Pergamon

0305-0548(94)00060-3

Computers Ops Res. Vol. 22, No. 7, pp. 669–680, 1995
Copyright © 1995 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0305-0548/95 $9.50+0.00

THE APPLICATION OF VALID INEQUALITIES TO THE MULTI-STAGE LOT-SIZING PROBLEM

Alistair Richard Clark[1]† and Vinicius Amaral Armentano[2]‡§

[1]UniSoma S.A., Campinas, São Paulo, Brazil and [2]Universidade Estadual de Campinas, São Paulo, Brazil

(Received April 1993; in revised form August 1994)

Scope and Purpose—Material Requirements Planning (MRP), widely used in industry, takes into account the component structure of manufactured products to determine a plan for the production of all components over a given planning horizon. However MRP does not take production and inventory costs into account and can produce needlessly expensive plans. The challenge of minimizing total costs has been formulated mathematically as a mixed integer program that is known to be very complex to solve optimally. This paper proposes and tests an efficient cutting plane algorithm to iteratively select valid inequalities that increase the lower bound provided by the formulation's linear programming relaxation. The algorithm, used within a Branch-and-Bound search, greatly improves the identification of good production plans with much stronger optimality guarantees.

Abstract—A capacitated multi-stage lot-sizing problem for general product structures with setup and lead times is considered. The problem is formulated as a mixed integer linear program and valid inequalities that are high dimension faces of the problem's convex hull are identified. A fast separation algorithm is developed to iteratively select those inequalities that cut off the solution of the linear programming relaxation. The resulting much improved lower bounds are used in a Branch-and-Bound algorithm. Computational test results are presented.

1. INTRODUCTION

Material Requirements Planning (MRP) is extensively used in manufacturing for the medium term planning of production in multi-stage systems. Given the end-item production plans specified in the Master Production Schedule (MPS), the MRP method uses an end-item's component structure to calculate when and in what quantity each component must be produced in order to satisfy the MPS. Such calculations are generally carried out on a simple lot-for-lot basis which can result in highly uneconomic production plans when component production setup costs or times are non-trivial. In this case, some merging of production lots will almost certainly reduce total costs, although the contribution of inventory costs will increase. The question of determining lot sizes so as to balance setup and inventory costs in MRP systems is known as the multi-stage lot-sizing problem and can be formulated mathematically as a mixed integer linear program (MILP). The problem is NP-hard [1] and optimal solutions have been obtained only for special cases or small

†Alistair R. Clark is a Production Systems Specialist at UniSoma S.A., an OR consulting and systems development company in Campinas, Brazil. He was formerly an academic adviser at IBM Brazil. His research interests include the modeling of manufacturing systems, scheduling, and combinatorial optimization. He studied Mathematics at the University of Sheffield, has a master's degree in Operational Research from Lancaster University, and a doctorate in Automation from the University of Campinas, Brazil. He has published in the *International Journal of Systems Science, IEEE Proceedings on Decision and Control,* and *Controle e Automação.*

‡Vinicius Amaral Armentano is a Professor in the Faculty of Electrical Engineering at the University of Campinas, São Paulo, Brazil. His research activities embrace the planning and scheduling of manufacturing systems using combinatorial optimization. He received a degree in Electrical Engineering from Mackenzie University, São Paulo, an M.Sc. in Operational Research from the University of Campinas, and a Ph.D. in Control Systems from Imperial College, London University, England. He has published in *Automatica, IEEE Transactions on Automatic Control, SIAM Journal on Control and Optimization, Annals of Operations Research, International Journal of Systems Science,* and other international journals.

§Author to whom correspondence should be addressed at: Departamento de Engenharia de Sistemas, FEE, Universidade Estadual de Campinas (Unicamp), Caixa Postal 6101, Campinas SP, 13081-970, Brazil.

problems [2–10]. Reviews of some of the research can be found in [1, 11]. When tight capacity constraints are included, it can be difficult to find just a feasible solution, let alone an optimal one [12].

This paper develops a solution method that selectively identifies high dimension valid inequalities as cuts of the problem and then enters a Branch-and-Bound search. The method can be used both for problems with and without capacity constraints.

Cuts methods have been used to solve integer programming problems by many researchers, starting with [13] over 30 years ago. In the case of the single-stage lot-sizing problem with no capacity constraints, Barany *et al.* [14] identified valid inequalities that are satisfied by all feasible solutions, but that can cut off the optimal solution of the linear programming relaxation of the formulation. The authors showed that, under certain loose conditions, these cuts are facets of the formulation's convex hull. This fact is important because the convex hull of the feasible region coincides with the polytope defined by all its facets. Hence an optimal solution of the linear program of all the convex hull's facets will also be an optimal solution of the MILP formulation. Barany *et al.* selected some of these facets *a priori* as being the strongest cuts, and added them to a multi-item single-stage problem with some simple capacity constraints. We do not know of any cutting plane work for the multi-stage lot-sizing problem or in the related field of multi-echelon inventory systems.

This article develops a similar approach for the multi-stage problem with component lead time and general product structures where a component may have multiple successor components. The inclusion of lead time into the general-structure model is not straightforward and, as will be shown in Section 2, involves the synchronization of the component production planning periods when the model is applied on a rolling horizon basis. In Section 3 we generalize the valid inequalities in [14] to this problem. At least under certain conditions, the inequalities define high dimensional faces of the formulation's convex hull which may be used as cuts. We develop a separation algorithm that attempts to identify the strongest cuts of a given problem in order to add them to the original constraints at the root node of a Branch-and-Bound search. Computational test results for uncapacitated and capacitated problems are presented in Section 4.

