Int J Adv Manuf Technol (2018) 94:3275–3285
DOI 10.1007/s00170-016-9797-4

ORIGINAL ARTICLE

A new module partition method based on the criterion and noise functions of robust design

Wei Wei[1] · He Liang[1] · Thorsten Wuest[2] · Ang Liu[3]

Received: 16 August 2016 / Accepted: 21 November 2016 / Published online: 5 December 2016

Abstract Against the sweeping trend of mass customization, the importance of product platform design is becoming increasingly recognized by the manufacturers. Module design is the foundation of product platform design, and module partition determines the effectiveness of module design. Traditionally, the vast majority of existing module partition methods ignored the design factor of customer preferences. This study proposes to employ the basic principles of robust design to make the module partition schemes less sensitive to the dynamically changing customer preferences by considering them as a noise factor. A criterion function and a noise function are each established based on the component-component correlation matrix and component-function contribution matrix, respectively. The criterion and noise functions, when combined, lead to a unique multi-objective optimization problem. Furthermore, an improved Pareto archive particle swarm optimization (PAPSO) algorithm is introduced to solve the multi-objective optimization problem in order to prevent the premature selections of non-optimal solutions. A case study is presented to showcase how the proposed new method is followed to conduct the module partition on an electric-traction drum shearer. The improved algorithm demonstrates highly competitive performance in comparison to the existing multi-objective optimization algorithms.

Keywords Module partition · Multi-objective optimization · Robust design · Criterion and noise function

✉ Ang Liu
ang.liu@unsw.edu.au

1 Advanced Manufacturing Technology and Systems Research Center, School of Mechanical Engineering and Automation, Beihang University, Beijing 100191, China

2 Industrial and Management Systems Engineering Department, Benjamin M. Statler College of Engineering and Mineral Resources, West Virginia University, Morgantown, WV 26506, USA

3 School of Mechanical and Manufacturing Engineering, University of New South Wales, Sydney, NSW 2052, Australia

1 Introduction

The manufacturing paradigm is rapidly transforming from mass production to mass customization. Against such a background, customers' growing demands for a greater variety of products have motivated the manufacturers to increasingly adopt the product platform strategy [1]. Modular design is the foundation of product platform design, which is widely considered to be a practically viable strategy to achieve mass customization in an economical manner [2]. Module design consists of module partition and module combination. The former and the latter are each related to a product's functional structure and physical architecture, respectively. It has been indicated by many previous studies that the design decisions regarding module partition impose significant impacts on a product's functionality and cost [3].

To date, the topic of module partition has been extensively investigated by many researchers. For example, Umeda proposed a method to first determine the modular structure by aggregating various attributes that are related to a product's lifecycle and then evaluate the geometric feasibility of the modules. Simpson et al. discussed the applicability of the multi-objective optimization approach in modular design, based on which they introduced a genetic algorithm-based approach [4]. Moon et al. prescribed a multi-objective particle swarm optimization (MOPSO) method to identify the most suitable product platform from a set of Pareto-optimal solutions [5]. Algeddawy et al. employed the design structure matrix (DSM) to cluster different product components into modules based on the principles of minimum external interfaces and maximum internal integration [6]. Özacar et al. proposed a method for modular ontology development, in which modules are designed as a combination of more abstract modules found in the upper levels of the hierarchy [7]. By combining the essences of flow analysis, DSM, and fuzzy clustering, Li et al. proposed an integrated product modularization approach to build flexible product platforms [8]. Jin Cheng proposed a nested genetic algorithm to address the nonlinear constrained interval optimization issues, and this method is proven to be especially effective for optimizing complex engineering structures under uncertainties [9]. Based on the combination of axiomatic design and fuzzy dendrogram, Rijun Wang proposed an integrated module division method, which uses the top-down product module partition strategy to make different modules functionally independent of each other and reduce the computational efforts required to complete the module division [10]. Based on the theory of function flow, Cong Xiao proposed a module partition method that combines customer requirements, function decomposition, and function realization with modular partition by the function chain [11]. Based on the functional flow, Shuying Gao proposed three basic rules of module partition for the complex engineering systems [12]. Shuangxia Pan proposed a module partition method, which comprehensively considered a variety of different design factors such as assembly, cost, and maintenance [13]. Renbin Xiao proposed a systematic approach to dispose undesirable couplings within product family design based on a two-level structure that includes a strategic level and an operational level [14].

Despite the obvious contributions of these previous studies, a remaining challenge is that the customer preferences are not properly considered in the module partition process. Against such a background, this paper presents a new method to support module partition based on the principles of robust design together with an improved Pareto archive particle swarm optimization (PAPSO) algorithm. Firstly, the criterion function and noise function for a robust design are established on the basis of component-component correlation and component-function dependency, respectively. Next, the existing PAPSO is improved by introducing some chaotic variables, which prevent the algorithm from selecting the premature (or non-optimal) solutions. Thereafter, the mappings between real coding and natural number coding are established, in order to solve the problem of discrete optimization for module partition through continuous optimizations. The proposed new method is characterized by its powerful searching capability for the optimal solution, in particular, when the number of modules is defined as an unknown variable. By doing so, it equips the resulting solution set with a stronger adaptability than the traditional methods.

2 Module partition based on robust design

A great majority of existing module partition methods heavily depend on the designer's empirical experiences to assign the weighting factors in consideration of customer preferences. On the one hand, different customers have diverse preferences, and the customer preferences are dynamically changing over time. On the other hand, the manufacturers cannot afford to keep adjusting the weighting factors purely driven by the customer preferences, because it results in huge development cost every time the weighting factors are reset. In order to reduce the effects of the fluctuation of customer preferences, a module partition model based on robust design is introduced in this section, as illustrated in Fig. 1. Generally speaking, robust design aims to make the performance of an engineered system (e.g., a product, device, equipment, and even process) insensitive to those variances, for example, the customer preferences [15].

