UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação
ISSN 0103-2569

A Heuristic for Multilevel Capacitated Lot Sizing with Parallel Machines

Maristela Oliveira dos Santos
Marcos Nereu Arenales
Regina Esther Berretta
Vinícius Amaral Armentano

Nº 129

RELATÓRIOS TÉCNICOS

São Carlos – SP
Dez./2000

A HEURISTIC FOR MULTILEVEL CAPACITATED LOT SIZING WITH PARALLEL MACHINES

Maristela Oliveira dos Santos[1], Marcos Nereu Arenales[1], Regina Esther Berretta[2] and Vinícius Amaral Armentano[3]

{mari; arenales@icmc.sc.usp.br}
berretta@bestway.com.br
vinicius@densis.fee.unicamp.br

[1]Departamento de Ciências de Computação e Estatística
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo Campus de São Carlos
Caixa Postal 668
13560-970, São Carlos, SP
[2]Centro de Ciências Exatas e Tecnológicas,
Universidade São Francisco, 13251-900 Itatiba, SP, Brazil

[3]Faculdade de Engenharia Elétrica e de Computação,
Universidade Estadual de Campinas, CP 6101, CEP 13083-970, Campinas, SP, Brazil

Abstract: This article discusses how a production plan is determined for a number of ordered products (final items) and their components to meet the demand in each period with a finite horizon. A setup time is considered to start production on any machine in any period. The objective is to minimize the costs involved in this process. The problem is formulated as a mixed integer programming model and a heuristic based on the short term memory component of the tabu search method is proposed. By relaxing capacity constraints, an initial but possibly infeasible solution is obtained by a dynamic programming algorithm. Production shifts are performed between periods and machines in an attempt to find a feasible solution. If a feasible solution is found, an improvement procedure is applied to try to obtain improved quality solutions. For small instances, heuristic solutions are compared with optimal solutions obtained by CPLEX 4.0. On the other hand, for larger instances for which CPLEX 4.0 cannot find an optimal or even feasible solution, the quality of the solutions is evaluated by a lower bound provided by Lagrangian relaxation.

Keywords: Production Planning, MRP, Multilevel Lot-sizing, Heuristics.

Resumo: Este artigo considera um problema que consiste em determinar um plano de produção que atenda a demanda dos itens finais e de seus componentes em cada período de um horizonte finito de planejamento. Um tempo de preparação é considerado para começar a produção em qualquer máquina e período. O objetivo é determinar um plano de produção que minimize os custos de produção, preparação e de estoque. O problema é formulado como um programa inteiro misto e um método heurístico baseado em estratégias de memória de curto prazo da busca tabu é proposto. Relaxando as restrições de capacidade, uma solução inicial, possivelmente infactível, é obtida utilizando um algoritmo de programação dinâmica, o qual é aplicado primeiramente para determinar a produção dos itens finais e, então, de seus componentes. Em seguida, transferências de produção entre períodos e máquinas tentam encontrar uma solução factível. Se uma solução factível é determinada, um procedimento de melhoria é aplicado para tentar obter soluções de qualidade melhores. Para os exemplos de dimensões pequenas, as soluções heurísticas foram comparadas com as soluções ótimas obtidas pelo pacote CPLEX 4.0. Para os exemplos maiores, os resultados obtidos foram analisados considerando o limitante inferior obtido através da técnica da Relaxação Lagrangiana.

Palavras-chaves: Planejamento da Produção, MRP, Dimensionamento de lotes Multiestágio, Heurística.

1. INTRODUCTION

This article addresses the problem of lot-sizing in a multistage production system, in which each stage consists of parallel machines with a finite capacity. The multistage lot-sizing problem consists of determining the production plan that supplies the demand of a number of products (final items) and their components, without exceeding the available production capacity. The demand for final items and their components is given in each period of a finite horizon and backlogging is not allowed. The objective is to minimize production, setup and inventory costs. Setup time and cost for each item are considered to start production on any machine in a given period. General and serial product structures are considered.

Lot-sizing problems arise, usually, in MRP systems (Material Requirements Planning), which define the lot sizes of final items and its components in order to supply the demand. However, generally, MRP systems do not include capacity constraints for production, or the costs involved. If the MRP solution is infeasible, the Capacity Requirements Planning (CRP)

procedure is applied to adjust the production plan (see Tempelmeier, 1997; Maes and Wassenhove, 1991).

The literature contains several good surveys that consider general aspects of lot-sizing problems, i.e., Billington *et al.* (1983), Bahl *et al.* (1987) and Kuik *et al.* (1994). In addition, Drexel and Kimms (1997) present several lot-sizing and scheduling models.

From the standpoint of computational complexity, when capacity is limited and setup costs are non-zero, the optimization problem is NP-Hard (Florian *et al.*, 1980). Maes *et al.* (1991) showed that the feasibility problem is NP-Complete when setup times are considered. Due to this computational complexity, most solution methods are heuristic. Among the authors who address this issue are Katok *et al.* (1998), França *et al.* (1997), Tempelmeier and Derstroff (1996), Tempelmeier and Helber (1994), and Billington *et al.* (1986). However, our approach to parallel machines differs from that of the above cited authors. Some authors who have discussed the parallel machine environment are Lasdon and Terjung (1971), Carreno (1990), Sabbag (1993) and Toledo (1998), but they all involve monostage systems. In this paper, we discuss the above cited articles on the multilevel lot-sizing problem.

Billington *et al.* (1986) consider the lot-sizing problem based on setup cost, setup time, and a single constrained resource. They propose a branch-and-bound strategy with Lagrangian relaxation to solve sub-problems and use a smoothing procedure to eliminate unfeasibilities. Tempelmeier and Helber (1994) work with multistage lot-sizing problems with multiple constrained resources and setup cost (no setup times). The procedure presented by Tempelmeier and Helber (1994) combines a modified cost approach by Blackburn and Millen (1982) and Heinrich and Schneeweisss (1986), and the Dixon and Silver (1981) procedure for single-stage capacitated lot sizing. Tempelmeier and Derstroff (1996) propose a heuristic approach based on Lagrangian relaxation for multistage lot-sizing problems with multiple constrained resources, setup cost and setup time. They use the Langrangian relaxation of the multi-level inventory balance constraints and capacity constraints. Thus, they use subgradient optimization to compute a lower bound. A feasible solution is built by an approach that shifts production from overloaded periods.