2. THE MULTI-STAGE PROBLEM WITH COMPONENT LEAD TIME

In order to formulate the multi-stage lot-sizing problem, let the component structure of a product be represented by an acyclic graph whose nodes correspond to the components. The nodes are numbered from 1 to N such that if component i is a sub-component of component j then j is called a successor of i and $i>j$. The end-item is node 1. Denote $S(i)$ as the set of all immediate successor components to component i, r_{ij} as the number of units of component i needed by one unit of component $j\in S(i)$, and d_{it} as the independent demand for component i in period t. A decision to produce a batch (lot) x_{it} of component i in a period t incurs a fixed setup cost s_{it} and a unit production cost c_{it}. A unit holding charge h_{it} is incurred for the inventory I_{it} of component i in stock at the end of period t. Component i has production lead time $L(i)$ such that the production x_{it} is only available for consumption in period $t+L(i)$.

A component i is called *primary* if it has no predecessor components. Let $M(1)$ be the maximum sum of the lead times $L(\cdot)$ over a path of successor components from a primary component until, but not including, the end-item. If the planning of the production of the end-item for any period before period $M(1)+1$ is attempted, the end-item production will be limited by the amount of inventory at the beginning of period 1 of at least one component. In order to have proper freedom of planning, the periods over which component production is planned must be synchronized [15]. To do so, let n_{ij} be the number of distinct paths from component i to component j, and $P^n(i,j)$ be the nth path from component i to component j, for $n=1,\ldots,n_{ij}$. For example, in the product structure in Fig. 1, it can be seen by inspection that $M(1)=6$ and that there are 13 paths, as shown in Table 1.

Define $K^n(i,j)$ as the sum of the lead times $L(\cdot)$ on the path $P^n(i,j)$ [excluding $L(j)$], and $T(i)$ as:

$$T(i)=M(1)-\max_{n=1,\ldots,n_{i1}}\{K^n(i,1)\}\qquad \text{for } i=1,\ldots,N.$$

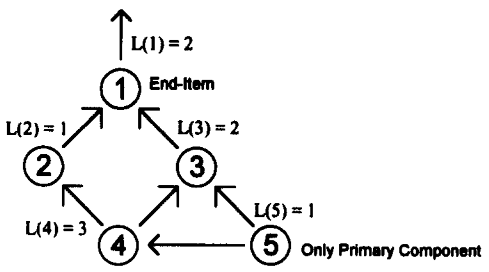

Fig. 1. A general product structure with non-zero lead times.

Table 1. Component paths of the general structure in Fig. 1

	$j=1$	$j=2$	$j=3$	$j=4$
$i=2$	$2\to1$			
$i=3$	$3\to1$			
$i=4$	$4\to2\to1$	$4\to2$		
	$4\to3\to1$		$4\to3$	
$i=5$	$5\to3\to1$		$5\to3$	
	$5\to4\to2\to1$	$5\to4\to2$		$5\to4$
	$5\to4\to3\to1$		$5\to4\to3$	

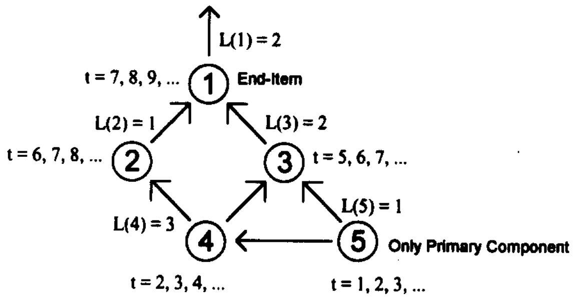

Fig. 2. A general product structure with its synchronized planning periods.

$T(i)+1$ is the period in which, on a rolling horizon basis, the planning of component i must be started in order to synchronize its production with that of the other components, in particular with that of the end-item. In the example of Fig. 1, the definition of $T(i)$ gives

$$T(1)=6-\max\{K^1(1,1)\}=6-0=6$$

$$T(2)=6-\max\{K^1(2,1)\}=6-1=5$$

$$T(3)=6-\max\{K^1(3,1)\}=6-2=4$$

$$T(4)=6-\max\{K^1(4,1),K^2(4,1)\}=6-\max\{4,5\}=1$$

$$T(5)=6-\max\{K^1(5,1),K^2(5,1),K^3(5,1)\}=6-\max\{3,5,6\}=0$$

as shown in Fig. 2.

$T(i)+1$ can also be interpreted as the last period in which a unit of component i may be produced in order to be able to use that unit to produce the end-item in period $M(1)+1$. The $T(i)$ concept incorporates MRP back schedule logic in multi-stage lot-sizing models [16] and further details may be found in [15].

The lot-sizing problem is to find a plan that minimizes the total of setup, production and inventory costs over a planning horizon of T periods. Thus the problem can be formulated as

$$F:\quad \text{minimize} \sum_{i=1}^{N} \sum_{t=T(i)+1}^{T(i)+T} [s_{it}y_{it}+c_{it}x_{it}+h_{i,t+L(i)}I_{i,t+L(i)}]$$

subject to

$$I_{i,L(i)+t-1}+x_{it}-I_{i,L(i)+t}=d_{i,L(i)+t}+\sum_{j\in S(i)} r_{ij}x_{j,L(i)+t} \qquad i=1,\ldots,N \quad t=T(i)+1,\ldots,T(i)+T \qquad (*)$$

$$x_{it}\leqslant M_{it}y_{it} \qquad i=1,\ldots,N \quad t=T(i)+1,\ldots,T(i)+T$$

$$x_{it}\geqslant 0;\ I_{i,L(i)+t}\geqslant 0;\ y_{it}=0 \text{ or } 1 \qquad i=1,\ldots,N \quad t=T(i)+1,\ldots,T(i)+T$$

with the following notation:

$S(i)$ = the set of all immediate successor components to component i,
$P(i)$ = the set of immediate predecessors of component i; if $P(i)=\emptyset$, then i is called *primary*,
r_{ij} = the number of units of component i needed by one unit of component $j\in S(i)$,
d_{it} = the independent demand for component i in period t,
c_{it} = the unit production cost of component i in period t,
s_{it} = the fixed setup cost incurred if component i is produced in period t,
h_{it} = the unit holding charge of component i at the end of period t,
x_{it} = the lot-size of component i in period t (a decision variable),
y_{it} = a dummy variable with value 1 if component i is produced in period t and 0 if not,
I_{it} = the inventory stock of component i at the end of period t,
$L(i)$ = the production lead time of component i, measured as an integer multiple of production planning periods, ensuring that the lot x_{it} is available for consumption only at the beginning of period $t+L(i)$,
$T(i)+1$ = the period in which, on a rolling horizon basis, we start the planning of component i over the T periods $T(i)+1,\ldots,T(i)+T$ in order to synchronize its production with that of the other components, in particular with that of the end-item.

