universität wien

DIPLOMARBEIT

Titel der Diplomarbeit

„Solution Methods for Lot-Sizing Problems – Multi-Level Models with and without Linked Lots“

Verfasser

Wolfgang Summerauer

angestrebter akademischer Grad

Magister der Sozial- und Wirtschaftswissenschaften (Mag. rer. soc. oec.)

Wien, im November 2008

Studienkennzahl lt. Studienblatt:	A 157
Studienrichtung lt. Studienblatt:	Internationale Betriebswirtschaft
Erster Betreuer:	O. Univ.-Prof. Dipl.-Ing. Dr. Richard F. Hartl
Zweiter Betreuer:	Univ.Ass. Dipl.-Ing. Dr. Christian Almeder

Contents

1 Introduction

In this thesis we center our attention on the lot sizing problem, which is part of the material requirements planning (MRP). In many production processes it costs money and takes time to setup a machine for a certain product. A lot-sizing problem therefore identifies the optimal timing and batch size of production. More precisely, it tries to minimize the inventory, setup, and production costs while meeting the required demand. Since production decisions and costs directly affect a company's efficiency and competitiveness in the market, the lot sizing problem is of utmost importance for every producing firm.

There are many different models and methods for solving various lot sizing problems. This thesis mainly deals with the multi-level capacitated lot-sizing problem (MLCLS), and then expands the model by adding the possibility of linking a setup state from one period to the next. The MLCLS problem is NP-complete (see Maes and McClain, 1991, for a proof). The solution approach used for the MLCLS problems is a hybrid algorithm from Pitakaso et al. (2006) which decomposes the given problem into multiple smaller subproblems. These subproblems are then solved by CPLEX, a commercial LP/MIP-Solver developed by ILOG. An Ant Colony Optimization (ACO) algorithm, a probabilistic metaheuristic that mimics the behavior of ants, is then applied to determine the lot-sizing sequence and to improve the decomposition. The ACO algorithm used in this thesis is a MAX-MIN ant system (MMAS) developed by Stützle and Hoos (1997). Our approach works very well with medium-sized problems, but is not able to compete with the other approaches when solving large-sized test instances.

As stated before, the capacitated lot-sizing problem is then expanded by adding a linkage property to the model. We use the same ant-based approach to solve the capacitated lot-sizing problem with linked lot sizes (CLSPL) which combines the characteristics of big- and small-bucket models. The CLSPL is a big-bucket model that allows the preservation of a setup state from one period to the next. We implement the CLSPL formulation suggested by Stadtler and Suerie (2003). Since this CLSPL model exchanges the common production variable by a simple plant location (SPL) formulation, we also test these two formulations for effectiveness. Our approach to solve the CLSPL problem is then tested with single-level and multi-level test instances.

The remainder of this thesis is organized as follows. Section 2 provides a detailed literature review of the different lot-sizing problems. Furthermore, an overview of the solution approaches for the lot-sizing problem is given. In the third Section the mathematical formulation is defined while the fourth Section explains the decomposition. Section 5 describes the MMAS algorithm and is followed by computational results for the MLCLS problem in Section 6. The solution approach for the linkage property is given in Section 7. Results for the CLPSL problem are presented in Section 8. Finally, the thesis finishes with a summary and further possible research in Section 9.

2 The Lot-Sizing Problem

2.1 Uncapacitated Lot-Sizing Problem

The first formulation of a dynamic lot-sizing problem dates back to Wagner and Whitin (1958) who introduced a single-item uncapacitated lot-sizing problem (ULS). In order to optimally solve the underlying problem a linear programming (LP) model is required. A LP model tries to optimize a certain objective function subject to some linear constraints. More precisely, the problem below is a mixed integer programming (MIP) model, which means that not all the variables have to be integer. The MIP model is as follows:

$$\min \sum_{t=1}^{T} (s_t y_t + h_t I_t + c_t^x x_t), \tag{1}$$

subject to

$$I_t = I_{t-1} + x_t - E_t, \quad \forall t, \tag{2a}$$

$$x_t \leq \left(\sum_{\tau=t}^{T} E_t \right) y_t \quad \forall t, \tag{2b}$$

$$I_t \geq 0, \quad x_t \geq 0, \quad y_t \in \{0,1\}, \quad \forall t. \tag{2c}$$

The model contains the following variables: x_t stands for the production quantity in period t, y_t is the setup variable, and I_t represents the inventory level in period t. Therefore, the objective function (1) tries to minimize the overall production(c_t^x), inventory(h_t), and setup (s_t) costs. The first constraints (2a) make sure that the external demand E_t is satisfied by either production in period t or by inventory from previous periods. Moreover, the constraints determine how much inventory is stored for future demands. The second constraints (2b) state that whenever production occurs a setup has to be made. Finally, the last constraints (2c) are the usual non-negativity and binary constraints.