The model is constructed based on the IDEF0 functional modeling method, with respect to input, output, mechanism, and control. The inputs are different components of a product, whereas the outputs are different schemes of modular partition. Various design factors are incorporated into the model as its mechanism (or rule) of clustering different components towards a module, which is in line with how the design factors are employed in those traditional module partition methods. The dynamically changing customer preferences are considered as a noise factor that affects the robustness of the module partition schemes. A robust product platform differs from a traditional product platform in a way that not only the controllable variables are considered but also those uncontrollable variables (for example, customer preference) are treated as noise factors. By doing so, the module partition targets obtain the optimal response outputs (i.e., module partition schemes) by considering both noise factors and design factors in a synthetic manner.

The underlying assumption of robust design for module partition, which is important to understand the intellectual merit of this work, is that, if the variances of customer preferences can be considered beforehand, the manufacturers can respond to the dynamic market competition, by imposing minimum changes on the design in a more timely manner [16]. It should be noted that the notion of robust design used in this study is an extension of the traditional robust design paradigm. The premise is that the performance and quality of modular products during the design process, based on a product platform considering the variation, also described as noise, are affected by changes in customer preferences at the stage of module partition. The step-by-step process of the proposed module partition method for robust design is illustrated by Fig. 2.

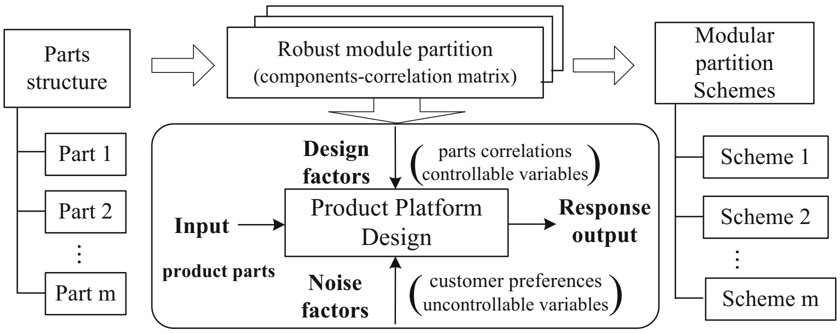
Parts structure
Part 1
Part 2
Part m
Robust module partition (components-correlation matrix)
Design factors
parts correlations controllable variables
Input
product parts
Product Platform Design
Response output
Noise factors
customer preferences uncontrollable variables
Modular partition Schemes
Scheme 1
Scheme 2
Scheme m

Fig. 1 Model of module partition based on robust design

3 Mathematical model of module partition based on robust design

That being explained above, the problem of establishing a robust product platform can be converted to building a mathematical model regarding the criterion function and noise function. The former indicates the product architecture, whereas the latter represents the customer preference. The mathematical model of the proposed module partition method is elaborated in this section.

3.1 Establish criterion function based on the component-component correlation matrix

As mentioned above, the effectiveness of module partition determines the quality of product platform design. This remains true when the generic principles of robust design are considered. The process of module partition based on robust design includes five steps:

1. Establish a component-component correlation matrix by employing the method of analytic hierarchy process (AHP) [17]
2. Develop a mathematical model based on the robust design principles
3. Find out the multi-objective optimization problem from the mathematical model
4. Deploy the PAPSO-C algorithm to produce the optimal solution
5. Produce the module partition schemes

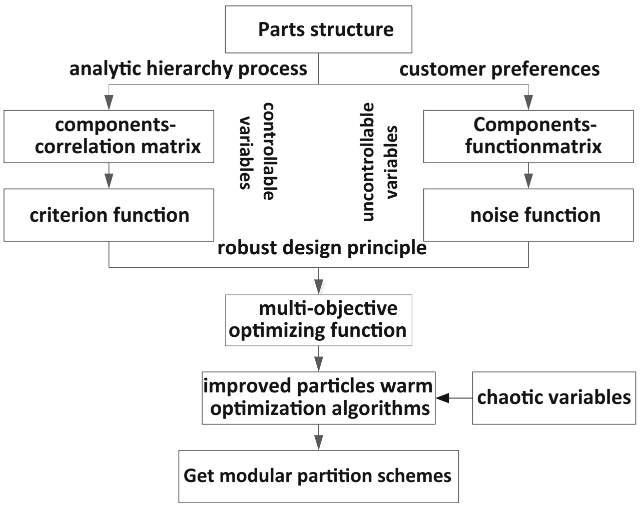
Parts structure
analytic hierarchy process
customer preferences
components-correlation matrix
controllable variables
uncontrollable variables
Components-functionmatrix
criterion function
noise function
robust design principle
multi-objective optimizing function
improved particles warm optimization algorithms
chaotic variables
Get modular partition schemes

Fig. 2 Process of module partition based on robust design

The weighting factor of each element is determined via two steps: (1) assign the weight factor to different elements at each layer and (2) aggregate those weight factors based on the primary and secondary weight values.

As illustrated in Fig. 3, a product component can be analyzed at three different layers: the target layer, the rule layer, and the indicator layer. The similarity component is decomposed in the same group. At the rule layer, w_1, w_2, and w_3 each represents the functional property, structural property, and auxiliary property, respectively. Functional property [18] represents the aggregation of all the functions that are associated with and affected by a certain component. Structural property indicates the dependency relationship among different components of a product, which is specified by the coupling degree and polymerization degree. Auxiliary property means other relevant properties of a product that play additional roles in enhancing customer's satisfaction with the product. The examples of auxiliary properties include but are not limited to "appearance," "material," "color," and "shape." Each of the three properties can be measured by different indicators. Therefore, at the bottom layer, w_{1l}, w_{2h}, and w_{3k} each represents the weight of different functional, structural, and auxiliary indicators, respectively.B_i, $D_{ij}(i=1, j=1, 2, \dots, L; i=2, j=1, 2, \dots, H; i=3, j=1, 2, \dots, K)$ represent the characteristic factors in the corresponding layer, respectively, which affect the distribution of the weight. The correlation degree between any two components must satisfy the following condition, as specified by Eq. (1)

$$w_1 + w_2 + w_3 = \sum\nolimits_{l=1}^{L} w_{1l} + \sum\nolimits_{h=1}^{H} w_{2h} + \sum\nolimits_{n=1}^{K} w_{3k} = 1 \tag{1}$$