Formulation F is a Mixed Integer Linear Program with NT 0/1 variables. The constraints (*) relate the production x_{it} of component i in period t with the stock $I_{i,L(i)+t}$ of component i at the end of period $t+L(i)$ and, to use MRP terminology, the sum of the independent demand $d_{i,L(i)+t}$ and the dependent demand $\sum_{j\in S(i)} r_{ij}x_{j,L(i)+t}$ for component i in period $t+L(i)$, i.e. offset by component i's lead time $L(i)$. M_{it} is a tight upper bound on x_{it} so that if x_{it} is positive then the constraint $x_{it}\leqslant M_{it}y_{it}$ causes the charging of the setup cost s_{it}.

Using the model on a rolling horizon basis, the decisions $\{x_{i1}\mid i=1,\ldots,N\}$ are implemented and time rolled forward one period. Being at the start of period 1, $\{x_{i1}\mid i=1,\ldots,N\}$ are "implemented" by fixing their values. What are actually produced in period 1 are the already fixed values of $\{x_{it}\mid i=1,\ldots,N;\ T(i)>0\}$ decided by previous rolling horizon applications of the model and the values $\{x_{it}\mid i=1,\ldots,N;\ T(i)=0\}$ of the current application.

Since the model is applied on a rolling horizon basis, x_{it} belongs to one of the following three categories:

(1) x_{it} is known and fixed if $t\leqslant T(i)$,
(2) x_{it} is a model variable if $T(i)+1\leqslant t\leqslant T(i)+T$,
(3) x_{it} is beyond the planning horizon if $t>T(i)+T$.

Thus $\{I_{i,L(i)+T(i)}\mid i=1,\ldots,N\}$ are known "initial" stock levels determined in the previous rolling horizon application of the model.

To generalize the valid inequalities of [14], it is necessary to introduce the concepts of echelon demand and stock. The echelon demand D_{it} and echelon stock E_{it} of component i at the end of period t are recursively defined as:

$$D_{it}=d_{i,L(i)+t}+\sum_{j\in S(i)} r_{ij}D_{j,L(i)+t} \quad \text{and} \quad E_{it}=I_{i,L(i)+t}+\sum_{j\in S(i)} r_{ij}E_{j,L(i)+t}.$$

In words, D_{it} is the total system demand for component i in period t, both independent and dependent, offset by lead time. Similarly E_{it} is the lead-time adjusted total system stock of component i, both as a stand-alone component, $I_{i,L(i)+t}$, and as part of successor components $\sum_{j\in S(i)} r_{ij}E_{j,L(i)+t}$. The relationship between echelon demand and stock: $E_{i,t-1}+x_{it}-E_{it}=D_{it}$ is as simple as that in the zero lead-time single-stage case. Further details about the lead-time related definitions of D_{it} and E_{it} can be found in [15]. The echelon stock concept was introduced by Clark and Scarf [17] and, in the context of zero lead times, has been used by a number of authors [6, 8–10].

Note that $\{E_{it} \mid i=1,\ldots,N;\ t=1,\ldots,T(i)\}$ are predetermined echelon stock levels, a subset of which, namely $\{E_{i,T(i)} \mid i=1,\ldots,N\}$, feature in the generalization of the valid inequalities in [14] for the multi-stage problem, as shown in the next section. For future use, define also:

$$d_{i,t_1,t_2}=\sum_{t=t_1}^{t_2} d_{it} \quad \text{and} \quad D_{i,t_1,t_2}=\sum_{t=t_1}^{t_2} D_{it}$$

i.e.

$$D_{i,t_1,t_2}=d_{i,t_1+L(i),t_2+L(i)}+\sum_{j\in S(i)} r_{ij}D_{j,t_1+L(i),t_2+L(i)}.$$

We now introduce a capacitated formulation F^C of F that includes representation of the capacity implications of lot setups, allows the sharing of capacity between component stages, and that also models multiple types of capacity. Let f_{ikt} be the amount of capacity resource k necessary to setup the production of component i in period t, for example, the setup time on a given machine. Similarly, let v_{ikt} be the unit amount of resource k necessary to produce component i in period t, in addition to the setup amount f_{ikt}. We assume that the availability b_{kt} of resource k in period t is fixed and paid for independently of its utilization, e.g. the regular workforce of a plant. In these circumstances, we have the constraints:

$$\sum_{\substack{i=1\\ i\mid t\leqslant T(i)+T}}^{N} [f_{ikt}y_{it}+v_{ikt}x_{it}]\leqslant b_{kt} \qquad \begin{array}{l} k=1,\ldots,K\\ t=1,\ldots,T(1)+T. \end{array}$$

These constraints are added to F to form the capacitated formulation F^C. The condition $i \mid t\leqslant T(i)+T$ is specified in the summation to exclude those x_{it} beyond component i's planning horizon. Note that the summation includes the already known values $x_{it} \mid t\leqslant T(i)$, given that this prefixed production consumes capacity. The constraint is enforced up to the period $t=T(1)+T$, the longest of the component planning horizons, namely that of the end-item.

3. VALID INEQUALITIES FOR THE MULTI-STAGE PROBLEM

Building on the work of Barany *et al.* [14], it is shown in [18] that under certain conditions the valid inequality:

$$\sum_{t\in S} x_{it}+\sum_{t\in Q\setminus S} D_{itq}y_{it}\geqslant D_{i,T(i)+1,q}-E_{i,T(i)} \qquad (V_i)$$

for $i\in\{1,\ldots,N\}$, $T(i)+1<q<T(i)+T$, $Q=\{T(i)+1,\ldots,q\}$and $S\subseteq Q$

is strong, defining a high dimensional face of the convex hull of F.

