The finite horizon economic lot sizing problem in job shops: the multiple cycle approach

Citation for published version (APA):

Document status and date:
Published: 01/01/2001

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.
Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.

Download date: 27. Oct. 2021
The finite horizon economic lot sizing problem in job shops: the multiple cycle approach

J. Ouenniche\textsuperscript{a,b,*}, J.W.M. Bertrand\textsuperscript{b}

\textsuperscript{a}Faculty of Business Administration, Lakehead University, 955 Oliver Road, Thunder Bay, Ont, Canada P4B-SE1
\textsuperscript{b}Department of Operations Planning and Control, Faculty of Technology Management, Eindhoven University of Technology, P.O. Box 513, Paviljoen H12, 5600 MB, Eindhoven, Netherlands

Abstract

This paper addresses the multi-product, finite horizon, static demand, sequencing, lot sizing and scheduling problem in a job shop environment where the planning horizon length is finite and fixed by management. The objective pursued is to minimize the sum of setup costs, and work-in-process and finished products inventory holding costs while demand is fulfilled without backlogging. We propose a new and efficient cyclic scheduling solution framework, called the multiple cycle (MC) method, based on the assumption that the cycle time of each product is an integer multiple of a basic period. This method relies on a decomposition approach which decomposes the problem into an assignment sub-problem, a sequencing sub-problem and a lot sizing and scheduling sub-problem. To evaluate its performance, the MC method was compared to the common cycle method and numerical results show that it performs better, as expected. However, the magnitude of improvement varies between 4\% and 8\% depending on the structure of the problems. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Sequencing; Lot sizing; Cyclic scheduling; Job shop

1. Introduction and problem definition

This paper addresses the problem of making sequencing, lot sizing and scheduling decisions for several products, say \( n \), manufactured through a job shop where all parameters are deterministic and constant over a given planning horizon length, say \( H \), which is assumed to be fixed by management. The objective pursued is to minimize the sum of setup costs, work-in-process (WIP) inventory holding costs and finished products inventory holding costs while a given demand is fulfilled without backlogging. We consider a job shop with \( m \) machines and assume that there is only one machine of each type and that each product \( i \) has a unique serial route through the shop which will be indicated by an ordered subset of machines and denoted by \( \rho(i,:) \); in other words, we assume that routing decisions have been made and, based on some criteria, a unique serial route has been chosen for each product. In addition, we assume that preemption is not allowed. To solve this problem, we assume that the cycle time of each product \( i \), say \( T_i \), is an integer multiple, say \( m_i \), of a basic period \( F \); that is, \( T_i = m_i F \) for all \( i \). In addition, we require that the basic period \( F \) be such that the planning horizon \( H \) is an integer multiple of the global cycle

---

*Corresponding author. Tel.: +1-807-343-8525; fax: +1-807-343-8443.
E-mail address: jamal.ouenniche@lakeheadu.ca (J. Ouenniche).
MF, that is, \( H = \xi MF \) where \( \xi \) is an integer and \( M \) denotes the least common multiple (LCM) of \( m_i \)'s. This problem can be formulated as a mixed non-linear program that simultaneously determines all the relevant decisions. However, for large or even medium size instances, the solution of this model may require a prohibitive amount of computational time. Consequently, in this paper, we propose a solution method, called the multiple cycle (MC) method, which decomposes the problem into three sub-problems; namely, an assignment sub-problem, a sequencing sub-problem and a lot sizing and scheduling sub-problem. The first two sub-problems are solved using heuristics while the third sub-problem is solved to optimality. Throughout this paper, we will use the notation given in Fig. 1.

To the best of our knowledge, the only contribution to this problem is reported in Ouenniche and Boctor [1]. Recall that the classical job shop problem has attracted the attention of many researchers (e.g., [3–10]). This problem consists of determining, for a given set of non-splittable jobs, the execution sequences and the starting dates so as to optimize some objective function. However, this problem is quite different from the one considered in this paper, as the classical job shop problem does not deal with lot sizing issues. Attention has also been paid
to the lot size issue in the stochastic version of the job shop problem but without simultaneously considering the sequencing and scheduling phase (see [11,12]). A recent exception to this is the work of Lambrecht et al. [13] who integrate lot sizing, release and sequencing decisions through a hierarchical approach with feedback adjustment mechanisms.

The remainder of this paper is organized as follows. In Section 2, we present the general framework of the MC method. In Section 3, we show how to assign products to basic periods. In Section 4 we show how to determine sequencing decisions and in Section 5 we present a mathematical formulation of the lot sizing and scheduling subproblem and show how to solve it. Finally, Section 6 provides some numerical results and Section 7 presents the conclusions of this research.

2. The multiple cycle heuristic

In this section, we propose a new and efficient heuristic to solve the multi-product, finite horizon, static demand, sequencing, lot sizing and scheduling problem in a job shop environment where the objective is to minimize the sum of setup costs, work-in-process inventory holding costs and finished products inventory holding costs while demand is fulfilled without backlogging. It is assumed that the planning horizon length \( H \) is finite and fixed by the management team as is usually the case in practice. This heuristic, called the multiple cycle (MC) method, assumes that cycle times \( T_i \) are integer multiples of a basic period, \( F \); that is, \( T_i = m_i F \) for all \( i \), where \( m_i \) denote the integer multiple of product \( i \). This leads to a global cycle of length \( MF \) where \( M \) denotes the least common multiple of \( m_i \)'s. In addition, we require that the basic period \( F \) be such that the planning horizon \( H \) is an integer multiple of the global cycle \( MF \); that is, \( H = \xi MF \) where \( \xi \) is an integer.

