Representative Benchmark for Concurrent Product and Process Configuration Problem: Definitions and Some Problem Instances

Luis Garcés Monge, Paul Pitiot, Élise Vareilles, Michel Aldanondo

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 52-13 (2019) 301–306

Representative Benchmark for Concurrent Product and Process Configuration Problem: Definitions and Some Problem Instances

Luis Garcés Monge[1, 3], Paul Pitiot[1, 2], Elise Vareilles[1], Michel Aldanondo[1],
[1]Université de Toulouse, IMT Mines Albi, Centre de Génie Industriel, Albi, France
[2] 3IL-CCI Rodez, France
[3]Instituto Tecnológico de Costa Rica, Cartago, Costa Rica

(lgarcesm@mines-albi.fr, paul.pitiot@mines-albi.fr, vareille@mines-albi.fr, aldanond@mines-albi.fr)

Abstract: This paper considers the Optimization of Concurrent Product and Process Configuration problems (O-CPPC) that satisfy various number of criteria, which rely on the customer's requirements and the objectives of the company. Various works have proposed evolutionary optimization algorithms dedicated to this concurrent configuration problem with generic model propositions due to this paper is relevant to the evaluation of these optimization algorithms. The aim of this paper is to define a set of instances of the generic model that represent a large family of problems. First, a background of the Optimization of Concurrent Product and Process Configuration problems is introduced. Next, some basic definitions of an O-CPPC generic model are analyzed. Then, the main general parameters to define an instance are presented (Product Structure, Process Structure, Model Size and Model Constraint Density) in order to propose some general evaluation tests. And finally, to be consistent with the previous works, some basic cases are described to show how to deal with this kind of problem in an organized way.

Keywords: instance, evaluation, generic model, product configuration, process configuration, concurrent configuration, optimization algorithms

1. INTRODUCTION

In the manufacturing industry, the concept of mass customization has established itself as an indispensable lever for gaining market share. The aim is to offer a high level of diversity of product or service to the customer while maintaining a good level of productivity (Pine (1993)). In order to develop and implement mass customization, many companies use configuration software (Felfernig et al (2014)), that enables companies to propose to their customers customized products from a huge variants and options of products (Wang et al. (2015)). Configuration software also allows to define the realization process of products by selecting project alternatives and its resources. In most configuration problems, the configured product / process must satisfy a certain number of criteria which rely on the customer's requirements (price, delivery time), and the objectives of the company (carbon footprint, production cost). This results in a multi-criteria decision making problem or in a multi-criteria optimization problem. The interest of this work is located on the Optimization of the Concurrent Product and Process Configuration (CPPC). Especially to finalize and evaluate metaheuristics published on the subject trying to avoid case dependency. It is stated that evaluation should not be done on a specific problem but on a family of problems provided by a configuration problem generator. The first step has been to define a benchmark for CPPC optimization problems (Pitiot et al. (2016)). A generic model of the concurrent problem was defined in order to create the selection of varied model instances to get the benchmark. This article focuses on the analysis of the main parameters required to create these kind of instances to evaluate optimization algorithms. The remaining of the document is organized as follows. In the second section a background of the optimization of concurrent product and process configuration problems are introduced. In the third section, a summary of basic definitions of an O-CPPC generic model is presented. Finally, in fourth section, the basic parameters for an instance of O-CPPC problem and some basic cases are described.