França *et al.* (1997) consider the lot-sizing problem as do Tempelmeier and Derstroff (1996), using a formulation in terms of echelon stock and proposing a heuristic method based on production shifts. Their results are compared with a lower bound obtained through Lagrangian relaxation. Katok *et al.* (1998) present a heuristic method that is an extension of the Coefficient Modification Heuristic of Harrison and Lewis (Harrison and Lewis, 1995) to handle setup costs and general assembly structures. They evaluate the performance of the

heuristic by comparing its solutions against the optimal solutions, obtained by OSL, for small instances. For medium size instances, they compare the heuristic with time-truncated OSL solutions.

This paper is organized as follows. Section 2 presents a mathematical formulation for Multistage Lot-sizing Problems (MLSP), in which each stage is composed of parallel machines with limited capacity. Setup costs and setup times are included. Section 3 proposes a heuristic while section 4 presents a summary of the computational results.

2. PROBLEM FORMULATION

The costs involved are production, inventory and setup. The available capacity is limited, and is taken up with production and setup times. Products and components are sometimes simply referred to as items, without loss of generality. The notation used is as follows:

Indices:

$t = 1,...,T$ - Number of periods.

$k = 1,...,K$ - Number of machines.

$i = 1,...,N$ - Number of items.

Sets

$S(i)$ - Set of immediate successors of item i.

$P(i)$ - Set of immediate predecessors of item i.

Parameters

a_{ij} – Number of units of item i required to produce one unit of item j.

d_{it} - independent demand of item i in period t.

h_{it} - unit inventory cost of item i in period t.

c_{itk} - unit production cost of item i in period t on machine k.

cs_{ik} - setup cost incurred if component i is produced on machine k.

b_{ik} - unit amount of resource used in the production of component i on machine k.

ts_{ik} - setup time of component i on machine k.

CAP_{tk} - amount of capacity available in period t on machine k (in units of time).

M - large number.

Variables

X_{itk} - lot size of item i in period t on machine k.

I_{it} - inventory of item i at the end of period t

Y_{itk} - 1, if component i is produced in period t on machine k; 0, otherwise.

The problem can be formulated as

(MLSP)

$$\text{Minimize } Z = \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K}\left(c_{itk}X_{itk} + cs_{ik}Y_{itk}\right) + \sum_{i=1}^{N}\sum_{t=1}^{T} h_{it}I_{it} \qquad (1)$$

Subject to:

$$\sum_{k=1}^{K} X_{itk} + I_{i(t-1)} - I_{it} = d_{it} + \sum_{j \in S(i)} a_{ij} \sum_{k=1}^{K} X_{jtk}, \qquad \forall\ i,t \quad (2)$$

$$\sum_{i=1}^{N}\left(b_{ik}X_{itk} + ts_{ik}Y_{itk}\right) \le CAP_{tk}, \qquad \forall\ k,t \quad (3)$$

$$X_{itk} \le M\,Y_{itk} \qquad \forall\ i,k,t \quad (4)$$

$$I_{it} \ge 0,\ X_{itk} \ge 0,\ Y_{itk} = 0,1 \qquad \forall\ i,k,t \quad (5)$$

The objective function (1) represents production, inventory and setup costs. Constraints (2) represent the inventory balance constraints. The capacity constraints (3) state that the capacity available for production and setup are limited. Constraints (4) ensure that X_{itk}=0, if Y_{itk}=0. Initial and final inventory (I_{i0} and I_{iT}, $\forall i$) are considered to be zero.

3. A HEURISTIC METHOD

The heuristic consists of four procedures. The first creates an initial solution for the problem without considering the capacity constraints. This solution is generally infeasible for the original problem. If this solution is infeasible (INFEA, in Figure 1), a smoothing procedure attempts to find a feasible solution (FEA, in Figure 1) starting from the initial solution. If a feasible solution is found, an improvement procedure tries to identify another solution with a lower cost. However, a feasible solution may be not found after the smoothing procedure, so a pertubation procedure is applied. This procedure identifies a new initial solution for the smoothing procedure, which is again applied. If no feasible solution is found after this procedure, the heuristic fails in its purpose of finding a feasible solution or the problem is, indeed, unsolvable. The main steps of the heuristic are shown in Figure 1. A similar heuristic approach is proposed by França *et al.* (1997) and Toledo (1998).

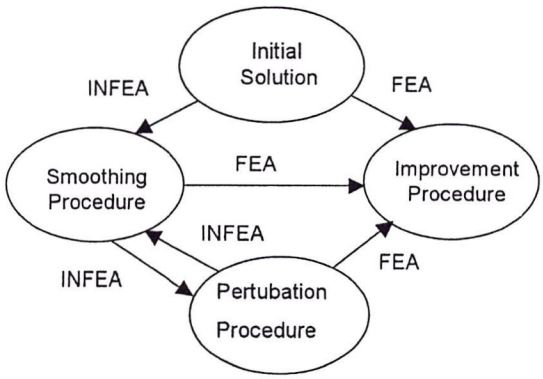

Figure 1. Heuristic method for the problem (1)-(5).

3.1. Initial Solution

The WW-algorithm (Wagner and Whitin, 1958) solves the problem with a single item and a single machine with no capacity constraints. Armentano and Toledo (1997) extended the WW- algorithm to parallel machines. This algorithm was used in our study to obtain an initial solution, by descending the product structure, in which the final items are located at the top (see Figure 2), as described below.

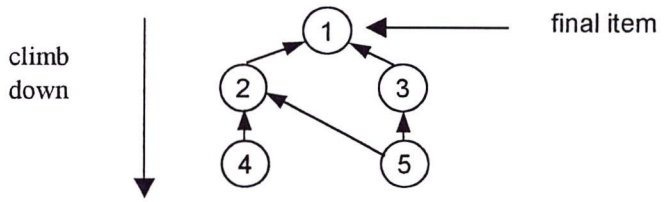

Figure 2. General product structure with 5 items.