In summary, for a given vector of time multiples \( (m_i; i = 1, \ldots, n) \), the MC method first assigns products to the basic periods of the global cycle using a simple heuristic (see Section 3). Then sequencing decisions \( \{\sigma^k_1, \ldots, \sigma^k_m; k = 1, \ldots, M\} \) are determined using either simulated annealing or tabu search (see Section 4). Both the assignment of products to the basic periods of the global cycle \( (P_1, \ldots, P_M) \) and sequencing decisions \( \{\sigma^k_1, \ldots, \sigma^k_m; k = 1, \ldots, M\} \) are used as an input to a mixed non-linear model (see Fig. 4) that simultaneously determines lot sizing and scheduling decisions as well as the integer multiple \( \xi \) (see Section 5). Actually, the sequencing, lot sizing and scheduling subproblem in embedded in the assignment subproblem to evaluate an assignment of products to the basic periods of the global cycle and the corresponding vector of time multipliers, whereas the lot sizing and scheduling subproblem is embedded in the sequencing subproblem to evaluate sequencing decisions; that is, the corresponding total cost.

In order to end up with a solution which is at least as good as the common cycle (CC) solution, if any exists, a bounding scheme is used to check whether a vector of time multipliers should be evaluated in detail or not by comparing a lower bound on the total cost corresponding to this vector of multipliers to the current upper bound on total cost; that is, the smallest total cost found so far where its initial value is set to the total cost of the CC solution (i.e., \( m_i = 1 \) for all \( i \)). If the lower bound is greater than the upper bound, this vector of multipliers needs no further evaluation as the corresponding schedule will necessarily have a total cost higher than that of the CC solution; else, the vector of time multipliers is evaluated in detail by calling upon the assignment routine. If the assignment obtained is feasible, then we compute sequencing, lot sizing and scheduling decisions as well as the corresponding total cost.

Within the MC method, we first determine a list of candidate vectors of multipliers where a candidate vector is defined as a vector of multipliers whose lower bound on total cost is not dominated by the CC solution. Then, the candidate vectors within this list are ordered in the ascending order of the corresponding lower bounds on total cost and the candidate vector on the top of the list (i.e., the one having the smallest lower bound on total cost) is evaluated; that is, we determine a feasible assignment of products to basic periods, sequencing, lot sizing and scheduling decisions as well as the integer multiple \( \xi \). If we fail to determine a feasible assignment or a feasible set of sequences for this
current candidate vector or the solution obtained is not better than the CC solution then, we proceed with the evaluation of the second candidate on the top of the list and so on. We calculate a lower bound on the total cost \( TC_{ib}(m_1, ..., m_n) \) corresponding to a given vector of setup costs \((m_i; i = 1, ..., n)\) as follows (see Section 5): 
\[
TC_{ib}(m_1, ..., m_n) = \min_{\xi=1, ..., \zeta, \alpha} \{TC_{ib}(M, \xi)\}
\]

The general framework of the MC method may be summarized as follows:

**Initialization step**

- Compute the optimal independent cycle time \( T^*_i \) of each product \( i \) as follows:

\[
T^*_i = \min \left[ \max \left( \frac{2s_i}{h_i r_i (1 - r_i / p_{i,\rho(i,a)})} + \frac{\sum_{j=2}^{\mu_i} h_i, \rho(i,j-1) r_i^2 (1 - 1 / p_{i,\rho(i,j)})}{1 - r_i / p_{i,\rho(i,j-1)}}, \text{maximum} \left\{ \frac{t_{i,\rho(i,j)}}{1 - r_i / p_{i,\rho(i,j)}} \right\} \right], \text{subject to}
\]

\[
\begin{align*}
(1) \quad & d_{i,\rho(i,j-1)} + \frac{r_i T_i}{p_{i,\rho(i,j-1)}} \leq d_{i,\rho(i,j)}, \\
& j = 2, ..., \mu_i \\
(2) \quad & t_{i,\rho(i,j)} + \frac{r_i T_i}{p_{i,\rho(i,j)}} \leq T_i, \quad \forall j, \\
(3) \quad & 0 \leq T_i \leq H,
\end{align*}
\]

This value \( T^*_i \) corresponds to the global minimum of the following mathematical program:

