ELSEVIER

Int. J. Production Economics 68 (2000) 319–331

international journal of production economics

www.elsevier.com/locate/dsw

An integrated Lagrangean relaxation-simulated annealing approach to the multi-level multi-item capacitated lot sizing problem

Linet Özdamar[a], Gülay Barbarosoglu[b,*]

[a]İstanbul Kültür University, Computer Engineering Department, İstanbul, Turkey
[b]Bogaziçi University, Industrial Engineering Department, İstanbul, Turkey

Received 16 March 1998; accepted 24 June 1999

Abstract

This study proposes a heuristic approach for the solution of the dynamic multi-level multi-item capacitated lot sizing problem (MLCLSP) with general product structures. The difficulty in solving MLCLSP is to provide capacity-feasible lot-sizes while maintaining the non-negativity of the inventories belonging to the items in the lower levels of the product structures. The proposed technique aims to resolve this issue by combining the capability of the Lagrangean relaxation to decompose the hard-to-solve problems into smaller subproblems and the intensive search capability of the simulated annealing. As the first attempt, two Lagrangean relaxation schemes are designed and different versions of simulated annealing are incorporated into relaxation designs as the Lagrangean heuristic. Then in order to improve the performance of the heuristic, a Phase-1 procedure is developed as a recursive algorithm to restore capacity feasibility. It is observed that the best results are obtained by executing first Phase-1 procedure and then simulated annealing approach with only improving moves in each Lagrangean cycle. The performance of these approaches is compared by using the benchmark problems available in literature. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Lot sizing with setup times; Lagrangean relaxation; Global Search; Simulated annealing

1. Introduction

This paper addresses the multi-level multi-item capacitated lot sizing problem (MLCLSP) which arises very commonly in multi-stage manufacturing systems. The main issue is to determine the optimal lot-sizes of all items appearing in different levels of the product structure where various resources are needed and shared by multiple items. In fact the model developed in this research includes setup times and setup costs with a general product structure. A lot of research has been directed towards solving lot sizing problems with different modeling features. The single stage capacitated lot sizing problem is shown to be NP-hard in [1], and in case of setup times even the problem of finding a feasible solution becomes an NP-complete problem as shown in [2]. Naturally it becomes even harder to solve the MLCLSP with setup times. In early studies addressing the solution of the MLCLSP, certain

* Corresponding author: Fax: 90-212-2651800.
E-mail address: barbaros@boun.edu.tr (Gülay Barbarosoglu).

modeling features have been excluded in order to simplify the solution procedure. Billington et al. [3] consider general product structures and a single capacity-constrained resource with setup times, and Billington et al. [4] assume a linear product structure and uniform processing times without setup times. Maes and Van Wassenhove [5,6] develop heuristics for the linear product structure case where capacity sharing by multiple levels is not permitted. Kuik et al. [7] also assume a linear product structure for which only a single capacitated stage is assumed to exist among multiple stages without setup times. Tempelmeier and Helber [8] develop a heuristic procedure for the general product structure case where the capacity of multiple resources can be shared by items appearing in different levels of the product structure whereas Tempelmeier and Derstroff [9] apply Lagrangean relaxation, Franca et al. [10] develop a constructive heuristic, and Segerstedt [11] uses dynamic programming for the general product structure with setup times.

This study aims at integrating Lagrangean relaxation and global search procedures in solving the MLCLSP. Lagrangean relaxation is employed to solve the single stage capacitated lot sizing problem in [12–16]. Furthermore, Billington [17] and Tempelmeier and Derstroff [9] use Lagrangean relaxation in the solution of the MLCLSP. Lagrangean relaxation possesses the capability to decompose a difficult problem into a number of smaller problems while the global search approaches are capable to improve both feasibility and optimality of a given solution by extensive search. Thus the global search concepts are used in the design of the Lagrangean heuristic and implemented in the relaxation in order to generate a feasible solution with the best objective function value starting from the solution obtained in each sub-gradient optimization iteration. Simulated annealing as discussed in [18] and [7] is selected as the search methodology to be implemented in this context.

Two different relaxation schemes are developed in this study: The hierarchical relaxation is obtained by relaxing only the capacity constraints while the second one called the non-restricted relaxation is developed by relaxing both inventory and capacity constraints. Similarly different designs for the Lagrangean heuristic are developed and compared. First a simulated annealing procedure which accepts the non-improving moves with certain probabilities are designed. Then in the design of the second procedure, the acceptance of non-improving moves is prohibited in the search, which is not restricted to the neighborhood of a given solution, but continues exploding the feasible region intensively. This approach is called simulated annealing with only improving moves. Then in order to enhance the solution of the Lagrangean heuristic, another procedure is developed to attain capacity feasibility in case of hierarchical relaxation before the simulated annealing procedures. This is called a Phase-1 procedure since it acts as an initial phase of the Lagrangean heuristic. Different combination of relaxation schemes and search procedures are tested by using the problems given in [9], and the results indicate that the hierarchical Lagrangean relaxation with Phase-1 and simulated annealing with only improving moves outperforms the other designs.

The paper is organized as follows: A generic MLCLSP formulation is provided in Section 2. The different Lagrangean relaxation schemes are described in Section 3 while the details of simulated annealing procedures and the recursive heuristic as the Lagrangean heuristic are provided in Sections 4 and 5, respectively. Computational results are given in Section 6, and Section 7 states the concluding remarks.