Initially, the capacity constraints (3) are relaxed. Production of the final items is initially determined (note that the right hand side of the constraints (2) consists only of the independent demand because $S(i)=\varnothing$). Next, the production of non-final items is determined by fixing the previously defined production of successive items, which appears on the right hand side of (2). Note that, after N problems with a single item and parallel machines are solved, their solutions jointly form a single solution to the original problem. This solution may be infeasible, violating the capacity constraints (3). If that is the case, a smoothing procedure is applied, as described below. Otherwise, in the case of a feasible solution, the improvement procedure described in section 3.3 is applied.

3.2. Smoothing Procedure

The initial solution obtained with the previous procedure is usually infeasible in terms of capacity constraints. If so, shifting production between periods and machines is performed to reach a feasible solution. This shifting may be based on a single item or on multiple items. Similar procedures have been presented by Trigeiro *et al.* (1987), Toledo (1998), França *et al.* (1997) and Tempelmeier and Derstroff (1996). The procedure consists of two main steps: backward and forward in time.

Let $\Delta_{kt} = \sum_{i=1}^{N} (b_{ik} X_{itk} + ts_{ik} Y_{itk}) - CAP_{tk}$ be the slack variables of the capacity constraints (3). The excess of resources used on machine k in period t is given by $Max(0,\Delta_{kt})$. Assume that $\Delta_{kt}>0$. To eliminate this excess, we consider a shift of an item i produced in the pair (k,t) (i.e., machine k, period t) to another pair (k',t') with $k \neq k'$ or $t \neq t'$. Let q be the quantity of item i shifted, which can be taken as:

$$q = min\{ Z_{itk}^{t'k'},\ Q_{itk}$$

where:

- $Z_{itk}^{t'k'}$ is the maximum quantity of item i produced on machine k that can be shifted from period t to machine k' and period t', respecting the balance inventory constraints (this quantity depends on whether the shift is backwards or forwards).
- $Q_{itk} = \frac{\Delta_{kt}}{b_{ik}}$ is the exact quantity of item i to be shifted in order to eliminate the excess of machine k and period t.

Next, the backward and forward steps are described in detail where the quantities $Z_{itk}^{t'k'}$ are determined.

- **Backward step**

The backward step analyzes the periods $T,...,2$ and machines $k=1,...,K$, in that order. The periods that are allowed to receive the production are chosen from among $\tau \leq t' \leq t$, where τ = *maximum {1, the last period in which there is production of component i before period t}*.

When a shift of q units of item i to period $t'<t$ has been done, the inventories of all the immediate predecessors of item i are reduced between periods $t-1$ and t', while the inventories of item i are increased with q units between the same periods. Thus, the quantity of item i to be shifted from period t and machine k to period t' and machine k' is limited by:

$$Z_{itk}^{t'k'} = min\left\{ X_{itk},\ \min_{j \in P(i)} \left\{ \frac{I_j}{a_{ji}},\ t' \leq \ \leq t-1 \right\} \right\} \quad \text{for } t' < t. \tag{6}$$

If $t=t'$, we execute only shifts between machines; therefore $Z_{itk}^{t'k'} = X_{itk}$. If $Z_{itk}^{t'k'} = 0$ for all items that are produced on *(k,t)*, then a multiple item shift (if items are not produced in *(k,t)*, they can be produced in any $k'' \neq k$ in period *t*) is attempted in order to allow for one item shift of the an item produced in *(k,t)*. Thus, in the backward step, all the items that are immediate predecessors or not of the item *i* produced in *(k'',t)*, $k'' \neq k$ should be analyzed. The transference of multiple items is only made between adjacent periods, and is interrupted as soon as the transference of the first item in the violated pair (k,t) becomes possible.

- **Forward Step**

Now, the original periods analyzed are $t=1,...,T-1$ and the target period can be one in the interval $t \leq t' \leq \tau$, where τ = *minimum{T, first period in which there is production of item i after period t}*.

The shift of *q* units of item *i* to period $t'>t$ causes a reduction of the inventory of this item of *q* units in period *t* until the period $t'-1$, while the inventory of immediate predecessor items will increase during the same period. Thus, the quantity of item *i* to be shifted from period *t*, machine *k* into period *t'* and machine *k'* is limited by:

$$Z_{itk}^{t'k'} = min\left\{ X_{itk},\ \min_{t \leq \tau \leq t'-1} \{ I_{i\tau} \} \right\} \quad \text{for t'>t} \tag{7}$$

If $Z_{itk}^{t'k'} = 0$ for all items that are produced in the *(k,t)* pair, then an attempt is made to execute multiple items shifts. Thus, in the forward case, all the items that are immediate successors or not of item *i* produced in *(k'',t)*, $k'' \neq k$ are analyzed, as discussed in the backward step.

- **Choosing the triple: (item, machine, period)**

After the quantity of each item *i* that can be transferred, from the infeasible pair (k,t) into any other pair (k', t'), has been calculated, we are ready to determine the trio (i,k',t'). This choice is based on the cost variation caused by the shift and on the resources used in (k,t) and (k',t'). The triple (i,t',k') is chosen so as to minimize the following ratio (França *et al.*, 1997):

$$\text{ratio} = \frac{\Delta\text{cost} + FP.\text{Penalty}}{\Delta\text{redution}} \tag{8}$$

The cost variation (*Δcost*) caused by the shift of quantity *q* of item *i* from (k,t) into (k',t') can be expressed as

$$cost = \frac{q\left((c_{it'k'} - c_{itk}) + step\sum \hat{h}_i \right) + su1 - su2}{\text{current\_solution\_cost}}$$

where,

$$\hat{h}_i = h_i - \sum_{j \in P(i)} a_{ji} h_j \tag{9}$$

and, in case of:

backward step: τ=t',...,t-1 *and step=1,*

forward step: τ=t,...,t'-1 *and step=-1*

t'=t: step=0.

