6th IEEE International Symposium on Assembly and Task Planning – ISATP 2005, Montréal, Canada , July 19-21, 2005.

Composition of modules' stock using Simulated Annealing

Catherine da Cunha† Bruno Agard†

†Département de Mathématiques et de Génie Industriel
Ecole Polytechnique de Montréal
C.P. 6079, succ. Centre-ville
Montréal (Québec), H3C 3A7, Canada
Email: {catherine.dacunha,bruno.agard}@polymtl.ca

Abstract

An assemble to order policy considers a trade-off 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 fixed number of modules, a stock composition that minimizes the mean final assembly time is difficult to establish. This problem is highly combinatorial; it is not achievable to look for an optimal solution. A first step leads to the elaboration of heuristics. This article proposes to combine the results operated with this heuristics to the use of a simulated annealing algorithm. Computational results showed that this method could be efficient.

1 Introduction

Assemble to order is a production policy that enables to offer a large diversity of final products from a limited number of semi-finished elements, often called modules [1]. Those modules may be manufactured by anticipation (in a pre-assembly operation) and stocked. 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 [2]. For a supplier, to satisfy an order within respect of the delay is essential. Delays are indeed costly both for the retailer (through the penalty costs, charged for non-respect of due dates) and the supplier (through production re-scheduling costs, induced by delayed components).

The composition of the modules' stock (number of different modules stocked and constitution of each module) impacts the delivery time (through the final assembly time). This article exploits the performance of simulated annealing to design such a composition.

The problem of interest is described in section 2. The formalization is proposed in section 3. The use of simulated annealing is discussed in section 4. Computational results are exposed in section 5.

2 Problem presentation

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 her/his individual needs.

For a company, it necessitates to design and manufacture diversified products. This enlarges the possibilities to be closer to the requirements of each customer. In the same time, this apparent large diversity must be manage with the lowest technical diversity to guaranty small costs. That may be accomplished using product families [3].

The automotive industry is a leader considering the importance of the diversity that is offered to the market. This important diversity also impacts the automotive suppliers. For example, the functional diversity offered on the cars, has repercussions on the *electrical wire harnesses*. The diversity of a "middle range" electrical wire harness (for a middle range car like "Peugeot 306") supports 15 functions that may be available up to 9 versions. It follows a potentiality of 7 million different wire harnesses for a unique car model. This has to be compared with the annual sales volume which reaches 350 000 vehicles.

In such a context of large diversity, a make to stock strategy may not be a relevant alternative. It will necessitate stocking many items that would never be ordered. On the contrary a make to order strategy necessitates a longer lead time, it may not be acceptable for the customers.

An assemble to order strategy may take advantage of a multi site organization. This enables a relocated production of modules on sites where the production costs are the smallest. Reactivity and synchronization is supported with the sites that are close to the contractor, where the final assembly operations are done. This reactivity may be guaranteed by a pertinent definition of the modules and their selection.

This issue is present in different industrial sectors. When product diversity is large, groupings of basic components are often necessary. It may be operated with data-mining techniques [4]. In the computer business, for example, peripheral equipments as well as motherboards are pre-assembled whereas software are also pre-grouped [5].

A packaging to stock policy allows a better inventory management and maintains the capacity to answer specific demands by grouping of packages. Picking in relative small selections of door styles, finishes, functional storage solutions and decorative enhancements, permits to customize quasi-immediately large sets of different kitchens [6].

The question addressed in this paper considers the selection of a modules' composition that minimizes the final assembly time.

Several points of view may be considered for this problem: geometrical (the design of the modules takes into account the components' physical constraints [7]), ergonomic (the modules are adapted to the assembly tasks). Functional approaches are also of interest in [8], [9].

Modular design also includes strategic thoughts, such as the choice between an intern and an extern production [10]. The criticality of the technologies concerned or the firm's core competencies are then important criterions [11].

Different points of view may be combined; the design becomes then even more complex. Simulated annealing has been used to design modules taking into account both physical and functional relations among components [12].

This study focuses on a design that considers the minimization of the mean assembly time. This approach is pertinent for industries with short lead time constraints.