2. Formulation of the basic model

The multi-level multi-item lot-sizing problem with capacity constraints can be formulated as follows:

Problem P

$$z = \min \sum_{t=1}^{T} \sum_{i=1}^{N} (c_{it}X_{it} + h_{it}I_{it} + s_{it}Y_{it}) \tag{1}$$

subject to

$$I_{i,t-1} + X_{it} - \sum_{j \in S(i)} a_{ij}X_{jt} - I_{it} = d_{it},$$

$$i = 1,\dots,N,\ t = 1,\dots,T, \tag{2}$$

$$\sum_{i \in R(k)} (ts_{it} Y_{it} + p_{it} X_{it}) \leqslant b_{kt},$$
$$k = 1, \ldots, K, \; t = 1, \ldots, T, \qquad (3)$$

$$X_{it} - \left(\sum_{t=1}^{T} \sum_{i=1}^{N} d_{it} \right) Y_{it} \leqslant 0,$$
$$i = 1, \ldots, N, \; t = 1, \ldots, T, \qquad (4)$$

$$X_{it} \geqslant 0, I_{it} \geqslant 0, Y_{it} = 0, 1,$$
$$i = 1, \ldots, N, \; t = 1, \ldots, T, \qquad (5)$$

where

T	= number of periods in the planning horizon
N	= number of items
K	= number of resources
c_{it}	= unit production cost of item i in period t
h_{it}	= unit inventory holding cost of item i in period t
s_{it}	= production set-up cost for item i in period t
a_{ij}	= number of units of item i required for the production of one unit of item j
d_{it}	= external demand for item i in period t
ts_{it}	= set-up time for item i in period t
p_{it}	= unit processing time of item i in period t
b_{kt}	= available capacity of resource k in period t
$S(i)$	= set of direct successors of item i in the product structure (item j is said to be the successor of item i if item i is a component of item j in the product structure)
$R(k)$	= set of items which require resource k
X_{it}	= lot-size of item i in period t
Y_{it}	= a binary variable indicating set-up for item i in period t
I_{it}	= ending inventory of item i in period t

(Further notation will be introduced as the need arises.)

The model assumes that there are a total of N items, some of which are end items with known external demand requirements given for a planning horizon of T periods. The formulation also includes external demand for the component items that appear in the product structures. The product structures which specify the technological input–output relations are assumed to be general in the sense that $S(i)$ may have more than one element; on the other hand, the product structures are called linear or assembly product structures if $S(i)$ has a single element for all the items. The operations and accordingly the resources needed in each operation are known for each item. Capacity limitations are imposed on the availabilities of these K resources, each of which may be needed by different items at different levels of the product structure, which implies that capacity sharing is permitted in the model. However in order to simplify the computations to follow, it is assumed, without loss of generality, that each item requires a single resource.

The objective function given by (1) aims at minimizing the variable production costs, inventory holding costs and setup costs. Constraints (2) require that the production of each item in each period together with beginning inventories should meet the external demand and the dependent demand generated by the successor items since backlogging is not permitted in the model. The capacity limitations upon the resources are expressed by constraints (3). Constraints (4) ensure that setup is performed (i.e. $Y_{it} = 1$) whenever there is production in period t (i.e. $X_{it} > 0$).

3. Description of the Lagrangean relaxation

As it is well-known, Lagrangean relaxation aims at decomposing a combinatorial optimization problem into a number of easy-to-solve subproblems by dualizing the "difficult" coupling constraints. As it is discussed in [19], the Lagrangean relaxation can be used in place of a linear programming relaxation efficiently to provide a lower bound (for minimization problems) on the optimal value of the original problem. In (P), both inventory (2) and capacity (3) constraints are linking the production quantities of items over the planning horizon, so the first decision to be made is selecting between different relaxations. Since the relaxation performance is evaluated by both the computational efficiency of solving the resulting subproblems and the goodness of the bounds obtained, two different relaxations are proposed in this study. In the first relaxation which is called hierarchical Lagrangean relaxation (HL), only the capacity constraints (3) are relaxed, so the relaxed problem turns out to be an uncapacitated multi-level lot sizing problem, which is not still so easy to solve optimally. This research proposes to solve this relaxed problem in an hierarchical manner and modify the sub-gradient optimization to circumvent the loss of optimality in solving the sub-problems. In the second relaxation method called non-restricted Lagrangean relaxation (FL), both multi-level inventory (2) and capacity (3) constraints are relaxed, so that the model is decomposed into N independent uncapacitated lot sizing sub-problems, each of which can be solved by the classical Wagner–Whitin algorithm. The general Lagrangean relaxation methodology used in this study and in similar previous studies can be summarized as follows:

Step 1: Solve the subproblems resulting from relaxation and update the lower bound for the original problem.

Step 2: Using the solution obtained in Step 1, determine the sub-gradients for the relaxed constraints and carry out sub-gradient optimization to update the values of the Lagrange multipliers.

Step 3: Execute a Lagrangean heuristic on the solution obtained in Step 1 to get a feasible solution and update the upper bound. Repeat Steps 1 and 2 with the new values of the Lagrange multipliers obtained in Step 2 until a termination criterion is satisfied.

In the remaining part of this section, the two relaxations will be described in detail.

3.1. *Hierarchical Lagrangean relaxation* (*HL*)

The first relaxation scheme is obtained by dualizing only the capacity constraints (3) and including them into the objective function:

Problem HL(u)

$$z_L(u) = \min \sum_{t=1}^{T}\sum_{i=1}^{N}(c_{it}X_{it} + h_{it}I_{it} + s_{it}Y_{it}) + \sum_{t=1}^{T}\sum_{k=1}^{K}u_{kt}\left(\sum_{i\in R(k)}(ts_{it}Y_{it} + p_{it}X_{it}) - b_{kt}\right) \tag{6}$$

subject to (2), (4), (5).

Here $u = \{u_{kt}: u_{kt} \geqslant 0\}$ is a vector of Lagrange multipliers. Rearranging the terms in (6), it is easy to obtain

$$z_L(u) = \min \left\{\sum_{t=1}^{T}\sum_{i=1}^{N}(c_{it} + p_{it}u_{kt})X_{it} + h_{it}I_{it} + (s_{it} + ts_{it}u_{kt})Y_{it}\right\} - \sum_{t=1}^{T}\sum_{k=1}^{K}b_{kt}u_{kt} \tag{7}$$

subject to (2), (4), (5), and $u \geqslant 0$.