$$\text{su1} = \begin{cases} cs_{i,k'} & if \quad X_{i,t',k'} = 0 \\ 0 & if \quad X_{i,t',k'} > 0 \end{cases}$$

$$\text{su2} = \begin{cases} cs_{ik} & if \quad q = X_{itk} \\ 0 & if \quad q < X_{itk} \end{cases}$$

- *Penalty* is a non-negative term that can be interpreted as the cost of the excess in (k,t) and (k',t') given by:

$$Penalty = \frac{Exc_{kt}}{CAP_{kt}} + \frac{\text{Exc}_{k't'} - \max\{0, \ _{kt}\}}{CAP_{k't'}}$$

where,

$$\text{Exc}_{k,t} = maximum\{0, (\ _{kt} - (q.b_{ik} + ts_{ik}\ _1))\}$$

$$\text{Exc}_{k',t'} = maximum\{0, (\ _{k't'} + (q.b_{ik'} + ts_{ik'}\ _2)\}$$

$$\nu_1 = \begin{cases} 1 & if \quad q = X_{itk} \\ 0 & if \quad q < X_{itk}, \end{cases}$$

$$\nu_2 = \begin{cases} 1 & if \quad X_{it'k'} = 0 \\ 0 & if \quad X_{it'k'} > 0. \end{cases}$$

- *FP* in (8) represents a weight given to the *Penalty,* which is increased according to the difficulty of finding a feasible solution. *FP=n* in the *nth* backward and forward step.
- Finally, *Δreduction* is given by:

$$\Delta\text{reduction} = \frac{qb_{ik} + ts_{ik}\nu_1}{Cap_{tk}}.$$

If a feasible solution is not obtained when the backward step is ended, the forward step is then applied. Next, if the forward step has been concluded and a feasible solution has still

not been found, the backward step is again applied. This procedure, called the *smoothing procedure*, is repeated until either a feasible solution is obtained or a maximum number of iterations is reached.

However, a drawback that has often been detected during the smoothing procedure is the occurrence of a cyclical pattern of shifts among machines in the same period or between forward and backward steps. To avoid the cyclical pattern among machines and between the forward and backward steps, we have used strategies based on the short term memory of tabu search. (Glover, 1989, 1990). The use of this kind of memory is used to restrict the search, preventing certain solutions (or moves) from the recent past to be revisited.

A move is the transfer of a production quantity of an item i from a machine k and a period t into a machine k' and period t'. The attributes that characterize this movement are formed by the elements i, t, k, t' and k'. The attributes that we have considered to characterize the executed movements are (i,t,k) and (i, t',k').

A three-dimensional matrix (i,t,k) is considered to classify the attributes as tabu. Each position of the matrix represents an attribute and the value stored indicates if the attribute is classified as tabu or not. The values of the matrix start with zero and, for each movement, the position regarding the attribute of this movement receives a value. This value is the time during which this attribute will be forbidden, i.e., when executing a forward and backward step in time, the movements related to these positions are prohibited during a certain number of iterations. We considered different ways to determine this value and found that the best strategy was to generate this time in the interval [1,5]. The rule of tabu activation adopted here was either to prevent the production of item i from leaving (t', k') and returning to (t, k) since this movement had already been made in the recent past (or leaving some other (t'',k'') to (t, k) or (t', k')).

It can be observed that, during the smoothing procedure, if there is no item i with production in the (k,t) pair overload, such as $Z_{itk}^{t'k'} > 0$, then it is due to two facts: (1) all the items that are produced in the (k,t) pair are forbidden to leave this pair (k,t), in other words, all movements are tabu; (2) there is no $Z_{itk}^{t'k'} > 0$ for items i that are produced in (k,t) that can possibly be transferred. In both cases, multiple item shifting is executed and, if it is possible to create a shift for an item that is forbidden to leave the (k,t) pair, then its transfer is allowed. We use this idea as our criterion of aspiration.

3.3. Pertubation Procedure

If the maximum number of iterations to execute the smoothing procedure is reached the first time round and no feasible solution is obtained, the pertubation procedure defines a new starting point for the smoothing procedure from the current solution. For each infeasible solution, the procedure tries to change the production levels and inventory in the first and second period. This strategy was adopted because it was found, during computational experiments, that most infeasible solutions presented violations only in the early periods, and most of them occurred in the first period.

In this procedure, the machines in the first period are considered in decreasing order of capacity, that is, beginning with the machine with the largest capacity and proceeding through the machines with smaller capacity. With the machines now ordered and beginning with the machine with largest production excess, one first attempts to shift production between machines from $t=1$ to $t=2$. If after this first attempt, if the analyzed machine is still violated, an attempt is made to shift items among the machines in $t=1$.

The shift of item i from $t=1$ to $t=2$ is only executed if item i is produced on at least one machine in period $t=2$. This strategy is adopted to guarantee no new setup in $t=2$. If more than one machine produces item i in period $t=2$, the machine with the largest slack is chosen. Shifts among machines in $t=1$ are made sequentially, i.e., from machines with larger excesses to machines with smaller excesses. The shifts among machines are executed without verifying the existence of slack in the destination machine.

After executing the alteration procedure, if the solution obtained is infeasible, then the smoothing procedure is repeated, but with a certain number of iterations. If the solution obtained at the end of the smoothing procedure is feasible, the improvement procedure is then applied to attempt to improve the quality of the solution obtained. Otherwise, the procedure fails in its purpose of determining a feasible solution.

3.4. Improvement Procedure

In this procedure, a feasible solution is used as the starting point and an attempt is made to find another solution with a lower cost. Similarly to the smoothing procedure, it is also based on production shifts forward and backward in time and among machines within a given period. Feasibility, however, is now maintained.

