Honda Research Institute EU
**Honda Research Institute Europe GmbH**
https://www.honda-ri.de/

# A review of concurrent optimization methods

## Xiaofen Lu, Ke Tang, Bernhard Sendhoff, Xin Yao

## 2014

**Preprint:**

This is an accepted article published in International Journal on Bio-inspired Computation. The final authenticated version is available online at: https://doi.org/[DOI not available]

# A Review of Concurrent Optimization Methods

Xiaofen Lu [1, 2], Ke Tang [2], Bernhard Sendhoff [3], Xin Yao [1, 2]

[1] Center of Excellence for Research in Computational Intelligence and Applications (CERCIA)
School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, U.K.

[2] USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications,
School of Computer Science and Technology,
University of Science and Technology of China (USTC), Hefei, Anhui 230027, China.

[3] Honda Research Institute Europe GmbH, Offenbach 63073, Germany.

E-mails: xxl332@cs.bham.ac.uk, ketang@ustc.edu.cn, bernhard.sendhoff@honda-ri.de, x.yao@cs.bham.ac.uk

**Abstract:**
During the past decades, products and their manufacturing processes have become much more complex than before. This have led to complex optimization problems that require optimizing the complicated product development process or the physical aspects of products with respect to a lot of design variables. A common approach to address these problems is to decompose the problem into a number of simpler sub-problems and optimize each sub-problem concurrently. This idea has been investigated in different research fields, including Concurrent Engineering (CE) and Cooperative Coevolution (CC). In this paper, main topics in CE and CC are reviewed, and the relationships between CE and CC are discussed along with some potential combinations between them.

**Keywords:** Concurrent Engineering, Cooperative Coevolution, Decomposition, Optimization

**Biographical notes:** Xiaofen Lu received the B.Eng. degree in Computer Science and Technology from the Nature Inspired Computation and Applications Laboratory (NICAL), School of Computer Science and Technology, University of Science and Technology of China (USTC), Hefei, China, in 2009. In 2009, she became a Ph.D. candidate majoring in Applied Computer Technology in NICAL, USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications (UBRI), School of Computer Science and Technology, USTC. In 2013, she became a Ph.D. candidate majoring in Computer Science in University of Birmingham, UK. Her current research interests include evolutionary computation, dynamic optimization, surrogate based optimization and various real-world applications.

Ke Tang received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from Nanyang Technological University, Singapore. Currently, he is a professor at the USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications (UBRI), School of Computer Science and Technology, University of Science and Technology of China (USTC), Hefei, China. He is also an Honorary Senior Research Fellow in the School of Computer Science, University of Birmingham, UK, and an associate editor of the IEEE Computational Intelligence Magazine and the Computational Optimization and Applications Journal. His major research interests include evolutionary computation and machine learning. He has authored/co-authored more than 70 refereed publications.

Bernhard Sendhoff obtained the PhD degree in Applied Physics in May 1998, from the Ruhr-Universität Bochum, Germany. From 1999 to 2002 he worked for Honda R&D Europe GmbH, and since 2003, he has been with Honda Research Institute Europe GmbH. Since 2007 he is Honorary Professor of the School of Computer Science of the University of Birmingham, UK. Since 2008, he is Honorary Professor at the Technical University of Darmstadt, Germany. Since 2011, he is President of the Honda Research Institute Europe GmbH.

Xin Yao is a Professor of Computer Science and the Director of CERCIA (the Centre of Excellence for Research in Computational Intelligence and Applications), University of Birmingham, UK. He is an IEEE Fellow and a Distinguished Lecturer of IEEE Computational Intelligence Society (CIS). His work won the 2001 IEEE Donald G. Fink Prize Paper Award, 2010 IEEE Transactions on Evolutionary Computation Outstanding Paper Award, 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), 2011 IEEE Transactions on Neural Networks Outstanding Paper Award, and many other best paper awards at conferences. He won the prestigious Royal Society Wolfson Research Merit Award in 2012 and was selected to receive the 2013 IEEE CIS Evolutionary Computation Pioneer Award. He was the Editor-in-Chief (2003-08) of IEEE Transactions on Evolutionary Computation, and is the President (2014-15) of IEEE CIS. His major research interests include evolutionary computation, neural networks ensembles, and their applications. He has more than 400 refereed publications in international journals and conferences.

Copyright © 200x Inderscience Enterprises Ltd.

# 1 Introduction

Complex modern products have led to challenging real-world optimization problems, which are formulated to optimize the product development process or design parameters with respect to the physical aspects of products. On one hand, the product development process can be composed of various design activities and thus hard to manage as a whole. On the other hand, the product structure can be very complicated and thus lead to high-dimensional optimization problems.

To address these problems, one common approach is to decompose the optimization problem into a number of simpler sub-problems and optimize each sub-problem concurrently. As a general problem-solving philosophy, this concept has been investigated in different areas. Specifically, this idea is usually referred to as Concurrent Engineering (CE) [1] in the domain of industrial design, while similar idea is explored in the Evolutionary Computation (EC) community under the name of Cooperative Coevolution (CC) [2]. Although CE and CC are naturally relevant to each other, their relationship has seldom been discussed before. Due to this consideration, this paper reviews the latest advances on CE and CC, and discusses the inherent relationships between them. In this way, we aim to identify promising future research directions for developing more powerful CE approaches based on CC techniques, as well as to achieve new insights of the advantages and drawbacks of CC for real-world problems and promote the research on CC itself.

