HAL
open science

Product and supply chain design using a taboo search

Radwan El Hadj Khalaf, Bruno Agard, Bernard Penz

To cite this version:

Radwan El Hadj Khalaf, Bruno Agard, Bernard Penz. Product and supply chain design using a taboo search. International Conference on Industrial Engineering and Systems Management, May 2009, Montréal, Canada. hal-00347196

HAL Id: hal-00347196

https://hal.science/hal-00347196v1

Submitted on 15 Dec 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

hal-00347196, version 1 - 15 Dec 2008

International Conference
on Industrial Engineering and Systems Management
IESM 2009
May 13-15
MONTRÉAL - CANADA

Product and supply chain design using a taboo search

Radwan EL HADJ KHALAF [a], Bruno AGARD [b], Bernard PENZ [a]

[a] *G-SCOP*
Grenoble INP – CNRS – UJF
46 avenue Félix-Viallet, 38031 Grenoble Cedex 1, France

[b] *Département de Mathématiques et Génie Industriel*
École Polytechnique de Montréal
C.P. 6079, succ. Centre-Ville, Montréal (Québec), H3C 3A7, Canada

Abstract

An assemble to order policy considers a tradoff between the size of product portfolio and assembly lead time. The concept of modular design is often used to implement the assemble to order policy. Modular design impacts assembly of products and the supply chain. In particular storage, transportation and production are affected by the selected modular structure.

For a determined assembly lead time, a module composition that minimizes assembly and production costs is difficult to establish. This problem is highly combinatorial, it is not achievable to look for an optimal solution. This article proposes a challenging modeling approach that consists in selecting a set of modules to be manufactured in distant facilities and shipped in a nearby location plant for a final assembly operation under time limits. A taboo search resolution approach is then investigated. Computational results showed that this method is promising.

Key words: product family, design, supply chain, optimization, taboo search.

1 Introduction

When commercial competition is strong, the customers have opportunities to choose between different products. Each customer may select the product that is the closest to his individual needs. For a company, it necessitates to design and manufacture diversified products. This enlarges the possibilities to be closer to the customer requirements. At the same time, this involves an increasing number of product variants and options. It follows a complex product diversity that must be well managed in order to guaranty competitive costs. [13].

In order to give an efficient answer to this problem without an expansive product proliferation, companies may focus on "mass customization" [19]. Mass customization deals with large products portfolio, flexible manufacturing systems and extended supply chain.

⋆ This paper was not presented at any other revue. Corresponding author Radwan EL HADJ KHALAF.
Email addresses: radwan.el-hadj-khalaf@g-scop.inpg.fr (Radwan EL HADJ KHALAF), bruno.agard@polymtl.ca (Bruno AGARD), bernard.penz@g-scop.inpg.fr (Bernard PENZ).

A product family is composed by similar products that differ by some characteristics such as options. For example, the basic car model may contains few options in order to minimize the sale price. Then, some options can be added to this basic model like air-conditioning, automatic gear box or diesel engines and so on.

Assemble to order policy enables to offer a large diversity of final products from a limited number of semi-finished components, often called modules [3]. Those modules may be manufactured in a pre-assembly operation and stored. The final assembly of the product from the modules is operated when an order is received. Lean production has induced major changes in the supplier-retailer relationship. Nowadays the lead time is a major issue of contracts [1]. For a supplier, satisfying an order within respect of the delay is essential. Delays are indeed costly for both the retailer and the supplier.

In this paper, we explore this production policy where modules are manufactured in distant location facilities for cost minimization. Those modules are shipped and assembled in a nearby location facility in order to have a short lead-time for the customers. Electric beam is an example of this category of product family. They are largely used in the car industry [13]. In section 2 we give an overview on the state of art of the different studies related to the problem at hand. We give a more detailed description of the problem in section 3. An Integer Linear Program model is given in section 4. A taboo search algorithm is investigated in section 5 and some computational experiments are given and analyzed in section 6. Finally concluding remarks and perspectives are proposed in section 7.