Since it is assumed that each item requires a single resource, (7) presumes that resource k is the particular resource required by item i. The relaxed problem HL(u) is an uncapacitated multi-level lot sizing problem which is not so easy to solve optimally either. Since it is well-known that $z_L(u) \leqslant z$, utmost effort should be directed towards solving HL(u) optimally in general; however in this study an approximate solution is found to HL(u), which will be further improved by global search approaches. Thus HL(u) is decomposed into N single-item lot sizing problems HL$_i$(u), $i = 1, \dots, N$, given by

Problem HL$_i$(u)

$$z_L^i(u) = \min \sum_{t=1}^{T}(c_{it} + p_{it}u_{kt})X_{it} + h_{it}I_{it} + (s_{it} + ts_{it}u_{kt})Y_{it} \tag{8}$$

subject to

$$I_{i,t-1} + X_{it} - I_{it} = d_{it} + \sum_{j\in S(i)}a_{ij}X_{jt}, \quad t = 1, \dots, T \tag{2'}$$

and (4), (5).

Here $\{X_{jt}: j\in S(i)\}$ denotes the optimal lot-sizing decisions of the successors of item i determined such that the sub-problems HL$_i$(u), $i = 1, \dots, N$, are solved in a hierarchical manner in accordance with the product structure. This approach implies that first HL$_i$(u) is solved for the end items by using the Wagner–Whitin algorithm. These optimal lot-sizes are then propagated to the immediate predecessors, so that HL$_i$(u) is solved for all of the predecessor items to determine the lot sizes so as to satisfy the requirements of all successors. Once the product structure is totally exploded, the cost of the complete solution is given by

$$z_0 = \sum_{t=1}^{T}\sum_{i=1}^{N}(c_{it}X_{it} + h_{it}I_{it} + s_{it}Y_{it}) \tag{9}$$

which is no longer an actual lower bound for the original problem (P), but provides an initial solution, possibly a capacity infeasible one, for the search procedures to follow. It is important to note that solving the relaxed problem in such a hierarchical manner guarantees that the solution is feasible with respect to inventory constraints.

Naturally the best choice for u is to solve the dual problem given by

Problem D

$$z_D = \max_u \ z_L(u)$$

One efficient method to solve (D) is to employ sub-gradient optimization without loss of generality. The systematic updating of the multipliers will be achieved according to the implementation given in [20]. Thus the capacity sub-gradients of HL(u) are determined by

$$\beta_{kt} = \sum_{i \in R(k)} (ts_{it} Y_{it} + p_{it} X_{it}) - b_{kt},$$

$$k = 1, \dots, K, \ t = 1, \dots, T. \tag{10}$$

At any iteration r, the values of the Lagrange multipliers are updated by

$$u_{kt}^{r+1} = u_{kt}^r + \mathrm{ss}^r \beta_{kt}^r, \quad k = 1, \dots, K, \ t = 1, \dots, T,$$

where

$$\mathrm{ss}^r = \lambda |z_{\text{upper}}^r - z_0^{r-1}| / \|\beta\|^2. \tag{11}$$

Here z_{upper}^r is an upper value on (P), z_0^{r-1} is the cost associated with the solutions of the subproblems in the previous iteration as computed by (9), $\|.\|$ denotes the Euclidean norm, $|.|$ denotes the absolute value and λ is taken as 2 in all iterations. A thorough analysis reveals that (11) is different from the standard step-size updating in sub-gradient optimization. Since HL(u) is solved only approximately, z_0 given by (9) is not necessarily a lower bound for z and may turn out to be greater than the upper bound in any iteration. Thus the absolute value of the distance between z_0 and the upper bound should be used in order to be able to continue the iterations. Furthermore whenever the step-size in any iteration r decreases below 0.0001, the step size is generated randomly between 0.0001 and 0.1001.The algorithm is terminated either at the end of a given number of Lagrange iterations or whenever a feasible solution is directly obtained by solving HL$_i$(u), $i = 1, \dots, N$.

3.2. Non-restricted Lagrangean relaxation

The second relaxation is obtained by dualizing both the inventory balance (2) and capacity (3) constraints and including them into the objective function.

Problem FL(u, v)

$$z_L(u, v)$$
$$= \min \sum_{t=1}^{T} \sum_{i=1}^{N} (c_{it} X_{it} + h_{it} I_{it} + s_{it} Y_{it})$$
$$+ \sum_{t=1}^{T} \sum_{i=1}^{N} v_{it} \left(I_{i,t-1} + X_{it} - \sum_{j \in S(i)} a_{ij} X_{jt} - I_{it} - d_{it} \right)$$
$$+ \sum_{t=1}^{T} \sum_{k=1}^{K} u_{kt} \left(\sum_{z \in R(N)} (ts_{it} Y_{it} + p_{it} X_{it}) - b_{kt} \right) \tag{12}$$

subject to (4) and (5).

Here again $u = \{u_{kt}: u_{kt} \geqslant 0\}$ is a vector of Lagrange multipliers associated with (3) and $v = \{v_{it}: v_{it} \text{ free}\}$ is a vector of Lagrange multipliers associated with (2). Rearranging the terms and letting V_i denote the set of immediate predecessors of item i, (12) is simplified to

$$z_L(u, v)$$
$$= \min \sum_{t=1}^{T} \sum_{i=1}^{N} \left(c_{it} + v_{it} - \sum_{j \in V(i)} a_{ji} v_{jt} + p_{it} u_{kt} \right) X_{it}$$
$$+ (h_{it} + v_{i,t+1} - v_{it}) I_{it} + (s_{it} + ts_{it} u_{kt}) Y_{it}$$
$$+ \sum_{i=1}^{N} I_{i0} v_{i1} - \sum_{t=1}^{T} \sum_{i=1}^{N} d_{it} v_{it} - \sum_{t=1}^{T} \sum_{k=1}^{K} b_{kt} u_{kt}. \tag{13}$$

Here $v_{i,T+1} = 0$ for all i. It is easy to show that FL(u, v) is decomposable into N uncapacitated single-item lot-sizing problems. Since the input–output relations between the items have disappeared through relaxation, it is possible that the production quantities obtained by solving (13) may lead to under-production even for a given item over the planning horizon. This situation may create poor solutions for which it might be very difficult to generate a feasible solution in the Lagrangean heuristic; thus inventory constraints are added into the single-item sub-problems to guarantee that production for each item meets the dependent demand. The single-item sub-problems $\mathrm{FL}_i(u, v)$, $i = 1, \ldots, N$, are given by