2. CONTEXTUALIZATION OF THE OPTIMIZATION OF CONCURRENT PRODUCT AND PROCESS CONFIGURATION PROBLEMS

The Concurrent Product and Process Configuration (CPPC) is the task of configuring a product and its related process at the same time. In order to do this task, it is necessary to link configurable product representation (product family) with configurable process representation (process family). Most of the academic works deal with the two domains in an independent way, either product configuration or process configuration. Some studies have shown the interest of the union of the two configuration domains like (Baxter (2007); Zhang et al. (2013); Hong et al. (2010); Li et al. (2006); Huang and Gu (2006)). CPPC is an interesting and increasingly studied industrial problem, since the idea is to

2405-8963 © 2019, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2019.11.192

avoid the sequential classical decision of configuring the product first and then the process by taking decision considering the two domains at the same time. The Optimization of Concurrent Product and Process Configuration (O-CPPC) is the task of combining the optimization with the configuration of a product and its related production process, in order to meet technical and particular customer requirements. Very few works consider the concurrence product / process with optimization directly. Likewise, the existing works dealing with the sequential association of Optimization plus Concurrent Product and Process Configuration are linked either with studies relevant to the mass customization business process like (Forza and Salvatore, (2002)) and (Hvam et al. (2002)), or with studies that associate interactive configuration and autonomous configuration completion (Amilhastre et al. (2002)) and (Ullman (2007)). The Optimization of Concurrent Product and Process Configuration (O-CPPC) problem is focused in configurable products with its respective process. A first effort combining the Optimization with Concurrent Product and Process Configuration was made by (Pitiot et al. (2013)), who presented a first efficient tool that was able to assist product configuration and process configuration concurrently, using an interactive constraint filtering system and an evolutionary optimization system.

3. BASIC DEFINITIONS OF A GENERIC MODEL OF O-CPPC PROBLEM

The goal of this work is to define a benchmark for O-CPPC optimization problems. A benchmark is a set of model instances of a specific optimization problem and which allows testing optimization algorithms and validation of their accuracy for the addressed problem. A generic model of O-CPPC problems was defined to create the selection of varied model instances to get the benchmark. Each part of the model (product configuration, process configuration and their coupling) is described using optimization constraint satisfaction problem (O-CSP) paradigm. That means that the problem is defined by a quadruplet <V,D,C,f> with "V" the set of variables, "D" the set of domains linked to each variable, "C" a set of configuration constraints that correspond to compatibilities between values of variables; and "f" a specific set of constraints, called evaluation constraints, that allow to calculate multivalued objective functions. A subset of "V" might also been identified as the set of decision variables, named Vd. In a decision aiding problem, Vd corresponds to the set of variables on which the stakeholder can act. Each decision variable is related to one or more objectives. Decision variables are discrete (numeric or symbolic), and product and process configuration corresponds to the selection of a value to set decision variables. The aim of O-CSP is to find the setting of decision variables that will maximize/minimize objectives. In this study, other variables of "V" will be called evaluation variables since allow to calculate the value of the objectives. The goal is now able to generate various instances of the O-CPPC problems that represent diversity and complexity of real industrial cases. That is why the main definitions of a generic model of O-CPPC are summarized. From the previous proposal, new concepts are added like product architecture, types of product architecture, assembly line and the typology of process structure in order to be more accurate in the definition of the instances. Being consistent with the previous work (Pitiot et al. (2016)), the summary of the O-CPPC problem is decomposed in four parts: Product Domain, Process Domain, Configuration Constraints and Evaluation Constraints.

3.1 Product Domain

In this section, the basic definitions of the product configuration domain are presented.

3.1.1 Product Definition

Product or System gathers a set of physical-functional modules in a one level decomposition.

3.1.2 Module Definition

A physical-functional module is a subset of a product that corresponds to a set of components which fulfills some functions of the product. Therefore, a physical-functional module is described in three parts: (1) a family of component (module_i_foc_j) that is a set of components that can fulfill some required functions in module i. (the family of Components are discrete decision variable in Vd); (2) a function is a fulfillment of a customer's requirement over various discrete functional levels; and (3) a functional description variable (module_i_fdv_j) that refers to a description of a function and is a set of possible functional level for a function in module i.