2 State of the art

Literature proposes various approaches for the design of product families. Some product design methodologies focus on the product architecture [4],[11], this is advantageously supported with modular design [10], component / product / process standardization [12], [17],[20]. Different methodologies concentrate on process standardization [17], process resequencing [15] or generic assembly routing [8],[9]. From another point of view, authors concentrate on supply chain design methodologies, by integrating customers and suppliers in uncertain environment [5], centered on stocks management [16] and for distant facility location [18].

In all these works, the design of product, process and supply chain are integrated two by two. However, Lee [14] shows that the product design choices (modular design, standardization or delayed differentiation) have a strong influence on the supply chain design. Hence it is important to determine simultaneously, even roughly, the product family design and its associated supply chain.

There is some recent works dealing with a global design modeling. Agard et al. [1] propose a genetic algorithm to minimize the mean of finished product assembly times for a given demand. Agard and Penz [2] propose a model for minimizing module production costs and a simulated annealing method based on a clustering approach. Lamothe et al. [13] use a generic bill of material representation in order to identify simultaneously the best bill of material for each product and the optimal structure of the associated supply chain.

3 Problem Presentation

Consider the following industrial context (Figure 1). The producer receives customers' orders for finished products containing options and variants. Each individual product is then manufactured from a set of modules that come from various suppliers.

Consider now that the producer has only a short delay (T) to respond to customer demands. This delay is less than the necessary time to assemble products from elementary components. In addition to this, the producer has to provide the product exactly like the customer demand (without extra options). This constraint comes from technical considerations or simply to avoid supplementary costs.

To satisfy customers, the producer brings pre-assembled components, called modules, from many suppliers which are located in distant facilities around the world. The suppliers' facilities are characterized by a very weak production costs. Then, the modules are assembled in the producer facility which we assume to be very close to the customers and thus characterized by its great reactivity and a reduced lead-time.

hal-00347196, version 1 - 15 Dec 2008

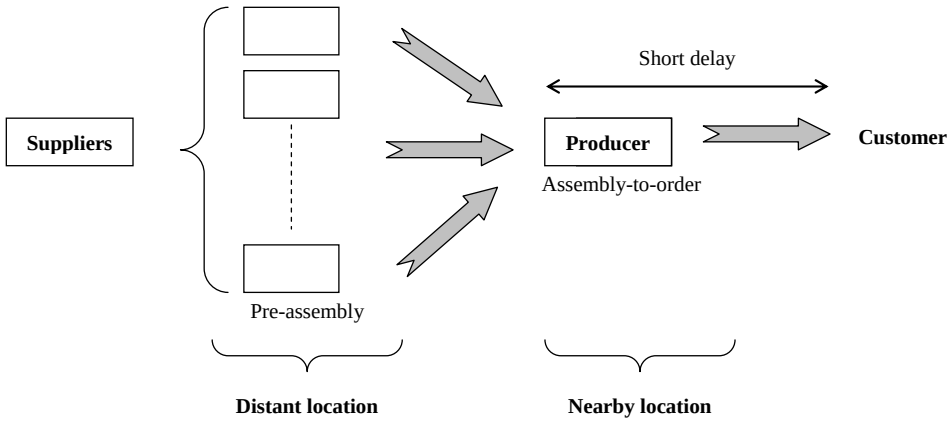

Fig. 1. Structure of the supply chain

4 Model and solution approach

4.1 Notations

Specifying the problem assumptions: a product or a module is considered as the set of functions that it must fill, then:

- a function F_k is a requirement that must be ensured by the finished product.
- a module M_j is an assembly of functions that could be added with other modules to make a finished product.
- a finished product P_i is an assembly of modules that corresponds exactly to at least one customer demand.

Let introduce the following notations:

- F = $\{F_1, ..., F_q\}$: set of q functions that can appeared in both finished products and modules;
- P = $\{P_1, ..., P_n\}$: set of n possible finished products that may be demanded by at least one customer. We note D_i the estimated demand of the product P_i during the life cycle of the product family.
- M = $\{M_1, ..., M_m\}$: set of m possible modules.
- S = $\{S_1, ..., S_s\}$: set of s distant production facilities where a site S_l has a production capacity Cap_l.
- CF_j : the management fixed cost of module M_j in the nearby facility.
- CV_j : the assembly variable cost of module M_j in the nearby facility.
- CF_{jl} : the management fixed cost of module M_j in facility S_l.
- CV_{jl} : the production variable cost of module M_j in facility S_l.
- w_j : the necessary time to assemble the module M_j in a finished product.
- C_{jl} : the load caused by producing module M_j in facility S_l.