3 Formalization

3.1 Notations

Consider the following notations: a_i is a component, $i \in [0, n]$, n is the number of components.

Suppose there are no exclusive or inclusive relationships between components: from n components it is then possible to have $2^n - 1$ different products P_i (the product without options is not considered). Let $\mathbb{P}$ be the set of final products.

Call $p(P_i)$ the probability that a demand is for product $P_i \in \mathbb{P}$. The final demand can then be summarized by the scalar $D = (p(P_1)p(P_2)\dots p(P_{2^n-1}))$.

A stock composition C is a selection of products that are used as modules, $C \subset \mathbb{P}$. Let $C[k] = 1$ when product P_k belongs to the composition, $C[k] = 0$ otherwise. Note $|C|$ the number of modules in stock ($|C| = \sum_{k=1}^{2^n-1} C[k]$).

Note $TA_D(C)$ the mean assembly time to assemble a demand D from the stock composition C.

A standardization of the products P_i is not accepted. No product could have any extra components. Also the products can not have any doublet (only different options are assembled).

3.2 Objective function

A stock composition C is evaluated with 2 criteria:

1. The number of modules, $|C|$
2. The mean assembly time, $TA_D(C)$.

If the number of modules $|C|$ is fixed equal to TS, optimization concerns the minimization of the mean assembly time $TA_D(C)$. Information about products demand defines a stochastic environment.

Mean assembly time minimization makes sense if there are many products. In a stochastic environment, mean time corresponds to the expected value of the assembly time.

The problem is, for a demand D, to select the stock composition of size TS that minimizes the mean assembly time. Assembly time is supposed to be proportional to the number of assembly operations. Mean assembly time is then assimilated to the expected value of the number of assembly operations.

It is thus necessary to evaluate the performance of a stock composition according to this criterion. Note $NA(P_i, C)$ the number of assembly operations necessary to the realization of the product P_i starting from a stock composition C. Equation 1 shows the relation between the number of assembly operations and the mean time.

$$TA_D(C) = \sum_i p(P_i) \times NA(P_i, C) \tag{1}$$

The determination of $NA(P_i, C)$ is explained in section 4.1.

The problem is formalized as follows:

$$\min_C TA_D(C) \tag{2}$$

S.T.

$$\begin{cases} TS \text{ modules in stock} \Leftrightarrow |C| = TS \\ \text{all the products are made without standardization} \end{cases}$$

4 Simulated annealing

The objective function can't be expressed in terms of elementary functions. Indeed the mean assembly time is the result of combined effects of the stocked modules (Cf. section 4.1). A gradient research was therefore not to be considered. Since its introduction in the 80's [13], simulated annealing proved to be a good solution for a broad range of problems [14].

To describe the application of the simulated annealing method: the state space (§ 4.2), the move set (§ 4.3) and the score function (§ 4.1) are explicated. The uphill moves occurrence (the schedule) is also given (§ 4.4).

4.1 Evaluation function

In order to evaluate the objective function, the term $NA(P_i, C)$ has to be evaluated.

To determine the number of assembly operations needed to obtain P_i from C, an assembly route-sheet has to be found. Indeed for each final product a set of modules of the stock which, by assembly, make it possible to obtain this given product has to be given.

This problem is known in the literature as **Set Cover** (**SC**). This problem is NP-hard [15]. Furthermore a result of 1995 showed that the best approximation factor is $O(log\, n)$ ([16], chap. 10).

The constraint of assembling only different options (Cf. §3.1) inducts a particular case also called **Set partition**, this problem is also NP-hard [15].

Here a greedy algorithm was chosen to evaluate the route-sheet. This approximation is quite efficient and enable to have rapidly a solution. Note the number of modules used to obtain the P_i from C, $\sharp modules(P_i, C)$. The solution given is at most $H(maxcompo)$ time the optimal, where $maxcompo$ is the number of components of the most complex module in the composition C and $H(n) = \sum_{i=1}^{n} \frac{1}{i}$ [17],[18].

$$NA(P_i, C) = \sharp modules(P_i, C) - 1 \tag{3}$$

4.2 Search space

The search space consists in the compositions guaranteeing the creation of all the final products without standardization. Note $\mathfrak{C}$ the set of the acceptable compositions.