If the optimal solution of the linear programming relaxation LP(F) of F is integer (i.e. all $y_{it}\in\{0,1\}$), then the solution will also be optimal for F. However this is unlikely and so we will generally implement the following algorithm:

Step 1. Let $n=1$ and LP^n(F) = LP(F).
Step 2. Solve LP^n(F). If the optimal solution is integer then it is also an optimal solution of F and so stop. Otherwise go to Step 3.
Step 3. Identify an inequality V_i that excludes the optimal solution found for LP^n(F). Such an inequality is a cut. Add this cut to LP^n(F) to form the linear program

LP$^{n+1}(F)$. If it is not possible to identify such a cut then go to Step 4. If it is possible then put $n=n+1$ and go to Step 2.

Step 4. Begin the Branch-and-Bound search.

Steps 1–4 describe an algorithm of the Strong Cutting Plane type [19,20]. In Step 5 we need a separation algorithm to identify a valid inequality V_i that cuts off the optimal solution found for LP$^n(F)$. There is no theoretical guarantee that such a cut exists. The ideal cut would be that which maximizes the value of the optimal solution of LP$^{n+1}(F)$, but it is not possible to identify such a cut without solving excessively many linear programs. We could think of choosing the inequality V_i that lies most distant in Euclidean space from the optimal solution found for LP$^n(F)$, but an algorithm that sought to do this would probably have to investigate all possible inequalities V_i and would be slow. A more efficient algorithm results if, like [20,21], we decide to choose that inequality which has the greatest difference (in the appropriate direction) between the constant right-hand side of the inequality and the left-hand side evaluated at the optimal solution found for LP$^n(F)$. The resulting algorithm will now be developed.

Fix i and consider the valid inequalities V_i for $T(i)+2\leqslant q\leqslant T(i)+T-1$, $Q=\{T(i)+1,\dots,q\}$, $S\subseteq Q$, $T(i)+1\in S$ and $Q\backslash S\neq\varnothing$. This subset of the inequalities can be re-expressed:

$$\sum_{t\in S} x_{i,T(i)+t}+\sum_{t\in Q\backslash S} D_{i,T(i)+t,T(i)+q}, y_{i,T(i)+t}\geqslant D_{i,T(i)+1,T(i)+q}-E_{i,T(i)}$$

for $2\leqslant q\leqslant T-1$, $Q=\{1,\dots,q\}$, $S\subseteq Q$, $1\in S$ and $Q\backslash S\neq\varnothing$. We need to identify sets Q and S that minimize:

$$\sum_{t\in S} x^*_{i,T(i)+t}+\sum_{t\in Q\backslash S} D_{i,T(i)+t,T(i)+q}y^*_{i,T(i)+t}-D_{i,T(i)+1,T(i)+q}+E_{i,T(i)}$$

where $x^*_{i,T(i)+t}$ and $y^*_{i,T(i)+t}$ belong to the optimal solution $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{I}^*)$ found for LP$^n(F)$. If we fix q at one of its $T-2$ possible values, then it only remains to identify the set S. To do so, first observe that V_i can be rewritten as:

$$\sum_{t\in S}[x_{i,T(i)+t}-D_{i,T(i)+t,T(i)+q}y_{i,T(i)+t}]+\sum_{t\in Q} D_{i,T(i)+t,T(i)+q}y_{i,T(i)+t}-D_{i,T(i)+1,T(i)+q}+E_{i,T(i)}\geqslant 0.$$

Secondly, for a given optimal solution $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{I}^*)$ and a fixed q, define, for $t\in Q$,

$$\alpha_t^Q=1 \text{ if } t\in S \text{ and } 0 \text{ if } t\in Q\backslash S. \qquad \text{(a variable)}$$

$$\pi_t^Q=x^*_{i,T(i)+t}-D_{i,T(i)+t,T(i)+q}y^*_{i,T(i)+t} \qquad \text{(a constant)}$$

$$D^Q=\sum_{t\in Q} D_{i,T(i)+t,T(i)+q}y^*_{i,T(i)+t}-D_{i,T(i)+1,T(i)+q}+E_{i,T(i)}+\pi_1^Q. \qquad \text{(a constant)}$$

Since $1\in S$, we know that $\alpha_1^Q=1$ (and it is because of this that we incorporate π_1^Q into D^Q). We need that $Q\backslash S\neq\varnothing$. Hence we must add the constraint $\sum_{t=2}^{q}\alpha_t^Q\leqslant q-2$ to force that at least one of $\alpha_2^Q,\dots,\alpha_q^Q$ be zero.

Thus we have the following mimimization problem:

$$\begin{aligned} Z_1(i,q)=\text{minimize} \quad & \sum_{t=2}^{q}\alpha_t^Q\pi_t^Q \qquad (+\text{the constant } D^Q)\\ \text{such that} \quad & \sum_{t=2}^{q}\alpha_t^Q\leqslant q-2\\ & \alpha_t^Q=0/1 \qquad t=2,\dots,q. \end{aligned}$$

This problem can be quickly solved by inspection of the signs of the π_t^Q for $t\in\{2,\dots,q\}$. If $\pi_t^Q\geqslant 0$ then let $\alpha_t^Q=0$ otherwise let $\alpha_t^Q=1$. However, if $\pi_t^Q<0$ for all $t\in\{2,\dots,q\}$, then let $\alpha_\tau^Q=0$ and $\alpha_t^Q=1$ for $t\in\{2,\dots,q\}\backslash\{\tau\}$ where $\tau=\arg\max\{\pi_t^Q\,|\,t=2,\dots,q\}$.