The rest of this paper is organized as follows. In Section 2, an introduction of CE will be given and some topics in CE will be detailed. In Section 3, recent advances on CC will be presented. In Section 4, the potential links between CE and CC and possible cross-fertilization will be discussed. Finally, Section 5 will conclude this paper.

# 2 Concurrent Engineering

In today's fast-changing and highly competitive market, efficient Product Development (PD) is the key for industries to survive. To improve their PD practices, one widely accepted approach in industries is through the use of "Concurrent Engineering" (CE) [3].

The generally accepted definition of CE was formulated in 1988 [4] as follows: "Concurrent engineering is a systematic approach to the integrated, concurrent design of products and related process, including manufacture and support. CE intends to cause developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements". Since then, researchers have given many interpretations of "Concurrent Engineering" in literature [5-8]. In [8], over 123 papers dealing with this subject have been listed. In general, CE emphasizes taking all elements of the product life cycle into consideration in the product design, and entails concurrent or overlapped design of the product and related processes instead of using a sequential development approach.

In traditional sequential engineering, the development process starts with user requirements and sequentially moves forward to detailed design, design for production and other additional steps until a product is finished. In this process, each design activity in the development process must be completed before a subsequent activity can begin. As a result, the design will be scrapped or heavily altered once the development cannot progress in a later activity. In contrast, CE allows all design activities to be conducted in parallel or with some overlap in an integrated environment. Overlapping allows a downstream design activity (such as design for production) to begin earlier before the upstream design activity (such as detailed design) is finished using some preliminary information, and thus the total development time can be reduced [9]. Furthermore, CE makes available the feedback of downstream design activities at an earlier time. Consequently, errors can be discovered earlier when the project is still flexible, and re-designs can thus be avoided. This would reduce the development time and cost, and improve the product quality [3]. Fig. 1 shows the sequential workflow and CE workflow.

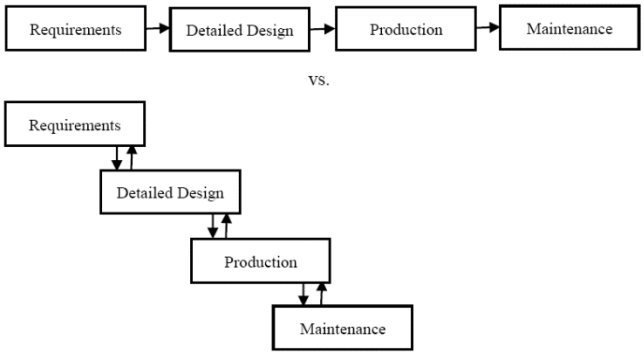



Fig. 1 Sequential engineering workflow (upper) and CE workflow (below)

## *2.1 Decomposition Methods in CE*

In CE, all design activities are encouraged to be conducted in parallel or with some overlap. This calls for effective decomposition methods through which the concurrency aspect of design (i.e., the degree to which design activities can be scheduled simultaneously) can be explored and thus concurrent design is enabled.

In [10], Kusiak and Larson categorized existing decomposition methods into three types: product decomposition, process decomposition and problem decomposition. Product decomposition is based on the physical elements of the product, i.e., product modularity and structure. For example, a car product based on the product modularity can be decomposed into several subsystems including engine, fuel system, and control system and so on. Based on the structural relationships,

Copyright © 200x Inderscience Enterprises Ltd.

each subsystem can be further decomposed into several body components. For example, the engine can be further decomposed into several body components such as piston components and crank mechanism. Modular subsystems can be designed in parallel as product modularity displays weak dependencies. In contrast, in structural decomposition, different components can have much stronger dependencies. The limitation of one subsystem in space or weight would also restrict the design of its constituents. Moreover, the physical or functional connection between more than two components would lead to existence of shared variables among them.

As CE takes all elements in the product life cycle into consideration at the beginning, the design of one part of the product requires different expertise from various engineering disciplines. Process decomposition decomposes the design based on engineering disciplines, which results in different perspectives sharing a common set of design variables or coupling variables that represents the part of the product.

Further, the design problems relevant to the decomposed products and processes can be divided into smaller sub-problems by problem decomposition. Constraint-parameter decomposition is one kind of problem decomposition methods [10]. It identifies relationships between parameters and constraints, and then a constraint or a set of constraints with strong dependencies can be implemented as a sub-problem. Other kinds of problem decomposition methods can be found in [10].

By using the above-mentioned decomposition methods, the design project will be decomposed into numerous interdependent design activities. When assigning these design activities into several teams of designers, grouping of design activities is required. Grouping of design activities aims to assign the same team with strong interdependent design activities and different teams with weak interdependent design activities. In [11], an "activity on nodes" (AON) network is used to represent the activities and their relationships. Each node denotes an activity and directional arcs denote the precedence relationship between activities. Then, grouping activities can be viewed as breaking down the network into a number of sub-networks. In [11], a heuristic algorithm was employed to solve this problem. It attempts to decompose design activities into groups of activities with least number of intergroup links. Also, it attempts to maintain the links between any pair of groups to be unidirectional.

In general, through the above-mentioned decomposition methods, a design problem can be decomposed into several smaller problems. The resulted sub-problems can be interrelated to one another by variable sharing (e.g. if decomposed by product and process decomposition). During the design process, the results of these smaller problems need to be recursively combined to formulate solutions to the larger problems to guarantee the overall optimization. Also, the consistency between the shared variable values used in the different sub-problems should be guaranteed.