Concerning the multi-item uncapacitated lot-sizing model, Wolsey (1989) for example analyzed the problem with start-up costs, while Pochet and Wolsey (1987) examined the problem with backlogging. Other authors for example developed simple heuristics to minimize the average setup cost and inventory cost over several periods (see Silver and Meal, 1973). Zangwill (1969) showed that the ULS is in effect a fixed charge network problem.

2.2 Capacitated Lot-Sizing Problem

A classical extension to the basic formulation is the multi-item capacitated lot-sizing problem (CLSP) (see Figure 1 in Section 2.4) where several items can be produced on one machine within one period over a given planning horizon. Production is therefore limited by the capacity constraint. The CLSP is NP-hard (see Bitran and Yanasse, 1982, for a proof). Trigeiro et al. (1989) extended the CLSP by adding setup times to the model.

2.3 Multi-Level Problems

The multi-level lot-sizing (MLLS) model deals with production processes that use various subassemblies and components to build a certain end item. Therefore, two kind of demands have to be considered in the inventory constraint: the primal (external) demand from the market place, and the secondary (internal) demand which is triggered when the production process starts to lot-size the ordered end item. Zangwill (1966) started with an uncapacitated multi-facility problem while Lambrecht and Vander Eecken (1978) extended the approach by adding capacity constraints at the last level. Further research for example was made by McClain and Thomas (1989), Tempelmeier and Helber (1994) and Harrison and Lewis (1996). Authors like e.g. Stadtler (2003) and Tempelmeier and Derstroff (1996) added set up times to the problem.

2.4 Small- and Big-Bucket Models

Another distinction in the literature is between small- and big-bucket models. Big-bucket models have the assumption that several products can be produced on the same machine in one period, while small-bucket models only allow a setup for one product on the same machine. However, in a small-bucket model it is possible to carry over a setup state for a certain item from one period to the next. Fleischmann (1990) proposed the discrete lot-sizing and scheduling problem (DLSP) where the linking of a setup state for one item is only possible if production uses the full capacity in the next period. In contrast, Karmarkar and Schrage (1985) and Salomon (1986) analyzed the continuous setup lot-sizing problem (CSLP) where production has not to use up the full capacity. The following LP model represents the CSLP formulation:

$$\min \sum_{i=1}^{P} \sum_{t=1}^{T} (s_i z_{it} + h_i I_{it} + c_i^x x_{it}), \tag{3}$$

subject to

$$I_{it} = I_{it-1} + x_{it} - E_{it}, \quad \forall i, t, \tag{4a}$$

$$\sum_{i=1}^{P} y_{it} \leq 1 \quad \forall t, \tag{4b}$$

$$a_i x_{it} \leq L_t y_{it} \quad \forall i, t, \tag{4c}$$

$$z_{it} \geq y_{it} - y_{it-1} \quad \forall i, t, \tag{4d}$$

$$I_{it} \geq 0, \quad x_{it} \geq 0, \quad y_{it}, z_{it} \in \{0, 1\}, \quad \forall i, t. \tag{4e}$$

The objective function (3) includes a new variable called start up variable z_{it}. Every time a machine is set up for which it was not set up in the previous period start up costs s_i occur. There is no change to the inventory constraints (4a). Constraints (4b) limit the setup per item and period to one. The next constraints (4c) restrain the production quantity by the available capacity L_t if a setup is made in that period. Constraints (4d) state that an item can only start up if the setup for that item in the current period is not equal to the setup in the previous period. The last constraints (4e) are again the usual binary and non-negativity constraints.

Note that the only difference between the DLSP and the CSLP is that in the DLSP constraints (3c) are formulated as an equality. Furthermore, the proportional lot sizing and scheduling problem (PLSP, see Figure 1) allows two items per period to use the same capacity, whereas there is no restriction concerning the consumption of the capacity (cf. Drexl and Haase, 1995.

Although the small-bucket model represents a more realistic scenario and allows for more accurate planning, it is certainly undesirable to divide the planning horizon into a huge number of small periods since it increases the complexity for the solution approach. To avoid the mentioned weakness of the small-bucket model and the possibly unrealistic simplifications of the big-bucket model, new model formulations are presented in the literature to combine both models. Fleischmann and Meyr (1997) introduced the general lot-sizing and scheduling problem (GLSP) where large time periods can be divided into several smaller time buckets of variable length, and the production in these periods is restricted to a single item. The model formulation we use in this thesis is the capacitated lot-sizing problem with linked lot sizes (CLSPL)(see e.g. Gopalakrishnan et al., 1995, 2001; Haase, 1994; Sox and Gao, 1999; Stadtler and Suerie, 2003) which is a big-bucket model that allows to carry-over setup states (see Figure 1).

