CESUN and MIT ESD, Cambridge

Siddique Z, Rosen DW, Wang N (1998) On the applicability of product variety design concepts to automotive platform commonality. In: ASME design engineering technical conferences design theory and methodology. ASME, Atlanta, DETC98/DTM-5661

Simpson TW (2004) Product platform design and customization: status and promise. Artif Intell Eng Des Anal Manuf 18(1):3–20

Simpson TW (2006) Product platform and product family design: methods and applications. Springer, New York, chap methods for optimizing product platforms and product families: overview and classification, pp 133–156

Simpson TW, Chen W, Allen JK, Mistree F (1996) Conceptual design of a family of products through the use of the robust concept exploration method. In: 6th AIAA/USAF/NASA/ISSMO symposium on multidisciplinary analysis and optimization, AIAA. Bellevue, AIAA-96-4161- CP, pp 1535–1545

Simpson TW, Maier JRA, Mistree F (2001) Product platform design: method and application. Res Eng Des 13(1):2–22

Simpson TW, Siddique Z, Jiao RJ (2006) Product platform and product family design: methods and applications. Springer, New York

Stone RB, Wood KL, Crawford RH (2000) A heuristic method to identify modules from a functional description of a product. Des Stud 21(1):5–31

Thevenot HJ, Simpson TW (2006) Commonality indices for product family design: a detailed comparison. J Eng Des 17(2):99–119

Uzumeri M, Sanderson SW (1995) A framework for model and product family competition. Res Policy 24(4):583–607

Wacker JG, Trelevan M (1986) Component part standardization: an analysis of commonality sources and indices. J Oper Manag 6:219–224

Yu TL, Yassine AA, Goldberg DE (2007) An information theoretic method for developing modular architectures using genetic algorithms. Res Eng Des 18:91–109