Concurrent configuration and planning problems: Some optimization experimental results

Paul Pitiot[1], Michel Aldanondo[1], Elise Vareilles[1], Paul Gaborit[1]

Abstract. This communication deals with mass customization and the association of the product configuration task with the planning of its production process while trying to minimize cost and cycle time. We consider a two steps approach that first permit to interactively (with the customer) achieve a first product configuration and first process plan (thanks to non-negotiable requirements) and then optimize both of them (with remaining negotiable requirements). The communication concerns the second optimization step. Our goal is to evaluate a recent evolutionary algorithm (EA). As both problems are considered as constraints satisfaction problems, the optimization problem is constrained. Therefore the considered EA was selected and adapted to fit the problem. The experimentations will compare the EA with a conventional branch and bound according to the problem size and the density of constraints. The hypervolume metric is used for comparison..

1 INTRODUCTION

This paper deals with mass customization and more accurately with aiding the two activities, product configuration and production planning, achieved in a concurrent way. According to the preferences of each customer, the customer requirements (concerning either the product or its production) can be either non-negotiable or negotiable. This situation allows considering a two-step process that aims to associate the two conflicting expectations, interactivity and optimality. The first interactive step, that sequentially processes each non-negotiable requirement, corresponds with a first configuration and planning process that reduces the solution space. This process is present in many commercial web sites using configuration techniques like automotive industry for example. Then, a second process optimizes the solution with respect to the remaining negotiable requirements. As the solution space can quickly become very large, the optimization problem can become hard. Thus, this behavior is not frequent in commercial web sites. Meanwhile some scientific works have been published on this subject (see for example [1] or [2]) and the focus of this article is on the optimization problem. In some previous conferences we proposed an interesting adapted evolutionary algorithm for this problem [3]. However, the presentation was rather descriptive and experimentations were not significant. Therefore, the goal of this paper is to compare this algorithm with a classical branch and bound. This initial section introduces the problem and the organization of the paper.

1.1 Concurrent configuration and planning processes as a CSP

Deriving the definition of a specific or customized product (through a set of properties, sub-assemblies or bill of materials, etc…) from a generic product or a product family, while taking into account specific customer requirements, can define product configuration [4]. In a similar way, deriving a specific production plan (operations, resources to be used, etc...) from some kind of generic process plan while respecting product characteristics and customer requirements, can define production planning [5]. As many configuration and planning studies (see for example [6] or [5]) have shown that each problem could be successfully considered as a constraint satisfaction problem (CSP), we have proposed to associate them in a single CSP in order to process them concurrently.

This concurrent process and the supporting constraint framework present three main interests. First they allow considering constraint that links configuration and planning in both directions (for example: a luxury product finish requires additional manufacturing time or a given assembly duration forbids the use of a particular kind of component). Secondly they allow processing in any order product and planning requirements, and therefore avoid the traditional sequence: configure product then plan its production [7]. Thirdly, CSP fit very well on one side, interactive process thanks to constraint filtering techniques, and on the other side, optimization thanks to various problem-solving techniques. However, we assume infinite capacity planning and consider that production is launched according to each customer order and production capacity is adapted accordingly.

In order to illustrate the addressed problem we consider a very simple example dealing with the configuration and planning of a small plane. The constraint model is shown in figure 1. The plane is defined by two product variables: number of seats (Seats, possible values 4 or 6) and flight range (Range, possible values 600 or 900 kms). A constraint Cc1 forbids a plane with 4 seats and a range of 600 kms. The production process contains two operations: sourcing and assembling. (noted Sourc and Assem). Each operation is described by two process variables: resource and duration: for sourcing, the resource (R-Sourc, possible resources “Fast-S” and “Slow-S”) and duration (D-Sourc, possible values 2, 3, 4, 6 weeks), for assembling, the resource (R-Assem, possible resources “Quic-A” and “Norm-A”) and duration (D-Assem, possible values 4, 5, 6, 7 weeks). Two constraints linking product and process variables

[1] Toulouse University - Mines Albi, CGI lab, Albi, France
email: somename@mines-albi.fr

modulate configuration and planning possibilities: one linking seats with sourcing, Cp1 (Seat, R-Sourc, D-Sourc), and a second one linking range with the assembling, Cp2 (Range, R-Assem, D-Assem). The allowed combinations of each constraint are shown in the 3 tables of figure 1. Without taking constraints into account, this model shows a combinatory of 4 for the product (2x2) and 64 for the production process (2x4) x (2x4) providing a combinatory of 256 (4 x 64) for the whole problem. Considering constraints lead to 12 solutions for both product and production process.