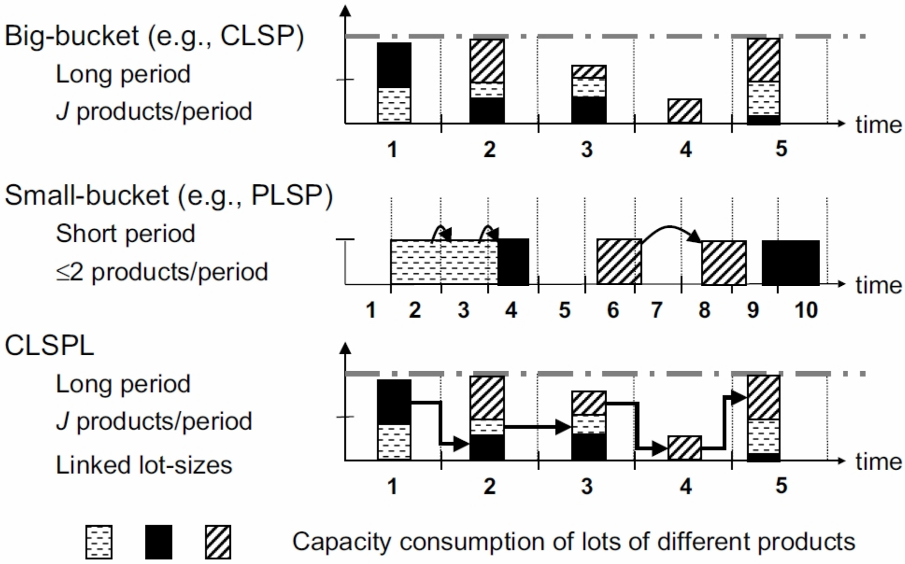

Figure 1: Three different formulations of the lot-sizing problem with linked lot sizes. The Figure is taken from Stadtler and Suerie (2003).

2.5 Solution Approaches

The lot-sizing problem is well known for being hard to solve, since even the single-item capacitated problem is NP-hard (see Florian et al., 1980, for a proof). For that reason a lot of research has been published on how to solve the problem efficiently in an alternative way. Since some formulations for the (mixed) integer programming problem yield to tighter bounds, various authors proposed strong valid inequalities and/or different model formulations. Tempelmeier and Helber (1994) analyzed a network or shortest path formulation, while Stadtler (1996) proposed a simple plant location formulation. Other contributions include heuristic algorithms with or without decomposition (e.g. Almada-Lobo et al., 2007; Stadtler and Suerie, 2003; Tempelmeier and Derstroff, 1996).

More recently, the use of metaheuristic became a well-established way of solving the underlying lot-sizing problem, such as the genetic algorithm (GA), simulated annealing (SA), tabu search (TS), and ant colony optimization (ACO). Xie and Dong (2002) used a GA, which belongs to the evolutionary algorithms and is based on the ideas of natural selection and genetics, to solve the CLSP. Furthermore, Dellaert and Jeunet (2000) solved the uncapacitated MLLS problem with a GA. Berretta and Rodrigues (2004) proposed a memetic algorithm, which is a less constrainted method of the GA, to solve multi-level capacitated lot-sizing problems. Their reported results for the small-sized instances could improve the solutions obtained by Tempelmeier and Derstroff (1996). Oezdamar and

Barbarosoglu (2000) proposed a Lagrangean relaxation-simulated annealing approach for the multi-level capacitated lot-sizing problem. SA is a metaheuristic which comes from annealing in metallurgy, and it is based on the heating and cooling of some material. The controlled slow cooling of the material allows the molecules to have enough time to restructure and build stabilized crystals with lower internal energy. Oezdamar and Barbarosoglu (2000) could improve the results for the small-sized instances obtained by Tempelmeier and Derstroff (1996) but not reach the results from Berretta and Rodrigues (2004). TS is a technique which uses memory structures to set potential solution 'taboo' so that this solution can not be visited again. Kimms (1996) for example used the TS to solve multi-level lot-sizing and scheduling problems. The ACO algorithm is based on the behavior of ants, which when searching for nourishment, walk randomly until they find some food, leaving pheromone trails behind. Authors like Pitakaso et al. (2006) and Almeder (2007) proposed ant-based algorithms to solve multi-item multi-level capacitated lot-sizing problems. Their approaches were tested with the instances provided by Tempelmeier and Derstroff (1996). Almeder (2007) delivers by far the best results for the medium-ranged instances, while the approach of Pitakaso et al. (2006) is superior to all the other approaches for the large-sized test instances. The time-oriented decomposition heuristic from Stadtler (2003) also provides very good results for the large-sized instances.

3 Mathematical Formulation

3.1 Standard Formulation

This Section provides a mathematical formulation for the MLCLS problem which originates from Stadtler (1996) and was used by Pitakaso et al. (2006). The indices, parameters and decision variables as well as the model itself are taken from Pitakaso et al. (2006).