Minimize
\[
\frac{s_i}{T_i} + h_i m_i r_i \left( 1 - \frac{r_i}{p_{i,\rho(i,a)}} \right) + \sum_{j=2}^{\mu_i} h_i, \rho(i,j-1) r_i^2 \left( \frac{1}{p_{i,\rho(i,j)}} - \frac{1}{p_{i,\rho(i,j-1)}} \right) \]
\[
+ \sum_{j=2}^{\mu_i} h_i, \rho(i,j-1) r_i (d_{i,\rho(i,j)} - d_{i,\rho(i,j-1)})
\]

subject to

\[
\begin{align*}
(1) \quad & d_{i,\rho(i,j-1)} + \frac{r_i T_i}{p_{i,\rho(i,j-1)}} \leq d_{i,\rho(i,j)}, \\
& j = 2, ..., \mu_i \\
(2) \quad & t_{i,\rho(i,j)} + \frac{r_i T_i}{p_{i,\rho(i,j)}} \leq T_i, \quad \forall j, \\
(3) \quad & 0 \leq T_i \leq H,
\end{align*}
\]

where (i) the objective function is the sum of setup cost, work-in-process inventory holding cost and finished product inventory holding cost of product \( i \) per unit of time, (ii) the first set of constraints state that, at each stage on the route of product \( i \), processing cannot take place before it is completed at its previous stage, (iii) the second set of constraints implies that, at each stage, the cycle time \( T_i \) must be sufficiently large to allow setup and processing operations, and (iv) the third constraint set lower and upper bounds on \( T_i \).

- Set \( m_i = 1 \) for all \( i \) and determine sequencing decisions (see Section 4) as well as lot sizing and scheduling decisions and the integer multiple \( \zeta \) (see Section 5). Store the common cycle (CC) solution as the best solution found so far and initialize the upper bound on total cost, denoted \( TC_{ub} \), to the optimal cost of the CC solution.

**Main step**

- Compute a list of candidate vectors of multipliers \( m_i; i = 1, ..., n \) as follows. Start with the initial vector of multipliers:

\[
m_i = \left\lceil \frac{T^*_i}{T_{\min}} \right\rceil i,
\]

where \( T_{\min} = \min_{i=1, ..., n} \{T^*_i\} \) and \( \lceil x \rceil \) denotes the smallest integer greater than or equal to \( x \). Compute the corresponding lower bound on total cost \( TC_{ib}(m_1, ..., m_n) \). If \( TC_{ib}(m_1, ..., m_n) \leq TC_{ub} \), then store it as a candidate vector. As long as the current vector of multipliers is not the unit vector, derive a new vector of multipliers by reducing by 1 the largest time multiplier within the current vector and compute the corresponding lower bound on total cost. Again, if \( TC_{ib}(m_1, ..., m_n) \leq TC_{ub} \), store it as a candidate vector.
vector. Note that if several products have the largest time multiplier, we reduce the multiplier of the product (among those having the largest multiplier) having the smallest optimal independent cycle time $T^\psi$.

- Order the list of candidate vectors in ascending order of their lower bound on total cost and evaluate in detail the candidate vector on the top of the list (i.e., the one having the smallest lower bound on total cost); that is, determine a feasible assignment of products to basic periods (see Section 3) and sequencing decisions (see Section 4), lot sizing and scheduling decisions as well as the integer multiple $\zeta$ (see Section 5). If we fail to determine a feasible assignment or a feasible set of sequences for the current candidate vector or the solution obtained is not better than the CC solution, proceed with the evaluation of the next candidate vector on the top of the list.

3. Assigning products to basic periods

Given a time multiple $m_i$ for each product $i$, we need to assign products to the basic periods of the global cycle so that Model I (see Fig. 4) has a non-empty set of feasible solutions. In this case, we say that the assignment is feasible. As all products having a time multiple $m_i = 1$ should be produced in each basic period of the global cycle, the only remaining question is how to assign those products having a time multiple $m_i > 1$. To overcome the computational burden of enumerating all assignments and choose among those that are feasible the best one according to some criterion, we propose hereafter a heuristic procedure to assign products to basic periods so that a necessary feasibility condition of Model I is met; that is, $\phi_k < 1$ where

$$\phi_k = \max_{1 \leq j \leq m} \left\{ \sum_{i \in P_k} m_i r_{i,j} \right\},$$

$$k = 1, \ldots, M,$$  \hspace{1cm} (3)

and $P_k$ denotes the subset of products assigned to basic period $k$. The assignment heuristic that we propose is outlined in the following steps:

**Step 1:** Sort the products in ascending order of $m_i$ and, within the products having the same multiplier $m_i$, in the descending order of $\lambda_i$ where

$$\lambda_i = \max_{j \in P(i, \cdot)} \{m_i r_{i,j} / p_{i,j}\}.$$

**Step 2:** Set $P_k = \{i/m_i = 1\}$ for all $k$. Among the products which are not yet assigned to any subsets $P_k$, choose the one having the largest value of $\lambda$ where $\lambda_i = \max_{j \in P(i, \cdot)} \{m_i r_{i,j} / p_{i,j}\}$, say product number $s$, and assign it to the basic periods $k, k + m_s, k + 2m_s, \ldots$ or equivalently to the subsets $P_k, P_{k+m}, P_{k+2m}, \ldots$ so as to minimize

$$\max_{k = 1, \ldots, M} \left\{ \max_{j = 1, \ldots, m} \left( \sum_{i \in P_k \cup \{s\}} m_i r_{i,j} / p_{i,j} \right) \right\}.$$  \hspace{1cm} (4)

This process is repeated as long as there are some non-assigned products. Notice that this step seeks to balance the load through all basic periods of the global cycle and through all stages. If $\phi_k < 1$ for all $k$, then the assignment obtained is feasible and the procedure stops; else, move products from an infeasible subset to a feasible one or exchange products between infeasible and feasible subsets so as to obtain a feasible assignment. If we still fail to obtain a feasible assignment, proceed with step 3.