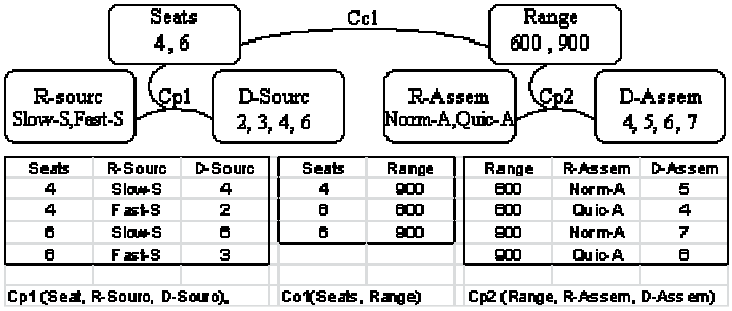

Figure 1 Concurrent configuration and planning CSP model

1.2 Optimizing configuration and planning concurrently

Given previous problem, various criteria can characterize a solution: on the product configuration side, performance and product cost, and on the production planning side, cycle time and process cost. In this paper we only consider cycle time and cost. The cycle time matches the ending date of the last production operation of the configured product. Cost is the sum of the product cost and process cost. We are consequently dealing with a multi-criteria optimization problem. As these criteria are in conflict, it is better for decision aiding to offer the customer a set of possible compromises in the form of Pareto Front.

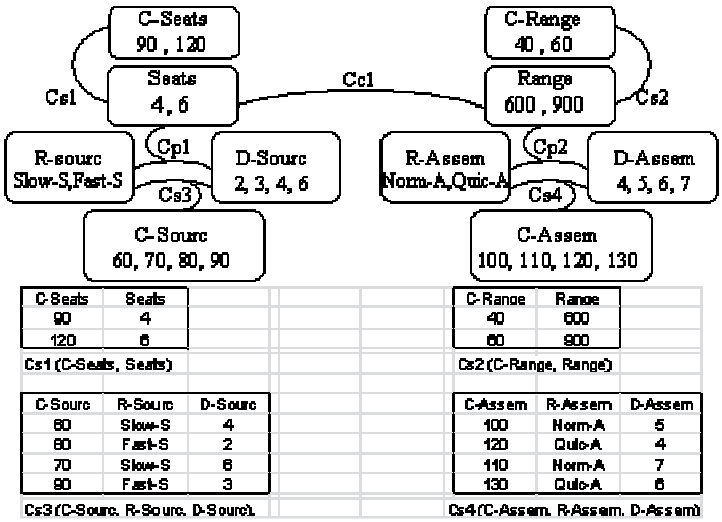

Figure 2 Concurrent configuration and planning model to optimize

In order to complete our example, we add a cost and cycle time criteria as represented in figure 2. For cost, each product variable and each process operation is associated with a cost parameter and a relevant cost constraint: (C-Seats, Cs1), (C-Range, Cs2), (C-Sourc, Cs3) and (C-Assem, cs4) detailed in the tables of figure 2. The total cost is obtained with a numerical constraint and the cycle time, sum of the two operation durations, is also obtained with a numerical constraint as follow:

Total cost = C-Seats + C-Range + C-Sourc + C-Assem.

Cycle time = D-Sourc + D-Assem

The twelve previous solutions are shown on the figure 3 with the Pareto front gathering the optimal ones. In this figure, all solutions are present. When non-negotiable requirements are processed during interactive configuration and planning, some of these solutions will be removed. Once all these requirements are processed, the identification of the Pareto front can be launched in order to propose the customer a set of optimal solutions.