The above problem is solved for each pair (i,q) in the set $\{(i,q)\,|\,i=1,\dots,N;\ 2\leqslant q\leqslant T-1\}$. We then choose the triple (i,q,S), and its corresponding V_i valid inequality, that supplies the solution to $\min\{Z_1(i,q)\,|\,i=1,\dots,N; 2\leqslant q\leqslant T-1\}$. The separation algorithm results in $\min\{Z_1(i,q)\,|\,i=1,\dots,N;$ $2\leqslant q\leqslant T-1\}\geqslant 0$ if and only if there does not exist an inequality V_i that cuts off the optimal solution found for $LP^n(F)$. However, if V_i cuts exist then the separation algorithm will indicate this through a negative solution $\min\{Z_1(i, q) \mid i=1,\ldots, N;\ 2\leqslant q\leqslant T-1\}<0$ and will identify one of the cuts.

We hope that the V_i cut identified by the separation algorithm results in an optimal solution for $LP^{n+1}(F)$ that is near to the optimal integer solution. The separation algorithm is incorporated into Step 3 of the principal algorithm.

A possible modification to the separation algorithm is to choose the triple (i, q, S), and its corresponding valid inequality, that minimizes $\{Z_1(i, q)/D_{i,T(i)+1,T(i)+q} \mid i=1,\ldots, N;\ 2\leqslant q\leqslant T-1\}$. The reasoning behind this criterion is that x_{it}, E_{it}, D_{it}, and D_{jt} for $j\in S(i)$ tend to be larger for higher values of i and so the difference between the left and right sides of the inequality V_i will also tend to be larger. If we divide the inequality by $D_{i,T(i)+1,T(i)+q}$, then the difference between the two sides will be of a similar order of magnitude for all i. If not, the separation algorithm will tend to choose inequalities with higher values of i.

The division by $D_{i,T(i)+1,T(i)+q}$ is an attempt to scale the inequalities. Such scaling is obviously crude, but has the advantage of still permitting the use of the fast separation algorithm developed above. The modification was adopted after initial computational tests showed its effectiveness.

4. COMPUTATIONAL TESTS

The objective of the computational tests was to assess the improvement in the performance of the Branch and Bound search due to the addition of the cuts selected by the separation algorithm.

The tests were carried out using products with 10 components over a 12-period planning horizon in the presence of two constraining resources. Ten sets of data were randomly generated for each of four contrasting product structures, making a total of 40 distinct data sets. Possible contrasting effects between high and low setup costs were considered by creating a further 40 high cost data sets where all setup costs were increased by a factor of 10. The resulting 80 problems were tackled with and without cuts in order to assess the impact of the cuts. Equivalent experiments on the same 80 problems with the addition of tight capacity constraints were also carried out.

Thus the tests investigated the effect of the following four factors:

(A) The addition or not of cuts. Two formulations were assessed:
   (1) F/F^C with no cuts;
   (2) F/F^C with all the cuts identified by the separation algorithm.

(B) The product structure. Four contrasting structures were investigated:
   (1) The flattest possible structure, i.e. $S(i)=\{1\}$ for $i=2,\ldots, 10$;
   (2) A typical assembly structure, namely: $S(2)=\{1\}$, $S(3)=\{1\}$, $S(4)=\{1\}$, $S(5)=\{2\}$, $S(6)=\{2\}$, $S(7)=\{4\}$, $S(8)=\{2\}$, $S(9)=\{3\}$ and $S(10)=\{7\}$;
   (3) The assembly structure extended to be a typical general structure, namely: $S(2)=\{1\}$, $S(3)=\{1\}$, $S(4)=\{1\}$, $S(5)=\{2\}$, $S(6)=\{2, 3\}$, $S(7)=\{4\}$, $S(8)=\{2, 3, 4\}$, $S(9)=\{3\}$ and $S(10)=\{5, 6, 7\}$;
   (4) A totally serial structure, i.e. $S(i)=\{i-1\}$ for $i=2,\ldots, 10$.

(C) The size of the setup costs s_{it}. The low costs were randomly sampled from the uniform distribution $U(5, 95)$. The high setup costs were ten times the low costs.

(D) The production capacity in terms of the resource availability b^{kt}. Capacity was constant over time, i.e. $b_{kt}=b_k$ for $k\in\{1, 2\}$ and $t\in\{1,\ldots, T(1)+T\}$. Two extremes were examined;
   (1) F, i.e. infinite capacity, represented by $b_1=b_2=\infty$;
   (2) F^C with tight capacity, the values of b_1 and b_2 being those that, during initial tests, represented the approximate tightest capacity for which the Branch-and-Bound search found a feasible solution. This capacity was found to be (1) tighter for the high setup costs, and (2) looser for the general structure, as shown in Table 2. The reasons for this could be that (1) high setup costs tend to result in feasible solutions with fewer setups over the planning horizon causing, in turn, less capacity usage overall than would be the case with low setup costs, and (2) general product structures have a greater number of inter-echelon constraints and hence less room for maneuver in moving production to other periods in order to become capacity feasible.

The variable production costs c_{it} were randomly sampled from the uniform distribution $U(1.5, 2.0)$, the end-item demands d_{1t} from $U(0, 180)$, the independent component demands $\{d_{it} \mid i=2,\ldots, N\}$

Table 2. Approximate tightest feasible capacities b_1 and b_2

	Low setup costs		High setup costs	
	b_1	b_2	b_1	b_2
Flat structure	4000	5000	3000	4000
Assembly structure	4000	5000	3000	4000
General structure	5000	6000	4000	5000
Serial structure	4000	5000	3000	4000

Table 3. Summary of the percentage precision guarantee test results

		Low setup costs				High setup costs			
		No cuts		With cuts		No cuts		With cuts	
		Infinite capac.	Tight capac.	Infinite capac.	Tight capac.	Infinite capac.	Tight capac.	Infinite capac.	Tight capac.
Flat	%	12.8	13.7	0.317	1.28	18.79	23.3	4.57	10.1
structure	m	10	9	10	9	10	10	10	10
Assembly	%	13.4	14.3	1.07	2.00	22.4	30.8	10.9	16.0
structure	m	10	10	10	10	10	10	10	10
General	%	11.08	12.8	3.17	5.08	33.9	36.2	11.8	21.5
structure	m	10	9	10	9	10	10	10	10
Serial	%	13.65	16.0	1.38	7.27	22.9	34.6	13.8	30.0
structure	m	10	9	10	9	10	10	10	10

from $U(0, 18)$, and the lead times $L(i)$ from the set $\{0, 1\}$. For simplicity, $r_{ij}=1$ for $i \in \{1,\ldots, N\}$ and $j \in S(i)$.