3.1.3 Product Architecture Definition

(Ulrich (1995)) defined the architecture of a product as "the scheme by which the function of the product is allocated to physical components". More precisely, the author defined product architecture as: (1) the arrangement of functional elements; (2) the mapping from functional elements to physical components; (3) the specification of the interfaces among interacting physical components. Other definition was presented by (Ulrich and Eppinger (2012)) as a "scheme by which the functional elements of the product are arranged (or assigned) into physical building blocks (chunks) and by which the blocks interact". Therefore, the arrangement of functional elements into physical chunks which become the building blocks for the product or family of products (Ulrich and Eppinger (2012)).

3.1.4 Typology of Product Architecture

For the Optimization of Concurrent Product and Process Configuration problem (O-CPPC), there are three basic types of product architecture:

a) Modular Architecture

The first distinction in the typology is between a modular architecture and an integral architecture. (Marti (2007)) and (Göpfert (1998)) characterized a modular system architecture by the property of near-decomposability, consisting of relatively autonomous subsystems. Therefore, in this architecture a module can be defined as "a special subsystem whose internal relationships are much stronger than the relationships with other subsystems" (Marti (2007); Göpfert (1998)). In addition, (Ulrich (1995)) emphasizes that a modular architecture includes a one-to-one mapping from functional elements in the function structure to the physical components of the product and specifies decoupled interfaces between components. More specifically, (Blackstone (2013)) explained that the modular architecture is a type of structure where functional modules correspond to a physical group of parts. The different physical pieces of parts have their own function, and there is an interaction between all modules (Blackstone (2013)).

b) Integral Architecture

(Ulrich (1995)) explained that an integral architecture includes a complex (not one-to-one) mapping from functional elements to physical components and coupled interfaces between components. In this type of architecture, the relationships among subsystems are more pronounced. As a result, the modules are more dependent on each other and less easily distinguished (Marti (2007); Göpfert (1998)). Due to the integral architecture is a type of modular architecture with strong relations between modules.

c) Platform Architecture

An increasingly popular method to reduce complexity in products is the product platform architecture. (Marti (2007)) and (Schuh and Schwenk (2001)) explained that the product platform is a special case of product modularization. The objective of modularization is decomposing a product into modules. Therefore, to define modules while establishing a platform means structuring the product's architecture according to a certain hierarchy (Marti (2007); Hofer (2001)). Essentially, this architecture divides the product architecture into a standardized part (the platform) and customized modules. (Blackstone (2013)) explained that in the platform architecture, there is a grouping of products to share common parts, components and characteristics (common platform). This kind of design can be used to reduce cost and time to market (Blackstone (2013)). Finally, (Robertson and Ulrich (1998)) defined a product platform in a concurrent way as "the collection of assets that are shared by a set of products", not confining it to the common physical structure shared across products. Therefore, these assets fall into one of the following four categories: components, processes, knowledge, and people/relationships (Robertson and Ulrich (1998)).

3.1.5 Definition of Configuration/Evaluation Pattern in Product Domain (Tpcep)

Coming from our experience on product configuration, a set of common generic sets of variables and constraints called Product Configuration/Evaluation Pattern (PCEP) were identified. Each PCEP gathers a set of decision variables [family of component (foc) and/or functional description variable (fdv)] and objective variables [selling price (sp)] linked by constraints configuration and evaluation constraints.

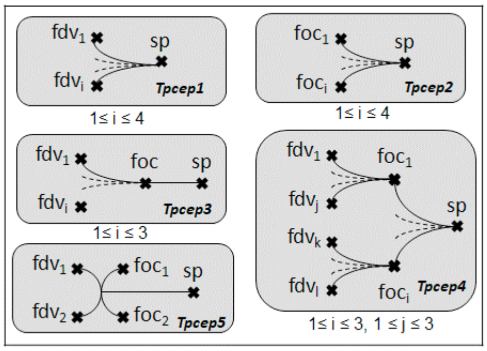

Fig. 1. Various types of PCEP (Tpcep)

The quantity and the type of Configuration/Evaluation Pattern in Product Domain is related with the structure of the specific product. For example, in Tpcep1 price is driven only by functional description variables, while at the opposite in Tpcep2 only family of component variables drive product price.