The correlation degree between two components is intended to quantitatively describe the functional, structural, and auxiliary properties of a product, which are the premises of module partition using the mathematical method. The correlation degree between two components can be expressed in the fashion of fuzzy relations, the value range of which is

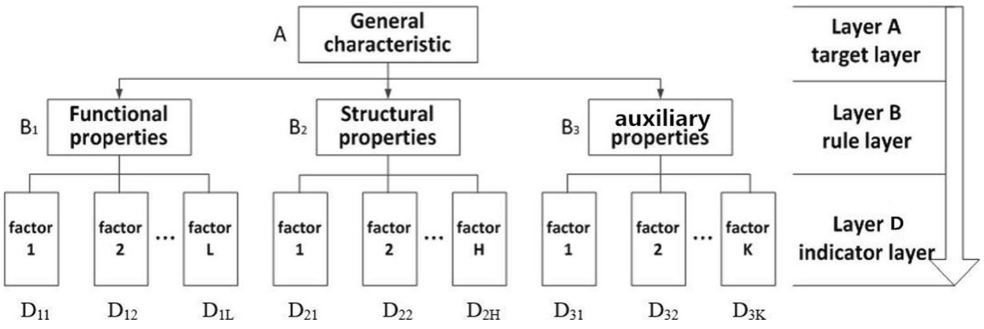

Fig. 3 Hierarchy of characteristics of a component

defined as [0,1] [19]. The correlation degree between any two components $C_i$and $C_j(i,j=1,2,\cdots,N;i\neq j)$ can be calculated via Eq. (2):

$$A(i,j)=\sum_{L=1}^{L}A_{1L}(i,j)w_{1l}+\sum_{h=1}^{H}A_{2h}(i,j)w_{2h}+\sum_{k=1}^{K}A_{3k}(i,j)w_{3k} \quad (2)$$

$A_{1L}(i,j), A_{2h}(i,j)$, and $A_{3k}(i,j)$ each represents the correlation degree between C_i and C_j with respect to the functional, structural, and auxiliary properties, respectively. $A(i, j)$ is the comprehensive correlation degree between component C_i and component C_j. The component-component correlation matrix can be established via Eq. (3):

$$S=\begin{array}{cccc|c} C_1 & C_2 & \cdots & C_N & \\ A(1,1) & A(1,2) & \cdots & A(1,N) & C_1 \\ A(2,1) & A(2,2) & \cdots & A(2,N) & C_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A(N,1) & A(N,2) & \cdots & A(N,N) & C_n \end{array} \quad (3)$$

Based on the above established component-component correlation matrix, the polymerization degree within one module and the coupling degree between different modules are the two most important indexes. Suppose there are N_u components in the module, $M_u, C_{i,j}(i\neq j)$ means the correlation degree between any two components, the correlation degree $A(i, j)$ between C_i and C_j is obtained by using AHP [20], and then the total polymerization degree F_1 of V modules can be calculated using Eq. (4):

$$F_1=\sum_{u=1}^{V}f_u=\sum_{u=1}^{V}\left[\sum_{i=1}^{N_{u-1}}\sum_{j=i+1}^{N_u}A(i,j)\Big/\frac{N_u(N_{u-1})}{2}\right] \quad (4)$$

The coupling degree between any two modules measures an aggregation of individual correlation degrees among all the components within the two modules. Suppose the module M_p includes N_p components, and the module M_q includes N_q components, and component C_i and component C_j each represents a component in module M_p and module M_q, respectively, then the coupling degree between module M_p and module M_q can be calculated using Eq. (5):

$$f_{pq}=\sum_{i=1}^{N_p}\sum_{j=1}^{N_q}A(i,j)\Big/f_{\max}=\sum_{i=1}^{N_p}\sum_{j=1}^{N_q}A(i,j)\Big/\left(N_pN_q\right) \quad (5)$$

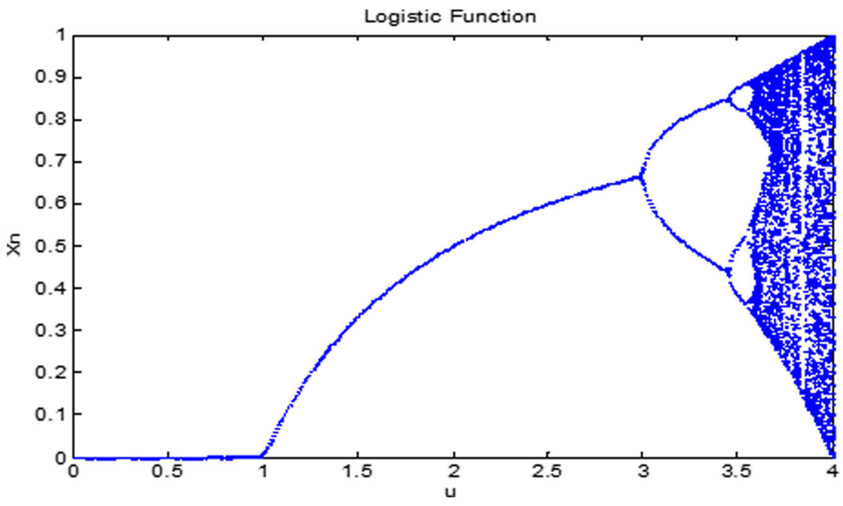

Fig. 4 Bifurcation diagram of logistic function

The total coupling degree of all modules can be calculated via Eq. (6):

$$F_2 = \sum_{p=1}^{V-1} \sum_{q=p+1}^{V} f_{pq} \tag{6}$$

According to the basic principle of module partition, when the polymerization degree goes high, the coupling degree goes low, and vice versa. Therefore, the criterion function can be expressed as Eq. (7):

$$\min F_0 = \frac{F_2}{F_1 \cdot V} \tag{7}$$

For certain modules that consist of no more than one component, M_u becomes equivalent to C_i, hence, $f_{\max} = \frac{N_u(N_u-1)}{2} = 0$. In other words, no polymerization degree can be derived. Therefore, before calculating the coupling degree with another module, it is important to determine whether the module consists of only one component. In the component-component correlation matrix, the value of each entry represents the correlation degree between any pair of two components. In any row of the matrix, if $\max\limits_{i,j \in n, i \neq j} A(i,j) \leq 0.4$, then the component C_i itself constitutes an individual module.