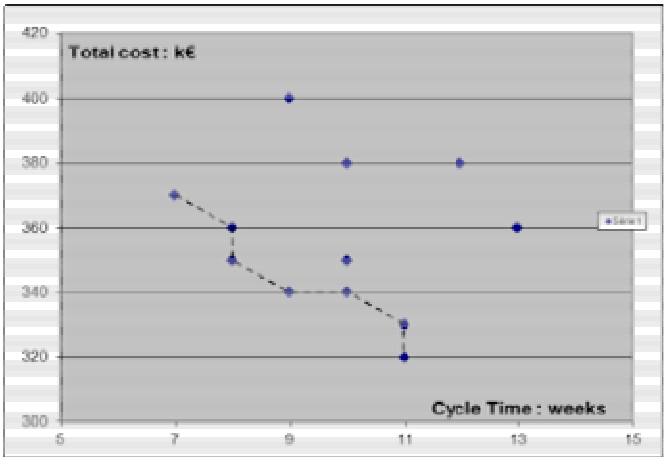

Figure 3 Optimal solutions on the Pareto Front

A strong specificity of this kind of problems is that the solution space is large. It is reported in [8] that a configuration solution space of more than $1.4 * 10^{12}$ is required for a car configuration problem. When planning is added, the combinatorial structure can become huge. Specificity lies in the fact that the shape of the solution space is not continuous and in most cases shows many singularities. Furthermore, the multi-criteria aspect and the need for Pareto optimal results are also strong problem expectations. These points explain why most of the articles published on this subject (as for example [9]) consider genetic or evolutionary approaches to deal with this problem. However classic evolutionary algorithms have to be adapted in order to take into account the constraints of the problem as explained in [10]. Among these adaptations, the one we have proposed in [3] is an evolutionary algorithm with a specific constrained evolutionary operators and our goal is to compare it with a classical branch and bound approach.

In the following section we characterize the optimization problem and briefly recall the optimization techniques. Then experimentation results are presented and discussed in the last section.

2 OPTIMIZATION PROBLEM AND OPTIMIZATION TECHNIQUES

2.1 Optimization problem

The problem of figure 2 is generalized as the one shown in figure 4. The optimization problem is defined by the quadruplet <V, D, C, f > where V is the set of decision variables, D the set of domains linked to the variables of V, C the set of constraints on variables of V and f the multi-valuated fitness function. Here, the aim is to minimize both cost and cycle time. The set V gathers: the product descriptive variables and the resource variables. The set C gathers constraints (Cc and Cp). Cost variables and operation durations are deduced from the variables of the set V thanks to the remaining constraints.

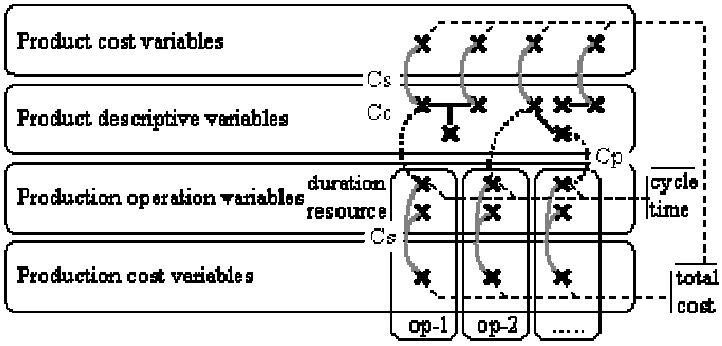

Figure 4 Constrained optimization problem

Experimentations will consider different problem sizes: different numbers of product variables, different number of production operations and different number of possible values for these variables. Different constraint densities (percentage of excluded combinations of values) will be also considered.

2.2 Optimization techniques

The proposed evolutionary algorithm is based on SPEA2 [11] with an added constraints filtering process that avoids infeasible individuals (or solutions) in the archive. This provides the six steps following approach:

1. Initialization of individual set that respect the constraints (thanks to filtering),
2. Fitness assignment (balance of Pareto dominance and solution density)
3. Individuals selection and archive update
4. Stopping criterion test
5. Individuals selection for crossover and mutation operators (binary tournaments)
6. Individuals crossover and mutation that respect constraints (thanks to filtering)
7. Return to step 2.

For initialization, crossover and mutation operators, each time an individual is created or modified, every gene (decision variable of V) is randomly instantiated into its current domain. To avoid the generation of unfeasible individuals, the domain of every remaining gene is updated by constraint filtering. As filtering is not full proof, inconsistent individuals can be generated. In this case a limited backtrack process is launched to solve the problem. For full details please see [3].