3.2 Process Domain

In the following section, basic definitions related to process configuration domain are presented.

3.2.1 Process Definition

A process is a sequence of operations that leads to obtain relevant product.

3.2.2 Operation Definition

An operation is a step of production process that corresponds to specified workload to be achieved using a specified type and quantity of resource. It has a duration ensuing from the choice of type and quantity of selected resources. Furthermore, it is important to model the temporal placement of each operation. In a constraint model, it is required three continuous variables for each operation i: starting date (operation_i_start), ending date (operation_i_end) and duration (operation_i_duration).

3.2.3 Resource Family Definition

Resource family is the set of resources that can achieve relevant works from a specific operation. To achieve an operation, the stakeholder could also act on the quantity of selected resources. For this reason, for each operation i, the generic model includes a discrete variable named quantity of resource (operation_i_qtr).

3.2.4 Definition of Assembly Line

The common method of production or manufacturing process for the configurable products are the assembly lines. As mentionetd by [Grzechca, 2011] an assembly line is "a manufacturing process in which parts are added as the semi-finished assembly that moves from wockstation to workstation where the parts are added in sequence until the final assembly is produced" (discrete process).

3.2.5 Typology of process structures

For our CPPC problem, the structure of the assembly process could be a "Serial" or "Converge" case. A Serial Case is a simple process in which operations or steps are executed in a strictly serial way. This means that one operation of the process finishes before the next starts, and only one step is active at any one instant. A Converge Case is a variation of the serial configuration structure where we can find two or more operations running at the same time and then converging to another main operation.

3.3 Configuration Constraints

Next, basic definitions related with Configuration Constraints are described.

3.3.1 Configuration Constraint Definition

Configuration constraint describes compatibility between values of a set of decision variables. As decision variables are discrete, it can correspond to a compatibility table.

3.3.2 Typology of Configuration Constraint

Configuration constraints take place in different location of the model. The configuration constraint is listed on table 1 and is illustrated with an example in figure 2.

Table 1. Example of Type of Configuration Constraints,

Type	Quantity	Constraint Density
——— intra-PCEP constraint	3	High
═══ intra-module constraint	2	High
— · — inter-module constraint	2	Medium
═══ coupling constraint	3	Medium
——— inter-operation constraint	2	Medium

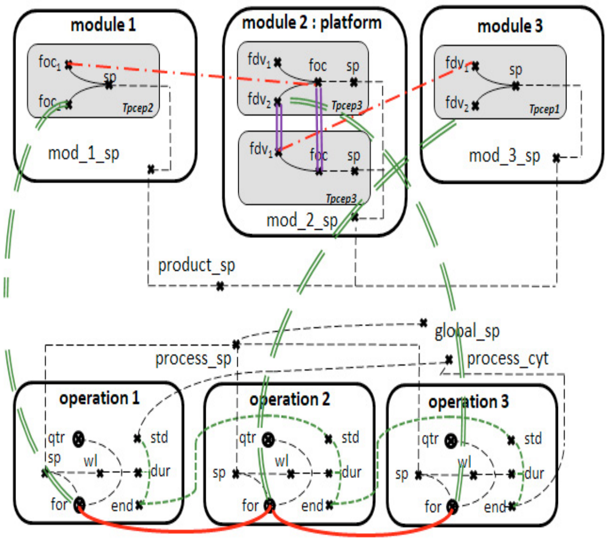

Fig. 2. Example with Configuration Constraint

Then, the quantity of evaluation constraints is calculated according to the number of operations that the model has. Table 2 shows the quantity of evaluations constraints for the example.

Table 2. Example of Evaluation Constraints

Type	Quantity
· — — — Evaluation (sp+duration)	16
- - - - - - Temporal	5

3.3.3 Definition of Configuration Constraint Pattern (Tcp)

There were identified various configuration shapes or behavior named type of configuration pattern (Tcp) presented on fig. 3.