Problem FL$_i$(*u*, *v*)

$$z_L^i(u, v) = \min \sum_{t=1}^{T} \left(c_{it} + v_{it} - \sum_{j \in V(i)} a_{ji} v_{jt} + p_{it} u_{kt} \right) X_{it} + (h_{it} + v_{i,t+1} - v_{it}) I_{it} + (s_{it} + ts_{it} u_{kt}) Y_{it} \tag{14}$$

subject to

$$I_{i,t-1} + X_{it} - I_{it} = d_{it} + \sum_{j \in S(i)} a_{ij} D_{jt}, \quad t = 1, \ldots, T$$

and (4) and (5).

Here D_{jt} is the dependent demand for any item j obtained by exploding the external demand of the items through the product structure. Consequently $\mathrm{FL}_i(u, v)$, $i = 1, \ldots, N$, can be easily solved by the Wagner–Whitin algorithm for each item independently. The cost of the current solution z_0 is determined by (9) where the inventory quantities in-between the levels are recursively computed by (2) and taken as zero whenever it turns out to be negative. Again z_0 is not an actual bound for the reasons explained above.

The sub-gradient optimization is implemented to solve the dual problem and to update the Lagrange multipliers where the sub-gradients are determined by

$$\alpha_{it} = I_{i,t-1} + X_{it} - \sum_{j \in S(i)} a_{ij} X_{ij} - I_{it} - d_{it}, \quad i = 1, \ldots, N, \ t = 1, \ldots, T, \tag{15}$$

$$\beta_{kt} = \sum_{i \in R(k)} (ts_{it} Y_{it} + p_{it} X_{it}) - b_{kt}, \quad k = 1, \ldots, K, \ t = 1, \ldots, T. \tag{16}$$

The Lagrange multipliers are recursively updated by

$$v_{it}^{r+1} = v_{it}^r + \mathrm{ss}^r \alpha_{it}^r, \quad i = 1, \ldots, N, \ t = 1, \ldots, T, \tag{17}$$

$$u_{kt}^{r+1} = u_{kt}^r + \mathrm{ss}^r \beta_{kt}^r, \quad k = 1, \ldots, K, \ t = 1, \ldots, T, \tag{18}$$

where the step-size ss^r is found by

$$\mathrm{ss}^r = \lambda |z_{\mathrm{upper}}^r - z_0^{r-1}| / \|\beta\|^2 + \|\alpha\|^2.$$

The same conventions and modifications are repeated in the non-restricted step-size updating. In both relaxation schemes, the multipliers are initialized from 0, and z_{upper}^r is computed by a simple heuristic in the very first iteration. One remaining issue is the design of the Lagrangean heuristic to generate a feasible solution, which is the main focus of this study and which will be covered in detail in the next section. In any iteration of (HL), the solution will be certainly inventory feasible, but it might turn out to be capacity infeasible. Similarly, the solution in each iteration of (FL) might turn out to be both capacity and inventory infeasible due to the relaxation of both (2) and (3). Thus there is a need to design different Lagrangean heuristics capable to restore feasibility depending upon the particular relaxation structure.

4. The Lagrangean heuristic

Since there is no guarantee that the solution of the dual problem (D) will generate a solution feasible to the original problem (P) because of duality gap, it is necessary to design a procedure which obtains a feasible solution out of the near-feasible Lagrangean solution in each sub-gradient optimization iteration. Generally the dual solution is input into a Lagrangean heuristic to obtain a feasible solution whose objective function value can be used as the upper bound for the problem (P) and which is improved as the procedure progresses. In this study it is important to note that, since z_0 is not necessarily a lower bound for (P), the Lagrangean heuristic should be designed to generate not only a feasible solution, but more importantly a feasible solution with an objective function value possibly smaller than z_0. Thus the generation of a feasible solution should not be restricted to the neighborhood of the current solution, but instead should possess the capability to carry out a global search which improves the upper bound and finds a more realistic z_0. To serve this purpose, a simulated annealing (SA) approach is designed to find a feasible solution with the minimum objective function value starting from the current Lagrangean solution as the initial solution. In SA, a move is generated by shifting some production quantity of an item from a given time period to another period either in order to restore both inventory and capacity feasibility, or just capacity feasibility, or just to improve z. SA is first applied such that the solutions worse than the current solution is accepted with a given probability. As another Lagrangean heuristic, SA is modified to accept only the improving moves where only solutions better than the current one are implemented. This approach is called simulated annealing with only improving moves (SAI).

In each iteration of the Lagrangean relaxation, SA search starts from the current solution of the relaxed sub-problems and tries to improve it by shifting the lot-size of only one item at a time. In fact the neighborhood around the current solution is defined by an hierarchy of decisions in the following manner:

(i) Decide randomly upon the shift type of whether to increase or decrease a lot.
(ii) Select randomly an item i and the time period t in which the first decision is to be implemented.
(iii) Determine the resource k required by item i in period t.
(iv) Determine the quantity to be shifted.

If the decision is made to "increase" in (i), then some production quantity of item i is shifted from period $t+1$ to period t, which is called one-step backward shifting. However if the decision is made to "decrease" in (i), some production quantity of item i is shifted from t to $t+1$. This is called one-step forward shifting. Thus the first decision is the selection between a backward and a forward shift. Then the time period and the item are selected randomly without considering overload/underload situation of the resources. The maximum quantity that can be shifted depends upon the feasibility state of the solution at the point where the search is initiated. In fact both search procedures can be executed upon three different types of solutions: all-feasible, all-infeasible, and only-inventory-feasible solutions.