**Step 3:** Set $P_k = \{i/m_i = 1\}$ for all $k$. Each product $i$ such that $m_i > 1$ is assigned to the basic periods $k, k + m_i, k + 2m_i, \ldots$ or equivalently to the subsets $P_k, P_{k+m}, P_{k+2m}, \ldots$, where $k \leq m_i$, that satisfies

$$\max_{j = 1, \ldots, m} \left( \sum_{i \in P_k \cup \{i\}} m_i r_{i,j} / p_{i,j} \right) < 1,$$  \hspace{1cm} (5)

$$t = k, k + m_i, k + 2m_i, \ldots, k + M - m_i.$$

If we succeed in assigning all products to basic periods, then the assignment obtained is feasible. On the other hand, if for a given product, say product number $s$, we fail to find a period $k$, within the first $m_s$ basic periods of the global cycle, which satisfies the above condition, we proceed as follows. Within the first $m_s$ basic periods, we look for a product $i \in P_k, 1 \leq k \leq m_s$, such that $m_i > 1$ which can be transferred to another period $t \leq m_i$ without violating the feasibility condition

$$\sum_{w \in P, w \leq t} m_w r_{w,j} / p_{w,j} + m_i r_{i,j} / p_{i,j} < 1, \quad j = 1, \ldots, m,$$

$$b = 0, 1, \ldots, M / m_i - 1,$$  \hspace{1cm} (6)
Choose an initial set of sequences/seed $\sigma_j$ and use Model I to calculate the corresponding minimum cost $c(\sigma_j)$, set $\sigma' = \sigma_j$ and $c' = c(\sigma_j)$;
Choose an initial temperature $\vartheta$ and set $\tau = \vartheta$;
Set temperature change counter $t = 0$;
REPEAT until $\tau < \varepsilon$
  Set repetition counter $r = 0$;
  REPEAT until $r = r'$
    Generate a neighbor set of sequences $\sigma'$ of the current seed $\sigma_j$ and use Model I to determine $c(\sigma')$;
    Calculate $\delta = c(\sigma_j) - c(\sigma')$;
    IF $\delta > 0$ or Random(0, 1) < $e^{\delta\tau}$ THEN set $\sigma_j = \sigma'$ and $c(\sigma_j) = c(\sigma')$;
    IF $c(\sigma') < c'$ THEN set $\sigma' = \sigma'$ and $c' = c(\sigma')$;
    $r = r + 1$;
  END REPEAT
  $t = t + 1$;
  $\tau = \alpha \tau$;
END REPEAT.

Fig. 2. The suggested adaptation of the simulated annealing algorithm.

and leaving enough time to allow assigning product $s$ to the periods $k, k + m_s, k + 2m_s, \ldots, k + M - m_s$ where $k \leq m_s$. Notice that this procedure may fail to find a feasible assignment. In this case, as shown in Section 2, we try another vector of time multipliers.

4. Sequencing

Given a time multiple $m_i$ for each product $i$ and a feasible assignment of products to the basic periods of the global cycle $(P_1, \ldots, P_M)$, we need to determine sequencing decisions $\{\sigma^k_j; j = 1, \ldots, m, k = 1, \ldots, M\}$ where $\sigma^k_j$ denote the processing order of those products which require processing at stage $j$ in basic period $k$. Note that sequencing decisions $\sigma^k_j$ should be determined so that Model I has a non-empty set of feasible solutions. In this case, we say that the set of sequences $\{\sigma^k_j; j = 1, \ldots, m, k = 1, \ldots, M\}$ is feasible. Also, note that a necessary feasibility condition of Model I is that if products $i$ and $u$ require processing at a given stage $j$ in the shop and both belong to $P_k$ for a given basic period $k$, then if product $i$ is ordered before product $u$ in $\sigma^k_j$, for all basic periods $t$ such that $i, u \in P_t$.

In this section, we propose two algorithms for sequencing. The first algorithm is an adaptation of the simulated annealing algorithm and the second one is an adaptation of tabu search.

4.1. A simulated annealing algorithm

Hereafter, we propose an adaptation of the simulated annealing (SA) algorithm for sequencing several products in a job shop environment so as to minimize total cost (see Fig. 2).