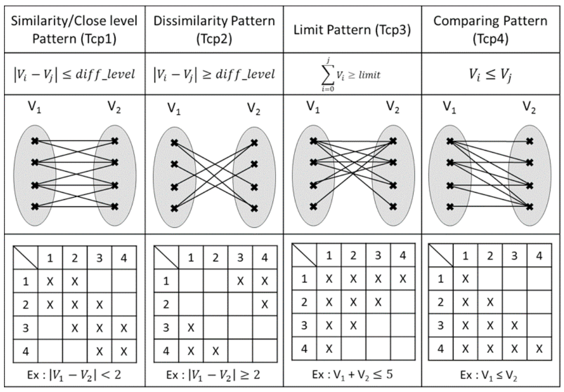
Similarity/Close level Pattern (Tcp1)
Dissimilarity Pattern (Tcp2)
Limit Pattern (Tcp3)
Comparing Pattern (Tcp4)
$|V_i - V_j| \leq diff\_level$
$|V_i - V_j| \geq diff\_level$
$\sum_{i=0}^{j} V_i \geq limit$
$V_i \leq V_j$
V_1
V_2
Ex : $|V_1 - V_2| < 2$
Ex : $|V_1 - V_2| \geq 2$
Ex : $V_1 + V_2 \leq 5$
Ex : $V_1 \leq V_2$

Fig. 3. Types of configuration constraint pattern

Most of these patterns assume that the values of variables were ordered in terms of performance or abilities. Examples on fig. 3, show configuration constraints between two variable but these could be extended to three or four variables. Compatibility tables are illustrated by compatibility matrix, where a cross in the matrix corresponds to a compatibility of a couple of values.

4. DEFINITION OF A SET OF INSTANCE OF OPTIMIZATION OF CONCURRENT PRODUCT AND PROCESS CONFIGURATION (O-CPPC)

For the definition of the instances, the basic parameters necessary to define a case and the different evaluation tests are presented.

4.1 Basic Parameters for an instance of O-CPPC problems

To define the cases to be evaluated, the following variables were considered:

4.1.1 Product Structure

As it was presented in section 3.1.4, the three typical product architectures will be used: a) Platform, b) Modular and c) Integrated. The idea is to evaluate if there are significant differences between the different product typologies.

4.1.2 Model Size

The idea is to evaluate the size of the concurrent model. It was considered the premise that the size of the model is related to the complexity of the problem. For example, the number of variables is related to the complexity of the problem. More variables we have the complexity of the problem is bigger. Likewise the variables, the quantity of the other parameters (modules, operations and constraints) are also related to the complexity of the problem. More quantity more complexity for the problem is faced. Therefore, the size of the model is related to the quantity of variables, the quantity of modules, the quantity of operations in the process, and the quantity of configuration constraints in the model. For this reason, there were defined four sizes of models:

Table 3. Model Sizes

Size	Quantity of Variables	Quantity of Modules	Quantity of Operations	Quantity of Configuration Constraints
Small	15	3	3	12
Medium	30	3	3	26
Intermediate	60	7	7	51
Large	100	10	10	82

4.1.3 Model Constraint Density

The constraint density for one constraint corresponds to the ratio of allowed tuples over every possible tuple. It coincides to the ratio of crosses in the compatibility matrix. Three levels of constraint density were defined: (1) low, (2) medium and (3) high. The low level corresponds to a 80% of ratio of allowed tuples, medium a level of 50% and high a level of 20%. The objective is to evaluate if there are differences between the three constraint density levels.

4.1.4 Process Configuration

In the evaluation tests, only serial cases for the process configuration will be considered, where the production process is a sequence of operations with no parallel or simultaneous operations. This assumption is made for simplicity only, and would be no problem to model simultaneous operations with precedence constraints (converge case).