Dimensions and indices:

P	number of products in the bill of material
T	planning horizon
M	number of resources
i	item index in the bill of material
t	period index
m	resource index

Parameters:

$\Gamma(i)$	set of immediate successors of item i
$\Gamma^{-1}(i)$	set of immediate predecessors of item i
s_i	setup cost for item i
c_{ij}	quantity of item i required to produce unit of item j
h_i	holding cost for item i
a_{mi}	capacity needed on resource m for one unit of item i
b_{mi}	setup time for item i on resource m
L_{mt}	available capacity for resource m in period t
c^o_m	overtime cost of resource m
G	sufficiently large number
h_i	holding cost for item i
E_{it}	external demand for product i in period t
I_{i0}	initial inventory of item i

Decision variables:

x_{it}	delivered quantity of item i at the beginning of period t
I_{it}	inventory level of item i at the end of period t
O_{mt}	overtime hours used on resource m in period t
y_{it}	binary variable indicating whether item i is produced in period t ($y_{it} = 1$) or not ($y_{it} = 0$)

$$\min \sum_{i=1}^{P} \sum_{t=1}^{T} (s_i y_{it} + h_i I_{it}) + \sum_{t=1}^{T} \sum_{m=1}^{M} c_m^o O_{mt}, \tag{5}$$

subject to

$$I_{it} = I_{it-1} + x_{it} - \sum_{j \in \Gamma(i)} c_{ij} x_{jt} - E_{it}, \quad \forall i, t, \tag{6a}$$

$$\sum_{i=1}^{P} (a_{mi} x_{it} + b_{mi} y_{it}) \leq L_{mt} + O_{mt}, \quad \forall m, t \tag{6b}$$

$$x_{it} - G y_{it} \leq 0, \quad \forall i, t, \tag{6c}$$

$$I_{it} \geq 0, \quad O_{mt} \geq 0, \quad x_{it} \geq 0, \quad y_{it} \in \{0, 1\}, \quad \forall i, t. \tag{6d}$$

The objective function (5) intends to minimize the total setup costs, holding costs and overtime costs. So whenever the available capacity is not sufficient, overtime may be used to meet the dynamic demand. The first equation (6a) in the model is the inventory balance equation, which assures that the inventory level of item i in period t is equal to the sum of the inventory level of the previous period, the amount produced in period t minus the internal demand needed to produce item i and the external demand. Constraints (6b) ensure that the available capacity and the overtime hours used are not exceeded by the capacity used for production and setup. Whereas constraints (6c) state that production in any period t for a certain item i is only possible if a setup is made in that period, with G representing the sum of the remaining demand. Due to performance reasons it is recommended to use a small value of G. The last constraints (6d) are the common non-negativity and binary constraints.

3.2 Simple Plant Location Formulation

The SPL formulation has been used by several authors to solve various lot-sizing problems. It was first introduced by Krarup and Bilde (1977) and then e.g. used by Rosling (1986) for assembly product structures and then considered by Maes and McClain (1991) for serial product structures. Here, we utilize the formulation proposed by Stadtler (1996) and Stadtler and Suerie (2003).

To properly implement the SPL formulation a few changes have to be considered in the LP model. First, the production variables x_{it} are exchanged by z_{ist} respective to

$$x_{it} := \sum_{s=t}^{T} D^n_{is} z_{its}, \quad \forall i, t. \tag{7}$$

The three-index production variable z_{its} can be seen as the portion of demand of item i produced for period s in period t. So basically a certain item i can only be produced if a 'plant location' has been made in period t for the present period or for any following period s. D^n_{is} represents the net demand of product i in period t and it is calculated according to Stadtler (1996):

$$\begin{aligned} &1, \ldots, P\colon \quad \delta = I_{i0} \\ &\left[1, \ldots, T\colon \quad D^n_{it} = \max\left\{ 0, E_{it} + \sum_{j=1}^{i-1} c_{ij} D^n_{jt} - \delta \right\}, \right. \\ &\left. \qquad\qquad\qquad \delta = \max\left\{ 0, \delta - E_{it} - \sum_{j=1}^{i-1} c_{ij} D^n_{jt} \right\}. \right] \end{aligned} \tag{8}$$

So D^n_{it} is either zero or the sum of the external and internal demand of item i in period t minus δ, which represents the remaining inventory at the beginning of period t.

The revised mixed-integer model formulation is identical to the model from Stadtler and Suerie (2003), except for the fact that the model below allows for overtime.

$$\min \sum_{i=1}^{P} \sum_{s=1}^{T-1} \sum_{t=s}^{T} h_i (t-s) z_{ist} D^n_{it} + \sum_{i=1}^{P} \sum_{t=1}^{T} s_i y_{it} + \sum_{t=1}^{T} \sum_{m=1}^{M} c^o_m O_{mt} \tag{9}$$

subject to

$$I_{it} = I_{it-1} + \sum_{s=t}^{T} z_{its} D_{is}^{n} - \sum_{j \in \Gamma(i)} \sum_{s=t}^{T} c_{ij} z_{jts} D_{js}^{n} - E_{it}, \quad \forall i, t, \tag{10a}$$

$$\sum_{i=1}^{P} \sum_{s=t}^{T} a_{mi} z_{its} D_{is}^{n} + \sum_{i=1}^{P} b_{mi} y_{it} \leq L_{mt} + O_{mt}, \quad \forall m, t, \tag{10b}$$

$$z_{its} \leq y_{it}, \quad \forall i, t, s = t, \ldots, T, \tag{10c}$$

$$\sum_{s=1}^{t} z_{ist} = 1, \quad \forall i, t, D_{it}^{n} > 0, \tag{10d}$$

$$I_{it} \geq 0, \quad O_{mt} \geq 0, \quad y_{it} \in \{0, 1\}, \quad z_{ist} \geq 0, \quad \forall i, t, s = t, \ldots, T. \tag{10e}$$