- **Forward Step**

Initially one chooses a target pair (k',t') with non-tight capacity, whose slack is given by: $-\Delta_{k't'}>0$. The target pairs to be analyzed are (k',t') so that $k'=1,...,K$ and $t'=2...,T$, in that order. After (k',t') has been selected, one considers the source pairs (k,t) so that $k=1...,K$ and

$t=1...,t'-1$. For all the items whose production is in t $(<t')$, one considers the quantity q of item i to be transferred from t to t', which is given by

$$q = min\left\{ \frac{max\{0, -(\Delta_{k't'} + v_2 . ts_{ik'})\}}{b_{i,k'}}, Z^{itk}_{k't'} \right\} \qquad (10)$$

The first term in (10) represents the maximum quantity of item i that can be transferred without violating the capacity in (k',t'). The second term represents the maximum quantity that can be transferred according to the smoothing procedure, given by (7). For each target pair (k',t') with non-tight capacity, the source pair to be chosen should be the one that provides the greatest reduction in the total cost of the current solution, given by $\Delta cost$. If the cost variation is negative, then the shift is performed that leads to a cost reduction in the current solution. If (k',t') remains in a state of non-tight capacity after the shift, one searches for another source pair that can reduce the cost of the current solution. If no such pair exists, then one considers the next target pair. The step continues until $(k',t')=(K,T)$.

- **Backward step**

Similar to the forward step, a target pair (k',t') is considered one with non-tight capacity, where $k'=1...,K$ and $t'=T-1...,1$, in that order. The source pairs are (k,t), so that $k=1,...,K$ and $t=T,...,t'+1$. For all the items for production in t $(>t')$, the quantity q of item i to be transferred from t to t' is analogous to the forward step, i.e., equation (10), where $Z^{t'k'}_{itk}$ is given by (6).

The two steps are performed successively until one backward step or forward step is achieved with no shift.

3.5. Lower Bound

The model in section 2 can be rewritten under a change of variable: $E_{it} = I_{it} + \sum_{j \in S(i)} a_{ij} E_{it}$, where, E_{it} is called *echelon stock variable*. The echelon stock concept was introduced by Clark and Scarf (1960) and implemented by Afentakis *et al.* (1984). Its cost is given by $\hat{h}_{it}$ (see (9)). Thus, the constraint (2) can be rewritten in terms of echelon stock variables as:

$$\sum_{k=1}^{K} X_{itk} + E_{i,t-1} - E_{it} = D_{it}.$$

where $D_{it} = d_{it} + \sum_{j \in S(i)} a_{ij} D_{jt}$ is called echelon demand.

Note that $I_{it} \geq 0$ which implies that,

$$E_{it} \geq \sum_{j \in S(i)} a_{ij} E_{it} \qquad (11)$$

The Lagrangian Relaxation is applied to constraint (11), as well as in the capacity constraint (3). When the Lagrangian relaxation of the constraints (3) uses the multipliers $\mu_{tk} \geq 0$ ($t=1,...,T$ and $k=1,...,K$), and the multipliers $\lambda_{it} \geq 0$ ($t = 1,\ldots,T$ e $i = 1,\ldots,N$) for the constraints (11), the problem item is decomposed. The Lagrangian problem can be written as follows:

$$L(\mu,\lambda) = minimize \sum_{i=1}^{N}\sum_{t=1}^{T}\left(ce_{it}E_{it} + \sum_{k=1}^{K} cp_{itk}X_{itk} \right) + \sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{i=1}^{N} csr_{itk}Y_{itk} - Const$$

Subject to:

$$\sum_{k=1}^{K} X_{itk} + E_{i(t-1)} - E_{it} = D_{it} \qquad i = 1,...,N \;\; t = 1,...T$$

$$X_{itk} - MY_{itk} \leq 0 \qquad i = 1,...,N \;\; k = 1,...K \;\; t = 1,...,T$$

$$E_{it} \geq 0,\; X_{itk} \geq 0,\; Y_{itk} \in \{0,1\} \qquad i = 1,...,N \; k = 1,...K \; t = 1,...,T$$

Where:

$$Const = \sum_{t=1}^{T}\sum_{k=1}^{K} \mu_{tk}\, Cap_{tk}\,, \qquad ce_{it} = \hat{h}_{it} - \lambda_{it} + \sum_{j \in P(i)} a_{ji}\,\lambda_{jt}$$

$$cp_{itk} = c_{itk} + b_{ik}\;\mu_{tk} \text{ e}, \qquad csr_{itk} = cs_{ik} + \mu_{tk}\, ts_{ik}$$

The problem remains a single-stage one, with parallel machines decomposed itemwise, and the subproblem for each item is solved by dynamic programming (Armentano and Toledo, 1997). The subgradient method (Camerini *et al.*, 1975) to solve the Lagrangian dual problem is limited to 200 iterations.

4. COMPUTATIONAL RESULTS

We have generated two groups of instances, named G1 and G2. Group G1 is composed of 240 instances of small dimension and, in some of these cases, the CPLEX 4.0 was able to find optimal solutions. Group G2 contains examples of medium dimension. In this case, we estimated the quality of the heuristic solutions with the lower bounds obtained from the dual Lagrangian problem. The characteristics of groups G1 and G2 are described in Table 1. Table 2 shows the intervals used to generate the examples in groups G1 and G2. These values are similar to those used by França *et al.* (1997) and Toledo (1998).

The heuristic was implemented in C language and the computational tests executed in a SUN Ultra-1 workstation.

Table 1. Characteristics of groups G1 and G2

	Group G1	Group G2
structure	Serial and general	Serial and general
N	5	10, 17,40
T	4	6,12,18
K	2,3,4	2,3,4
Setup cost	low and high	low and high
Capacity	loose	Normal
Seeds	20	10
Total	240	1080

Table 2. Parameters used to generate the instances.

Parameters	Interval
c_{itk}	U[1.5,2]
cs_{ik} for low setup cost	U[5,95]
cs_{ik} for high setup cost	U[50,950]
$\hat{h}_{it}$	U[0.2,0.4]
b_{ik}	U[2,3]
ts_{ik}	U[150,250]
d_{it} for end items	U[0,180]
d_{it} for components items	U[0,18]

($x \in U[a, b]$ means that x was uniformly randomly generated in interval [a,b])

The inventory costs were determined by $h_{it} = \hat{h}_{it} + \sum_{j \in P(i)} a_{ji}\, h_{jt}$, where $\hat{h}_{it}$ is the generated echelon holding cost. The a_{ij} parameter was considered to be equal to one. The general structures considered in this article are discussed by Clark and Armentano (1995) (containing 5 items), Clark (1990) (10 and 40 items), and Maes *et al.* (1991) (17 items). The CAP_{tk} capacity was determined as described below.