In "all-feasible" solutions, the idea is to improve the current objective function value without violating capacity and inventory feasibility; thus, the moves which might lead to some infeasibility in (2) and (3) are avoided. In these solutions, when the "decrease" decision is made, the maximum quantity of item i to be shifted from t to $t+1$ is determined by

$$\Delta_i = \max\{0, \min\{I_{it}, X_{it}, Q_{k,t+1}\}\}. \tag{19}$$

Here $Q_{k,t+1}$ is defined to be the maximum quantity of item i which can be shifted without violating the feasibility of resource k in period $t+1$ and given by

$$Q_{k,t+1} = -\beta_{k,t+1}/p_{i,t+1} \quad \text{if } X_{i,t+1} \neq 0$$

and (20)

$$Q_{k,t+1} = (-\beta_{k,t+1} - ts_{i,t+1})/p_{i,t+1} \quad \text{if } X_{i,t+1} = 0$$

so that the lot-sizing decisions of immediate successors in the product structure are not affected and the resource k in period $t+1$ does not become overloaded. However when the "increase" decision is made, the maximum quantity of item i to be shifted from period $t+1$ to period t should consider the availability of material from the preceding items and is given by

$$\Delta_i = \max\left\{0, \min\left\{\min_{j\in Vi} I_{jt}, X_{i,t+1}, Q_{kt}\right\}\right\}. \tag{21}$$

The change in the cost given by Δ_{cost} is determined by considering the net effect of inventory holding, production and set-up changes resulting from this move.

In "all-infeasible" solutions, it is aimed to generate a feasible solution with the best objective function value out of a solution which is infeasible with respect to both (2) and (3). When the "decrease" decision is made, the maximum quantity of item i to be shifted from t to $t+1$ is only limited by the lot-size in t and

$$\Delta_i = X_{it}. \tag{22}$$

When the "increase" decision is made, the maximum quantity of item i to be shifted from period $t+1$ to period t is determined by the lot-size in $t+1$ and

$$\Delta_i = X_{i,t+1}. \tag{23}$$

In these solutions, the infeasible moves are penalized by assigning penalties to negative inventory levels and capacity infeasibilities. These penalties are included into the cost of the move Δ_{cost} in order to reduce the attractiveness of the move.

In "only-inventory-feasible" solutions, the aim is to generate a capacity feasible solution out of a solution which is feasible with respect to (2), but infeasible with respect to (3), without violating inventory feasibility. Thus the moves which lead to inventory infeasibility are avoided. In case of "decrease" decisions,

$$\Delta_i = \min\{I_{it}, X_{it}\} \tag{24}$$

ensures that the inventory balances for the predecessors are preserved in a possible move. In case of increase decisions, the maximum quantity of item i to be shifted from $t+1$ to t given by

$$\Delta_i = \min\left\{\min_{j \in Vi} I_{jt}, X_{i,t+1}\right\} \tag{25}$$

avoids the possibility of having shortage of inventory for the predecessors of item i. Again capacity infeasibility is assigned a penalty cost and included into Δ_{cost} in order to reduce the possibility of performing such a move.

Once a neighbor is generated, it is important to determine whether it improves the cost or not. Whenever the move leads to an improvement over the current cost, it is directly implemented. However, when the current solution is worsened, the acceptance of non-improving moves is randomized by assigning a probability (PA) of accepting the move defined by

$$\text{PA}^r = \exp[-\Delta_{\text{cost}}/z^{r-1}E^r] \tag{26}$$

where z^{r-1} is the cost of the current solution and E^r is the annealing temperature in iteration r of the search. E^r is updated, whenever an unfavorable move is accepted, by

$$E^{r+1} = E^r/(1 + BE^r),$$

$$E^{r+1} = 1 \text{ whenever } E^{r+1} < 0.001.$$

This geometric cooling is introduced as simulated quenching in [21] and shown to converge very rapidly for similar problems. Efficient implementations of this temperature cooling scheme can also be found in [22,23]. Here B is the annealing parameter for which different values are tested in this study and the best results are obtained for $B = 0.01$. A general pseudo-code is given in Fig. 1 where Δ_i is defined by (19)–(25) depending on the situation.

Since Lagrangean relaxation is expected to converge to the global optimum, it is anticipated that the second approach SAI which prohibits the acceptance of unfavorable moves will accelerate the convergence rate as compared to SA. In SAI, the generation of a neighbour and the definition of

```
do{
    randomly make a decision to increase or decrease lot size;
    if decrease, then{
        select randomly item i and period t;
        find resource k required by item i;
        find $\Delta_i$;
        $\Delta_i \leftarrow$ random($\Delta_i$);
        calculate $\Delta_{cost}$;
        If $\Delta_{cost}$> 0, then{
                calculate PA by (26);
                }
        If $\Delta_{cost}$< 0 or a non-improving is accepted, then{
                transfer $\Delta_i$ from period t to period t+1.
    }
    if increase, then{
        select randomly item i and period t;
        find resource k required by item i;
        find $\Delta_i$;
        $\Delta_i \leftarrow$ random($\Delta_i$);
        calculate $\Delta_{cost}$;
        If $\Delta_{cost}$> 0, then{
                calculate PA by (26);
                }
        If $\Delta_{cost}$< 0 or a non-improving is accepted, then{
                transfer $\Delta_i$ from period t+1 to period t.
    }
    searchcount$\leftarrow$searchcount+1
}while (seachcount< maxcount )
```

Fig. 1. Pseudocode for SA procedure.

a move is made in a similar manner, but the acceptance of nonimproving moves is avoided. This implies that the move is not performed in SAI procedure if it does not lead to a cost-effective neighbor; i.e. if $\Delta_{\text{cost}} > 0$ and a new neighbor is generated to continue the search.

The SA and SAI procedures described above are separately used as the Lagrangean heuristic both within the hierarchical Lagrangean and nonrestricted Lagrangean relaxations. When used within the hierarchical Lagrangean relaxation, Δ_i's are determined by (24) and (25) since they attempt to restore capacity feasibility for "only-inventory-feasible" solutions obtained by solving sub-problems $\text{HL}_i(u)$, $i = 1, \dots, N$, hierarchically. When used within the nonrestricted Lagrangean relaxation, Δ_i's are determined by (22) and (23) and they attempt to attain a solution which satisfies (2) and (3) out of an "all-infeasible" solution obtained by solving sub-problems $\text{FL}_i(u)$, $i = 1, \dots, N$, independently.