The first change applies to the objective function (9), which now calculates the inventory costs by multiplying the holding costs of product i and the portion of demand of item i produced in t for s with the time difference between the period indices t and s. For constraints (10c), it is now possible to omit the parameter G since production variables z_{ist} will never take value above one. A new equation (10d) is needed to ensure that the required demand in each period is satisfied. As before, constraints (10e) are the typical non-negativity and binary constraints which in this case also assure that the variables z_{ist} never take values below zero. Concerning the other constraints, the variables x_{it} are replaced by z_{ist} according to (7).

4 Decomposition

4.1 Standard Formulation

Since obtaining an optimal solution for a MLCLS problem with real world instances is rather time consuming, the problem is divided into various subproblems, which are then solved exactly. In the absence of a realistic scenario this can be done by constructing subproblems for either items or periods. But with sizes ranging between 40-100 items in the bill of materials and 20-40 periods, it can be easily seen that the problem size would still be too big to be solved within a reasonable time. To avoid this flaw, the underlying approach combines both variants and splits the bill of material as well as the number of periods into several subproblems.

Prior to the decomposition, the items in the bill of material are sorted so that the successors of a certain item are always positioned somewhere before that item in the list. So if we number the items in the list from 1 to P and item i is a direct successor of item j, then $i < j$. This lot-sizing sequence allows us to schedule the products one after another, which can lead to numerous variations in the presence of a general system. Furthermore, it is advisable to introduce an overlap region for items and periods in a subproblem to consider the interdependencies between periods and items in different subproblems. Thus, only a certain number of items and periods are fixed after the computation of one subproblem.

To better explain the implications of the decomposition approach consider this simple example with five items and eight periods (see Figure 2). Every subproblem consists of three items and five periods with an overlap of two periods and one item. The decomposition starts with SP1 (subproblem 1), but after the calculation only region I (periods 1-3 and items 1-2) is fixed. Then, SP2 (periods 3-8 and items 1-3) is solved and region II is recalculated and finally fixed. This concludes the first level of the decomposition. In the second and last level, SP3 (periods 1-5 and items 3-5) region III and IV are solved but only III is fixed afterwards, and so on.

If we now consider a single subproblem it is crucial to assign correct capacity limits for every subproblem. Hence, it is not possible to use the general formulation from Section 2. By taking another look at Figure 2, it can be seen that after solving SP1 it is not certain if the available capacity is sufficient for SP3. Since the the model allows for infinite overtime, there are no infeasible solutions. But because overtime costs are very high, the solution will be poor, and therefore it is necessary to include the capacity consumption of previous subproblems. Another problem might occur if the demand

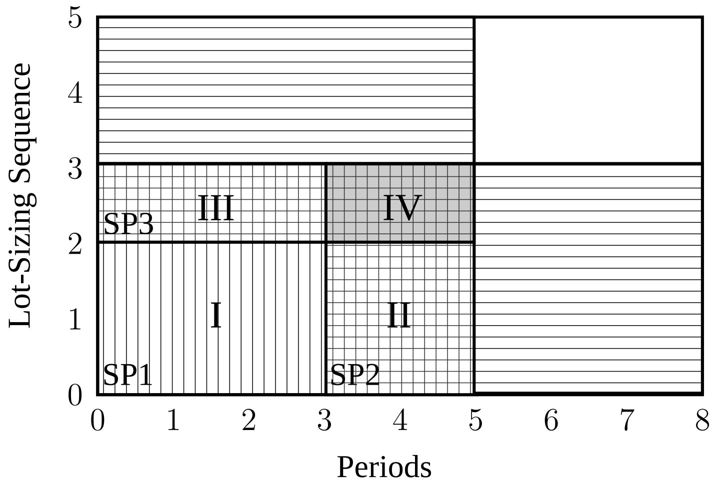

Figure 2: Every subproblem consists of three items and five periods with an overlap of two periods and one item. So after e.g. solving SP1 only region I (periods 1-3 and items 1-2) is fixed.

in one subproblem exceeds the available capacity in that time interval while it would be sufficient in previous periods. As a result, the capacity consumption for setup and production of an item have to be modified to take the capacity needs of its predecessors into account.