Under these assumptions, we can represent a product (or a module) by a binary vector of size q. Each element shows whether the corresponding function is required in the product (value = 1) or not (value = 0). The set M contains m modules. It may be a selection of modules defined by the engineering or all the possible modules issued from the whole combinatory.

The problem is now to determine the subset $M^{'} \in M$, of minimum cost, such that all products in P can be built in a constrained time window T with elements from $M^{'}$. Concerning the products, the goal is to determine which bill of material is the most suitable (Figure 2).

In order to take into account the process design, (1) the producer assembly line costs are considered - (2) finished product assembly time must be less than the available time to respect the time delivery for the customers. To take into account the supply chain design, distant location production costs are considered (transportation costs are implicitly considered in the production costs because we treat a problem with only one nearby location facility).

We have done a model of the problem using an Integer Linear Program formulation. Our objective consists in minimizing all the costs linked to the activity of the producer and suppliers. These costs are : fixed costs

hal-00347196, version 1 - 15 Dec 2008

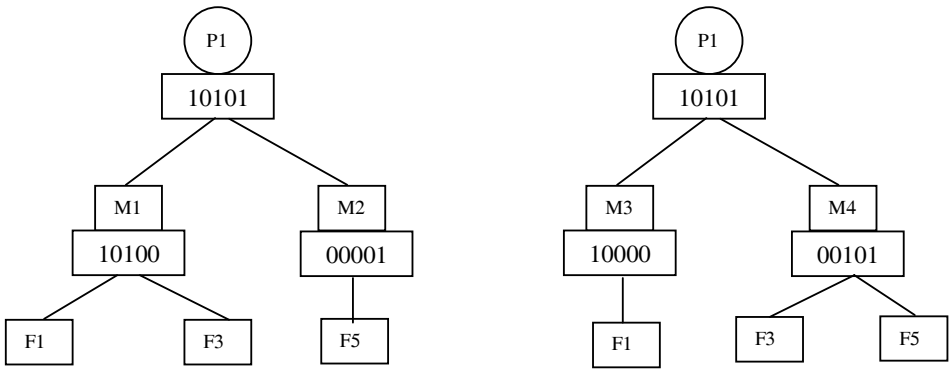

Fig. 2. alternative bills of material

due to module management in the nearby location facility, module assembly costs in the nearby location facility, fixed costs due to module management in the distant location facilities and module production costs in the distant location facilities.

4.2 The mathematical model

The model given in this section consists to optimize simultaneously costs occurring in the nearby location facility and the distant location facilities:

$$\text{Z} = \min \sum_{j=1}^{m} CF_j Y_j + \sum_{j=1}^{m} CV_j (\sum_{i=1}^{n} D_i X_{ij}) + \sum_{l=1}^{s} \sum_{j=1}^{m} CF_{jl} Y_{jl} + \sum_{l=1}^{s} \sum_{j=1}^{m} CV_{jl} Q_{jl}$$

s.t.

$$AX_i = P_i \qquad \forall i \in \{1, \cdots, n\} \tag{1}$$

$$\sum_{j=1}^{m} w_j X_{ij} \leq T \qquad \forall i \in \{1, \cdots, n\} \tag{2}$$

$$X_{ij} \leq Y_j \qquad \forall i \in \{1, \cdots, n\} \; \forall j \in \{1, \cdots, m\} \tag{3}$$

$$\sum_{l=1}^{s} Q_{jl} = \sum_{i=1}^{n} D_i X_{ij} \; \forall j \in \{1, \cdots, m\} \tag{4}$$

$$\sum_{j=1}^{m} C_{jl} Q_{jl} \leq Cap_l \quad \forall l \in \{1, \cdots, s\} \tag{5}$$

$$Q_{jl} \leq K_j Y_{jl} \qquad \forall j \in \{1, \cdots, m\} \; \forall l \in \{1, \cdots, s\} \tag{6}$$

$$Q_{jl} \geq 0 \qquad \forall j \in \{1, \cdots, m\} \; \forall l \in \{1, \cdots, s\} \tag{7}$$

$$Y_j, X_{ij}, Y_{jl} \in \{0, 1\} \; \forall i \in \{1, \cdots, n\} \; \forall j \in \{1, \cdots, m\} \; \forall l \in \{1, \cdots, s\} \tag{8}$$