Furthermore, it is determined to develop a heuristic procedure which will restore capacity feasibility a-priori before the global search commences. Thus a recursive procedure is designed to generate a feasible solution upon which both SA and SAI procedures can be implemented to bring along further improvement. Since this procedure acts as a Phase-I step upon the Lagrangean solution before the search procedures, it is called Phase-1 procedure. This way it will be possible to analyze the performance of two SA approaches when they are initiated from an "all-feasible" solution in which case Δ_i's are determined by (19) and (20).

5. Phase-1 procedure

This procedure is an iterative heuristic which carries out a search around a given solution either until a capacity feasible solution is obtained or a given number of iterations is reached without obtaining a feasible solution. The main steps of the procedure are summarized below:

(i) Determine randomly an overloaded resource k and time period t.
(ii) Select randomly an item i requiring resource k in period t.
(iii) Carry out backward shifting by transferring production quantities of item i and multi-item shifting of its predecessors if necessary from period t to some earlier period without violating inventories of predecessors.
(iv) If solution is still infeasible, carry out forward single-item shifting by transferring production quantities of item i from period t to some later period without violating the requirements of the successors.
(v) Continue with (ii) if there still remain unconsidered items requiring resource k in period t and resource k is still capacity infeasible.

Here backward shifting is performed before forward shifting in order to provide more freedom to higher level items which might require backward shifting. There exist three main differences between this procedure and the SAI and SA procedures. First of all, in the global search procedures only one-step backward and forward shifts are considered between two consecutive time periods; however in the Phase-1 procedure, starting from a selected period, first backward shifts are tried either until capacity infeasibility is totally resolved or the time origin is reached. Then forward shifts are made starting from the selected period onward. The second difference is related to the treatment of predecessors: Phase-1 procedure carries out multi-item shifting in the backward stage by modifying the lot-sizes of the predecessors if necessary while only single-item shifting is considered in the global search procedures. Finally in Phase-1, transfer lot-sizes are determined by the minimal quantity required to attain feasibility; however the global search procedures are free to transfer quantities up to the whole lot-sizes as long as other constraints permit.

In backward shifting, the maximum quantity of item i to be shifted from t to some earlier period tt, $tt = t - 1, \dots, 0$, is given by

$$\Delta_i = \max\{0, \min\{\Delta^*, X_{it}, Q_{k,tt}\}\} \qquad (27)$$

where Δ^* is the maximum quantity permitted by the predecessors of item i in the whole product structure. Since changes in lot-sizes of item i will influence the lot-sizes of all the predecessors, the effects of backward shifts should be propagated down the product structure without worsening capacity feasibility of other resources required by them. Thus the procedure explodes the product structure recursively starting from the item in question until the lowest level is reached. When it is attempted to increase the lot-size of item i in some earlier period in backward shifting, the inventories of immediate predecessors are checked to see if their inventories are sufficient to permit such an increase; if not, then it is attempted to increase the lot-sizes of the predecessors by the necessary quantities. This requires the explosion of their own immediate predecessors. Furthermore, as the lot-sizes of the predecessors whose inventories restrict the transfer of a lot to an earlier period are increased in the required periods, their respective lot-sizes in some future time period should be decreased by the same quantity. The recursive procedure which carries out this propagation to determine Δ^* is described in the procedure **increase** (**item**:j, **period**:s, **quantity**:Δ_j) triggered for some resource k overloaded in period s and given by the pseudo-code in Fig. 2.

In forward shifting of the procedure Phase-1, the maximum quantity of item i to be shifted from period t to some future period tt, $tt = t + 1, \dots, T$, is given by

$$\Delta_i = \max\left\{0, \min\left\{\min_{t \leqslant z \leqslant tt} I_{iz}, X_{it}, Q_{k,tt}\right\}\right\} \tag{28}$$

```
$\Delta_j'' = 0$; count=0; $\Delta_j' \leftarrow \Delta_j' - I_{js}$; r= resource of item j;
    do{
        count$\leftarrow$count+1;
        tt'= s+count;
        calculate $\nabla_j = \min\{Q_{rs}, X_{j,tt'}, \Delta_j'\}$;
       for t'=s to tt' do
           for m=1 to immeprecno[j] do
               if inv[immeprec[m,j]t']<$\nabla_j$ then
                   $\nabla_j \leftarrow$ increase(immeprec[m,j],t', $\nabla_j$ )
        increase $X_{js}$ by $\nabla_j$;
        decrease $X_{j,tt'}$ by $\nabla_j$;
        update all resource idle times and inventories;
        $\Delta_j' \leftarrow \Delta_j' - \nabla_j$; $\Delta_j'' \leftarrow \Delta_j'' + \nabla_j$;
    }while ($Q_{rs} > 0$ AND $\Delta_j' > 0$ AND tt'<maxperiod AND $\beta_{ks} > 0$)
  return( $\Delta_j''$ );
```

Fig. 2. Pseudo-code for procedure **increase** (**item**:j, **period**:s, **quantity**:Δ_j).

```
do{
        select randomly a resource k and time period t such that $\beta_{kt} > 0$;
        do{
            select randomly an item i on (k,t);
            tt $\leftarrow$ t-1
           do{
               calculate $\Delta_i$;
                 for s=tt to t do
                     for m=1 to immeprecno[i] do
                         if inv [immeprec[m,i],s] < $\Delta_i$ , then
                             $\Delta_i \leftarrow$ increase ( immeprec[m,i],s, $\Delta i$);
               transfer $\Delta_i$ from t to tt;
              } while(tt $\geq$ 0 AND $X_{it} > 0$ AND $\beta_{kt} > 0$)
            tt $\leftarrow$ t+1
            if $\beta_{kt} > 0$, then
           do{
               calculate $\Delta_i$;
               transfer $\Delta_i$ from t to tt;
               tt $\leftarrow$ t+1
              }while(tt$\leq$T AND $X_{it} > 0$ AND $\beta_{kt} > 0$)
         } while(all items on (k,t) are not considered AND $\beta_{kt} > 0$)
            phascount $\leftarrow$ phascount+1
    }while (phascount< maxcount AND solution capacity infeasible).
```

Fig. 3. Pseudo-code for Phase-1 procedure.

which implies that only single-item shifting is tried for item i without violating the inventories of successors and worsening capacity feasibility. The pseudo-code for Phase-1 procedure is given in Fig. 3. Here immeprecno[i] is used to denote the number of the immediate predecessor of item i, and inv(immeprec[m,i],s] denotes the inventory level of the mth immediate predecessor of item i in period s.