3.2 Establish noise function based on the component-function contribution matrix

By calculating the corresponding contribution of various components to each function, a component-function matrix A can be used to deal with the noise factors in robust design, towards establishing a noise function. Through adjusting the noise function, the result of module partition can be made less sensitive to the variation of customer preferences. In Eq. (8), $C_i(i=1,2,\cdots,m)$ and $F_j(j=1,2,\cdots,p)$ each represents a product's different components and functions, respectively. Furthermore, $Ctb_{ij}(i=1,2,\cdots,m; j=1,2,\cdots,p)$ represents the contribution of the component i to the function j, and $0 \leq Ctb_{ij} \leq 1$. The contribution degrees are assigned by the expert designers based on soliciting individual customer preferences and then aggregating the individual preferences towards a collective preference.

$$A = \begin{array}{ccccc} & C_1 & C_2 & \cdots & C_m \\ F_1 & Ctb_{11} & Ctb_{12} & \cdots & Ctb_{m1} \\ F_2 & Ctb_{12} & Ctb_{22} & \cdots & Ctb_{m2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ F_p & Ctb_{1p} & Ctb_{2p} & \cdots & Ctb_{mp} \end{array} \qquad \sum_{j=1}^{p} Ctb_{ij} = 1 \tag{8}$$

where the sum of the contribution degrees of any component i to all the functions is equivalent to "1" as Eq. (8) depicts.

The noise function can be established according to the contribution degrees (i.e., to what degree a particular component contributes to the realization of a certain function). When dividing modules according to the noise function, the internal polymerization within each module should be minimized, and the external coupling among different modules should be maximized. $M_k(k=1,2,\cdots,R)$ means different modules, and $Q_k(k=1,2,\cdots,R)$ means the quantity of components within a module. For the kth virtual module, the index d_k (i.e., the inner degree of relevance within the module) can be calculated using Eq. (9).

$$d_k = \frac{\sum_{i=1}^{Q_k}\sum_{j=1}^{Q_k}\sum_{p=1}^{P}\left(Ctb_{ip}-Ctb_{jp}\right)^2}{\sum_{i=1}^{Q_k}\sum_{j=1}^{Q_k}\sum_{p=1}^{P}1} = \frac{\sum_{i=1}^{Q_k}\sum_{j=1}^{Q_k}\sum_{p=1}^{P}\left(Ctb_{ip}-Ctb_{jp}\right)^2}{\frac{PQ_k(Q_k+1)}{2}} \tag{9}$$

The parameter P defines the number of customized features according to the customer preferences. Ctb_{ip} and Ctb_{jp} each represents the contribution degree of component i and component j to the same customized feature P, respectively. A small d_k suggests a high relevance within the modules. For all the modules $M_k(k=1,2,\cdots,R)$, the total degree of relevance in the modules can be calculated via Eq. (10):

$$d = \sum_{k=1}^{R} d_k \tag{10}$$

Finally, the noise function can be established using Eq. (11)

$$d_{\text{total}} = \min\left(\sum_{k=1}^{R} d_k\right) \tag{11}$$

4 Module partition based on an improved particle swarm optimization algorithm

4.1 Mathematical model of an improved particle swarm optimization algorithm

The particle swarm optimization (PSO) algorithm is selected to solve the above formulated multi-objective optimization problem, because of its featured advantages of simple rule, quick convergence speed, and few adjustable parameters. This section explains how and in what ways the PSO algorithm is improved for the module partition based on robust design.

In the past, PSO has been widely validated to be an effective approach for the continuous optimization problems [17]. The algorithm was inspired by the social behaviors of flocking birds. To date, some variations of PSO have been proposed in order to solve different kinds of discrete-valued optimization problems (DOP). When extending a single PSO to a multi-objective PSO (MOPSO), the most critical task is to find the best route for each particle within the swarm. Moreover,

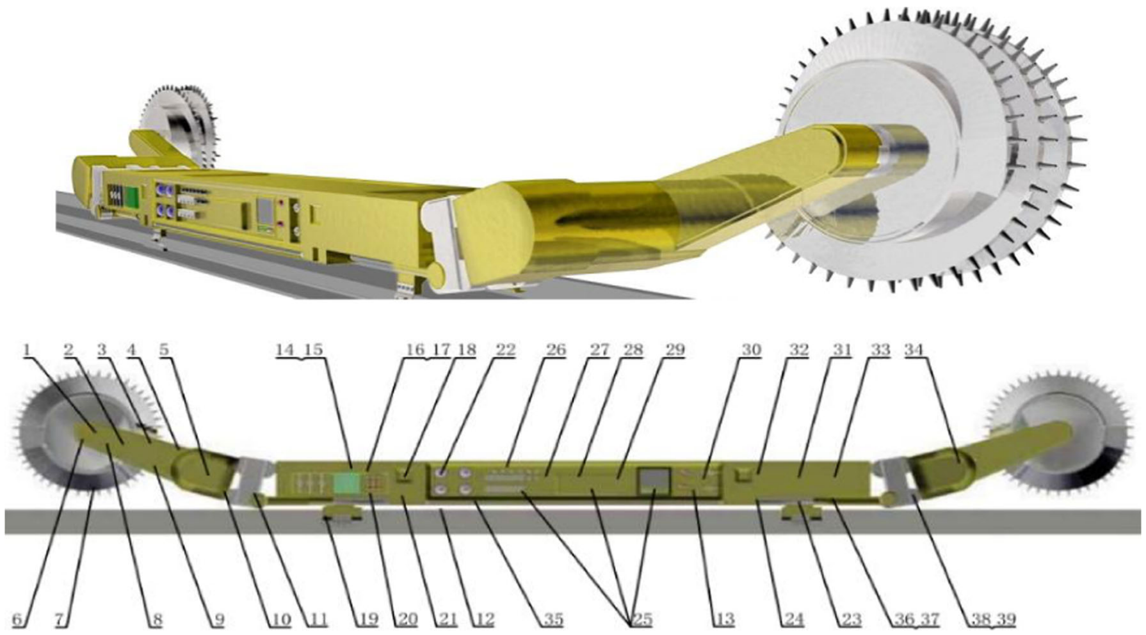

Fig. 5 Illustration of an electric-traction drum shearer

unlike a single-objective problem, a multi-objective problem demands a global optimal solution that can be selected from a set of Pareto optimal solutions [21, 22]. That being explained above, MOPSO is intended precisely to solve the multi-objective optimization problem [23, 24].