Within this algorithm, $\varepsilon$ takes a relatively small value, $r'_{\text{max}}$ (the maximum number of neighbors to be evaluated at a given temperature) depends on $t$ (the temperature change counter) as follows: $r'_{\text{max}} = \beta t$ where $\beta$ is a constant $(\beta = 10, 20, 30)$, and the temperature $\tau$ is decreased according to a temperature reduction coefficient $\alpha$; that is, $\tau = \alpha \tau$ where $\alpha = 0.70$ and $\tau^0 = 5$. In addition, the initial sequencing decisions $\sigma_0 = \{\sigma^k_j; j = 1, \ldots, m, k = 1, \ldots, M\}$ are randomly generated as follows. First, a so-called “global ordering” of products is chosen at random for each stage $j$, say $\sigma_j$. Then, for each basic period $k$, $\sigma^k_j$ is defined such that if products $i$ and $u$ require processing at stage $j$ and $i$ is ordered before $u$ in $\sigma_j$, then $i$ is also ordered before $u$ in $\sigma^k_j$ for all basic periods $k$ to which both $i$ and $u$ are
Choose an initial set of sequences $\sigma_i$ and use Model I to calculate the corresponding minimum cost $c(\sigma_i)$, set the tabu list $TL_j^l = \emptyset$ for all $j$ and $k$, $\sigma' = \sigma_i$, $c' = c(\sigma)$ and the iteration counter $l = 0$;

REPEAT until $l = l_{\text{max}}$

1. Find the least-cost neighbor set of sequences $\sigma'$ of the current set of sequences $\sigma_i$.
   - If $\sigma'$ is not tabu, set $\sigma_i = \sigma'$;
   - If $\sigma'$ is tabu but better than the solution found so far, set $\sigma_i = \sigma'$; ELSE, find the non-tabu least-cost neighbor set of sequences $\sigma'$ and set $\sigma_i = \sigma'$;

2. Update the tabu lists $TL_j^l$ and increment $l$ by 1;

END REPEAT.

Fig. 3. The suggested adaptation of the tabu search algorithm.

assigned. To generate a neighbor set of sequences, say $\sigma'$, we first choose, at random, a machine number, say $j$, and two positions, say $l_1$ and $l_2$, in $\sigma_j$ then we move the product in position $l_1$ in $\sigma_j$ to position $l_2$ and construct $\sigma_j'$ for all basic periods $k$ accordingly. To evaluate a given set of sequences $\sigma$, Model I is solved to determine the minimum cost that can be achieved while using $\sigma$.

4.2. A tabu search algorithm

Within this adaptation of the tabu search (TS) algorithm (see Fig. 3), a tabu list $TL_j$ is kept for each machine $j$. Each element of $TL_j$ consists of a vector $(i, l_1, l_2)$ where $i$ denotes a product number, $l_1$ denotes the current position of product $i$ in the global sequence $\sigma_i$, and $l_2$ is the position to which product $i$ will be moved. Furthermore, in order to avoid cycling, the size of each tabu list is randomly chosen within the interval [6,10] (see [14]). In addition, the initial set of sequences $\sigma_0$ and neighbor sets of sequences $\sigma'$ are obtained in the same way as suggested within our adaptation of the simulated annealing algorithm. Finally, the maximum number of cycles, denoted $I_{\text{max}}$, is set to 5, 10 and 15, respectively.

5. Optimal lot sizing and scheduling: Mathematical formulation

Given the planning horizon length $H$, a vector of time multipliers $(m_i; i = 1, \ldots, n)$, an assignment of products to the basic periods of the global cycle $(P_1, \ldots, P_M)$ and a production sequence $\sigma_i^k$ for each machine $j$ ($j = 1, \ldots, m$) and basic period $k$ ($k = 1, \ldots, M$), the purpose of Model I (see Fig. 4) is to determine the length of the basic period $F$, the starting times $d_{ij}$ and the integer multiple $r$. Notice that $d_{ij}$ is the processing start time of product $i$ on machine $j$ and that the corresponding setup should be finished before the starting date $d_{ij}$. Also, as backlogging or shortages are not allowed, the processing time of product $i$ on machine $j$, $\pi_{ij} = r m_i F / p_{ij}$.

The objective of Model I is to minimize the sum of set-up costs, WIP inventory holding costs and finished products inventory holding costs per unit of time (see [1]). This objective function is to be minimized subject to six constraints (see Fig. 4). Constraints (1) state that, at each stage, no product can be processed before it is completed at its previous stage. Constraints (2) state that, within each basic period of the global cycle and at each stage, no product can be processed before the completion of its predecessor in the production sequence and setup is made. Constraints (3) state that the processing finish time of each product on the last machine on its route is within the basic period $F$. Constraints (4) state that, for each basic period and at each stage, processing the first product cannot start before setup is made. Constraints (5) require that the basic period $F$ be such that the schedules obtained or the global cycle pattern of decisions obtained could be repeated exactly an integer number of times during the finite planning horizon $H$. Finally, constraints (6) are the non-negativity constraints.
To solve this model, we propose the following iterative procedure:

**Initialization step**

Set \( z^* = z_{\xi} \) and \( F^* = F_{\xi} \), where \( z^* \) denotes the best objective function value found till now, \( F^* \) is the basic period value corresponding to \( z^* \), \( z_{\xi} \) is the objective function value corresponding to \( \xi \) and \( F_{\xi} = H/\xi M \).

**Iterative step**

Increment \( \xi \) by 1, solve Model I for the new value of \( \xi \) and update \( z^* \) and \( F^* \) if necessary.

**Stopping criterion**

Stop if Model I has no feasible solution for the current value of \( \xi \) or \( \xi > \xi_{ub} \).