## *2.2 Coordination in CE*

Through decomposition, the optimization problem can be decomposed into simpler sub-problems that can be solved concurrently. However, these sub-problems may not be solved in a completely independent way. As mentioned above, sub-problems generated by decomposition may have common design variables or be linked through interactions (i.e. coupling variables). In this situation, to obtain an optimal system design, the consistency on the common design variables and on the coupling variables should be ensured while solving each sub-problem separately, which is called coordination [12].

As a family of methods for the above coordination, Multidisciplinary Design Optimization (MDO) have been developed with regard to multiple coupled disciplines [13]. A standard example of MDO application problems exists in aeroelastic design [12], as shown in Fig. 2. Aeroelastic design aims to design an airplane wing considering both structural and aerodynamic aspects. Suppose the wing is not sufficiently stiff. The aerodynamic analysis requires the deformed wing shape that needs to be computed through the structural analysis, while the structural analysis in turn is based on the aerodynamic pressure distribution over the wing surface that needs to be computed through the aerodynamic analysis. It can be seen that the aerodynamic and structural analyses are coupled through the pressure distribution and deformed shape variables. For this example, a consistent state of the pressure distribution and deformation should be achieved in the final wing design.

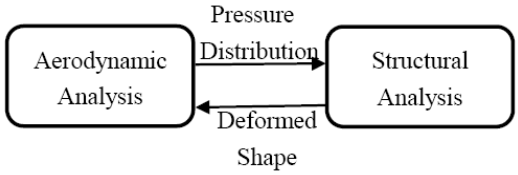



Fig.2 Aeroelastic analysis, taken from [12]

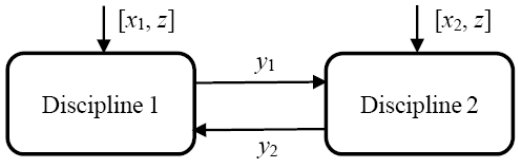



Fig. 3 Two-discipline system

For a system with two coupled disciplines, a more detailed description is shown in Fig. 3, where $x_1$, $x_2$, $z$ are design variables, $z$ denotes the shared design variables or global design variables, and $y_1$, $y_2$ are the coupling variables (also the analysis outputs) that are interdependent. The challenge in optimizing such a system concurrently (i.e., simultaneously optimizing each discipline sub-system) is how to manage the shared

variables $z$ and the coupling variables $y_1$ and $y_2$. In the literature, different methods has been proposed to address this problem. In this paper, three popular methods, Collaborative Optimization (CO) [14], Concurrent Subspace Optimization (CSSO) [15], and Bi-Level System Synthesis (BLISS) [16], are introduced.

### *2.2.1 Collaborative Optimization (CO)*

In CO, a system level optimization is conducted to optimize the global design variables $z$ and coupling variables $y$ to minimize the system objective while satisfying the interdisciplinary compatibility, and each disciplinary level optimization is responsible for minimizing the discrepancies between it and other disciplines while satisfying its specific local constraints [17]. In CO, each local disciplinary level optimization can be conducted in parallel, only communicating with the system level optimization each iteration, which plays a coordination role.

Assume a system has $n$ disciplines. The CO formulation can be stated at the system level as:

$$\begin{aligned} &\min_{z^{SL},\, y^{SL}} \; f(z^{SL}, y^{SL}) \\ s.t. \;\; &J_i\left(z_i^{SL}, z_i^*, y_i^{SL}, y_i(x_i^*, y_j^{SL}, z_i^*)\right) = \| z_i^{SL} - z_i^* \|_2^2 + \\ &\qquad \| y_i^{SL} - y_i \|_2^2 = 0 \\ &\qquad \text{for}\, , i, j = 1, \dots, n, j \neq i \end{aligned} \tag{1}$$

where $J_i$ represents the compatibility constraints, one for each discipline, and * represents the solution obtained in the $i$-th discipline optimization.

The $i$-th disciplinary level optimization problem takes the following form as:

$$\begin{aligned} &\min_{z_i, x_i} J_i(z_i, z_i^{SL}, y_i(x_i, y_j^{SL}, z_i), y_i^{SL}) = \| z_i - z_i^{SL} \|_2^2 + \\ &\qquad \| y_i - y_i^{SL} \|_2^2 \\ &s.t. \quad g_i\left(x_i, z_i, y_i(x_i, y_j^{SL}, z_i)\right) \leq 0 \end{aligned} \tag{2}$$

where $z_i^{SL}$ and $y_j^{SL}$ are the solutions obtained in the system level optimization and $g_i$ represents the specific disciplinary constraints.

It can be seen that the decomposition used in CO can lead to much lower dimensional sub-problems only if the number of disciplinary variables is much larger than the global variables. Thus, CO is appropriate for solving design problems with loosely coupled analyses of individual and large dimension [13].