Because PAPSO is more efficient in solving the integer programing problem, the mapping from the real coding to the natural number coding is established in order to solve the discrete optimization problem for module partition by means of the continuous optimization method. Although the basic arithmetic operation is a relatively simple operation, it still shows a strong ability to perform global searching with a reasonably fast speed of convergence [25]. This method is intended to solve a two-objective optimization problem, which concerns both a criterion function based on designer's experience and a noise function based on customer's preferences, towards a more robust scheme of module partition.

5 Coding and transcoding

In the interest of the convenience of coding and transcoding, as well as the applicability of particle swarms, the natural number (integer) is used. Using the natural numbers has many advantages. First, the natural numbers are more straightforward to comprehend, which is important for practical applications of the proposed method. Next, the PAPSO algorithm is proven to be more effective and efficient in solving the integer problem. Thirdly, using the natural numbers is in the best interest of shortening computing time and improving the calculation accuracy. Specifically, every component belongs to only one module, and the "module coding" of a particular component is represented by two digits. Every digit is defined from zero to four. If necessary, the digit may increase beyond four to deal with a larger number of components. All components of a product are numbered sequentially, which constitutes a linear vector. For example, suppose a product is composed of ten components, and the module partition scheme is $M_0:[1,2,5], M_1:[3,6,7], M_2:[4,9], M_3:[8,10]$, then the corresponding coding string is expressed as: ||00|00|01|02|00|01|01|03|02|03||.

The adjacent four digitals are merged together to constitute the substrings. Therefore, the above coding string can be transformed to ||0000|0102|0001|0103|0203||. Every substring has 625 combinations. Arrange h_1、 h_2、 h_3、 h_4 as four digitals of a substring and then transcode them in accordance with the following formula: $z_i = 0.01(5^3h_1 + 5^2h_2 + 5h_3 + h_4)$. Finally, the vector $z = \{0, 0.27, 0.01, 0.28, 0.53\}$ is obtained after transcoding the sample string.

6 The updating of particle velocity and position

In PSO, suppose that a population has N particles, then each decision variable $X_n^{(k+1)}$ has a velocity that concludes a location of the next step. The position of each particle can be calculated using Eq. (12) [26]. The component vector in generation k about the ith particle is described as $X_i^{(k+1)}$. Finally, the velocity of a particle is defined by Eq. (13) [25].

$$X_i^{(k+1)} = X_i^k + \lambda V_i^{(k)} \tag{12}$$

$$v_{id}^{(k+1)} = w v_{id}^{(k)} + c_1 r_1 \left(p\text{best}_{id} - x_{id}^{(k)}\right) + c_2 r_2 \left(g\text{best}_{id} - x_{id}^{(k)}\right) \tag{13}$$

Table 1 List of components of an electric-traction drum shearer

No.	Component	No.	Component	No.	Component
1	Cutting motor	14	Traction motor	27	Pressure relay
2	Rocker gear box	15	Transmission shaft 3	28	Oil level detector
3	Transmission shaft 1	16	Transmission shaft 4	29	Pressure gauge
4	Double planetary gear reducer 1	17	Double planetary gear reducer 2	30	Reversing valve handle
5	Transmission shaft 2	18	Transmission shaft 5	31	Hoisting ring
6	Drum base	19	Guide foot	32	Cylinder
7	Cutting drum	20	External traction shell	33	Check valve
8	Brake 1	21	Filling fittings	34	Cooler
9	Picks sprayer	22	Induction motor	35	Motor cover
10	Clutch handle	23	Height adjustment pump	36	Hose guardrail
11	Brake 2	24	Oil tank	37	Towing device
12	Main frame part 1	25	Valve block	38	Magnetic filter
13	Main frame part 2	26	Accumulator	39	Pipe accessories

X_i^k means the position of the *i*th particle in generation *k*, and $V_i^{(k)}$ means the moving speed of the *i*th particle in generation *k*. λ means a constraining factor that limits the velocity's magnitude, and the interval of λ is [0,1]. In this case, the value of λ is assigned to be "1." Equation (12) can be used as a general equation to obtain the location of a particle in generation $k + 1$ based on the location and velocity in generation *k*. The velocity of each particle is updated to fly into the personal best ($p\text{best}_{id}$) and the global best ($g\text{best}_{id}$) for the *d*th component of the *i*th particle. The parameter *w*, which represents the inertia, controls the trade-off between the global search and the local search, and its value is set to be "0.4." r_1 and r_2 are two uniformly distributed random numbers in the range [0,1]. Two constants, c_1 and c_2, manipulate the weighting factors of the personal best value and global best value. In this study, both c_1 and c_2 are assigned to be "1."

In order to exclude those premature solutions of PAPSO, a logistic function is used to improve the inertia weight *w* with chaotic variables [27]:

$$X_{n+1} = X_n \times \mu \times (1 - X_n) \tag{14}$$

where $\mu \in [0, 4]$ represents the status variable, whereas $X \in [0, 1]$ represents the system control parameter. The function values obey distribution of pseudorandom if $3.5699456 < \mu \leq 4$, which improves the search capability of particles. The logistic property of chaos can prevent the algorithm from falling into local optimum. The bifurcation diagram of the logistic function is illustrated by Fig. 4.