The global search procedures SAI and SA and Phase-1 heuristic are used in combination with the two Lagrangean relaxation schemes discussed in Section 3 resulting in six different solution approaches for problem (P). These can be listed as follows:

(1) hierarchical Lagrangean relaxation (HL) + simulated annealing with only improving moves(SAI);
(2) hierarchical Lagrangean relaxation (HL) + simulated annealing (SA);
(3) hierarchical Lagrangean relaxation (HL) + phase-1(P1) + simulated annealing with only improving moves (SAI);
(4) hierarchical Lagrangean relaxation (HL) + phase-1(P1) + simulated annealing (SA);

(5) non-restricted Lagrangean relaxation (FL) + simulated annealing with only improving moves (SAI);
(6) non-restricted Lagrangean relaxation (FL) + simulated annealing (SA).

Tempelmeier and Derstroff [9] also used Lagrangean relaxation to solve the multi-level multi-item capacitated lot sizing problem (P). They first eliminated the inventory variables I_{it} from the formulation in order to transform the problem into a form which can be solved by a procedure similar to Federgruen and Tzur [24]. Then they relaxed both the material balance and capacity constraints, so their relaxed model is similar to the non-restricted Lagrangean relaxation developed in this study. They also employed sub-gradient optimization procedure to update the values of Lagrange multipliers. As for the Lagrangean heuristic to find the upper bound, they designed a finite loading heuristic which is initiated from an "inventory-feasible" solution. The "inventory-feasible" solution was obtained by solving the relaxed problem only for the end items and then exploding the optimal lot sizes of the end items over the product structure. Although this is similar in a sense to the hierarchical Lagrangean relaxation done in this study, $\mathrm{HL}_i(u)$ as defined by (8) and (2′) is solved for all the items in the order specified by the product structure, not only for the end items. The finite loading heuristic in [9] then performs backward and forward shifting for the overloaded resources in the solution obtained by the explosion of the product structure. The main difference between their Lagrangean heuristic and Phase-1 procedure designed in this study is that they carry out multi-item shifting in both backward and forward shifting while Phase-1 carries out multi-item shifting only in backward shifting. In order to compare the performance of the procedures developed in this study against the approach of Tempelmeier and Derstroff [9], their Lagrangean relaxation scheme and Lagrangean heuristic are re-coded in their basic forms and the results are denoted as (TD). The performance of the approaches among themselves and against that of Tempelmeier and Derstroff [9] will be provided in the computational study.

6. Computational study

The six approaches mentioned above are tested on Class *A* and Class *B* problems given in [9] because optimum solutions are reported in [9] only for these problems. Class *A* problems consist of 1500 small problems with 10 items, 4 time periods, 3 resources, and no setup times. Class *B* problems consist of 600 problems generated from Class *A* problems with setup times. Here the number of Lagrangean iterations is taken as 100 for Class *A* and 150 for Class *B* problems in all the procedures including the re-coded version of the TD approach, so that a fair comparison among all the procedures can be achieved. The global search procedures are executed with 5000 iterations during the first quarter of Lagrangean iterations, 3000 iterations during the second and third quarters, and 2000 iterations during the last quarter of Lagrangean iterations. Phase-1 procedure is executed at most 10 times to find an "all-feasible-solution" in all instances it is included

The quality of the solutions are compared by computing the deviation from optimality (Optimum-UB)/Optimum for Class *A* and *B* problems. The average deviations and the average computation times taken on a Pentium 133 MHz machine are provided for each problem class in Table 1. The analysis of the deviation results across the six algorithms developed in this study reveals that HL relaxation performs better than FL relaxation for any Lagrangean heuristic implemented, and SAI produces better results than SA. The fact that SAI performs better than SA for any relaxation type

Table 1
Average deviations and computation times (s)

	Class *A*		Class *B*	
	dev	cpu	dev	cpu
HL + SAI	8.12	14.78	7.38	13.03
HL + SA	23.94	29.73	20.94	68.33
HL + P1 + SAI	4.63	5.88	4.21	8.54
HL + P1 + SA	6.89	28.49	6.04	37.77
FL + SAI	13.61	11.77	12.27	9.02
FL + SA	39.13	54.34	38.70	91.94
TD	11.67	1.49	12.06	1.45

indicates that the acceptance of non-improving moves should not be randomized. As it is expected, the convergence of Lagrangean relaxation to the global optimum is accelerated by SAI. Furthermore, Phase-1 procedure improves the solution quality in any combination, which implies that the global search procedures perform better when restricted within feasible solution space. In fact, HL + P1 + SAI approach outperforms the other approaches in all problem types. The same conclusions can be made for the computation times; SA takes the longest time in all combinations and the inclusion of the Phase-1 procedure decreases the computation time considerably. This again verifies the relative performance of the search procedures SA and SAI; they improve the solution better when they search in the feasible space mainly due to the fact that infeasible moves are prevented a priori, thus eliminating penalty calculations as well. Among the six approaches developed in this study, HL + P1 + SAI takes the shortest time.