The key idea of the Branch and Bound algorithm is to explore a search tree but using a cutting procedure that stops exploration of a branch when a better branch has already been found. The first tool is a splitting procedure that corresponds to the selection of one variable of the problem and to the instantiation of this variable with each possible value. The second tool is a node-bound evaluation procedure. The filtering process is used to achieve this task with a partial instantiation and is able to evaluate if the partial instantiation is consistent with the constraints of the problem, and, if this is the case, to provide the lower bound of each criterion cycle time and cost. When the search reaches a leaf of the search tree, or complete instantiation, the filtering system gives the exact evaluation of the solution. Thus, the values of leaf solutions can be used to compute the current Pareto front and then to cut remaining unexplored branches that are dominated by any aspect of the Pareto front solution (e.g. the upper bounds of the leaf solution dominate the minimal bounds of the branch to cut).

3 EXPERIMENTATIONS

The optimization algorithms were implemented in C++ programming language and interacted with a filtering system coded in Perl language. All tests were done using a laptop computer powered by an Intel core i5 CPU (2.27 Ghz, only one CPU core is used) and using 2.8 Go of ram. These tests compared the behavior of our constrained EA algorithm with the exact branch-and-bound algorithm.

3.1 First experimentation: problem size and constraint densities

An initial first model, named "full model" is considered. It can be consulted and interactively used at http://cofiade.enstimac.fr/cgi-bin/cofiade.pl select model ‘Aircraft-CSP-EA-10’. It gathers five product variables with a domain size between 4 and 6, six production operations with a number of possible resources between 3 and 25. Without constraints consideration, the solution space of the product model is 5,184, and the planning model is 96,000. The size of the global problem model is 497,664,000. A second model, named “small model”, has been derived from the previous one with the suppression of a high combinatory task and a reduction of one domain size. This reduces the planning problem size to 12,000 and global model 6,220,800.

In order to evaluate the impact of constraints density, two versions of each model were produced: one with a "weak density" of constraints (20% of possible combinations are excluded in each constraint Cc and Cp) and the other with a "high density" of constraints (50% excluded). These values are frequently met in industrial configuration situations. This provides four models characteristics in table 1.

Table. 1 Problems characteristics

Solution quantity	Without constraints	Low density	High density
Small model	6 220 800	595 000	153 000
Full model	497 664 000	47 600 000	12 288 000

For the small models, evolutionary settings are tuned to: population size: 50; archive size: 40; P_{mut}: 0.4; P_{cross}: 0.8. The ending criterion used is a time limit of half an hour. For the full models, we adapt settings for a wider search: population size: 150; archive size: 100; P_{mut}: 0.4; P_{cross}: 0.8. The ending criterion used is a time required by the BB algorithm. In order to analyze the two optimization approaches, we compare the hypervolume evolution during optimization process. Hypervolum metric has been defined in [12]. It measures the hypervolume of space dominated by a set of solutions and is illustrated in Figure 5.

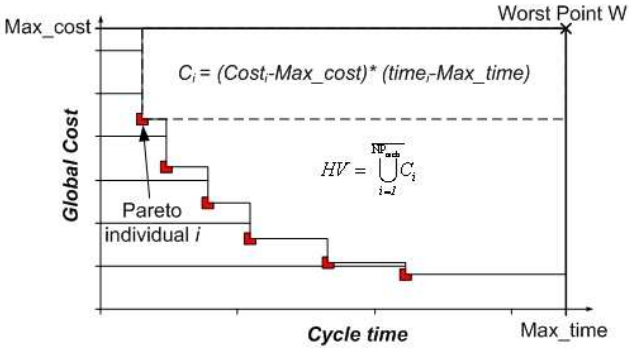

Figure 5 Hypervolume linked to a Pareto front

In our two criteria case, it is the upper right area of figure 5. It thus allows evaluation of both convergence and diversity properties because the fittest and most diversified set of solutions is the one that maximizes the hypervolume.

Results are presented in figure 6 where EA curves are average results for 30 executions. Both algorithms start with a lapse of time where performance is null. For the BB algorithm, this corresponds to the time needed to reach a first leaf on the search tree, while for the EA; it corresponds to the time consumed to constitute the initial population.

For the small models (first two curves), the BB algorithm reaches the optimal Pareto front much faster compared with EA performance. On the other hand, the EA is logically better than the BB algorithm on the full model. For example, on the low-constrained model, the BB algorithm took 20 times longer to reach a good set of solutions (less than 0.5% of the optimal hypervolume).

The impact of constraints density could also be discussed. As it can be seen, the BB algorithm performance is improved when the density of constraints is high. This is because the filtering allows more branches to be cut on the search tree, in such way that the algorithm reaches leaf solutions and, consequently, optimal solutions more quickly. The EA performance moves in the opposite way. The more the model is constrained, the more the random crossover operation will have to backtrack to find feasible solutions, and thus the time needed by the algorithm will be consequent.