7 Archive maintenance

External archives are used to store the non-inferior solutions during the searching process. Because the capacity of the

Table 2 The component-component correlation matrix

	#1	#2	#3	#4	…	#37	#38	#39
#1	1	0.75	0.68	0.78	…	0	0	0.02
#2	0.75	1	0.82	0.59	…	0.29	0	0
#3	0.68	0.82	1	0.69	…	0.22	0	0.08
#4	0.78	0.59	0.69	1	…	0.12	0	0.08
#5	0.64	0.67	0.69	0.77	…	0.22	0	0.08
	⋮	⋮	⋮	⋮	⋱	⋮	⋮	⋮
#36	0	0.27	0	0	…	0.87	0.73	0.91
#37	0	0.29	0.22	0.12	…	1	0.69	0.77
#38	0	0	0	0	…	0.69	1	0.82
#39	0.02	0	0.08	0.08	…	0.77	0.82	1

Table 3 The component-function contribution matrix

	#1	#2	#3	…	#37	#38	#39
Rocker arms	0.89	0.87	0.92	…	0	0	0
Drums	0	0	0	…	0	0	0
Traction drive	0	0	0	…	0	0	0
External traction	0	0	0	…	0	0	0
Pumping station	0	0	0	…	0	0	0
Height oil vat	0	0	0	…	0	0	0
Main frame	0	0	0	…	0.11	0.07	0.13
Auxiliary components	0.11	0.13	0.08	…	0.89	0.93	0.87

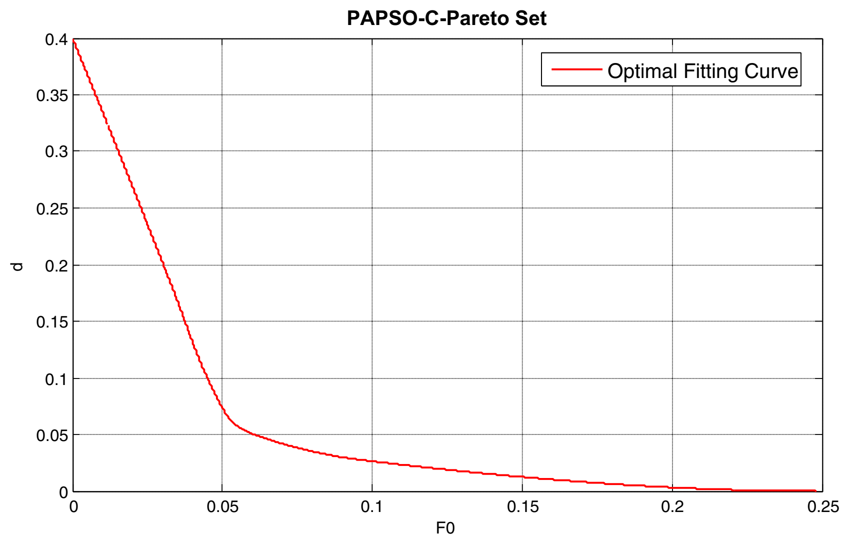

Fig. 6 PAPSO-C Pareto set

external archives is limited, they should be regularly maintained in order to determine whether a new solution can be stored into the archives without exceeding the capacity (i.e., the number of non-inferior solutions reaches the limitation). Each solution contained in the archive A_t should be the global optimum result of a certain particle in order to avoid approaching the local optimum result of the Pareto front [28].

8 Mutation

The non-inferior solutions in the external archives should be regularly updated in order to prevent the algorithm from stopping, which is often caused when the solutions are crowded within a small local region. In order to improve the convergence and expand the solution space, the mutation of solutions is necessary after maintaining the external archives.

8.1 Description of PAPSO-C used in module partition

$$\text{objective vector functions}: \begin{cases} \min F_0 = \dfrac{F_2}{V \cdot F_1} \\ \min d = \left(\sum_{k=1}^{R} d_k\right) \end{cases} \tag{15}$$

Each variable is the function of the correlation matrix $S=(A(i,j))_{N\times N}$ or the function matrix $A=(Ctb_{ij})_{m\times p}$, and the searching space is the domain of the corresponding function [29]. The smaller the objective function value is, the better the module partition scheme is.

Each step of the algorithm is specified as the following:

Step 1: $t=0$, initialize the particle swarm S_t, calculate the vector of objective function of each particle, and store the non-inferior solutions into archive A_t.

Step 2: Set the initial G_t and P_t, which respectively means the *p*best and *g*best of the particle at time *t*.

Step 3: Update the velocity and position of the particles according to Eqs. (12) and (13) within the searching space in order to obtain S_{t+1}, and then update P_t of the particles.

Step 4: Maintain the archives, obtain A_{t+1}, and then calculate the G_t of each particle.

Step 5: $t=t+1$, stop searching if the termination conditions become satisfied, or go back to repeat step 3.

The following method can be followed to prevent the particles from flying out of the boundaries in step 3:

$$\text{If } x_{tj}^i + v_{tj}^i > \beta_j, v_{(t+1)j}^i = \theta\left(\beta_j - x_{tj}^i\right), x_{(t+1)j}^i = x_{tj}^i + v_{(t+1)j}^i,$$

$$\text{If } x_{tj}^i + v_{tj}^i < \alpha_j, v_{(t+1)j}^i = \theta\left(x_{tj}^i - \alpha_j\right), x_{(t+1)j}^i = x_{tj}^i - v_{(t+1)j}^i,$$

$\theta < 1, v_t = (v_{t1}, v_{t2}, \cdots, v_{tn}), x_t = (x_{t1}, x_{t2}, \cdots, x_{tn}), x_{tj} \in [\alpha_j, \beta_j]$, $[\alpha_j, \beta_j]$ is the domain of the *j*-th decision variable, $j=1, 2, \cdots, n$.

9 Case study

9.1 Module partition of an electric-traction drum shearer

A case study is presented in this section to demonstrate how the proposed new method can be used to solve real-world design problems. The case study is regarding a complex engineered system, the electric-traction drum shearer, a heavy machine commonly used in the mining industry. Because of the demanding conditions of longwall mining, robustness is

Table 4 The optimal result of module partition

Module	Component	Main function
1	[1–6, 10]	Rocker arm
2	[7–9]	Cutting drums
3	[12, 13]	Main frame
4	[11, 14–18, 21]	Traction gearbox
5	[19, 20]	Traction drive
6	[22–30]	Pumping station
7	[31,32,33]	Height oil vat
8	[34,35,36,37,38,39]	Auxiliary components

an important quality of a drum shearer. A typical shearer consists of 39 components, as illustrated in Fig. 5 and summarized in Table 1.