The capacity for $t \geq 2$ is calculated in such a way that its value is related to the amount of resources necessary to support a lot-for-lot solution. The demand for each item generated in each period is equally divided among the machines. The lot-for-lot method is applied to each machine and the average is calculated based on the number of periods and machines, i.e.,

$$cap = \frac{\sum_{t=2}^{T}\sum_{j=1}^{K}\sum_{i=1}^{N}\left(\frac{D_{it}}{K}\right)b_{ik} + ts_{ik}}{K(T-1)}.$$

The capacities for the periods $t \geq 2$ were fitted using the expression (12), while the capacities of the machines in $t = 1$ were determined so that a feasible solution existed in this period, considering the solution given by the lot-for-lot policy.

$$cap_{tk} = ((10 - K)0.11)\, cap \quad t \geq 2 \tag{12}$$

The procedure to determine these capacities in the first period consists of producing the items using the machines with the longest setup times so as to generate larger capacities in this period. As the machine capacities are positive, if there is no production in machine k' in period $t=1$, that is, $cap_{1\,k'} = 0$ because no item is being produced on this machine, then one sets $cap_{1\,k'} = \sum_{k} cap_{1\,k} \Big/ K$.

The capacities for the instances of group G2 were generated as shown above. The capacities for the instances of group G1 were generated using the same strategies as G2, but the value of cap_{tk} was increased by 20%.

We have considered at most 100 iterations for the smoothing procedure during the first smoothing attempt. An iteration consists of performing backward and forward steps of the smoothing procedure. If this number of iterations is reached and no feasible solution is found, then the pertubation procedure is applied. After the pertubation procedure has been applied, the smoothing procedure is again executed for 50 iterations at the most. If feasibility is not achieved, the method fails or a feasible solution does indeed not exist.

The results for each group are presented separately. Initially, the results for group G1 are given in Table 3, which shows the values of GapO, GapL and GapFea defined as:

$$\text{GapO} = \left(\frac{Z_H - Z_{opt}}{Z_{opt}} \right) * 100,$$

$$\text{GapL} = \left(\frac{Z_H - ZR_{LB}}{ZR_{LB}} \right) * 100, \text{ and}$$

$$\text{GapFea} = \left(\frac{Z_H - Z_{FC}}{Z_{FC}} \right) * 100,$$

where Z_H denotes the objective function value given by (1) returned from the proposed heuristic, Z_{opt} is the optimum objective function value, ZR_{LB} is the lower bound on the minimum value of the objective function obtained by the subgradient method applied to the dual Lagrangian problem, and Z_{FC} is the value of the objective function evaluated based on the feasible (but not optimal) solution produced by CPLEX 4.0.

Note that GapO is only calculated if the CPLEX 4.0 was able to find an optimal solution, and that GapFea is calculated only if the CPLEX 4.0 failed to find an optimal

solution, but identified a feasible solution. The heuristic is also analyzed based on its ability to find feasible solutions, and FEA denotes the percentage of feasible solutions obtained by the proposed heuristic. We have also calculated the percentage of the capacity used. The percentage of resources used for production and setup by each machine in each period is given by:

$$\text{cap\_used} = 100 * \left(\left(\sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{i=1}^{N} (b_{ik} X_{itk} + ts_{ik} Y_{itk}) \Big/ cap_{kt} \right) \Big/ (K * T) \right)$$

The amount of capacity used for machine preparation is given by:

$$\text{setup\_used} = 100 * \left(\left(\sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{i=1}^{N} ts_{ik} Y_{itk} \Big/ (K * T) \right) \Big/ \text{cap\_used} \right)$$

and,

$$\text{FEA} = 100 * \left(\frac{\text{Number of instances that the heuristic found feasible solution}}{\text{Number of generated instances}} \right)$$

These results for group G1 are given in Table 3, where OTM means the percentage of optimal solutions obtained by CPLEX 4.0. Table 3 shows that the lower bounds for instances of high setup cost are rather poor, which may lead to a false conclusion about the lower quality of the heuristic solutions. For example, GapL for serial structure and high setup costs is 27.1, while GapO is 12.7. This finding is also discussed in the literature (Billington *et al.*, 1986, Clark and Armentano, 1995).

Table 3. Results for instances in group G1

Structure	K	Low setup cost					High setup cost				
		OTM	GapO	GapL	GapFea	Cap_used	OTM	GapO	GapL	GapFea	Cap_used
Serial	2	65.0	0.6	1.7	0.4	64.6	90.0	19.8	33.5	13.2	60.1
	3	15.0	0.7	1.6	0.1	58.6	25.0	8.9	22.5	9.2	51.9
	4	0.0	-	-	-1.1	53.5	15.0	9.4	25.4	-2.0	45.7
mean		**40.0**	**0.7**	**1.7**	**-0.2**	**58.9**	**43.3**	**12.7**	**27.1**	**6.8**	**52.6**
General	2	75.0	0.9	2.0	0.6	65.4	85.0	13.0	22.9	11.3	59.2
	3	25.0	1.3	2.5	-0.3	58.8	30.0	7.2	17.4	4.4	52.5
	4	0.0	-	-	-0.3	54.6	15.0	13.7	26.1	2.4	46.1
mean		**50.0**	**1.1**	**2.3**	**0.0**	**57.5**	**43.3**	**11.3**	**22.1**	**6.0**	**52.6**

The heuristic and CPLEX 4.0 obtain 100% of feasible solutions in problems with low and high setup cost. Note that CPLEX was unable to find optimal solutions for instances with low setup costs and four machines, with both general and serial structures. On the other hand, it found 15% of optimum solutions for instances with high setup costs and four machines. This may be ascribed to the fact that production is more spread out among the machines for the instances with low setup costs, which possibly rendered the work of CPLEX more

difficult. Average capacity utilization was more than 50%. The heuristic spent less than 0.11 seconds to solve an instance in this group while CPLEX spent an average of 410 seconds (using the default parameters in CPLEX 4.0).