To randomly, but realistically, generate the conventional stock holding costs $\{h_{i,L(i)+t} \mid i=1,\ldots, N;\ t=T(i)+1,\ldots, T(i)+T\}$, values of the echelon stock holding costs e_{it} [15] were first sampled from $U(0.2, 0.4)$. The values of e_{it} were then used to calculate the costs $\{h_{i,L(i)+t} \mid i=1,\ldots,N;\ t=T(i)+1,\ldots, T(i)+T\}$ via the identity $h_{i,L(i)+t}=e_{it}+\sum_{j\in P(i)} r_{ji}h_{jt}$ [15].

The distributions for s_{it} (high costs), e_{it}, d_{it}, and r_{ij} are the same as those for s_i, e_i, $d_t(d_{1t})$ and r_{ij} of [7] who used $d_{it}=0$ for $i \in \{2,\ldots, N\}$, $L(i)=0$ for $i \in \{1,\ldots, N\}$, and ignored the c_{it} since all costs were assumed constant over time. The setup resource requirements f_{i1t} and f_{i2t} were randomly sampled from the uniform distributions $U(150, 250)$ and $U(200, 300)$ respectively, and the unit resource requirements v_{i1t} and v_{i2t} from $U(1.5, 2.5)$ and $U(2.0, 3.0)$ respectively. The initial stock levels $\{I_{it} \mid i=1,\ldots, N;\ t=L(i),\ldots, T(i)+L(i)\}$ and the prefixed production levels $\{x_{it} \mid i=1,\ldots,N;\ t=1,\ldots, T(i)\}$ were randomly and realistically determined, and took into account that the latter will consume part of the availability b_{kt} of the resource k in period $t \in \{1,\ldots, T(1)\}$.

The computer programs were coded in the C programming language, and linked with IBM's Optimization Subroutine Library (OSL) [22] to solve the linear and mixed integer programs. An IBM 3090 mainframe (with no vector facility) was used.

First, if cuts were to be added, the cut identifying algorithm of Section 3 was executed. An optimal solution was then searched for using OSL's EKKMSLV subroutine. If after 20,000 Branch-and-Bound (BB) nodes an optimal solution was not found, then a percentage precision guarantee of the distance of the BB incumbent from optimality was adopted as the measure of the success of the BB search, defined as

$$\text{Precision Guarantee} = \frac{\text{incumbent after 20,000 nodes} \times 100\%}{\text{least lower bound of the active nodes}} - 100\%.$$

Table 3 summarizes the results of the test runs. Each cell contains two numbers, namely the mean value of the percentage precision guarantee (%) over m experiments and the value of m itself. In some tight capacity cells, m is 9 because occasionally no feasible solution was found within 20,000 BB nodes. The significance of the differences between these mean values should be judged from Tables 4, 5 and 6 which show the mean differences between matched values and Wilcoxon Test results.

Table 4. Percentage point precision guarantee degradation with high setup costs

		Infinite capacity		Tight capacity	
		No cuts	With cuts	No cuts	With cuts
Flat structure	$\bar{d}$	6.0	4.3	10.1	9.42
	W	10	3	7	2
	m	10	10	9	9
	signif.	10%	2%	10%	1%
Assembly structure	$\bar{d}$	9.1	9.8	16.5	14.0
	W	0	0	0	0
	m	10	10	10	10
	signif.	1%	1%	1%	1%
General structure	$\bar{d}$	22.8	8.6	24.0	17.4
	W	0	0	0	0
	m	10	10	9	9
	signif.	1%	1%	1%	1%
Serial structure	$\bar{d}$	9.3	12.4	17.8	20.4
	W	1	0	0	0
	m	10	10	9	9
	signif.	1%	1%	1%	1%

Table 5. Percentage point precision guarantee degradation with the addition of tight capacity

		Low setup costs		High setup costs	
		No cuts	With cuts	No cuts	With cuts
Flat structure	$\bar{d}$	0.75	0.93	4.5	5.5
	W	3	0	3	1
	m	7	6	9	9
	signif.	10%	5%	2%	1%
Assembly structure	$\bar{d}$	0.99	0.92	8.4	5.1
	W	1	1	0	9
	m	10	10	10	10
	signif.	1%	1%	1%	10%
General structure	$\bar{d}$	0.435	1.4	2.3	9.7
	W	1	10	11	0
	m	9	9	10	10
	signif.	1%	> 10%	10%	1%
Serial structure	$\bar{d}$	2.36	5.9	11.7	16.2
	W	0	0	0	1
	m	9	9	10	10
	signif.	1%	1%	1%	1%

Table 6. Percentage point precision guarantee improvement with the addition of cuts

		Infinite capacity		Tight capacity	
		Low setup costs	High setup costs	Low setup costs	High setup costs
Flat structure	$\bar{d}$	12.5	14.2	12.4	13.2
	W	0	0	0	0
	m	10	9	9	9
	signif.	0.5%	0.5%	0.5%	0.5%
Assembly structure	$\bar{d}$	12.3	11.5	12.3	14.8
	W	0	0	0	0
	m	10	10	10	10
	signif.	0.5%	0.5%	0.5%	0.5%
General structure	$\bar{d}$	7.91	22.0	7.72	14.6
	W	0	0	0	0
	m	10	10	9	10
	signif.	0.5%	0.5%	0.5%	0.5%
Serial structure	$\bar{d}$	12.3	9.16	8.86	4.58
	W	0	5	1	17
	m	10	10	8	10
	signif.	0.5%	1%	1%	>5%

Table 3 shows that in the tests better incumbent precision guarantees were obtained with the low setup costs than with the high ones. This is statistically confirmed by Table 4 which shows, on the first line in each cell, the mean $\bar{d}$ of the differences in the guarantee caused by multiplying the low setup costs by a factor of 10. The statistical significance of $\bar{d}$ was evaluated applying the non-parametric Wilcoxon test for differences between two matched samples [23]. The second line in each cell shows W, the sum of the ranks with the less frequent sign while the third line shows the sample size m used to assess the significance of W. The Wilcoxon test requires that zero differences be excluded from the sample significance assessment. In addition, OSL was unable to find feasible solutions for a few tightly capacitated problems. For these two reasons the sample size m is occasionally less than 10. The fourth line indicates the two-tailed significance level of W [24]. Tables 5 (two-tailed) and 6 (one-tailed) are laid out in the same manner.