The weights of the functional property (ω_1), structural property (ω_2), and auxiliary property (ω_3) are assigned based on the designers' domain knowledge and empirical experience. Specifically, the weights are determined to be $\omega_1 = 0.6$ $\omega_2 = 0.25$ $\omega_3 = 0.15$. The correlation degree $A(i,j)$ of any pair of two components is calculated using the Eq. (2).

For example, the correlation degree between the component #18 and component #21 is calculated as $A(18,21) = \sum_{L=1}^{6} A_l(i,j) w_l = 0.6 \times 0.6 + 0.6 \times 0.25 + 0.4 \times 0.15 = 0.57$

By thoroughly calculating the correlation degree of every possible pair of two components from component #1 to component #39, a full component-component correlation matrix R is established, as shown in Table 2.

The component-function contribution matrix indicates the contribution of the 39 components to the ten major functions of a drum shearer. Based on a thorough literature review and in consideration of the experts' opinions, the component-function contribution matrix is established, as shown in Table 3. Please note that these experts' opinions indirectly represent the customer preferences.

The next step is to establish a criterion function using Eq. (7) and a noise function using Eq. (11). As a result, a multi-objective optimization problem is formulated, which can be solved using Eq. (15).

9.2 Module partition based on an improved particle swarm optimization algorithm

The PAPSO-C algorithm was developed using the MATLAB and run with a population of 100 particles and 200 generations. After 20 calculation cycles, the result of the optimal Pareto front is shown in Fig. 6:

The abscissa values are the values of criterion function calculated based on the component-component correlation matrix, while the ordinate values are the values of noise function calculated based on the component-function contribution matrix. The curve shows that the Pareto front is convex, with an inflection point (f_0, d_0) at $F_0 = 0.055$. The curve is initially steep within the interval $(0, f_0)$, while it becomes gentle within the interval (f_0, f_{max}). As suggested by the low sensitivity of the criterion function, the noise function has a relatively low impact on the result of module partition. Based on a comprehensive consideration of the two-objective optimization result and the robust design, it is determined by the designers that the optimal result of module partition can be achieved near the inflection point when $F_0 = 0.058$ and $d = 0.05476$. The final result of module partition is summarized in Table 4.

In summary, the component-component correlation matrix is established by using the traditional methods. In the meantime, the basic principles of robust design are complied with to establish the component-function contribution matrix by considering customer preferences as a noise factor that affects the design decisions regarding module partition. Furthermore, the

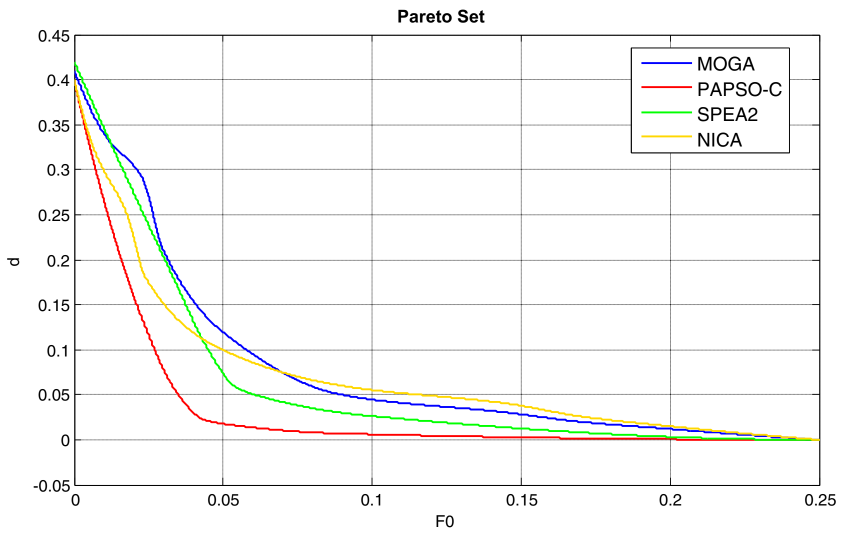

Fig. 7 Comparison of the Pareto sets among the four algorithms

Table 5 Performance comparison among the four algorithms

	PAPSO-C	NICA	SPEA2	MOGA
Running time	Shortest	Long	Short	Short
Approximation to Pareto front	Best	Excellent	Good	Good
Uniformity	Excellent	Good	Acceptable	Good
Breadth	Good	Excellent	Good	Good
Covering space of Pareto front	Good	Best	Excellent	Good

multi-objective particle swarm optimization is improved by introducing the notion of chaotic variables, which serves to prevent selecting those premature solutions. By doing so, the module partition is conducted in consideration of functional independence, structural integrity, as well as enhanced robustness against dynamically changing customer preferences.

9.3 Comparison and analysis of the algorithms

For a two-objective optimization problem, the algorithm this paper favors (i.e., PAPSO-C) is compared to the other three commonly used algorithms, which are the multi-objective genetic algorithm (MOGA) [30], the strength Pareto evolutionary algorithm (SPEA2) [31], and the multi-objective artificial immune clone algorithm (NICA) [32], with respect to the resulted optimal Pareto fronts as illustrated by Fig. 7.

The four algorithms are compared against a variety of different metrics (i.e., running time, approximation to Pareto front, uniformity, breadth, and covering space of Pareto front). The comparison results are summarized in Table 5. In particular, the PAPSO-C algorithm significantly outperformed the other three algorithms with respect to the metrics of "running time" and "approximation to Pareto front." Furthermore, the PAPSO-C algorithm also demonstrated competitive performance in regard to the metrics of "uniformity" and "breadth" of distribution.