Table 4 shows the results for instances in group G2, only for GapL. Table 5 indicates the percentage of feasible solutions (FEA) obtained for this group of instances.

Table 4. GapL for instances in group G2

structure	K	Low setup cost			High setup cost		
		N=10	N=17	N=40	N=10	N=17	N=40
Serial	2	3.9	3.0	1.8	43.9	36.8	29.9
	3	2.9	2.0	1.4	38.9	33.7	23.4
	4	3.1	1.9	1.8	42.8	32.7	23.3
mean		**3.3**	**2.3**	**1.7**	**41.6**	**34.4**	**25.5**
General	2	3.8	1.7	3.3	28.4	17.7	21.1
	3	3.8	1.7	3.0	34.1	17.0	18.8
	4	5.0	1.6	3.6	36.2	15.5	20.1
mean		**4.2**	**1.7**	**3.3**	**32.9**	**16.7**	**20.0**

Once again one can see that instances with high setup costs have a higher GapL. One reason for this could be the fact that the initial solution is obtained by a procedure that considers setup cost information but no capacity information. Therefore, the smoothing procedure is rendered difficult for this solution as a starting point with a high level of violation on capacity constraints. Another reason for high GapL may be the low value of the lower bound, already found in instances for which optimal solutions were available. Note also that GapL decreases as N increases.

Table 5. FEA obtained in group G2

Structure	K	Low setup cost			High setup cost		
		N=10	N=17	N=40	N=10	N=17	N=40
Serial	2	93.3	100.0	93.3	90.0	100.0	93.3
	3	100.0	100.0	100.0	100.0	100.0	93.3
	4	96.7	100.0	100.0	100.0	100.0	93.3
mean		**96.7**	**100.0**	**97.8**	**96.7**	**100.0**	**93.3**
General	2	63.3	100.0	53.3	63.3	100.0	36.7
	3	86.7	100.0	86.7	90.0	100.0	53.3
	4	93.3	100.0	86.7	86.7	100.0	63.3
mean		**81.1**	**100.0**	**75.6**	**80.0**	**100.0**	**51.1**

The results shown in Table 5 demonstrate that the heuristic's performance in finding feasible solutions was quite good. A greater number of feasible solutions was found for the instances with low setup costs. One reason for this could be that the production of the initial

solutions obtained by these instances is more spread out among periods and machines as a result of the low setup costs. For example, consider the instances with 40 items, general structure and high setup cost. The heuristic was able to find only 51.1% of a feasible solution while, for instances with low setup costs, the heuristic obtained 75.6% .

Table 6 shows the capacity utilization (cap_used) for the instances in group G2, where each entry is an average value for the instances with 6, 12 and 18 periods. The table also shows the average time spent on setup.

Table 6. Capacity utilization (cap_used) for instances in Group G2

Structure	K	Low setup cost			High setup cost		
		N=10	N=17	N=40	N=10	N=17	N=40
Serial	2	80.2	85.0	91.1	70.7	76.6	88.1
	3	73.5	79.2	87.9	62.6	69.6	85.2
	4	71.4	77.3	88.1	60.7	67.2	85.4
mean		**75.0**	**80.5**	**89.0**	**64.7**	**71.1**	**86.3**
setup_used		**39.3**	**34.8**	**24.8**	**25.0**	**25.4**	**23.4**
General	2	81.4	75.5	88.2	72.1	63.7	80.3
	3	76.7	69.0	84.3	67.4	58.2	77.1
	4	75.4	67.0	84.5	65.9	55.8	77.8
mean		**77.9**	**70.5**	**85.7**	**68.5**	**59.2**	**78.4**
setup_used		**33.8**	**36.7**	**27.6**	**22.2**	**21.6**	**19.8**

Note, in Table 6, that the instances with only two machines present a greater capacity utilization. This consumption decreases by increasing the number of machines. The instances with 40 items are apparently tighter, because they show a larger capacity utilization than the instances with 10 and 17 items. However, this did not affect the performance of the heuristic in obtaining feasible solutions for instances with 40 items and serial structures (see Table 5). One can also note that the average capacity utilization for instances with high setup costs are lower than for instances with low setup costs. This behavior is due to the fact that the heuristic depends on the initial solution determined by the algorithm, which tends to perform a smaller number of setups. Consequently, the average use of capacity will be lower than it is for instances with low setup costs.

5. CONCLUSIONS

In this paper, the multistage lot-sizing problem where each stage is composed of parallel machines with limited capacity was addressed. To the best of the authors' knowledge there is no proposed approach for tackling this problem. A heuristic method was proposed and tested on a number of randomly generated instances. Computational results have shown that

the heuristic presents a good performance regarding its ability to find feasible solutions and also good quality solutions.

The heuristic found feasible solution for all small instances. Moreover, the difference between the optimal solution value and the Lagrangian lower bound suggest the existence of a duality gap, which seems to be larger for high setup costs.

For large instances, the heuristic found 97.5% and 81.4% of feasible solutions for serial structures (low and high setup costs) and for general structures, respectively. Although the gap relative to the Lagrangian lower bound for instances with low setup cost is very low, this was not the case when setup costs are high. Table 7 summarizes such observations.

Table 7. Average values of FEA and GapL in Group G2

	Serials structures		General structures	
	FEA	GapL	FEA	GapL
Low setup cost	98.2	2.4	85.6	3.1
High setup cost	96.7	33.8	77.0	23.2
mean	**97.5**	**18.1**	**81.3**	**13.2**

It should also be pointed out that the average running time spent by the heuristic to solve the larger instances of group G2 was quite low. For example, the average running time was less than 20 seconds and the greatest running time was 103.81 seconds for an instance with 40 items, 18 periods, 4 machines and general structure.