Where $X_{ij} = 1$ if module M_j is used in the bill of material of product P_i, 0 otherwise. $Y_j = 1$ if module M_j is selected (belongs to $M^{'}$), 0 otherwise. A is the binary matrix whose the column j is the vector M_j. X_j is the column vector composed by the variables X_{ij}. $Y_{jl} = 1$ if module M_j is produced in facility S_l, 0 otherwise. Q_{jl} is the quantity of module M_j produced in facility S_l.

The objective function minimizes the full costs occurring in supply chain, where $(\sum_{i=1}^{n} D_i X_{ij})$ is the total demand of module M_j.

Constraint (1) shows that finished products P_i must be assembled exactly like the customer demand. Constraint (2) indicates that products must be assembled within the time window T in order to respect the delivery time. Constraint (3) translates the relation between X_{ij} and Y_j variables. If a module is used in the bill of material of some products then it belongs to $M^{'}$. Constraint (4) indicates that the production of a module M_j must satisfy the requirements. Constraint (5) shows that production in facility S_l must not exceed its capacity. Constraint (6) expresses the relation between Q_{jl} and Y_{jl} variables. A

hal-00347196, version 1 - 15 Dec 2008

module M_j can be produced in S_l only if M_j is assigned to S_l ($Y_{jl} = 1$). K_j is big constants bounded by $\sum_{i=1}^{n} D_i X_{ij}$.

The problem described here contains the set partitioning problem (constraints 1). We then conclude that it is NP-hard in the stong sense.

5 Resolution with a taboo search algorithm

The Taboo Search (TS) is a methaheuristic approach designed to find a near optimal solution of combinatorial optimization problems [6],[7]. TS can be sketched as follows. There is a set F of feasible solutions. A move is defined as an operation or a function which transforms a solution $x_1 \in F$ into another solution $x_2 \in F$. For any solution $x \in F$, a subset of moves applicable to it is defined as its neighborhood $N(x) \subseteq F$. TS starts from an initial solution. At each step the neighborhood $N(x)$ of a given solution x is searched in order to find a neighbor $x^{'}$. The move, which lead to the neighbor $x^{'}$, is performed and then the newly obtained solution is set as the origin for the next step. In order to prevent cycling, a structure called the taboo list of length L (fixed or variable) is introduced to prevent returning to a solution visited in the last L iteration. The taboo list is often interpreted as a limited queue of length L containing forbidden moves.
For this problem, it is difficule to define moves that transform a feasible solution to another one. So, in order to generate a new feasible solution from the current one, we use a group of moves (Figure 3) which are used consecutively until finding a feasible solution. These moves are described in the elimination and insertion neighborhoods.

Before introducing the description of the taboo search algorithm, let us introduce the following terminology.

- A module M_j is compatible with a product P_i if it does not contain extra functions for this product. So a finished product can only be assembled from compatible modules (because we require an exact assembly of each demanded product).
- The degree of a module M_j is the number of finished products that are compatible with the module M_j.
- $M^{'}$ is the subset of modules that are selected in the current solution.
- A product P_i is not feasible if it does not admit a bill of materials from modules in $M^{'}$.
- The logistic costs of a module are costs (fixed and variable) generated by the production of the module in the distant location facilities.

5.1 Taboo search algorithm

The algorithm (Figure 3) begins by an initialization phase (steps 1 & 2) which consists to generate an initial solution. Step (1) is the determination of the bill of materials for each product. This is ensured by solving the integer linear program formed by constraints (1), (2), (3) and (8). We note here that we determine the bill of materials of products one by one. We then solve the above integer linear program by freezing the variable i (which represents the finished products) at each time. Due to the size of the problem, it is possible to do it exactly with an efficient integer linear programming solver. The choosen modules are assigned to distant facilities (step 2) by solving the integer linear program formed by constraints (4), (5), (6), (7) and (8). This program is easily solved by the ILP solver because the number of choosen modules is always very small compared with the whole combinatory and also the number of distant facilities is generally small in real supply chains.