Note in Table 4 that the precision guarantee differences between high and low setup costs are all significant at the two-tailed 1% level (except in three of the flat structure cells, particularly the two no-cuts cells which are significant at merely 10%). Thus the incumbent precision guarantees are substantially better in the presence of lower setup costs. Pooling all 154 pairs of the data summarized in Table 4 gives a mean difference of 13.38%.

Some mean differences in Tables 4, 5 and 6 are not exactly equal to their differences between the appropriate means in Table 3, due to different sample sizes. For example, in Table 3 the guarantee for flat/tight-capacity/no-cuts with low and high setup costs are respectively 13.7 with a sample size of 9 and 23.3 with a sample size of 10. The mean difference of 10.1 in Table 4 is calculated from the 9 matching pairs.

Table 3 also indicates that generally worse precision guarantees are obtained with tight capacity than with infinite capacity. Table 5 shows the mean and Wilcoxon test results of the percentage point degradation in the guarantee caused by adding the tight capacity constraints. Over half the differences are significant at the two-tailed 1% level, and all except one at the 10% level. This indicates that tight capacity disproportionally increases the incumbent in relation to the lower bound provided by the relaxed optimal solution.

Hence the incumbent precision guarantees are better when there is infinite production capacity. Pooling all 147 pairs of the data summarized in Table 5 gives a mean difference of 4.88%.

Finally and most importantly, observe that Table 3 indicates that better guarantees appear to be obtained with the use of cuts than without their use. This is partially confirmed by Table 6 which shows the mean and Wilcoxon test results of the improvement in the precision guarantee caused by the cuts. Note that the differences are all at the one-tailed 1% level (except for the serial structure with high setup costs) and that they do not appear to be influenced by capacity, product structure or setup costs.

Thus the incumbent precision guarantees are substantially improved by the addition of the cuts. Pooling all the data summarized in Table 6 gives a mean difference of 11.9%.

In certain cases the addition of the cuts enabled an optimal solution to be identified. Table 7 shows how many optimal solutions were found, without and with cuts respectively, for each cell's ten problems.

Note that the BB search generally identified very few optimal solutions within 20,000 nodes, and that the addition of cuts greatly improved the number of optimal solutions only in the case of the flat structure with low costs and infinite capacity. However, the problem lower bound increased

Table 7. Number of optimal solutions identified within 20,000 nodes (a) without cuts and (b) with cuts

		Infinite capacity		Tight capacity	
		Low setup costs	High setup costs	Low setup costs	High setup costs
Flat structure	(a)	0	1	0	1
	(b)	8	3	3	1
Assembly structure	(a)	0	0	0	0
	(b)	0	0	0	0
General structure	(a)	0	0	0	0
	(b)	0	0	0	0
Serial structure	(a)	0	0	0	0
	(b)	2	0	0	0

Table 8. Lower bound improvement in percentage points (a) during the Branch-and-Bound search with no cuts added, (b) during addition of cuts before the Branch-and-Bound search, and (c) during the Branch-and-Bound search after addition of cuts

		Infinite capacity		Tight capacity	
		Low setup costs	High setup costs	Low setup costs	High setup costs
Flat structure	(a)	0.07	2.18	0.09	2.07
	(b)	8.65	9.48	8.68	9.88
	(c)	0.81	2.91	0.36	1.83
Assembly structure	(a)	0.20	1.33	0.20	0.78
	(b)	9.00	9.81	9.23	9.31
	(c)	0.28	0.52	0.01	0.57
General structure	(a)	0.12	0.48	0.11	0.58
	(b)	7.08	13.06	8.63	13.2
	(c)	0.08	0.41	0.03	0.24
Serial structure	(a)	0.35	0.97	0.27	0.62
	(b)	9.09	8.95	9.43	8.70
	(c)	0.37	0.51	0.04	0.28

Table 9. CPU times in seconds (a) of the cuts addition phase, and (b) of the Branch-and-Bound search

		Infinite capacity		Tight capacity	
		Low setup costs	High setup costs	Low setup costs	High setup costs
Flat structure	(a)	386	67	527	96
	(b)	488	771	1349	1145
Assembly structure	(a)	418	80	702	126
	(b)	1467	980	2069	1322
General structure	(a)	493	270	887	488
	(b)	1553	1336	1915	1798
Serial structure	(a)	417	94	701	156
	(b)	1434	1043	1842	1356

by an average of 9.51 percentage points during the addition of cuts and then by 0.58 percentage points on average during the BB search after cuts were added. This mean increase of over ten percentage points in the lower bound compares very favorably with a mean increase of only 0.65 percentage points during the BB search with no cuts added.

To see this in more detail Table 8 shows the mean lower bound improvements in percentage points (a) during the BB search with no cuts added, (b) during the addition of cuts and (c) during the BB search after cuts were added. The coefficient of variation of the improvement (b) during the addition of cuts in each cell was generally very small with a mean value of less than 0.05 over the sixteen cells and never more than 0.104. In summary, the lower bound improvement due to the cuts is substantial and consistent with small variability.

Table 9 shows the mean CPU times (in seconds) of the cuts addition phase (Steps 1–3 of the general algorithm) and of the BB search (Step 4). Note that the cuts phase always took less time than the BB search and usually substantially so. For technical reasons concerning OSL, the problem was primally reoptimised from scratch each time a new cut was added, resulting in the CPU times of Table 9(a). We did not use the dual simplex method which, of course, offers a much more efficient method of regaining feasibility and optimality each time the separation algorithm adds a new cut to the relaxed problem.