Acknowledgments: The authors are grateful to Dr. Jatinder Gupta for his useful suggestions. This research was supported by FAPESP and CNPq.

6. REFERENCES

Afentakis P., B. Gavish and U. Karmakar (1984). Computationally Efficient Optimal Solutions to the Lot-Sizing Problem in Multistage Assembly Systems. *Management Science*, **30** (2): 222-239.

Armentano, V. A. and F. M. B Toledo (1997). Dynamic Programming Algorithms for the Parallel Machine Lot Sizing Problem. *Pesquisa Operacional*, **17**:137-149.

Bahl, H. C., L. P. Ritzman and J. N. D.Gupta (1987). Determining Lot Sizes and Resource Requirements : A Review. *Operations Research*, **35**(3): 329-345.

Berretta, R. E. (1997). Heurísticas para Otimização do Planejamento da Produção em Sistemas MRP. Tese de Doutorado, FEEC-UNICAMP.

Billington, P. J., J. O. McClain and L. J. Thomas (1983). Mathematical Programming Approaches to Capacity MRP Systems: Review, Formulation and Problem Reduction. *Management Science*, **29**(10): 1126-1141.

Billington, P. J., J. O. McClain and L. J. Thomas (1986). Heuristics for Multilevel Lot-Sizing with a Bottleneck. *Management Science*, **32**(8): 989-1006.

Blackburn, J. D. and R. A Millen (1982). Improved Heuristics for Multi-Stage Requerements Planning Systems. *Management Science*, **28**(1):44-56.

Camarini, P. M., L. Fratta and F. Maffioli (1975). On Improving relation methods by modified Gradient Techniques. *Mathematical Programming Study* **3**: 26-54.

Carreno, J. J. (1990). Economic Lot Scheduling for Multiple Products on Parallel Identical Processors. *Management Science*, **36**: 348-358.

Clark, A. R. (1990). Problemas Multiestágios de Dimensionamento de Lotes com Tempo Não-Zero de Produção e Capacidade Finita. Tese de doutorado, Faculdade de Engenharia Elétrica, UNICAMP.

Clark, A. R. and V. A. Armentano (1995). A Heuristic Resource-Capacitated Multi-Stage Lot-Sizing Problem with Lead Times. *Journal of the Operational Research Society*, **46**(10): 1208-1222.

Clark A. J. and H. Scarf (1960). Optimal Policies for a Multi-Echelon Inventory Problem. *Management Science*, **6**: 475-490.

Dixon, P. S. and E.A. Dilver (1981). A Heuristic Solution Procedure for the Multi-item, Single Level, Limited Capacity Lot-Sizing Problem. *Jounal of Operational Management*, **2**: 23-39.

Drexel A. and A. Kimms (1997). Lot sizing and Scheduling – Survey and Extension. *European Journal of Operation Research*, **99**: 221-235.

Florian, M., J. K. Lenstra and A. H. G. Rinnoy Kan (1980). Deterministic Production Planning Algorithms and Complexity. *Management Science*, **26** (7): 669-679.

França, P. M., V. A. Armentano, R. E. Berretta and A. Clark (1997). A Heuristic Method for Lot-Sizing in Multi-Stage Systems. *Computers & Operations Research,* **24** (9): 861-874.

Glover, F. (1989). Tabu Search – Part I. *Orsa Journal on Computing*, **1** (3): 190-206.

Glover, F. (1990). Tabu Search – Part II. *Orsa Journal on Computing*, **2** (1): 4-32.

Harrison, T. P. and H. S. Lewis (1995). Lot Sizing in Serial Systems with Multiple Constrained Resources. *Management Science*, **41**(11): 19-36.

Heinrich, C. and C. Scheneeweiss (1986). Multi-Stage Lot-Sizing for General Production Systems. Editors, *Lecture Notes in Economics and Mathematical Systems*, **266**: 150-181, Springer Verlag, Heidelberg, Germany.

Katok, E., S. H Lewis and T. P. Harrison (1998). Lot-Sizing in General Assembly Systems with Setup Costs, Setup Times and Multiple Constrained Resources. *Management Science*, **44** (6):859-877.

Kuik, R., M. Salomon and L. N. Van Wassenhove (1994). Batching Decisions: Structure and Models, *European Journal of Operation Research*, **75**: 243-263.

Lasdon, L. S. and R. C. Terjung (1971). An Efficient Algoritm for Multi-Item Scheduling, *Operations Research*, **19**: 946-969.

Maes, J., J. O. McClain and L. N. Van Wassenhove (1991). Multilevel Capacitated LotSizing Complexity and LP based Heuristics, *European Journal of Operational Research*, **53**: 131-148.

Maes J. and L.N. Van Wassenhove (1991). Capacitated Dynamic Lotsizing Heuristic for Serial Systems, *International Journal Production Research*, **29**(6): 1235-1249.

Tempelmeier, H. (1997). Resource-Constrained Material Requeriments Planning - MRP rc, *Production Planning & Control*, **8**(5): 451-461.

Tempelmeier, H. and M. Derstroff (1996). A Lagrangian-based Heuristic for Dynamic Multilevel Multiitem Constrained Lotsizing with *Setup* Times, *Management Science*, **42**(5): 738-757.

Tempelmeier, H. and S. Helber (1994). A Heuristic for Dynamic Multi-item Multi-level Constrained Lotsizing for General Product Structures, *European Journal of Operation Research*, **75**: 296:311.

Toledo, F. M. B. (1998). Dimensionamento de Lotes em Máquinas Paralelas, Tese de Doutorado, FEEC-UNICAMP.

Trigeiro W. W., L. J. Thomas and J. O. McClain (1989). Capacitated Lot Sizing With Setup Times, *Management Science*, **35**(3): 353-366.

Wagner, H. M. and T. M. Whitin (1958). Dynamic Version of the Economic Lot Size Model, *Management Science*, **5** (1): 89-96.

Sabbag, Z. (1993). Planejamento da Produção em Máquinas Paralelas Sob Restrições de Capacidade, Tese de Mestrado, FEE-UNICAMP.