Note that once \( \xi \) is fixed, the resulting linear model is solved using the commercial mathematical programming code CPLEX [15]. In order to reduce the number of linear models to be solved, we calculate a lower bound on the total cost, say \( TC_{lb}(M, \xi) \), corresponding to given values of \( M \) and \( \xi \) as follows:

\[
TC_{lb}(M, \xi) = \sum_{i} \frac{s_i}{m_iF_{\xi}} + \sum_{i} \left[ h_i \frac{m_ir_i}{2} \left( 1 - \frac{r_i}{p_{i,\rho(i,j)}} \right) + \sum_{j=2}^{\mu_i} h_{i,\rho(i,j)} \frac{m_r^2}{2} \left( \frac{1}{p_{i,\rho(i,j)}} - \frac{1}{p_{i,\rho(i,j-1)}} \right) \right] F_{\xi} + \sum_{i} \sum_{j=2}^{\mu_i} h_{i,\rho(i,j)} r_i (d_{i,\rho(i,j)} - d_{i,\rho(i,j-1)}) .
\]

(7)

where the products are assumed to be scheduled independently and the processing start time of each product at each stage on its route is assumed to start as soon as it is completed at the previous stage; that is

\[
d_{i,\rho(i,j)} - d_{i,\rho(i,j-1)} = \frac{m_r F_{\xi}}{p_{i,\rho(i,j-1)}} .
\]

(8)
Consequently,

\[
TC_{lb}(M, \xi) = \sum_{i} \frac{s_i}{m_i F_\xi} + \sum_{i} \left[ h_i \frac{m_i r_i}{2} \left( 1 - \frac{r_i}{p_{i, \rho(i, u_i)}} \right) + \sum_{j=2}^{\mu_i} h_i, \rho(i, j-1) \frac{m_i r_i^2}{2} \left( \frac{1}{p_{i, \rho(i, j)}} + \frac{1}{p_{i, \rho(i, j-1)}} \right) \right] F_\xi,
\]

where

\[
F_{lb} = \frac{\sum_{i} s_i/m_i}{\sum_{i} \left[ h_i (m_i r_i/2) \left( 1 - \frac{r_i}{p_{i, \rho(i, u_i)}} \right) + \sum_{j=2}^{\mu_i} h_i, \rho(i, j-1) (m_i r_i^2/2) \left( \frac{1}{p_{i, \rho(i, j)}} + \frac{1}{p_{i, \rho(i, j-1)}} \right) \right]}. \tag{10}
\]

Obviously, \( F_{lb} \) is a lower bound on \( F^* \), the optimal value of \( F \), as it is the smallest value of \( F \) that minimizes the objective function of Model I under the first set of constraints only (i.e.; the other constraints are relaxed). Thus, if \( TC_{lb}(M, \xi) > TC_{ub} \), Model I does not need to be solved for the current values of \( M \) and \( \xi \) as the resulting total cost will be necessarily higher that the total cost of the best solution found so far. Furthermore, the largest value of \( \xi \) for which Model I need to evaluated; that is, the corresponding lower bound on total cost \( TC_{lb}(M, \xi) \) computed and Model I solved if necessary, is determined as follows. As \( \xi(F_{lb}) \geq \xi(F^*) \) where \( \xi(F_{lb}) = H/MF \) rounded to the nearest integer, \( \xi_{ub} \) is an upper bound on \( \xi \):

\[
\xi_{ub} = \left\lceil \frac{H}{MF_{lb}} \right\rceil. \tag{11}
\]

6. Performance evaluation

The performance of the proposed adaptations of the simulated annealing (SA) and tabu search (TS) algorithms is studied through their solutions for a total of 270 randomly generated problems with 10 products and 5 machines. To assess their performance, these methods were compared to the solutions obtained by their counterpart methods under a common cycle assumption (see [1]).

6.1. Test problems

We may expect that the difference in improvement over the CC solution will depend on the structure of the problems; in particular, the standard deviation of the optimal independant cycle time \( T^p \) of products (low, medium and large) and the machines utilization rates (low, medium and high); therefore, we generated 9 classes of test problems each consisting of 30 problems that differ with respect to these two factors.

Each of the 270 test problems was randomly generated as follows. First, annual demand rates \( r_i \) are drawn from a uniform distribution between 100 and 1000. Second, the route of each product is randomly generated. Third, production rates \( p_{ij} \) were calculated such that (i) enough time will be available for setup and processing operations and (ii) machines utilization levels vary between 45% and 90% as follows:

\[
p_{ij} = n_j r_i \Omega_{ij}, \quad \text{rounded up to the nearest integer, } \tag{12}
\]

where \( n_j \) is the number of products which require processing at stage \( j \) and \( \Omega_{ij} \) is a real number generated from a uniform distribution over the interval \([1.66, 2.22]\) if machines utilization levels vary between 45% and 60%, \([1.33, 1.66]\) if machines utilization levels vary between 60% and 75% and \([1.11, 1.33]\) if machines utilization levels vary between 75% and 90% where the bounds of each interval are calculated as follows. As \( \Omega_{ij} \) are determined such that

\[
a < \sum_{i} \frac{r_i}{p_{ij}} \leq b, \quad \forall j, \tag{13}
\]

where \( a \) and \( b \) denote the chosen lower and upper bounds on machine utilization level, it follows that:

\[
\frac{1}{b} < \Omega_{j} < \frac{1}{a}, \quad \forall j. \tag{14}
\]
where
\[ \Omega_j = \sum \frac{1}{n_j \omega_{ij}}, \quad \forall j. \] (15)

Fourth, the inventory holding costs were generated based on the value added at each stage which is assumed to depend on the processing times. Finally, setup times are chosen from a uniform distribution between 2 and 12 hours (the planning horizon is assumed to equal 1600 working hours) and setup costs are uniformly generated within the interval that makes the largest optimal independent cycle time \( T_i^* \) lies between 2 and 30 times the smallest one.