This adaption of the problem will lead to additional variables and parameters which are described below. The notation and the model with its description are again taken from Pitakaso et al. (2006).

k	index of the subproblem
T_s^k	starting time period of subproblem k
T_e^k	last time period of subproblem k
P_s^k	number of first item of subproblem k
P_e^k	number of last item of subproblem k
A_{mi}^k	modified capacity needed for production of one unit of item i on resource m
B_{mit}^k	modified capacity needed for setup of production of item i in period t on resource m
S_{it}^k	modified setup cost for item i in period t of subproblem k
X_{it}	lot size for product i in period t (already determined in previous subproblems)
Z_{ist}	lot size for product i in period t produced in period s (already determined in previous subproblems)
Y_{it}	binary variable indicating whether item i is scheduled to be produced in period t

(already determined in previous subproblems)

avC_{mt}^k available regular capacity of resource m in period t for subproblem k

The mixed-integer problem for subproblem k is then

$$\min \sum_{i=P_s^k}^{P_e^k} \sum_{t=T_s^k}^{T_e^k} (S_{it}^k y_{it} + h_i I_{it}) + \sum_{t=T_s^k}^{T_e^k} \sum_{m=1}^{M} c_m^o O_{mt}, \tag{11}$$

subject to (each constraint must hold for all $i = P_s^k, \ldots, P_e^k$, $t = T_s^k, \ldots, T_e^k$, and $m = 1, \ldots, M$)

$$I_{it} = I_{it-1} + x_{it} - \sum_{\substack{j \in \Gamma(i) \\ j < P_s^k}} c_{ij} X_{jt} - \sum_{\substack{j \in \Gamma(i) \\ j \geq P_s^k}} c_{ij} x_{jt} - E_{it}, \tag{12a}$$

$$\sum_{i=P_s^k}^{P_e^k} (a_{mi} x_{it} + b_{mi} y_{it}) \leq L_{mt} + O_{mt} - \sum_{i=1}^{P_s^k - 1} (a_{mi} X_{it} + b_{mi} Y_{it}), \tag{12b}$$

$$\sum_{\tau=T_s^k}^{t} \sum_{i=P_s^k}^{P_e^k} (A_{mi}^k x_{i\tau} + B_{mi\tau}^k y_{i\tau}) \leq \sum_{\tau=T_s^k}^{t} (avC_{m\tau}^k + O_{m\tau}), \tag{12c}$$

$$x_{it} - G y_{it} \leq 0, \tag{12d}$$

$$I_{it} \geq 0, \quad x_{it} \geq 0, \quad O_{mt} \geq 0, \quad y_{it} \in \{0, 1\}. \tag{12e}$$

Various authors have proposed numerous methods to deal with setup costs when solving a multi-level lot-sizing problem by a series of single level lot-sizing problems (e.g. Dellaert and Jeunet, 2003; McLaren, 1977). The method used here is a randomized cumulative Wagner-Whitin (RCWW) method from Pitakaso et al. (2007) which is an extension of Dellaert and Jeunet (2003). The difference between these methods is that Pitakaso et al. (2007) use sequence-dependent time-varying setup costs (STVS), so the modified setup costs for every product depend on the actual position in the production sequence.

The reason for using modified setup costs in the objective function (11) is due to the fact that lot-sizing an item in a current period results in additional lot-sizes for some predecessors in previous periods. So the setup costs of an item are adapted by a fraction of the setup costs of all its predecessors. This leads to two cases: (i) the predecessor of an item already has a positive demand, which results in no additional costs; (ii) there is no positive demand for a predecessor and therefore we have to add the modified setup costs. To calculate the modified setup cost we introduce a new variable

T_{ijt} is a binary variable which equals to 1 if a lot size of item i in period t leads to a positive demand for predecessor j ($j \in \Gamma^{-1}(i)$) in period t; T_{ijt} equals to 0 if there is already a planned lot-size for item j in period t (resulting from a different successor of item j).

and calculate the modified setup costs as following:

$$S_{it}^{k} = s_i + r_i \sum_{\substack{j \in \Gamma^{-1}(i) \\ j > P_e^k}} T(i,j,t) \frac{S_j}{|\Gamma(j)|}, \tag{13a}$$

where S_j is calculated recursively by

$$S_i = s_i + \sum_{j \in \Gamma^{-1}(i)} \frac{S_j}{|\Gamma(j)|}, \tag{13b}$$

and r_i by

$$r_i = R \cdot \left(1 + \frac{P - 2\Phi_i + 1}{P - 1} \cdot u\right), \tag{13c}$$

Hence, the value of variable T_{ijt} depends on how the products are scheduled in the lot-sizing sequence. To avoid adding the setup costs for a single item multiple times, the modified setup costs are divided by the immediate successors in the present subproblem. This is due to the fact that not every new lot for an item also creates a positive demand for a certain predecessor if some item with the same predecessor has already been scheduled. The random variable r_i decides how much the setup costs of the predecessors influence the modified setup costs. The parameters $R \in \{0, 0.5\}$ and $u \in \{-1, 1\}$ are uniformly distributed random variables, whereas $\Phi_i \in \{1, \ldots, P\}$ is the position of item i in the lot-sizing sequence. Thus, for the first item (= end item) scheduled in the lot-sizing sequence we obtain $r_i = R(1 + u)$, whereas for the last item we obtain $r_i = R(1 - u)$. Therefore R determines the average value of r over all items and u is the slope.