### *2.2.2 Concurrent Subspace Optimization (CSSO)*

In CSSO, each subspace optimization is performed concurrently using approximation of the coupling variables responses in other disciplines. Concretely, CSSO uses an iterative optimization process. It begins with an initial set of design points. For each design point, a complete multidisciplinary analysis is performed to provide data points to construct discipline approximation models, which can provide an estimate of each discipline's coupling variables of design points. The multidisciplinary analysis is to calculate the objective function values of design points by generating a consistent state of coupling variables for each design point. Once the approximation model for each discipline analysis is built, each subspace designer can optimize the global design variables and its local design variables by evaluating them based on these discipline approximation models. The $i$-th discipline optimization can be stated as:

$$\begin{aligned} &\min_{z, x_i} \quad f\left(z, y_i(x_i, y_j^{app}, z_i), y_j^{app}\right) \quad i, j = 1, \dots, n \; j \neq i \\ &\quad s.t. \; g(x_i, z, y_i(x_i, y_j^{app}, z_i), y_j^{app}) \leq 0 \end{aligned} \tag{3}$$

where $y_j^{app}$ represents the other discipline approximate state responses. After the subspace designers have developed improved designs, a complete multidisciplinary analysis will be performed again to evaluate each improved design point, which will be used to update the approximated models. The last step in each iteration is the system level optimization, which is conducted based on the discipline approximation models as follows:

$$\begin{aligned} &\min_{x, z} \quad f(x, z, y^{app}) \\ &s.t. \; g(x, z, y^{app}) \leq 0 \end{aligned} \tag{4}$$

The generated design points will also be used to update the approximation model and then the optimization enters the next iteration.

Compared to CO, the CSSO method provides multidisciplinary analysis feasibility at each cycle while CO allows consistency constraint violation at intermediate steps and tries to satisfy feasibility of design constraints. But, CSSO needs to deal with all the design variables at the system level optimization, which still be a high dimensional problem.

### *2.2.3 Bi-Level Integrated System Synthesis (BLISS)*

Like CSSO, BLISS begins with an initialized set of points and then a complete multidisciplinary analysis is performed to each design point. Based on these evaluated design points, BLISS can obtain the total derivatives through the global sensitivity equations method [18] to predict the effects of each set of variables on the objective function. Based on the derivatives, each discipline sub-problem is optimized by varying their local variables $x$, while holding the global variables $z$ constant and minimizing the disciplinary objective under local constraints, and the system level optimization only optimizes the global variables.

The optimization of the $i$-th discipline take the following form as:

$$\begin{aligned} &\min \ d(f,x_i)^T \Delta x_i \\ &s.t. \ g(x_i) \le 0 \end{aligned} \quad (5)$$

where $d(f,x_i)^T$ is the local total derivative of the objective function with respect to the local variables and disciplines, which includes the indirect effects of these variables on other disciplines. The system level objective takes the form:

$$\begin{aligned} &\min \ d(f,z)^T \Delta z \\ &s.t. \ g(z,y(x,z),x) \le 0 \end{aligned} \quad (6)$$

Once the new values for local and global variables are generated, the complete multidisciplinary analysis will be performed again and the total derivatives will be calculated. Then, the whole process will be iterated again.

### *2.3 Overlapping Issues in CE*

In CE, overlapping allows a downstream design activity to start earlier using some preliminary information of the upstream design activity. However, the preliminary information will evolve as the upstream design activity proceeds. This will cause longer duration or rework of the downstream design activity and thus longer development time [19-22]. To reduce the negative effect of overlapping, increased information exchange between design activities is required.

There are several well-known studies along this direction. In these studies, the change of the information of the upstream design is modelled as a function of time, while the reduction (or duration) of the downstream progress is modelled as a function of the change of the upstream information. Then a mathematical model can be built and the optimal solution to this model is the best information exchange strategy.

In [19], Krishnan et al. introduced the concepts of "upstream evolution" and "downstream sensitivity". By describing the exchanged information as a collection of design parameters, the notion of degree of upstream evolution is used to measure how close the design parameter is to its final value. The downstream sensitivity means the relationship between the duration of downstream iteration after information exchange and the magnitude of the change in the upstream information value. Using evolution and sensitivity, it was advocated that the best situation for overlapping occurs when upstream evolution is fast and downstream sensitivity is low. Also, a simple mathematical model was developed to decide the number and beginning time points of downstream iterations. This model was demonstrated on a door panel development [19]. In [20], Roemer et al. studied the trade-off between product development reductions and cost increases caused by overlapping of product design stages. An efficient algorithm was also proposed to determine an appropriate overlapping strategy. In [21], a performance generation model was proposed to manage coupled design activities subject to resource, performance, and deadline constraints. In [22], a system dynamics model was presented for managing overlapped iterative product development.

However, according to [23], frequent communication requires additional time and cost. Therefore, for the development project with costly communication, overlapping and communication policies should be coordinated to improve project performance.

In [24], an analytical model for overlapped upstream and downstream tasks is presented for minimizing time-to-market. In this model, preliminary information is modelled as engineering changes that happen to the upstream task. These changes are released to downstream tasks in batch form and impose rework on the downstream task. Based on the model, an optimal concurrency and communication policy was derived.

More recently, an analytical model was presented in [25] to determine the optimal overlapping and communication policies, with the goal of maximizing project performance. In this model, the arrival of upstream modifications was modelled as a nonhomogeneous Poisson process, and the reduction of the downstream progress caused by arriving modifications was modelled. Using this model, an appropriate overlapping level and the corresponding communication policy were derived.

Nonetheless, there is still a long way to go for researchers concentrating on overlapping management. The above-mentioned studies consider only two dependent activities, little work was devoted to multiple dependent activities. When using the above-mentioned models, it is required that the engineers are very familiar with the development project to reliably evaluate model parameters, which is almost impossible when starting a new project.