### 6.2. Computational results

Each of the 270 test problems was solved two times: using the proposed simulated annealing adaptation and the tabu search algorithm. Tables 1–3 provide for the SA algorithm, the minimum, the maximum, the mean and the standard deviation of their percentage of improvement in total cost over the common cycle (CC) solution for \( \beta = 10, 20 \) and 30, respectively. Also, Tables 4–6 provide for the TS algorithm, the minimum, the maximum, the mean and the standard deviation of their percentage of improvement in total cost over the common cycle (CC) solution for \( \beta = 5, 10, 15 \), respectively. Finally, Tables 7–12 provide some computational times (not CPU times) required on a personal computer with an Intel II processor running at 400 MHz.

From Tables 1–6, we can see that the MC method outperforms the CC method, as expected.

### Table 2
Percentage improvement in total cost over the CC solution – SA
\( \beta = 20 \)

<table>
<thead>
<tr>
<th>Machine utilization</th>
<th>Variance ( (T_i^*) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Min 4.4355</td>
<td>4.6099</td>
</tr>
<tr>
<td>Max 10.4793</td>
<td>10.0140</td>
</tr>
<tr>
<td>Mean 6.4163</td>
<td>7.2523</td>
</tr>
<tr>
<td>Std-dev. 1.5849</td>
<td>1.6033</td>
</tr>
<tr>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>3.1636</td>
</tr>
<tr>
<td>9.5219</td>
<td>11.7555</td>
</tr>
<tr>
<td>5.2492</td>
<td>6.3931</td>
</tr>
<tr>
<td>2.0922</td>
<td>1.9690</td>
</tr>
<tr>
<td>High</td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>1.7450</td>
</tr>
<tr>
<td>8.5469</td>
<td>11.4388</td>
</tr>
<tr>
<td>4.1812</td>
<td>5.7775</td>
</tr>
<tr>
<td>2.5283</td>
<td>2.2359</td>
</tr>
</tbody>
</table>

### Table 3
Percentage improvement in total cost over the CC solution – SA
\( \beta = 30 \)

<table>
<thead>
<tr>
<th>Machine utilization</th>
<th>Variance ( (T_i^*) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Min 4.4355</td>
<td>4.2946</td>
</tr>
<tr>
<td>Max 10.1618</td>
<td>10.4844</td>
</tr>
<tr>
<td>Mean 6.3359</td>
<td>7.0962</td>
</tr>
<tr>
<td>Std-dev. 1.5176</td>
<td>1.6620</td>
</tr>
<tr>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>3.1636</td>
</tr>
<tr>
<td>9.2088</td>
<td>11.1541</td>
</tr>
<tr>
<td>5.1562</td>
<td>6.1191</td>
</tr>
<tr>
<td>1.9612</td>
<td>1.8801</td>
</tr>
<tr>
<td>High</td>
<td></td>
</tr>
<tr>
<td>0.0000</td>
<td>1.7450</td>
</tr>
<tr>
<td>8.2139</td>
<td>11.5004</td>
</tr>
<tr>
<td>3.9965</td>
<td>5.4503</td>
</tr>
<tr>
<td>2.1933</td>
<td>2.1182</td>
</tr>
</tbody>
</table>
However, there is a large variance in the improvement; that is, in some cases there is no improvement at all whereas in other cases, the improvement is substantial. This difference in improvement is due to differences in the structure of the problems; in particular, the standard deviation of the optimal independent cycle times $T^*_i$ and the machines utilization rates. The numerical results show that the improvement in total cost over the CC solution ranges from an average of 4% for problems with high capacity utilization and small variance of $T^*_i$’s to an average of 8% for problems with a low capacity utilization and high variance of $T^*_i$’s. This difference in improvement may be explained as follows. If all $T^*_i$ values are within a relatively small range, the MC method cannot produce