The obligation to be able to assemble the $2^n - 1$ final products without standardization enables a characterization of the acceptable compositions:

$$C \in \mathfrak{C} \Leftrightarrow (a_i \in C \quad i = 1, \ldots, n \, and \, |C| = TS) \tag{4}$$

Equation 4 translates the fact that every single final product can be assembled from the basic components.

The problem is then:

$$\min_{C \in \mathfrak{C}} TA_D(C) \tag{5}$$

4.3 Move set

The proposed generation of neighbours only produces acceptable configurations.

Call neighbour of a composition C each composition C' such as:

$$\begin{cases} \exists i, j \in [[1, 2^n - 1]] \\ \forall k \in [[1, 2^n - 1]] \setminus \{i, j\} \quad C'[k] = C[k] \\ C'[i] = (1 - C[i]) \\ C'[j] = (1 - C[j]) \\ a_l \in C' \quad l = 1, \ldots, n \end{cases}$$

This definition guarantees that a neighbour does belong to the search space.

ALGORITHM ***neighbour***
INPUTS composition C
OUTPUTS composition C'

```
For k := 1 to 2^n − 1
  C'[k] = C[k]
End For
newcomposition← FALSE
While newcomposition = FALSE
  // choose 2 different integers in [[1; 2^n − 1]]
  i ← Uniform{1, .., 2^n − 1}
  j ← Uniform{1, .., 2^n − 1} \ i
  C'[i] = (1 − C[i])
  C'[j] = (1 − C[j])
  If a_l ∈ C'  l = 1, ..., n
    newcomposition← TRUE
  Else
    C'[i] = C[i]
    C'[j] = C[j]
  End If
End While
Return C'
End
```

The following algorithm (***accepted***) gives the "move pattern". It express how a neighbour becomes the actual solution.

ALGORITHM ***accepted***
INPUTS increase δTA_D, temperature T
OUTPUTS boolean
  If $\delta TA_D < 0$
    Return TRUE
  Else
    $A \leftarrow e^{-\frac{\delta TA_D}{T}}$ // *the degree of uphill movement*
    $d \leftarrow U[0,1]$
    If $d < A$
      Return TRUE
    Else
      Return FALSE
    End If
  End If
End

4.4 Schedule

The schedule must be controlled in such a way that the process converges to an optimum. Here an exponential cooling scheme was used. The parameter α does then belong to [0,1[. The initial temperature T_0 is problem specific, it is estimated by conducting an initial search in which all increases are accepted. The average objective increase observed $\Delta^+ TA_D$ is calculated. T_0 is given by:

$$T_0 = -\frac{\Delta^+ TA_D}{ln(x_0)}$$

with x_0 the initial average increase acceptance probability [13].

ALGORITHM ***newtemperature***
INPUTS actual temperature T
OUTPUTS new temperature $newT$
  $newT \leftarrow T \times \alpha$
  Return $newT$
End

4.5 Implementation

Let's now present the implementation of the simulated annealing algorithm.

ALGORITHM ***simulated annealing***
INPUTS initial solution C_0, $\sharp iterations$
OUTPUTS best solution TA_D^*
  $C \leftarrow C_0$
  $T \leftarrow T_0$
  $TA_D^* \leftarrow TA_D(C)$
  $m \leftarrow 0$
  While $m < \sharp iterations$
    $C_1 \leftarrow$ ***neighbour*** (C)
    $\delta TA_D \leftarrow TA_D(C_1) - TA_D(C)$
    If ***accepted*** $(\delta TA_D, T)$
      $C \leftarrow C_1$
    End If
    $T \leftarrow$ ***newtemperature***(T)
    $m \leftarrow m + 1$
    If $TA_D(C) < TA_D^*$
      $TA_D^* \leftarrow TA_D(C)$
    End If
  End While
  Return C
End

4.6 Initial composition

Consider a 2 stages approach: the first stage being the research of a "better than random" initial configuration, the second a simulated annealing beginning from this configuration.