4.2 Some examples of instances of O-CPPC problems

Once the product structure, the problem size, the model constraint density and the process configuration has been defined, the next step is to design the diagram of the model that will be a guide for programming. According to the standard (section 3.3.2), at the top, it is shown the product domain with the detail of the modules (Variables, Type of Configuration/Evaluation Pattern, and Configuration Constraints). Then at the bottom, it is presented the detail of the process domain (the sequence steps of the production process, the variables related to time and the variables related to the resources). Finally, the evaluation constraints related with time and cost and the coupling constraints that connect the two domains is presented. To exemplify this, three problem cases are described, each one with a different product architecture and maintaining the same size (medium) and the same constraint density (medium). Fig. 4 shows the first basic example with three modules. There are only constraints between the platform (module 2) and the other modules (inter-module constraints). The internal constraints (intra-module and intra-PCEP constraints) are stronger than the constraints between modules.

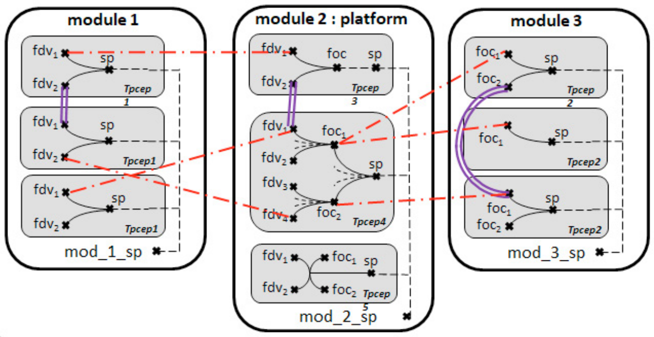

Fig. 4. Example of Platform Structure

Then fig. 5 shows an example of MODULAR_MEDIUM_MEDIUM. The only difference is that there are inter-module constraints between any pair of modules. These constraints have low density while intra-module constraints have high density.

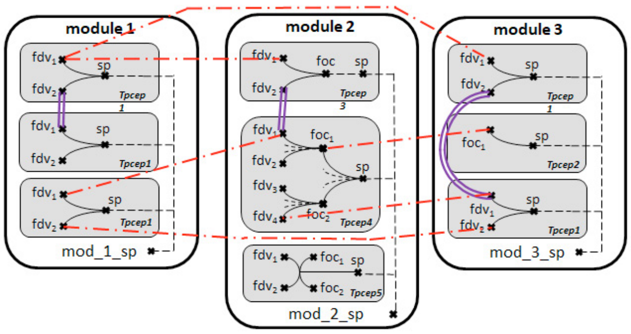

Fig. 5. Example of Modular Structure

Finally, fig. 6 illustrates the example of INTEGRATED_MEDIUM_MEDIUM. This model is similar to the previous one except by the constraint densities that are opposite. Inter modules constraints are high density and intra-module ones are low.

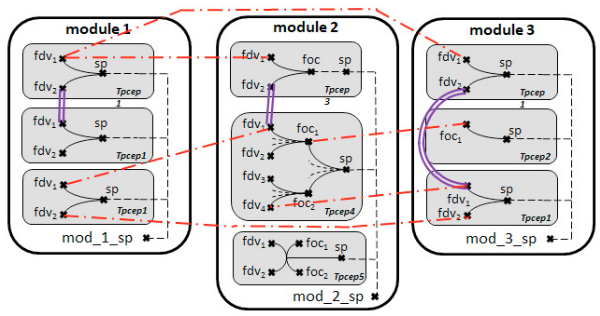

Fig. 6. Example of Product Integral Structure

CONCLUSIONS

The goal of this paper was to extend our research perspectives on Concurrent Product and Process Configuration problems. There is no standard problem to analyse optimization algorithms. Thus the existing works dealing with the optimization of Concurrent Product and Process Configuration (CPPC) are theoretical proposals, in the best cases, an evaluation with a single problem was found. Coming from our experience in the topic, the main parameters necessary to define a set of instances to evaluate CPPC optimization algorithms are analysed. Next works will be concerned by the evaluation of classical optimization algorithms with respect to these parameters that will avoid case dependency.