Iteration phase (steps 3 to 8) consists in constructing a new subset of modules, allowing to define the bill of materials of all finished products, from the current subset. This can be done by eliminating a module (step 3) from the current subset $M^{'}$ and then inserting new modules (step 5) until the feasibility of all finished products is prooved.

Once the new subset is constructed and new bills of materials for infeasible products are determined (step 4 permits to check and update bill of materials of finished products), the new solution is evaluated and recorded if it is better than the current best solution (step 7). This processus is iterative until reaching

hal-00347196, version 1 - 15 Dec 2008

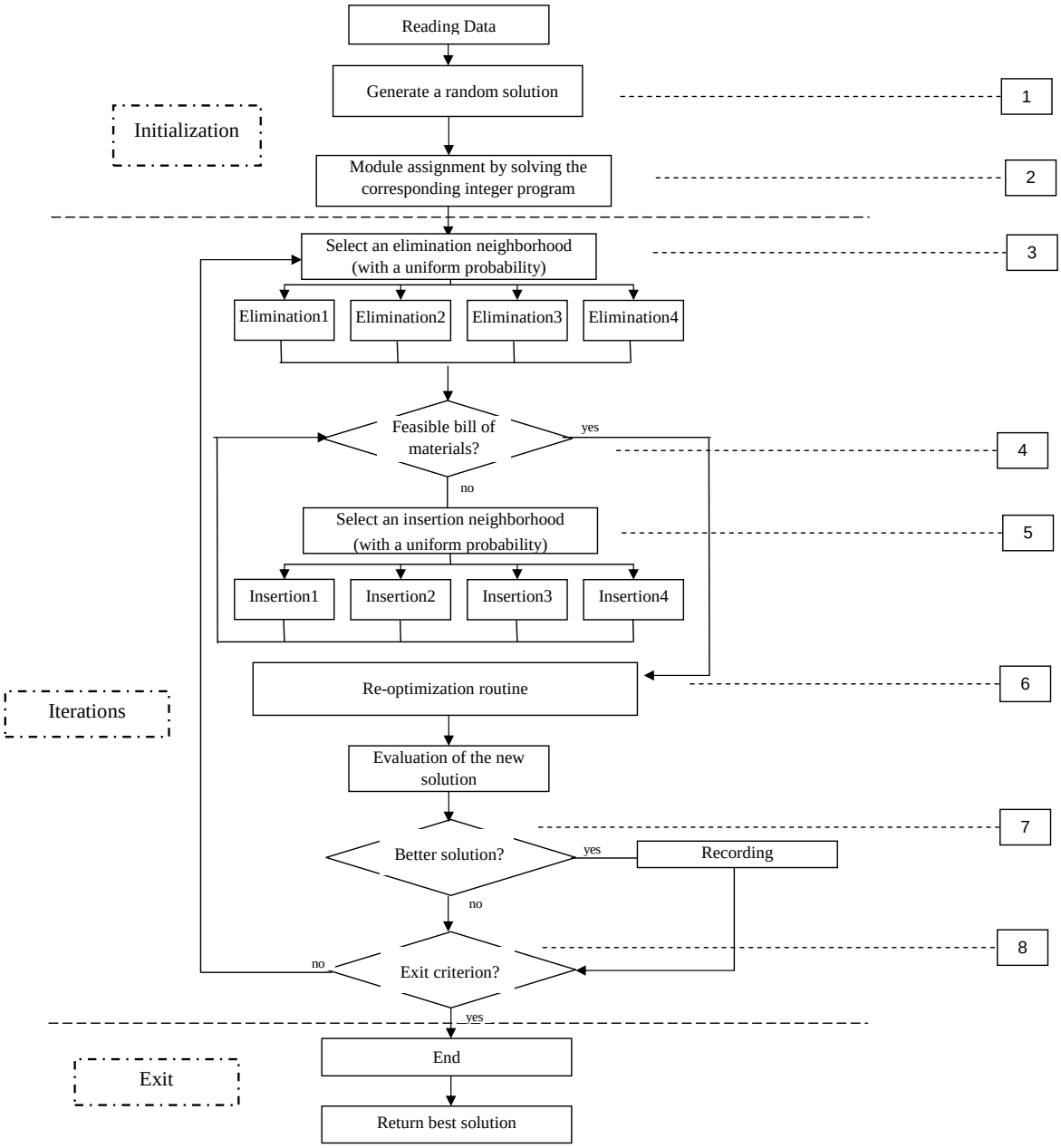

Fig. 3. The Taboo search algorithm

the exit criterion. The algorithm returns then the best solution recorded (Exit phase) that is, the best subset $M^{'}$, the corresponding product bills of materials and the corresponding module assignments.

Elimination and insertion operations are controlled by taboo lists of a specific length. Thus, we keep in memory the latest modules that have been eliminated (resp. inserted) and we forbid their insertion (resp. elimination) within a specific number of iterations.

The following will give a more detailed description of the different routines and neighborhoods used in the algorithm.

5.2 Bill of materials feasibility control

After elimination (resp. insertion) of a module from (resp. in) $M^{'}$, this routine takes the infeasible products one by one and check their bill of material feasibility by solving the integer linear problem formed by constraints (1), (2), (3) and (8). If the product admits a bill of materials by the new subset $M^{'}$ then its bill of materials will be updated, otherwise it will be still considered as an infeasible product and its

hal-00347196, version 1 - 15 Dec 2008

feasibility will be checked again in the next iteration.

5.3 Re-optimization routine

This routine consists only in cleaning the new subset $M^{'}$ by eliminating the modules that do not belong to any bill of materials. It consists also in updating some data used by the other routines (such that module degrees and quantity needed of each module). It consists also to assign the modules of $M^{'}$ to the distant location facilities by solving the integer linear program formed by constraints (4), (5), (6), (7) and (8), the objective function here is formed only from logistic costs.

5.4 Elimination Neighborhoods