The inventory balance equation (12a) is slightly modified to consider the production quantity fixed in some previous subproblems. To ensure that the available capacity is sufficient in the current subproblem the right-hand side of capacity constraint (12b) is now reduced by the amount of resources already used in previous subproblems.

A cumulative capacity constraint (12c) is introduced to the model which should guarantee that the global solution will only use overtime if it is inevitable. The summation term on the left-hand side contains the accumulated capacity needs for every item in the subproblem. The idea behind the modified capacity needs A^k_{mi} and B^k_{mit} is that if some item is scheduled, we also have to consider the resources needed for its predecessors in some previous period. They are calculated recursively as follows:

$$A^k_{mi} = a_{mi} + \sum_{\substack{j \in \Gamma^{-1}(i) \\ j > P^k_e}} A^k_{mj}, \tag{14}$$

$$B^k_{mit} = b_{mi} + \sum_{\substack{j \in \Gamma^{-1}(i) \\ j > P^k_e}} T(i,j,t) \frac{\tilde{B}_{mj}}{|\Gamma(j) \cap \Delta(i)|}, \tag{15a}$$

where $\tilde{B}_{mj}$ is the accumulated capacity needed for setup of product i on resource m

$$\tilde{B}_{mi} = b_{mi} + \sum_{j \in \Gamma^{-1}(i)} \tilde{B}_{mj}. \tag{15b}$$

Note that the concept of equation (15a) is quite similar to the calculation of the time-varying modified setup costs in (13a). The accumulated capacity needed for setup of product i on resource m is divided by the number of immediate successors which are located in the same group as item i. There are three groups: (i) already lot-sized items; (ii) items in the present subproblem k; and (iii) items which are not yet scheduled.

$$\Delta(i) = \begin{cases} \{i, \ldots, P^k_s - 1\}, & \text{if } i < P^k_s, \\ \{P^k_s, \ldots, P^k_e\}, & \text{if } P^k_s \leq i \leq P^k_e, \\ \{P^k_e + 1, \ldots, P\}, & \text{if } i > P^k_e. \end{cases} \tag{15c}$$

The available capacity avC^k_{mt} is calculated by subtracting the already used resources and the yet to schedule resource consumption from the total available capacity L_{mt}.

$$avC^k_{mt} = L_{mt} - \sum_{i=1}^{P^k_s - 1} (A^k_{mi} X_{it} + B^k_{mit} Y_{it}) - \sum_{\substack{i=P^k_e+1 \\ \Gamma(i)=\emptyset}}^{P} (A^k_{mi} E_{it} + B^k_{mit} Y^E_{it}), \tag{16a}$$

where

$$Y^E_{it} = \begin{cases} 1, & E_{it} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{16b}$$

Due to the fact that a single subproblem does not necessarily contain all available periods, a demand backward shifting from Pitakaso et al. (2006) is introduced to balance the demand in different subproblems. See also Berretta and Rodrigues (2004), Franca et al. (1994), Trigeiro et al. (1989), and Xie and Dong (2002) for various methods for production backward shifting. The demand shifting used here only operates between subproblems of the same level.

/* **Decomposition** */
Choose the number of items I_s and periods T_s included in the subproblems
Set the present level of subproblems p to 1
while p is not the last level **do**
    Perform the capacity modifications (13a)-(16b) for every subproblem
    Start the demand shifting procedure and adjust the demands for each subproblem
    for each subproblem in level p **do**
        Calculate the subproblem with the one containing the first period
        Fix solution in the non-overlapping region
        Utilize the solution (inventory levels) to calculate the next subproblem
    end
    Fix solution for each non-overlapping item
    Update new demand for the next level $p+1$
    $p = p + 1$
end

Figure 3: Pseudo-code of the decomposition.

The procedure always starts with the last subproblem of the present level containing the last period. In order to quantify the demand for every subproblem, capacity constraint (12c) is modified so that the external demands E_{it} and the internal demands

$\sum_{j\in\Gamma(i)} c_{ij}X_{jt}$ replace the production quantities and setups Y_{it}. Whenever there is a external or internal demand for item i in period t the demand shifting assumes a positive value for Y_{it}. Starting with the first period in the subproblem, the procedure shifts any excess demand to the closest period outside the current subproblem. The demand shifting then continues with the previous subproblem and stops when it reaches the first subproblem. The pseudo-code in Figure 3 summarizes the decomposition approach.