The first step to find the optimal solution was to look for the "better solution" characteristics. Our bias was to suppose that good solutions were composed of small and frequently used modules. Therefore several heuristics based on those 2 ideas were implemented. Those heuristics were tested and proved to perform well [19].

5 Computational results

To test the proposed methods, numerous tests were lead. The instances were constituted of 5 components (n=5) i.e. 31 different final products.

5.1 Simulated annealing evolution

Figure 1 presents, for a given demand, the evolution of the solution TA_D^* (Y axis) in function of the number of iterations (X axis). It is then possible to compare the evolution pattern for the 3 initial points (random, frequency, and size). The curves represent 2 values of TS: $TS = 12$ and $TS = 20$. The tests were lead with $\alpha = 0.5$ and an initial average increase acceptance probability x_0 of 33%.

The initial mean assembly times obtained with the heuristics are better than the ones obtained by random. The curves show that the obtained compositions are better (smaller mean assembly time) when the number of iterations enlarges. After an important number of iterations, the random initial composition may lead to a better solution than the ones having an heuristical initial composition.

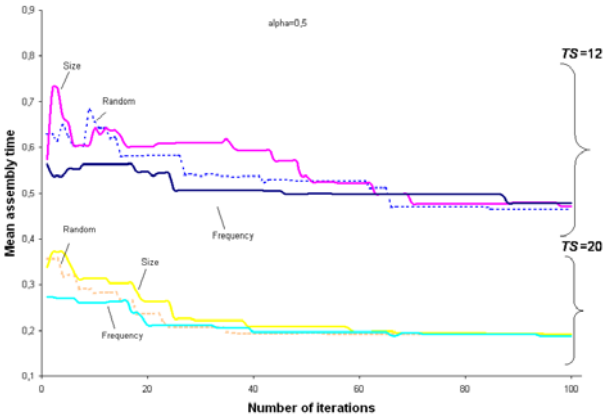

Figure 1: Evolution of $TA_D(C)$ for $\alpha = 0.5$.

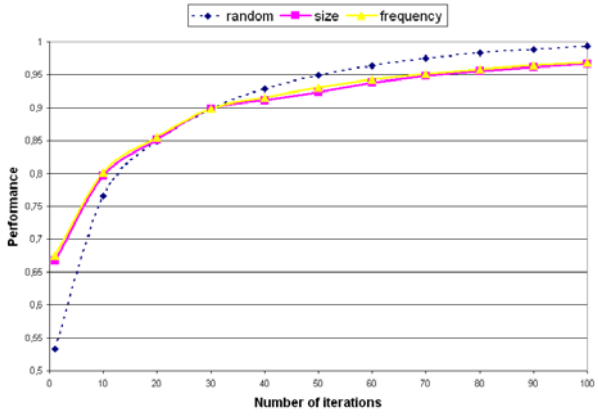

Figure 2: Mean performances in function of number of iterations.

5.2 Performances

In this section, the aggregated results of 100 instances are presented. The performance of the simulated annealing algorithm is studied, considering the nature of the initial point.

The performance of a modules' composition is evaluated with consideration for the mean assembly time extremes. It is defined as follows:

$$performance(C) = \frac{TA_{max} - TA(C)}{TA_{max} - TA_{min}}$$

where TA_{max} and TA_{min} are respectively the worst and the better solution found.

This indicator was chosen for its synthetic characteristics. The composition (TA_{min}) has then a performance of 1 while the less performing composition (TA_{max}) has a performance of 0. The performance of a method is evaluated trough the mean (on the TS) of the compositions it determines.

Figure 2 gives the mean performance of the simulated annealing algorithm for the 3 different initial values in function of the numbers of iterations.

The results, summarized in table 1, show that the performances of the obtained compositions are better when the number of iterations enlarges.

Surprisingly, when the number of iterations is larger than 50, the performance achieved with a random initial point is better than the ones obtained with the 2 stages approach. This may be explicated by the fact that solutions given by the heuristics based on size or frequency are in a local "valley" from which it is difficult to escape.