Four elimination neighborhoods are considered:

- Elimination 1: Eliminates a module of small degree.
- Elimination 2: Eliminates a module that generates great logistic costs.
- Elimination 3: Eliminates a module of big degree (because it generates certainly great variable costs).
- Elimination 4: Random elimination.

The selection of the elimination neighborhood is determined by a random draw, that is a random number between 1 and 4 is drawn (using a uniform probability law) and then according to the draw, one of the above neighborhoods is used to determine the module to eliminate.

5.5 Insertion Neighborhoods

Four insertion neighborhoods are considered:

- Insertion 1: Inserts a module of small logistic costs.
- Insertion 2: Inserts a module of a big degree.
- Insertion 3: Inserts a module of big degree (by regarding only infeasible products).
- Insertion 4: Inserts a module that allow the bill of materials feasibility for an infeasible product.

As elimiantion neighborhoods, the insertion neighborhood used to insert modules is determined by a random draw of a uniform probability law.

5.6 The Exit Criterion

Different exit criterions are considered simultaneously: computational time and number of iterations without improvement of the objective function.

6 Computational experiments

6.1 Data sets and experimental conditions

This paper shows performances of this approach to solve small instances and to compare those results with optimal solutions, further tests is conducting at this time on larger instances. For this, five small instances have been generated on which the module set, the finished product set, the distant facility set, the demand D_i, the assembly operating times w_j, the distant facility capacities, the module production loads, the distant facility costs and the nearby facility costs are fixed.

The problem data was fixed as follow: $q = 8$, $n = 30$, where each product has at least 3 functions and at most 6, $m = 255$ (all possible combinatory of modules) and $s = 2$. The assembly operating times w_j were fixed to 1, T was varied from 4 to 5.

Three cost files were randomly generated. For problem "Cost 1": assembly costs are preponderant than logistic ones while for problem "Cost 2" both costs are almost equivalent and for problem "Cost 3" logistic costs are preponderant.

hal-00347196, version 1 - 15 Dec 2008

The tests have been carried in C++ with Ilog Cplex9.0 library. They have been solved in a DELL station / 2.8 GHZ/ 1Go RAM. The computational time is fixed to 30 minutes for the taboo algorithm.