REFERENCES

Amilhastre, J., Fargier, H., and Marquis, P. (2002). Consistency restoration and explanations in dynamics csps-application to configuration. Artificial Intelligence, 135 (1-2), 199-234.

Baxter, D. (2007). An engineering design knowledge reuse methodology using process modelling. Research in Engineering Design, 18 (1), 37-48.

Blackstone, J. (2013). APICS Dictionary. Fourteen Edition. APICS.

Felfernig, A., Hotz L., Bagley C. and Tiihonen J. (2014) Knowledge-based Configuration: From Research to Business Cases. Elsevier/Morgan Kaufmann.

Forza, C., and Salvador, F. (2002). Managing for variety in the order acquisition and fullfiment process: the contribution of product configuration systems. International Journal of Production Economics, 76 (1), 87-89.

Göpfert, J. (1998). Modulare Produktentwicklung: Zur gemeinsamen Gestaltung von Technik und Organisation. Wiesbaden: Deutscher Universitäts-Verlag.

Grzechca, W. (2011). Assembly Line –Theory and Practice. InTech Open Access Publisher. Croatia

Hong, G., Xue, D., and Tu, Y. (2010). Rapid identification of the optimal product configuration and its parameters based on customer-centric product modeling for one-of-a-kind production, in: Computers in Industry Vol.61 n°3, 270–279.

Hofer, A.P. (2001). Management von Produktfamilien: Wettbewerbsvorteile durch Plattformen (Doctoral dissertation, University of St. Gallen, 2001). University of St. Gallen.

Huang, H. Z., and Gu, Y.K. (2006). Development mode based on integration of product models and process models. Concurrent Engineering: Research and Applications. 14, 1. 27-34.

Hvam, L., Riis, J. and Malis, M. (2002). A multiperspective approoach for the design of comfiguration systems. In: ECAI configuration worshops proceedings. Lyon France, 56-62.

Li, L., Chen, L., Huang, Z. and Zhong, Y. (2006). Product configuration optimization using a multiobjective GA, in: I.J. of Adv. Manufacturing Technology vol. 30, 20-29.

Marti, M. (2007). Complexity Management. Optimizing Product Architecture of Industrial Products. Deutscher Universitäts-Verlag.

Pine, B.J, (1993). Mass Customization: The New Frontier in Business Competition. Harvard Business School Press, Boston, MA.

Pitiot, P., Garcés, L., Aldanondo, M. and Vareilles, E. (2016). Benchmark for configuration and planning optimization problems: Proposition of a generic model. Proceedings of the 18Th International Configuration Workshop whitin CP 2016 Conference. Toulouse France. 89–96.

Pitiot, P., Aldanondo, M., Vareilles, E., Gaborit, P., Djefel, M. and Carbonnel, S. (2013). Concurrent product configuration and process planning, towards an approach combining interactivity and optimality, in: I.J. of Production Re-search Vol. 51 n°2, 524-541.

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

Schuh, G., and Schwenk, U. (2001). Produktkomplexität managen: Strategien, Methoden, Tools. München: Hanser.

Ullman, J.R. (2007). Partition search for non-binary constraint satisfaction. Information Sciences, 177 (18), 3639-3678.

Ulrich, K.T., and Eppinger, S.D. (2012). Product Design and Development. Chapter 10, 5th Edition, Irwin McGraw-Hill.

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. Research Policy 24. Elsevier Science B.V.

Wang, L., Sheng Zhong, S., and Jian Zhang, Y. (2015). Process Configuration based on generative constraint satisfaction problem. Journal of Intelligence Manufacturing. Volume 28, Issue 4, 945-952. (2015)

Zhang, L., Vareilles, E. and Aldanondo, M. (2013). Generic bill of functions, materials, and operations for SAP2 configuration, International Journal of Production Research, Vol. 51, n°2, 465-478.