Except the afore-mentioned topics, other studies in CE that address some engineering issues also exist, including how to represent information and propagate information among different design teams or different CAE systems [26-28], and how to implement CE in a distributed environment [29].

## 3 Cooperative Coevolution (CC)

CC employs a divide-and-conquer strategy for tackling complex optimization problem. In CC, the original problem is usually decomposed into several sub-problems that can be solved concurrently. The idea of CC has been introduced into Evolutionary Algorithms (EAs) to solve high-dimensional problems and other complex optimization problems.

### *3.1 CCEA*

As a class of population-based global optimizers, Evolutionary Algorithms (EAs) have achieved great success in various real-world applications, such as music

composition [30], aircraft design [31], travelling salesman problems and flow shop scheduling problems [32], and image colour extraction [33] and so on. However, it has been shown that the performance of EAs deteriorates as the dimension of the problem increases [34]. To have better scalability to high-dimensional problems, one technique usually employed in EAs is Cooperative Coevolution (CC). Through CC, the decision variables of the problem is decomposed into several subcomponents. These subcomponents will be evolved in parallel in different subpopulations, each with a specified EA. Fitness evaluation for each subpopulation individual is carried out by combining it with representative individuals (usually the current best individuals) from the remaining subpopulations. This framework is called Cooperative Coevolutionary Algorithm (CCEA) [35].

An early work on CCEA was conducted by Potter and Jong in [2] to solve function optimization problems. In this paper, assuming a function optimization problem consisting of $n$ decision variables, a natural decomposition was used to decompose the $n$ decision variables of the problem into $n$ subcomponents, each with a decision variable. Then, a population of individuals for each subcomponent (called subpopulation) was generated and coevolved using a traditional Genetic Algorithm (GA). The fitness of a subpopulation individual is assessed by combining it with the current best subcomponents from the remaining subpopulations. Although promising experimental results have be achieved, it has been shown in [2] that such decomposition methods lost efficiency on non-separable problems. For non-separable problems, subsets of decision variables are usually interdependent and an ideal decomposition is to group those interacting variables into the same subcomponent. This calls for development effective methods to capture the interacting variables and group them into one subcomponent. Existing decomposition methods can be categorized into two types: random grouping and adaptive grouping.

### *3.1.1 Random Grouping*

In [36, 37], Yang et al. employed a random grouping scheme to decompose a high-dimensional vector with a predefined group size. In this method, the whole optimization process is decomposed into several cycles. At the beginning of each cycle, the random grouping method is repeated. In this random grouping method, every decision variable has the same probability to enter any subcomponent. It has been shown that through this scheme high probabilities can be obtained to group two interacting variables into the same component for at least one or two cycles. Experimental results in [37] also showed that the resulted algorithm outperformed previous CC techniques for large scale non-separable optimization problems. Furthermore, Yang et al. have introduced a self-adaptive mechanism in [38] to adapt the group size.

However, when using random grouping scheme, it has been proved in [34] that the probability of grouping interacting variables in one subcomponents will decrease quickly as the number of interacting variables increases. To address this issue, increased frequent random grouping of variables is required without increasing the total number of fitness evaluations. In order to maximize the frequency of random grouping, the subcomponent optimizers only ran for one iteration in [34]. That is, random grouping happens at every generation. This method resulted in faster convergence without sacrificing solution quality.

### *3.1.2 Adaptive Grouping*

In [39], a CCEA with Adaptive Variable Partitioning (CCEA-AVP) was proposed, which aims to automatically find out ideal decomposition while evolving each subpopulations. In CCEA-AVP, all variables are evolved in a single population for the first 5 generations. Then, the top 50% solutions of the population are used to compute the correlation coefficient between each pair of variables, and variables having a correlation coefficient greater than a specified threshold (0.6 in [39]) are partitioned into the same subpopulation. This correlation-based partition process will be repeated at every subsequent generation. Through correlation-based partition, the determination of the number of group size can be avoided, but the setting of the correlation coefficient threshold becomes a key issue of CCEA-AVP.

In [40], a new technique (called delta method) for capturing interacting variables was proposed. In this method, the amount of change (called delta value) in each of the decision variables in every iteration is measured. As the improvement interval will shrink significantly on a non-separable function, the delta method groups the variables with smaller delta values in one subcomponent at the expectation that variables interacting with each other will have smaller delta values. Experimental results in [38] showed that delta method is capable of grouping interacting variables.

## *3.2 Two-Level Coevolutionary Model*

In the framework of CCEA, the decision variables are decomposed into several subcomponents which are evolved separately in isolated subpopulations and fitness evaluation of each subpopulation individual is conducted by combining it with representative individuals from the remaining subcomponents. This class of divide-and-conquer methods in CC is called single-level coevolutionary methods [41]. Another class is called two-level coevolutionary methods [41]. In two-level coevolutionary methods, modules (i.e., sub-solutions) and systems (i.e., the complete solutions) are evolved in two separate populations. Each module should cooperate

with others in the module population to form a complete solution. Fitness of each individual in the module population is determined based on its contribution to various systems in the system population.