6.2 Result analyses

Following figures 4, 5 and 6 show the mean values of the numerical results per instance file, cost file and delay value. The second column represents the mean objective function value of initial solution. The third one represents the mean optimal value of the objective function, the forth one represents the mean objective value of the best solution found by the taboo algorithm and finally the last column represents the gap rate between both values expressed as a percentage format.

Instance file	Initial Solution	Final Solution	Optimal Solution	Gap
1	41241	29712	28791	3,20%
2	37163	28374	27184	4,38%
3	38824	29009	27738	4,58%
4	41034	29816	28296	5,37%
5	34941	29041	27843	4,30%

Fig. 4. Objective function gap rate per instance file

Cost file	Initial Solution	Final Solution	Optimal Solution	Gap
1	21395	15087	14156	6,58%
2	25838	19479	18552	5,00%
3	68689	53005	51203	3,52%

Fig. 5. Objective function gap rate per cost file

T	Initial Solution	Final Solution	Optimal Solution	Gap
4	46306	31150	29092	7,07%
5	30975	27231	26848	1,42%

Fig. 6. Objective function gap rate per T

These first tests reveal that in average the gap rate is quite reasonable (7% at most) and fairly uniform per instance or cost files (figure 4 and 5). This is a good sign because it shows that our taboo search algorithm is stable and the neighborhoods are well defined. However, examination of figure 6 indicates that the gap is increasingly smaller when T increases which is logically normal because the problem becomes less difficult.

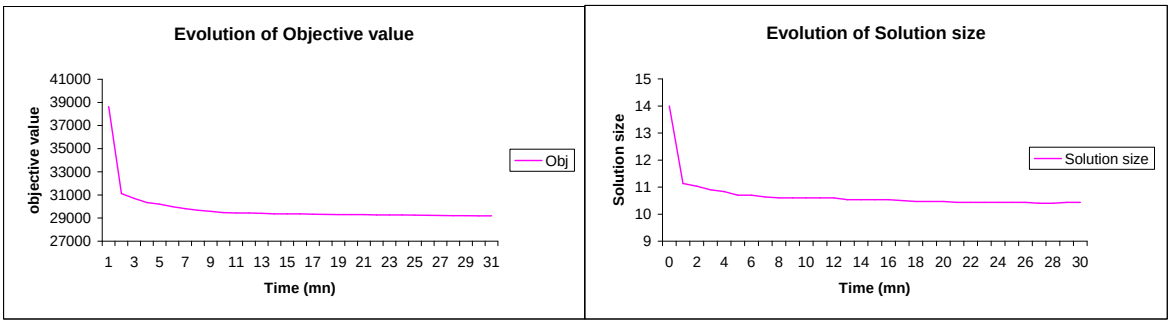

Fig. 7. Evolution of objective value and solution size according to computational time

Figure 7 shows the evolution of the objective function value and the solution size (the average on all instances) according to the computational time. The solution size is the number of distinct modules in $M^{'}$. This figure indicates that the taboo algorithm succeeds very quickly to improve greatly initial solution (after about 7 minutes). Then, it continues to improve the solution because the interesting modules have been already detected. The algorithm needs more time to investigate more solution space. In the other hand, we see that the solution size follows approximately the same evolution shape as the objective

hal-00347196, version 1 - 15 Dec 2008

function value. In fact, many modules in initial solution are generally used in litte bill of materials and then, they increase greatly the fixed costs (either of assembly phase and production phase). The taboo algorithm detects them very quickly through the first elimination neighborhood and consequently the solution size decreases greatly at the first iterations.

We can conclude that thirty minutes is a very suitable computational time for this instance size. However, this does not exclude that we can reach better solutions if we extend the algorithm run.

7 conclusion

This paper was dedicated to a difficult industrial problem arising when companies try to offer a large variety of products to consumers. In this problem, a choice of components (modules) has to be efficient. These modules are produced for stock, and used in the last stage, in the assembly line. Several authors considered this problem, using different assumptions - a function can appear twice in a final product, a final product can be substituted by another one containing more functions - but few papers consider the problem in which each final product must correspond exactly to the demand.