<table>
<thead>
<tr>
<th>Machine utilization</th>
<th>Variance ($T^*_i$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low</td>
</tr>
<tr>
<td><strong>Low</strong></td>
<td>Min 4.7901</td>
</tr>
<tr>
<td></td>
<td>Max 9.7441</td>
</tr>
<tr>
<td></td>
<td>Mean 6.5819</td>
</tr>
<tr>
<td></td>
<td>Std-dev. 1.4776</td>
</tr>
<tr>
<td><strong>Medium</strong></td>
<td>2.0317</td>
</tr>
<tr>
<td></td>
<td>9.8378</td>
</tr>
<tr>
<td></td>
<td>5.7657</td>
</tr>
<tr>
<td></td>
<td>1.8404</td>
</tr>
<tr>
<td><strong>High</strong></td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>11.8310</td>
</tr>
<tr>
<td></td>
<td>5.0355</td>
</tr>
<tr>
<td></td>
<td>3.3386</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine utilization</th>
<th>Variance ($T^*_i$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low</td>
</tr>
<tr>
<td><strong>Low</strong></td>
<td>Min 3.7209</td>
</tr>
<tr>
<td></td>
<td>Max 9.3030</td>
</tr>
<tr>
<td></td>
<td>Mean 5.9621</td>
</tr>
<tr>
<td></td>
<td>Std-dev. 1.3996</td>
</tr>
<tr>
<td><strong>Medium</strong></td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>8.2753</td>
</tr>
<tr>
<td></td>
<td>4.6212</td>
</tr>
<tr>
<td></td>
<td>1.7949</td>
</tr>
<tr>
<td><strong>High</strong></td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>10.5804</td>
</tr>
<tr>
<td></td>
<td>3.8006</td>
</tr>
<tr>
<td></td>
<td>2.5374</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine utilization</th>
<th>Variance ($T^*_i$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low</td>
</tr>
<tr>
<td><strong>Low</strong></td>
<td>Min 7</td>
</tr>
<tr>
<td></td>
<td>Max 32</td>
</tr>
<tr>
<td></td>
<td>Mean 16</td>
</tr>
<tr>
<td></td>
<td>Std-dev. 6</td>
</tr>
<tr>
<td><strong>Medium</strong></td>
<td>49</td>
</tr>
<tr>
<td></td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>10</td>
</tr>
<tr>
<td><strong>High</strong></td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>239</td>
</tr>
<tr>
<td></td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>49</td>
</tr>
</tbody>
</table>

Table 4  
Percentage improvement in total cost over the CC solution – TS ($I_{max} = 5$)  

Table 5  
Percentage improvement in total cost over the CC solution – TS ($I_{max} = 10$)  

Table 6  
Percentage improvement in total cost over the CC solution – TS ($I_{max} = 15$)  

Table 7  
Computer time required by SA ($\beta = 10$)
a significant cost improvement as the products time multipliers are likely to have the same value which is the ideal situation for the CC method to produce the best solution. In addition, if the $T^*$ values cover a large range, substantial cost improvements are likely to be obtained, as the products time multipliers are now quite different which privileges the MC method. On the other hand, it is easy to see that when the demands involved lead to high machine utilization rates, it becomes more difficult to construct a feasible schedule with many different multipliers’ values. Consequently, the MC method may fail to obtain a large cost improvement.

The MC method is quite efficient. Computational times are on average 2.5 times the computational times required to solve the same problems with the CC method. For many situations in practice, this extra computational time seems to be justified by the improvement in performance. Finally, we observe that SA and TS perform
Table 12

<table>
<thead>
<tr>
<th>Machine utilization</th>
<th>Variance ($T^2$)</th>
<th>Low</th>
<th>Medium</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td></td>
<td>Min 63</td>
<td>59</td>
<td>73</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>194</td>
<td>209</td>
<td>335</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>113</td>
<td>132</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td>Std-dev.</td>
<td>34</td>
<td>39</td>
<td>50</td>
</tr>
<tr>
<td>Medium</td>
<td></td>
<td>55</td>
<td>71</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>295</td>
<td>258</td>
<td>269</td>
</tr>
<tr>
<td></td>
<td></td>
<td>138</td>
<td>138</td>
<td>126</td>
</tr>
<tr>
<td></td>
<td></td>
<td>43</td>
<td>40</td>
<td>47</td>
</tr>
<tr>
<td>High</td>
<td></td>
<td>72</td>
<td>45</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td></td>
<td>316</td>
<td>362</td>
<td>368</td>
</tr>
<tr>
<td></td>
<td></td>
<td>149</td>
<td>160</td>
<td>152</td>
</tr>
<tr>
<td></td>
<td></td>
<td>51</td>
<td>77</td>
<td>86</td>
</tr>
</tbody>
</table>

equally well; however, TS seems to be more time consuming.

7. Conclusions

In this paper, we proposed a new and efficient heuristic, called the multiple cycle (MC) method, to solve the multi-product, finite horizon, static demand, sequencing, lot sizing and scheduling problem in a deterministic job shop environment where the planning horizon length is fixed by the management team. This heuristic assumes that products cycle times are integer multiples of a basic period and requires that the basic period be such that the planning horizon length is an integer multiple of the global cycle length. The MC method decomposes the problem into three sub-problems: an assignment sub-problem, a sequencing sub-problem and a lot sizing and scheduling sub-problem. The first two sub-problems are solved using heuristics while the third one is solved to optimality using mathematical programming. Actually, the sequencing, lot sizing and scheduling subproblem in embedded in the assignment subproblem to evaluate an assignment of products to basic periods and the corresponding vector of time multipliers where-as the lot sizing and scheduling subproblem is embedded in the sequencing subproblem to evaluate sequencing decisions. The MC method was compared to the common cycle method and numerical results show that it performs much better. However, the magnitude of performance depends on the problem structure.

References