The Symbiotic Adaptive Neuro-Evolution (SANE) system [42] employed a two-level coevolutionary method to coevolve a population of neurons that cooperate to form a functioning neural network. Neurons in SADE play different but overlapping roles (i.e., functional components of neural networks). The task of SANE is to find a portion of neurons to form an effective functioning neural network by combining them while maintaining high levels of diversity in its final populations to face changes in the task environment. To carry out this task, SANE also evolve a population of network blueprints except the population of neurons and the relationship between the network blueprints population and neuron population is shown in Fig. 4. Each member of neuron population denotes a series of connection weights between the input layer and the output layer within a neural network, while each member of the blueprint population denotes a series of pointer to specific neurons that are used to form a two-layer feed forward neural network. The neuron population searches for good neurons (building blocks) for the neural network while the blueprint population searches for good combinations of neurons. Each population is evolved separately except fitness evaluation. The evaluation phase in SANE is as follows: each blueprint is used to select the corresponding neurons to form a neural network and receives the fitness of the resulting neural network, and each neuron receives the summed fitness of the best 5 networks in which it participated. It has been shown in [42] that SANE is more efficient and maintains higher levels of diversity than the common network-based population approaches.

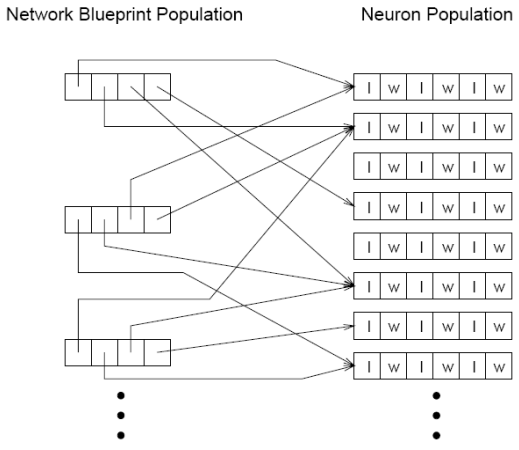



Fig. 4 An overview of the network blueprint population in relation to the neuron population, taken from [42]

In [41], a two-level co-evolutionary model was employed to co-evolve RBF Networks and Gaussian neurons in separate population. The main difference between this model and SANE is the use of RBF Networks instead of Multi-Layer Perceptrons (MLPs). Moreover, three different fitness evaluation strategies of neurons are compared in this paper.

In [43], a two-level co-evolutionary model was used to design and optimize modular neural networks with sub-task specific modules. Different modular structure denotes different decomposition of the task, which can be obtained by connecting each module to different set of inputs. As appropriate task decomposition is required for two-level co-evolutionary model, the structure of the module is allowed to change in [43] by adding or deleting an input to a module while simultaneously solving the sub-tasks. It is expected through this way the system can automatically discover natural decompositions of complex problems during the search process. Experimental results showed some good decompositions that one would not think of actually emerged. Furthermore, two-level co-evolutionary model was employed to obtain a neural network ensemble in [44], in which each module denotes a neural network and automatic decomposition is conducted by changing the structure of each neural network while optimizing them.

Except the above-mentioned work in CC, other studies include how to divide the computational budget in CCEA among all of the subcomponents [45], how to apply the random grouping method to large-scale capacitated arc routing problems [46-47], how to introduce CC into other EAs such as PSO [48].

## 4 Relationships between CE and CC

In this section, the similarity and differences between CC and CE are discussed. It can be seen that both CE and CC employ decomposition methods. CE advocates concurrent design of the product and related processes. Thus, decomposition is employed to explore the concurrency aspect of the design to enable concurrent design. In CC, high dimensional problems are decomposed into several lower dimensional problems that can be optimized separately. Also, from the perspective of the whole framework, the framework of CC is similar to that of the BLISS method in MDO as each of CC and BLISS optimizes each subpopulation with other components fixed. Differently, problem decomposition in CE is known a priori. That is, the decomposition is directly based on the structure of the product, human experts or domain analysis and keeps unchanged during the optimization process. However, in CC, which variables are interrelated is not known a priori. Thus, different methods to capture variable interaction during the search process were proposed to boost the efficiency of CC.

However, CE may still face a high-dimensional problem after decomposition. In this case, CC can be employed as an optimizer for solving large-scale problems in CE. Further, in some real-world application of CE, which design variables are related to each discipline may not be known a priori. In this situation,

the methods employed in CC to capture interrelated variables may be exploited. For MDO, different sub-tasks have common variables in real-world applications and different methods have been developed to manage them. In contrast, for CC, the generated subcomponents through decomposition do not have any common variable in existing studies. Assuming we have a two-objective optimization problems, these two objective functions have a set of common variables and a large set of their separate variables. How to decompose and optimize such a problem in the framework of CC arises a new problem for CC. New CC framework with overlapped subcomponents needs to be developed along this direction. Moreover, the MDO methods may be exploited to manage the common variables between different subcomponents.

## 5 Conclusion

In this paper, two concurrent optimization topics, CE and CC, are reviewed and their relationships are discussed. Potential combinations between CE and CC exist, including employing the grouping methods in CC to find common or uncommon variables among different disciplines and developing new CC framework with overlapped subcomponents to address some specific problems.

## 6 Acknowledgement

This work was supported in part by the 973 Program of China under Grant 2011CB707006, the National Natural Science Foundation of China under Grants 61175065 and 61329302, the Program for New Century Excellent Talents in University under Grant NCET-12-0512, the Science and Technological Fund of Anhui Province for Outstanding Youth under Grant 1108085J16, the Honda Research Institute Europe GmbH, and the European Union Seventh Framework Programme under Grant 247619.