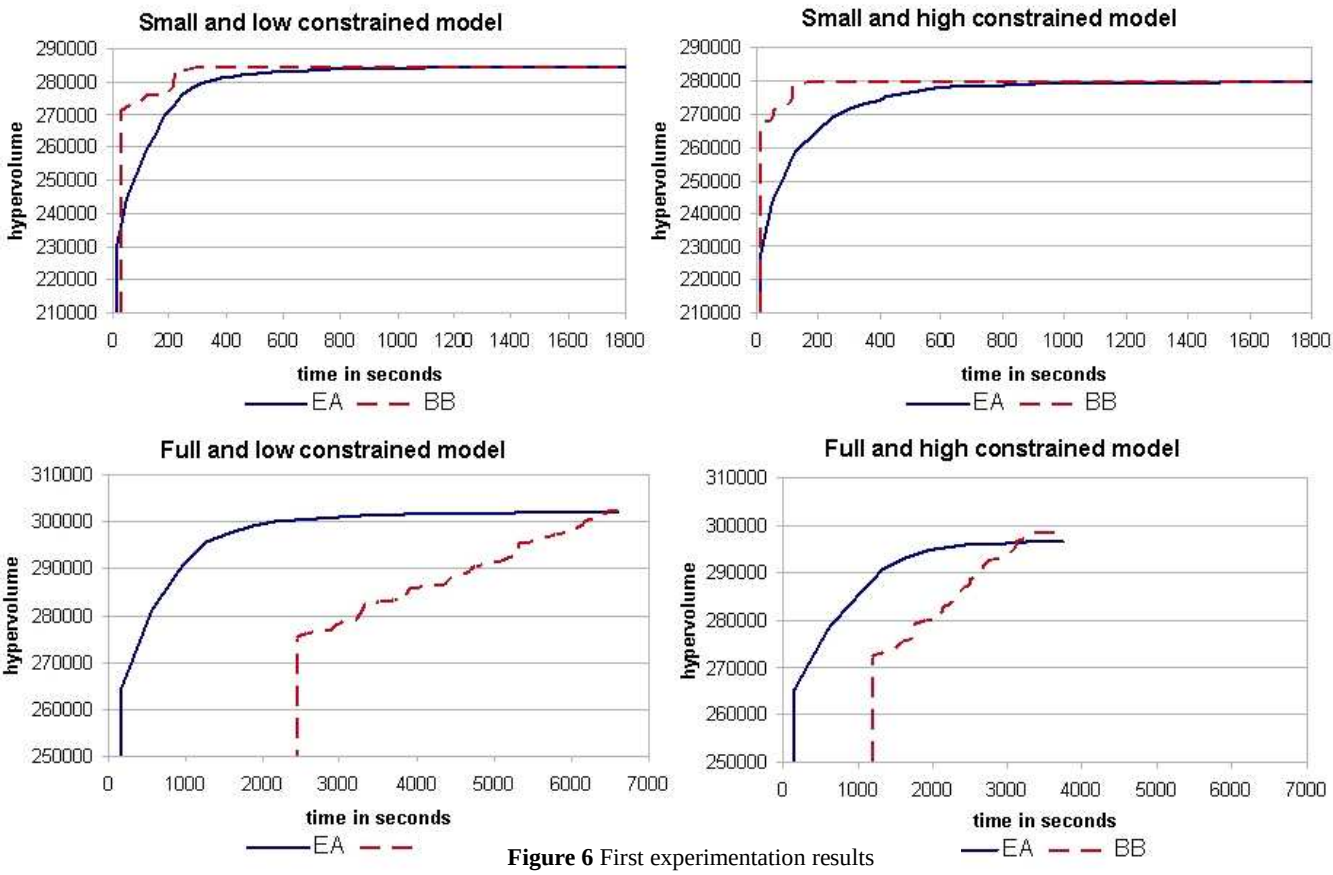

Figure 6 First experimentation results

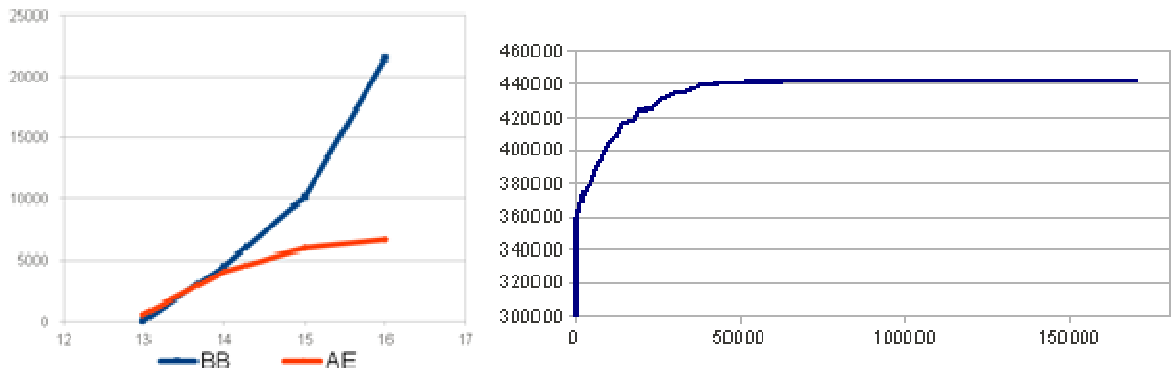

Figure 7 Experimentation results dealing with problem size

3.2 Experimentations on problem size

In order to try to identify the problem size where EA is more suitable than BB, we have modified the low constrained model as follows. We consider now a model gathering six product variables and six production operations with three possible values for each, and sequentially add either a product variable and or an operation. The range of study is between 12 and 16 decision variables with three possible values for each. Relevant solution spaces without constraint vary between $1.6*10^6$ and $43*10^6$.

The results are shown in the left part figure 7. The vertical axis corresponds with the computation time and the horizontal one with the number of decision variables. For BB curves, it shows the time to reach the optimal solution. For the EA curve it shows the time required for nine EA runs over ten to reach the optimal solution. Order of magnitude are close for both around 13 or 14 variables corresponding with a solution space around $2*10^6$ to $5*10^6$ comparable with our previous small model size.

As we already mention, industrial models are frequently larger than that. We therefore try our EA approach with a low constrained model with 30 variables and a solution space around 10^{16}. The stopping criterion is "2 hours without improvement". The right part of Figure 7 shows that the optimization process has stopped after 48 hours. It can be noticed that 90% of the final score was obtained after 3 hours and 99% in 10 hours. This allows underlining the good performance of our approach when facing large low constrained problems. Of course the idea is to use BB, if the first interactive configuration step has led to a rather small problem, less than 13 or 14 variables in our case, and EA otherwise.

Finally we also try to break optimization in two steps. The idea is: (i) compute quickly a low quality Pareto, (ii) select the area that interest the customer (iii) compute a Pareto on the restricted area. The restricted area is obtained by constraining the two criteria total cost and cycle time (or interesting area) and filtering these reductions on the whole problem. The search space is greatly reduced and the second optimization much faster. This is shown in figure 8 where the left part shows the single step process with 10 and 60 minutes Pareto and the right part shows the restricted area with the two previous curve and the one corresponding with a 10 minutes Pareto launched on the restricted area. It shows that the sequence of two optimization steps of 10 minutes provide a result equivalent to a 60 minutes optimization process.