If the time at disposal for the simulated annealing complexion is sufficient to allow a important number of iterations, priority should then be given to a random initial point. In the opposite case, the 2 stages approach should be prefered.

	random	***frequency***	***size***
initial value	0.55	0.7	0.7
final value after 10 iterations	0.76	0.79	0.78
final value after 20 iterations	0.84	0.85	0.85
final value after 50 iterations	0.94	0.93	0.93
final value after 100 iterations	0.99	0.97	0.97

Table 1: Mean performances, $\alpha = 0.5$.

6 Conclusion

This article proposed to use a simulated annealing algorithm to address the problem of module design focusing on minimizing the mean assembly time. This method proved to perform well on the numerous instances tested. The importance of the initial population was also stressed. Indeed, computational results stress that when the number of iterations allowed is important, a random initial value performs better. Nevertheless, when iterations are limited a "better than random" initial configuration should be used.

Further research may concern the evaluation of the costs generated by this design. Different costs may be integrated: transportation cost, inventory costs, production costs. Also different production costs may be considered depending on the design of the supply chain (relocated pro-

duction for the modules).

References

[1] Martin Starr, "Modular production - a new concept," *Harvard business review*, pp. 131–142, nov.-dec. 1965.

[2] Gérard Cachon, *Supply Chain Management: Design, coordination and operation*, vol. 11 of *Handbooks in Operations Research and Management Science*, chapter Supply chain coordination with contracts, pp. 229–340, Ed. Graves, Steve and de Kok, Ton, 2003.

[3] Bruno Agard, "Conception et fabrication des familles de produits : Etat de l'art," *Journal Européen des Systèmes Automatisés*, vol. 38, no. 1-2, pp. 59–84, 2004.

[4] Bruno Agard and Andrew Kusiak, "Data mining for subassembly selection," *ASME Transactions: Journal of Manufacturing Science and Engineering*, vol. 126, no. 3, pp. 627–631, 2004.

[5] Carliss Baldwin and Kim Clark, "Managing in an age of modularity," *Harvard business review*, pp. 84–93, September-october 1997.

[6] Diana Twede, Robert Clarke, and Jill Tait, "Packaging postponement: a global packaging strategy," *Packaging technology and science*, vol. 13, no. 3, pp. 105–115, 2000.

[7] Karl Ulrich and Karen Tung, "Fundamentals of product modularity," in *Proceedings of the ASME Winter annual meeting symposium on issues in design/Manufacturing integration*, 1991.

[8] Chun-Che Huang and Andrew Kusiak, "Modularity in design of products and systems," *IEEE Transactions on systems, man and cybernetics*, vol. 28, no. 1, pp. 66–77, January 1998.

[9] Andrew Kusiak and Chun-Che Huang, "Development of modular products," *IEEE transactions on components and manufacturing technology*, vol. 19, pp. 523–538, 1996.

[10] Juliana Hsuan Mikkola, "Modularity, outsourcing, and inter-firming learning," *R& D Management*, vol. 33, no. 4, pp. 439–454, 2003.

[11] Amiya Chakravarty and Nagraj Balakrishnan, "Achieving product variety through optimal choice of module variations," *IIE Transactions*, vol. 33, no. 7, pp. 587–598, 2001.

[12] Victor Kreng and Tseng-Pin Lee, "Modular product design with grouping genetic algorithm-a case study," *Computers and industrial engineering*, vol. 46, no. 3, pp. 443–460, 2004.

[13] Scott Kirkpatrick, Charles Gelatt, and Mario Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[14] Ralph Otten and Lukas van Ginneken, *The annealing algorithm*, Kluwer academic publishers, 1989.

[15] Michael Garey and David Johnson, *Computers and Intractability: A guide to the theory of NP-completeness*, W. H. Freeman and Company, San Francisco, 1979.

[16] Dorit Hochbaum, Ed., *Approximation algorithms for NP-hard problems*, PWS publishing company, 1997.

[17] David Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, vol. 9, pp. 256–278, 1974.

[18] Vasek Chvatal, "A greedy heuristic for the set covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233– 235, 1979.

[19] Catherine da Cunha, *Définition et gestion de produits semi-finis en vue d'une production de type assemblage à la commande*, Ph.D. thesis, INP Grenoble, 2004.