## 7 References

[1]A. Kusiak, "Concurrent Engineering: Automation, Tools, and Techniques", John Wiley & Sons, Inc. New York, New York, USA, 1993.

[2]M. Potter, and K. De Jong, "A cooperative coevolutionary approach to function optimization", *Parallel Problem Solving from Nature PPSN III*, Springer, pp. 249-257, 1994.

[3]R. Addo-Tenkorang, "Concurrent Engineering (CE): A Review Literature Report", in *Proceedings of the World Congress on Engineering and Computer Science*, San Francisco, 19-21 October 2011, pp. 1074-1080.

[4]R. I. Winner, J. P. Pennell, H. E. Bertrand, and M. M. G. Slusarczuk, "The Role of Concurrent Engineering in Weapon System Acquisition", IDA Report R-338, Institute for Defense Analyses, Alexandra VA, 1988.

[5]D. E. Carter, and B. S. Baker, "Concurrent Engineering: The Product Development Environment for the 1990s", Addison-Wesley Publishing Company, Reading, Massachusetts, 1992.

[6]J. L. Turino, "Managing Concurrent Engineering: Buying Time to a Market: A Definitive Guide to Improved Competitiveness in Electronics Design and Manufacturing", Van Nostrand Reinhold, New York, 1992.

[7]H. R. Parsaei, and W. G. Sullivan, "Concurrent Engineering: Contemporary Issues and Modern Design Tools", Chapman & Hall, London, 1993.

[8]H. C. Zhang, D. Zhang, "Concurrent Engineering: An Overview from Manufacturing Engineering Perspectives", *Concurr. Eng. Res. Applications Int. J.*, 3(3):221-236, 1995.

[9]A. A. Yassine, R. S. Sreenivas, J. Zhu, "Managing the exchange of information in product development", *European Journal of Operational Research*, 184(1):311-326, 2008.

[10]A. Kusiak, and N. Larson, "Decomposition and Representation Methods in Mechanical Design", *Transactions of the ASME Journal of Mechanical Design*, 117:17-24, 1995.

[11]A. Kusiak, and K. Park, "Concurrent Engineering: Decomposition of Design Activities", in *Proceedings of Rensselaer's 2nd International Conference on Computer Integrated Manufacturing*, pp. 557-563, 1990.

[12]J. T. Allison, "Optimal Partitioning and Coordination Decisions in Decomposition-based Design Optimization", *Desertion for the Degree of Doctor of Philosophy*, 2008.

[13]S. Kodiyalam, and J. S-Sobieski, "Multidisciplinary Design Optimization-Some Formal Methods, Framework Requirements, and Application to Vehicle Design", *International Journal of Vehicle Design*, 25(1):3-22, 2001.

[14]R. Braun, P. Gage, I. Kroo, and J. Sobieszczanski-Sobieski, "Implementation and Performance Issues in Collaborative Optimization", *in Proceedings 5th AIAA/USAF MDO symposium*, AIAA pp. 96-4017, Bellevue, WA, Sept. 1996.

[15]R.S. Sell, S. M. Batill, and J. E. Renaud, "Response Surface Based Concurrent Subspace Optimization for Multidisciplinary System Design", in *Proceedings of the 34th AIAA Aerospace Sciences Meeting*, AIAA 96-0714, Reno, Nevada, 1996.

[16]J. Sobieszczanski-Sobieski, S. A. Jeremy, and Jr. R. R. Sandusky, "Bi-Level Integrated System Synthesis (BLISS)", *National Aeronautics and Space Administration Technical Manual* NASA/TM-1998-208715, August 1998.

[17]R. E. Perez, and H. T. T. Liu, and K. Behdinan, "Evaluation of multidisciplinary optimization approaches for aircraft conceptual design", in *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Albany, NY, 30 August-1 September, 2004.