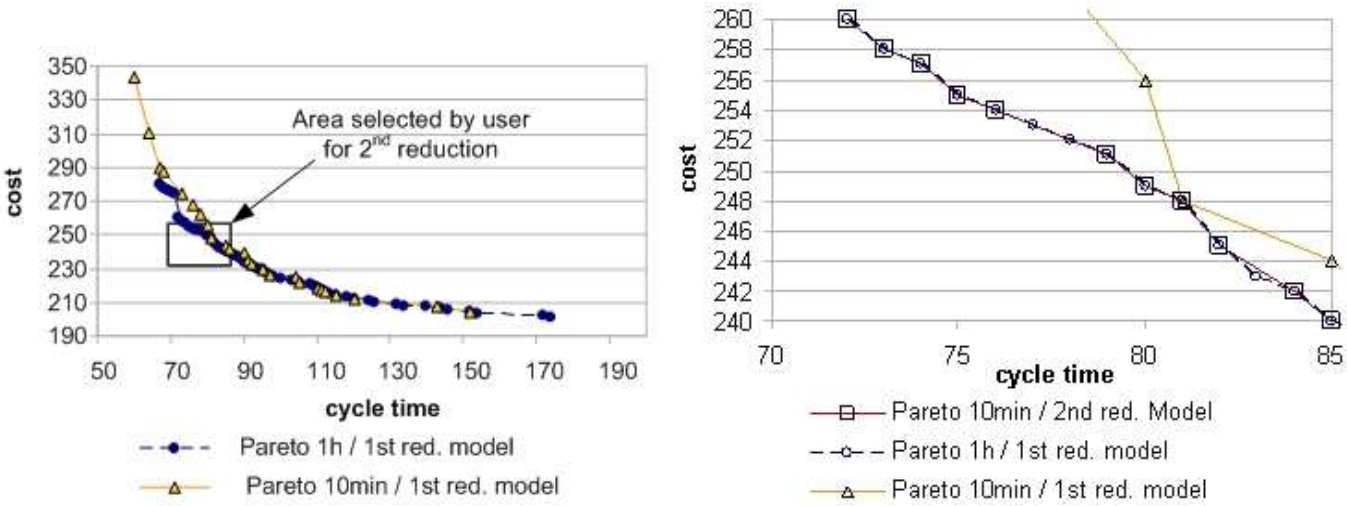

Figure 8 Experimentations with a two steps optimization process

4 CONCLUSIONS

The goal of this communication was to propose a first evaluation of an adapted evolutionary algorithm that deals with concurrent product configuration and production planning. The problem was recalled and the two optimization approaches (Evolutionary algorithm and branch and bound) where briefly presented. Various experimentations have been presented. A first result is that: (i) the proposed EA works fine when the size of the problem gets large compare to the BB, (ii) when problem tends to be more constrained the tendency goes to the opposite. When problem is low constrained (90% of excluded solutions) with 13-14 decision variables with 3 values each, they perform equally. When the problem gets larger, BB cannot be considered and EA can provide good quality results for the same problem with up to 30 variables (around 10^{16} solutions - 90% rejected). Finally some ideas about a two steps optimization process have shown that the proposed approach is quite promising for large problems. These are first experimentation results and we are now working on comparing our proposed EA with some penalty function approaches.

ACKNOWLEDGEMENTS

We would like to thank the referees for their comments, which helped improve this paper considerably.

REFERENCES

[1] Hong G., Hu L., Xue D., Tu Y, Xiong L,. Identification of the optimal product configuration and parameters based on individual customer requirements on performance and costs in one-of-a-kind production. Int. J. of Production Research, 46(12) 3297-3326 (2008)

[2] Aldanondo M., Vareilles E., Configuration for mass customization: how to extend product configuration towards requirements and process configuration, Journal of Intelligent Manufacturing, vol. 19 n° 5, p. 521-535A (2008)

[3] Pitiot P., Aldanondo M., Djefel M., Vareilles E., Gaborit P., Coudert T., Using constraints filtering and evolutionary algorithms for interactive configuration and planning. IEEE press, IEEM 2010, p.1921-1925, Macao China (2010)

[4] Mittal S., Frayman F., Towards a generic model of configuration tasks, proc of IJCAI, p. 1395-1401(1989)

[5] Barták R., Salido M., Rossi F., Constraint satisfaction techniques in planning and scheduling, in: Journal of Intelligent Manufacturing, vol. 21, n°1, p. 5-15 (2010)

[6] Junker U., Handbook of Constraint Programming, Elsevier, chap. 24, p. 835-875 (2006)

[7] Aldanondo M., Vareilles E., Djefel M.. Towards an association of product configuration with production planning, Int. J. of Mass Customisation, vol.3 n°4, p. 316-332 (2010)

[8] Amilhastre J., Fargier H., Marquis P., Consistency restoration and explanations in dynamic csps - application to configuration, in: Artificial Intelligence, vol.135, p. 199-234 (2002)

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

[10] Coello Coello C., Theoretical and numerical constraint-handling techniques used with EAs : A survey of the state of art, Computer Methods in Applied Mechanics and Engineering, vol. 191, n°11-12, p. 1245-1287 (2002)

[11] Zitzler E., Laumanns M., Thiele L., SPEA2: Improving the Strength Pareto Evolutionary Algorithm, Technical Report 103, Swiss Fed. Inst. of Technology (ETH), Zurich (2001)

[12] Zitzler E., Thiele L., Multiobjective Optimization Using Evolutionary Algorithms - A Case Study, 5th Conf. Parallel Problem Solving from Nature, Springer, p. 292-301 (1998)