The CPU times at the cuts addition phase for high setup costs are in line with number of cuts identified by the separation algorithm as shown in Table 10 below.

Table 10 shows the mean number n of cuts identified by the separation algorithm for the test problems, i.e. there remained no cuts that excluded the optimal solution to $LP^n(F)$. The number of cuts identified for low setup costs was always more than the number for the same problem with high setup costs, in fact over three times as many on average. We have no theoretical explanation for this.

The small percentage lower bound improvements of the BB search suggest that a further application of the cuts would be to improve the precision guarantee of a heuristic solution, avoiding a time-consuming BB search. Initial tests show the cuts to be very useful in this respect, especially given the difficulty of obtaining an optimal solution for the problem. However to be practically

Table 10. Number of cuts identified

	Infinite capacity		Tight capacity	
	Low setup costs	High setup costs	Low setup costs	High setup costs
Flat structure	213.8	54.7	208.4	60.0
Assembly structure	227.0	61.9	234.0	69.7
General structure	232.0	129.5	250.0	148.0
Serial structure	222.1	72.1	229.1	79.9

applicable, this requires an efficient method of regaining feasibility and optimality each time the separation algorithm adds a new cut to the relaxed problem. This can be done with the efficient dual simplex reoptimization facilities available in much mathematical programming software.

5. CONCLUSIONS

This article has generalized the single-stage lot-sizing valid inequalities of [14] to the multi-stage problem with lead times and developed computational tests that showed the inequalities' great strength in improving lower bounds and improving incumbent guarantees in a Branch-and-Bound search. The separation algorithm enabled quick iterative selection of new strong cuts. The valid inequalities can be extended to multi-end-item multi-stage lot-sizing problems.

Future research directions would include investigating whether it is advantageous to drop non-active cuts and/or add new cuts at the Branch-and-Bound nodes.

Acknowledgements—This work was supported by the Conselho Nacional de Desenvolvimento Cientifico (CNPq), UniSoma S.A and IBM Brazil.

REFERENCES

1. H. C. Bahl, L. P. Ritzman and J. N. D. Gupta, Determining lot-sizes and resource requirements: a review. *Ops Res.* **35**, 237–249 (1987).
2. L. A. Johnson and D. C. Montgomey, *Operations Research in Production Planning, Scheduling and Inventory Control.* Wiley, New York (1974).
3. W. I. Zangwill, A backlogging model and a multi-echelon model of a dynamic economic lot-size production system—a network approach. *Mgmt Sci.* **15**, 506–527 (1969).
4. H. Konno, Minimum concave cost production system: a further generalization of multi-echelon model. *Math. Prog.* **41**, 185–193 (1988).
5. E. Steinberg and H. A. Napier, Optimal multi-level lot-sizing for requirements planning systems. *Mgmt Sci.* **26**, 1258–1271 (1980).
6. P. Afentakis, B. Gavish and U. Karmarkar, Computationally efficient optimal solutions to the lot-sizing problem in multistage assembly systems. *Mgmt Sci.* **30**, 222–239 (1984).
7. P. Afentakis and B. Gavish, Optimal lot-sizing algorithms for complex product structures. *Ops Res.* **34**, 237–249 (1986).
8. W. B. Crowston, M. H. Wagner and J. F. Williams, Economic lot size determination in multi-stage assembly systems. *Mgmt Sci.* **19**, 517–527 (1973).
9. W. B. Crowston and M. H. Wagner, Dynamic lot-size models for multi-stage assembly systems. *Mgmt Sci.* **20**, 517–527 (1973).
10. L. B. Schwarz and L. Schrage, Optimal and system myopic policies for multi-echelon production/inventory assembly systems. *Mgmt Sci.* **21**, 1285–1294 (1975).
11. S. K. Goyal and A. Gunasekaran, Multi-stage production-inventory systems. *Eur. J. Opl Res.* **46**, 1–20 (1990).
12. J. Maes, J. O. McClain and L. N. Van Wassenhove, Multilevel capacitated lotsizing complexicity and LP-based heuristics. *Eur. J. Opl Res.* **53**, 131–148 (1991).
13. R. E. Gomory, Outline of an algorithm for integer solutions to linear programs. *Bull. Am. Math. Soc.* **64**, 275–278 (1958).
14. I. Barany, T. J. Van Roy and L. A. Wolsey, Strong formulations for multi-item capacitated lot-sizing. *Mgmt Sci.* **30**, 1255–1261 (1984).
15. A. R. Clark and V. A. Armentano, Echelon stock formulations for multi-stage lot-sizing with lead times. *Int. J. Syst. Sci.* **24**, 1759–1775 (1993).
16. T. E. Vollman, W. T. Berry and D. C. Whybark, *Manufacturing Planning and Control Systems*, 2nd edition. Dow-Jones/Irwin, IL (1988).
17. A. P. Clark and H. Scarf, Optimal policities for a multi-echelon inventory problem. *Mgmt Sci.* **6**, 475–490 (1960).
18. A. R. Clark and V. A. Armentano, The application of valid inequalities to the multi-stage lot-sizing problem. Technical Report #11, Faculty of Electrical Engineering, Universidade Estadual de Campinas, Brazil (1993).
19. G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization.* Wiley, New York (1988).
20. T. J. Van Roy and L. A. Wolsey, Solving mixed integer programming problems using automatic reformulation. *Ops Res.* **35**, 45–57 (1987).
21. T. J. Van Roy and L. A. Wolsey, Valid inequalities for mixed 0–1 programs. *Discr. Appl. Math.* **14**, 199–213 (1986).
22. IBM, *Optimization Subroutine Library (OSL) User's Guide*, Release 2 (1991).
23. J. E. Freund and R. E. Walpole, *Mathematical Statistics*, 3rd edition. Prentice-Hall, Englewood Cliffs, NJ (1980).
24. J. White, A. Yeats and G. Skipworth, *Tables for Statisticians*, 3rd edition. Thorns, Cheltenham (1979).