10 Conclusion and future work

In conclusion, this study contributes to the understanding and practice of module partition for product platform design in two ways:

1. The basic principles of robust design are incorporated into the module partition process. The primary objective is to make the module partition schemes less sensitive to the customer preferences, which are always dynamically changing and can be hardly manipulated by the manufacturers. Firstly, a criterion function is established based on the component-component correlation matrix. Next, a noise function is established based on the component-function contribution matrix in consideration of customer preferences as a noise factor. Finally, the criterion function and the noise function together constitute a multi-objective optimization problem.
2. An improved particle swarm optimization algorithm is proposed to address the above formulated multi-objective optimization problem. Specifically, the chaotic variables are introduced in order to exclude those premature solutions without significantly increasing the algorithm's complexity. The improved algorithm significantly outperformed some traditional algorithms in terms of both running time and approximation to Pareto front.

With respect to the future works, the proposed method will be applied and validated through another more complex product, for instance, a certain product from the automotive, aerospace, or shipbuilding industries.

Acknowledgments This research is supported by the National Natural Science Foundation of China (Grant number 51675028).

References

1. Tchertchian N, Millet D, Pialot O (2013) Modifying module boundaries to design remanufacturable products: the modular grouping explorer tool. J Eng Des 24(8):546–574
2. FJ Emmatty, S. P. Sarmah. (2012). Modular product development through platform-based design and DFMA. J Eng Des 23(9):696-714.
3. Umeda Y, Fukushige S, Tonoike K (2008) Product modularity for life cycle design. J. CIRP Annals: Manufacturing Technology, 57, 13-16.
4. Simpson TW, D'souza B (2006) Assessing variable levels of platform commonality within a product family using a multi-objective genetic algorithm. 9th AIAA/ISSMO symposium on multidisciplinary analysis and optimization. Atlanta 9:1–10
5. Moon SK, Park KJ, Simpson TW (2014) Platform design variable identification for a product family using multi-objective particle swarm optimization. Res Eng Des 25:95–108
6. Algeddawy T, Elmaraghy H (2013) Optimum granularity level of modular product design architecture. CIRP Ann Manuf Technol 62: 151–154
7. Özacar T, Öztürk O, Ünalir MO (2011) ANEMONE: An environment for modular ontology development. Data Knowl Eng 70:504–526
8. Li ZK, Cheng ZH, Feng YX et al (2013) An integrated method for flexible platform modular architecture design. J Eng Des 24:25–44
9. Cheng J, Liu Z, Wu Z et al (2016) Direct optimization of uncertain structures based on degree of interval constraint violation. Comput Struct 164:83–94
10. Wang R (2009) Based on the axiomatic design and the integrated module partition method of fuzzy tree graph. Journal of agricultural machinery 40(4):179–183
11. Xiao C (2009) Complex product module partition method based on the function flow study. Mechanical design and manufacturing 12: 249–251
12. Gao S (2005) Based on function module partition method in flow model and case study. Machine with hydraulic 9:22–23
13. Pan S (2003) Mass customization production mode of module partition method research. Journal of mechanical engineering 39(7):1–6

14. Xiao R, Cheng X (2016) A systematic approach to coupling disposal of product family design (part 1): methodology. Procedia CIRP 53:21–28
15. Rathod V, Yadav OP, Rathore A et al (2013) Optimizing reliability-based robust design model using multi-objective genetic algorithm. Comput Ind Eng 66:301–310
16. Gremyr I, Siva V, Raharjo H et al (2014) Adapting the robust design methodology to support sustainable product development. J Clean Prod 79:231–238
17. Zhengyu L (2005) Research on evaluation method of module division scheme. Construction machinery 36(5):24–26
18. Guo W, Liu G, Zhang L (2010) Research on product green module partition method for full life cycle. Journal of Hefei University of Technology 33(10):1441–1445
19. Wang H, Wang J, Sun B (2005) Optimization design of modular product family based on core platform. Comput Integr Manuf Syst 11(2):162–167
20. Salil D, Bopaya B, Lovell MR (2012) Material and process selection in product design using decision-making technique (AHP). European Journal of Industrial Engineering 43:322–346
21. Ehsan T, Maryam M (2011) A novel method in fuzzy data clustering based on chaotic PSO. Internet technology and secured transactions, pp. 335–340
22. Alvarez-Benitez JE, Everson RM, Fieldsend JE (2005) A MOPSO algorithm based exclusively on pareto dominance concepts, evolutionary multi-criterion optimization. Lect Notes Comput Sci 3410: 459–473
23. Su S, Yu H, Wu Z, Tian W (2014) A distributed coevolutionary algorithm for multiobjective hybrid flowshop scheduling problems. Int J Adv Manuf Technol 70:477–494
24. Cheng J, Liu Z, Tan J (2013) Multiobjective optimization of injection molding parameters based on soft computing and variable complexity method. Int J Adv Manuf Technol 66:907–916
25. Jongbin I, Jungsun P (2013) Stochastic structural optimization using particle swarm optimization, surrogate models and Bayesian statistics. Chin J Aeronaut 01:112–121
26. Wu X (2011) Uniform search particle swarm optimization (PSO) algorithm. Electronic journals 39(6):1261–1266
27. Pluhacek M, Senkerik R, Zelinka I et al (2013) New adaptive approach for chaos PSO algorithm driven alternately by two different chaotic maps—an initial study. J. Advances in Intelligent Systems and Computing 210:77–87
28. Feng YX, Zheng B, Li ZK (2010) Exploratory study of sorting particle swarm optimizer for multiobjective design optimization. J. Mathematical and Computer Modeling 52:1966–1975
29. An WG (2007) Interactive multi-objective optimization design for the pylon structure of an airplane. Chin J Aeronaut 06:524–528
30. Fan WH, Xu HY, Xu X (2009) Simulation on vehicle routing problems in logistics distribution. J COMPEL - The international journal for computation and mathematics in electrical and electronic engineering 28:1516–1531
31. Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the performance of the strength Pareto evolutionary algorithm, technical report 103. Zurich, Computer Engineering and Communication Networks Lab (TLK), Swiss Federal Institute of Technology (ETH)
32. Hart E, Timmis J (2005) Application areas of AIS: the past, the present and the future. Proc 4th Inter Conf Arti Immune Sys, ICARIS 2005, Springer, Lect Notes in Comp Sci 3627:483–497