We presented a new challenging model that takes into account the product family design, the process design and the supply chain design at the same time. The product family design consideration is the determination of an efficient modules set allowing to assemble products and to avoid function redundancy. The process design consideration is delivery time constrained. Finally, the supply chain is capacity constrained for all the distant location facilities. The objective functions consists in optimizing the costs occurring in the activity of the producer and the suppliers.

We presented a taboo search algorithm with interesting neighborhoods. Tests on small instances and comparisons with optimal values show the performance of our algorithm and encourage us to develop more tests in order to improve its performance.

Further tests are conducted at this time. More big instances are being generated, we intend to verify until which size our taboo is able to run and until which level it is able to improve the initial solution.

References

[1] B.Agard, B. Cheung, and C. da Cunha. Seletion of a module stock composition using genetic algorithm. *In 12th IFAC Symposium on Information Control Problems in Manufacturing - INCOM 2006*, Saint-Etienne, France, May 17-19 2006.

[2] B. Agard, B. Penz. A simulated annealing method based on a clustering approach to determine bills of materials for a large product family. *To appear in International Journal of Production Economics*, DOI 10.1016/j.ijpe.2008.12.004,(2007).

[3] O.Briant and D.Naddef. The optimal diversity management problem. *Operations Research*, 52(4):515-526, 2004.

[4] J.B. Dahmus, J.P. Gonzalez-Zugasti and K.Otto. Modular product architecture. *Design studies*, 22:409-424, 2001.

[5] E.W. Davis, Global outsourcing : have U.S. managers thrown the baby out with the bath water ? *Business Horizons*, pages 58-65, 1992.

[6] Glover F., McMilan C., Novick B. Interactive Decision Software and Computer Graphics for Architectural and Space Planning. *Annals of Operations Research*, 5 :557-573, 1985.

[7] Glover F. Taboo Search - PartI *ORSA Journal on Computing*, 1(3) :190-206, 1989.

[8] S. Gupta and V. Krishnan. Product familly-based assembly sequence design methodology. *IIE Transactions*, 30:933-945, 1998.

[9] D.W He and A.Kusiak. Design of assembly systems for modular products. *IEEE Transactions on Robotics and Automation*, 13(5):646-655, 1997.

[10] C.C Hung and A. Kusiak. Modularity in design of products and systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 28(1):66-77, 1998.

[11] J. Jiao and M. Tseng. A methodology of developing product family architecture for mass customization. *Journal of Intelligent Manufacturing*, 10:3-20, 1999.

[12] S. Kota, K. Sethuraman, and R. Miller. A metric for evaluating design commonality in product families. *Journal of Mechanical Design*, 122:403-410, 2000.

[13] J. Lamothe, K. Hadj-Hamou, M. Aldanondo: An optimization model for selecting a product family and designing its supply chain. *European Journal of Operational Research*, 169:1030-1047, 2006.

[14] H.L. Lee. Product universality and design for supply chain management. *Production Planning and Control*, 6(3):270-277, 1995.

hal-00347196, version 1 - 15 Dec 2008

[15] H.L. Lee. Effective inventory and service management throught product and process redesign. *Operations Research*, 44(1):151-159, 1996.

[16] H.L. Lee and C.Billington. Managing supply chain inventory : Pitfalls and opportunities. *Sloan Management Review*, 33(3):65-73, 1992.

[17] H.L. Lee and C.S. Tang. Modelling the costs and benefits of delayed product differentiation. *Management Science*, 43(1):40-53, 1997.

[18] T. Van Roy. Cross decomposition algorithm for capacity facility location. *Operations Research*, 34:145-163, 1986.

[19] B.J. Pine II. Mass Customization: The new Frontier in Business Competition. *Harvard Business School Press*, Boston, 1993.

[20] U.W Thonemann and M. Brandeau. Optimal commonality in component design. *Operations Research*, 48(1) :1-19, 2000.

hal-00347196, version 1 - 15 Dec 2008