[18]J. Sobieszczanski-Sobieski, “Sensitivity of Complex, Internally Coupled Systems”, *AIAA Journal*, 28(1):153-160, 1990.
[19]V. Krishnan, S. D. Eppinger, and D. E. Whitney, “A model-based framework to overlap product development activities”, *Management Science*, 43(4):437-451, 1997.
[20]T. A. Roemer, R. Ahmadi R, and R. H. Wang, “Time-cost trade-offs in overlapped product development”, *Operations Research*, 48(6): 858-865, 2000.
[21]N. R. Joglekar, A. A. Yassine, S. D. Eppinger, and D. E. Whitney, “Performance of coupled product development activities with a deadline”, *Management Science*, 47(12):1605-1620, 2001.
[22] J. Lin, K. H. Chai, Y. S. Wong, and A. C. Brombacher, “A dynamic model for managing overlapped iterative product development”, *European Journal of Operational Research*, 185(1):378-392, 2008.
[23]K. R. Haberle, R. J. Burke, R. J. Graves, “A note on measuring parallelism in concurrent engineering”, *International Journal of Production Research*, 38(8):1947-1952, 2000.
[24]C. H. Loch, and C. Terwiesch, “Communication and uncertainty in concurrent engineering”, *Management Science*, 44(8):1032-1048, 1998.
[25]J. Lin, Y. Qian, W. Cui, and Z. Miao, “Overlapping and Communication policies in product development”, *Production, Manufacturing and Logistics*, 201:737-750, 2010.
[26]J. G. McGuire, D. R. Kuokka, J. C. Weber, J. M. Tenenbaum, T. R. Gruber, and G. R. Olsen, “SHADE: Technology for Knowledge-based Collaborative Engineering”, *Concurrent Engineering*, 1(3):137-146, 1993.
[27]S. A. Kelsey, and A. S. Larry, “A Representation for Design Information during the Product Definition Process”, *Concurrent Engineering*, 3(2):107-111, 1995.
[28]J. Hwang, D. Mun, S. Han, “Representation and Propagation of Engineering Change Information in Collaborative Product Development using a Neutral Reference Model”, *Concurrent Engineering*, 17(2):147-157, 2009.
[29]B. A. Wujek, J. E. Renaud, S. M. Batill, and J. B. Brockman, “Concurrent Subspace Optimization Using Design Variable Sharing in a Distributed Computing Environment”, *Concurrent Engineering*, 4(4):361-377, 1996.
[30]C. Johnson, and J. Cardalda, “Genetic algorithms in visual art and music”, *Leonardo*, 35(2):175–184, 2002.
[31]D. Simon, “Biogeography-based optimization,” IEEE Transaction on Evolutionary Computation, 12(6):702–713, 2008.
[32]G. Schaefer, and L. Nolle, "Optimal image colour extraction by differential evolution", *International Journal of Bio-Inspired Computation*, 2(3/4):251-157, 2010.
[33]A. Chowdhury, A. Ghosh, S. Sinha, S. Das, and A. Ghosh, “A novel genetic algorithm to solve travelling salesman problem and blocking flow shop scheduling problem”, *International Journal of Bio-Inspired Computation*, 5(5):303-314, 2013.
[34]M. N. Omidvar, X. Li, X. Yao and Z. Yang, “Cooperative Co-evolution for Large Scale Optimization Through More Frequent Random Grouping,” *Proc. of the 2010 IEEE Congress on Evolutionary Computation (CEC2010)*, Barcelona, Spain, 18-23 July 2010. pp. 1754-1761.
[35]R. P. Wiegand, “An Analysis of Cooperative Coevolutionary Algorithms”, *PhD Dissertion*, George Mason University, Fairfax, VA, 2004.
[36]Z. Yang, K. Tang, and X. Yao, “Differential Evolution for High-Dimensional Function Optimization,” in *Proceedings of the IEEE 2007 Conference on Evolutionary Computation,* pp. 3523-3530, 2007.
[37]Z. Yang, K. Tang, and X. Yao, “Large Scale Evolutionary Optimization Using Cooperative Coevolution”, *Information Sciences*, 178:2985-2999, 2008.
[38]Z. Yang, K. Tang, and X. Yao, “Multilevel Cooperative Coevolution for Large Scale Optimization,” in *Proceedings of the IEEE 2008 Conference on Evolutionary Computation (CEC’08)*, pp. 1663-1670, 2008.
[39]T. Ray, X. Yao, “A Cooperative Coevolutionary Algorithm with Correlation Based on Adaptive Variable Partitioning”, in *Proceedings of the IEEE 2007 Conference on Evolutionary Computation,* pp.983-989, 2009.
[40] M. N. Omidvar, X. Li and X. Yao, “Cooperative Co-evolution with Delta Grouping for Large Scale Non-separable Function Optimization”, in *Proceedings of the 2010 IEEE Congress on Evolutionary Computation (CEC2010)*, Barcelona, Spain, 18-23 July 2010. pp. 1762-1769.
[41]V. R. Khare, X. Yao and B. Sendhoff, “Credit Assignment among Neurons in Co-evolving Populations”, in *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, Lecture Notes in Computer Science, Vol. 3242, pp.882-891, Springer, September 2004.
[42]D. E. Moriarty, and R. Miikkulainen, “Forming Neural Networks through Efficient and Adaptive Coevolution”, *Evolutionary Computation*, 5(4):373-399, 1997.
[43]V. R. Khare, X. Yao, B. Sendhoff, Y. Jin and H. Wersing, “Co-evolutionary modular neural networks for automatic problem decomposition”, in *Proceedings of the 2005 Congress on Evolutionary Computation (CEC’05)*, pp.2691-2698.
[44]V. R. Khare, X. Yao and B. Sendhoff, “Multi-network evolutionary systems and automatic problem decomposition”, *International Journal of General Systems*, 35(3):259-274, June 2006.
[45]M. N. Omidvar, X. Li and X. Yao, “Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms”, in *Proceedings*

*of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO 2011)*, pp.1115-1122, 12-16 July 2011, Dublin, Ireland, ACM Press, New York, NY, USA.
[46] Yi Mei, "Decomposing Large-Scale Capacitated Arc Routing Problems using a Random Route Grouping Method", *Proc. of the 2013 IEEE Congress on Evolutionary Computation (CEC2013)*, pp. 1013-1020.
[47] Yi. Mei, X. Li and X. Yao, "Cooperative Co-evolution with Route Distance Grouping for Large-Scale Capacitated Arc Routing Problems", *IEEE Transactions on Evolutionary Computation*, accepted on 31 July 2013.
[48] Xiaodong Li, and Xin Yao, "Cooperative Coevolving Particle Swarms for Large Scale Optimization", *IEEE Transaction on Evolutionary Computation*, 16(2):210-224, 2012.