Tempelmeier and Derstroff [9] report that average deviation from optimality obtained by TD procedure is 1.47% (varying between 0.01% and 2.99%) for Class *A* problems and 1.30% (varying between 0.00% and 2.87%) for Class B problems. Although the TD procedure was re-coded in its basic form as it is described in [9], these results could not be reproduced in the re-coded form of the TD procedure as implemented within the context of this study. Based upon the results generated in this study, HL + P1 + SAI outperforms the TD approach since the inclusion of global search into the Lagrangean relaxation approach has improved the results as initially expected. As for the computation times, the average computation time of TD is reported in [9] as 1.4 CPU seconds for Class *A* and *B* problems where the number of Lagrange iterations is 50. The computation times for the re-coded TD are indicated as 1.49 and 1.45 CPU seconds for Class *A* and *B* problems, respectively, in Table 1. These are consistent with the ones reported in [9] since the Lagrangean iterations are taken as 100 in this implementation where a faster hardware is utilized. Although the computational time of HL + P1 + SAI is slightly higher than that of TD, it is still very fast in finding high quality solutions. The fact that the computational times are low enables the practitioner to execute these procedures on-line as the parameters of the model change dynamically. As long as the practitioner has access to real-time inventory data, the production plan generated by these procedures can be frequently updated especially in response to demand fluctuations in fast-moving items.

7. Conclusion

This study develops a heuristic approach for solving the MLCLSP which is very frequently encountered in real life and which is still of great interest to the operations researchers. The advancements in the area of metaheuristics has mainly motivated this study which aims to integrate simulated annealing concepts with the classical Lagrangean relaxation methodology. In fact this research is one of the first attempts integrating these two well-known approaches. Different procedures are designed and tested upon the benchmark problems available in literature, and the results are compared with the Lagrangean relaxation procedure proposed in [9]. The preliminary results indicate that the solutions obtained in this study are satisfactory with respect to the deviations from the known optimal solutions and the computational times. Hence the integration of Lagrangean relaxation schemes with meta-heuristics prove to be mutually beneficial for both approaches.

References

[1] G.R. Bitran, H.H. Yanasse, Computational complexity of the capacitated lot size problem, Management Science 28 (1982) 1174–1186.

[2] M. Garey, D. Johnson, Computers and Intratractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, CA, 1979.

[3] P.J. Billington, J.O. McClain, L.J. Thomas, Heuristics for multi-level lot sizing with a bottleneck, Management Science 32 (1986) 989–1006.

[4] P.J. Billington, J. Blackburn, J. Maes, R. Millen, L.N. Van Wassenhove, Multi-item lot sizing in capacitated multi-stage serial systems, IIE Transactions 26 (1994) 12–18.

[5] J. Maes, L.N. Van Wassenhove, A simple heuristic for the multi-item single level capacitated lot sizing problem, Operations Research 4 (1986) 265–273.

[6] J. Maes, J.O. McClain, L.N. Van Wassenhove, Multi-level capacitated lot sizing complexity and LP-based heuristics, European Journal of Operational Research 53 (1991) 131–148.

[7] R. Kuik, M. Salomon, L.N. Van Wassenhove, J. Maes, Linear programming, simulated annealing and tabu search heuristics for lot sizing in bottleneck assembly systems, IIE Transactions 11 (1993) 319–326.

[8] H. Tempelmeir, S. Helber, A heuristic for dynamic multi-item multi-level capacitated lot sizing for general product structure, European Journal of Operational Research 75 (1994) 296–311.

[9] H. Tempelmeir, M. Derstroff, A Lagrangean-based heuristic for dynamic multi-item multi-level constrained lot sizing with setup times, Management Science 42 (1996) 738–757.

[10] P.M. Franca, V.A. Armentano, R.E. Berretta, A.R. Clark, A heuristic method for lot sizing in multi-stage systems, Computers and Operations Research 24 (1997) 861–874.

[11] A. Segerstedt, A capacity-constrained multi-level inventory and production control problem, International Journal of Production Economics 45 (1996) 449–461.

[12] J.M. Thizy, L.N. Van Wassenhove, Lagrangean relaxation for multi-item capacitated lot sizing problem: A heuristic implementation, IIE Transactions 17 (1985) 308–313.

[13] W.W. Trigerio, L.J. Thomas, J.O. McClain, Capacitated lot sizing with setup times, Management Science 35 (1989) 353–366.

[14] W.H. Chen, J.M. Thizy, Analysis of relaxations for the multi-item capacitated lot sizing problem, Annals of Operations Research 26 (1990) 29–72.

[15] S. Lozano, J. Larraneta, L. Onieva, Primal-dual approach to the single level capacitated lot sizing problem, European Journal of Operational Research 51 (1991) 354–366.

[16] M. Diaby, H.C. Bahl, M.H. Karwan, S. Zionts, A Lagrangean relaxation approach for very-large-scale capacitated lot sizing, Management Science 38 (1992) 1329–1340.

[17] P.J. Billington, 1983. Multi-level lot sizing with a bottleneck work center, Ph.D. Dissertation, Cornell University, Ithaca, NY, 1983.

[18] A. Kirkpatrick, C.D. Gelatt Jr., M.P. Vechi, Optimization by simulated annealing, Management Science 220 (1983) 671–680.

[19] M.L. Fisher, The Lagrangean relaxation method for solving integer problems, Management Science 27 (1981) 1–18.

[20] M. Held, P. Wolfe, H.P. Crowder, Validation of subgradient optimization, Mathematical Programming 6 (1974) 62–88.

[21] L. Ingberg, Simulated annealing: Practice versus theory, Journal of Mathematical Computational Modelling 18 (1994) 29–57.

[22] L. Ozdamar, I.S. Birbil, Hybrid heuristics for the capacitated lot sizing and loading problem with setup times and overtime decisions, European Journal of Operational Research 110 (1998) 525–547.

[23] L. Ozdamar, M.A. Bozyel, Simultaneous lot sizing and loading of product families on parallel facilities of different classes, International Journal of Production Research 36 (1998) 1305–1324.

[24] A. Federgruen, M. Tzur, A simple forward algorithm to solve general dynamic lot sizing models with n periods in $O(n \log n)$ or $O(n)$ time, Management Science 37 (1991) 909–925.