4.2 Simple Plant Location Formulation

As for the basic model in Section 3.2, the variables x_{it} are replaced by z_{ist} according to (7). Furthermore, the calculation of the inventory costs in the objective function is replaced by the standard calculation taken from the model in Section 3.1.

$$\min \sum_{i=P_s^k}^{P_e^k} \sum_{t=T_s^k}^{T_e^k} (S_{it}^k y_{it} + h_i I_{it}) + \sum_{t=T_s^k}^{T_e^k} \sum_{m=1}^{M} c_m^o O_{mt}, \tag{17}$$

subject to (each constraint must hold for all $i = P_s^k, \ldots, P_e^k$, $t = T_s^k, \ldots, T_e^k$, and $m = 1, \ldots, M$)

$$I_{it} = I_{it-1} + \sum_{s=t}^{T_e^k} z_{its} D_{is}^n - \sum_{\substack{j\in\Gamma(i)\\ j<P_s^k}} \sum_{s=t}^{T_e^k} c_{ij} Z_{jts} D_{js}^n - \sum_{\substack{j\in\Gamma(i)\\ j\geq P_s^k}} \sum_{s=t}^{T_e^k} c_{ij} z_{jts} D_{js}^n - E_{it}, \tag{18a}$$

$$\sum_{i=P_s^k}^{P_e^k} \sum_{s=t}^{T_e^k} (a_{mi} z_{its} D_{is}^n + b_{mi} y_{it}) \leq L_{mt} + O_{mt} - \sum_{i=1}^{P_s^k-1} \sum_{s=t}^{T_e^k} (a_{mi} Z_{its} D_{is}^n + b_{mi} Y_{it}), \tag{18b}$$

$$\sum_{\tau=T_s^k}^{t} \sum_{i=P_s^k}^{P_e^k} \sum_{s=\tau}^{T_e^k} (A_{mi}^k z_{i\tau s} D_{is}^n + B_{mi\tau}^k y_{i\tau}) \leq \sum_{\tau=T_s^k}^{t} (avC_{m\tau}^k + O_{m\tau}), \tag{18c}$$

$$\sum_{s=T_s^k}^{t} z_{ist} = 1, \qquad \text{if} \quad D_{it}^n \geq 0, \tag{18d}$$

$$z_{ist} \leq y_{it} \qquad \forall s = t, \ldots, T_e^k, \tag{18e}$$

$$I_{it} \geq 0, \quad O_{mt} \geq 0, \quad y_{it} \in \{0,1\}, \quad z_{ist} \geq 0 \qquad \forall s = 1, \ldots, T_e^k. \tag{18f}$$

The next steps of the decomposition are equal to those of the standard formulation in the previous Subsection. In addition, tests have shown that the standard formulation yields to better results than the SPL formulation (see Section 6 for details).

5 Ant Colony Algorithm

5.1 General Description

The Ant Colony Optimization (ACO) algorithm was introduced by Dorigo (1992) in his PhD thesis to solve discrete optimization problems. It is a probabilistic technique that was originally applied to the traveling salesman problem and the quadratic assignment problem. The idea is based on the behavior of ants, which when searching for nourishment, walk randomly until they find some food. Then, on the way back to their colony, the ants leave trails of pheromone behind. If the food is far away from the colony, the pheromone trail evaporates quickly, while a shorter path has a higher pheromone concentration since more ants follow this way. The concept of evaporation prevents the algorithm to converge towards a locally optimal solution. In other words, a lack of evaporation would attach too much importance to the first ants and bias the next generation of ants, therefore limiting the search space. Following this real life concept, the ACO algorithm creates a population of artificial ants which generate and improve a solution to a certain instance of a combinatorial optimization problem. For the next generation of ants a global memory is updated. After the initialization of the pheromone information the framework of the ACO algorithm can be typically summarized in the following three steps:

- Step 1: Ants construct solutions according to pheromone and heuristic information.
- Step 2: Application of local search methods to the solution of the ants.
- Step 3: Update of the pheromone information.

A detailed explanation of the ACO algorithm is now given with the example of the traveling salesman problem (TSP). Given a number of cities (nodes) and the associated costs of traveling from one city to another, the goal of the TSP is to minimize the total costs under the assumption of visiting every city once and of returning to the starting city. The TSP can therefore be represented as a complete graph. In the ACO algorithm the desirability of visiting city j after city i in iteration m is given by the pheromone information $\tau_{ij}(m)$. This information is used in the construction phase (Step 1) and updated in Step 3. The algorithm starts with randomly placing a number of artificial ants on cities. In every construction step the selection of the next feasible city is biased towards a probabilistic decision. This decision includes the pheromone information $\tau_{ij}(m)$ and the visibility in the ant system framework η_{ij} (or heuristic information). The inverse arc length of visiting city j after city i is a prudent choice for the visibility. So the ant will therefore favor an arc which has a high pheromone value and where city j is close to city i. The probability of visiting city j after city i can be mathematically formulated by:

$$p_{ij}^k(m) = \begin{cases} \frac{\tau_{ij}(m)\eta_{ij}}{\sum_{l\in N_i}\tau_{il}(m)\eta_{il}}, & \text{if } j \in N_i^k, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

The set N_i^k includes the feasible cities that can be visited by ant k and has not yet been visited. After the solution has been created a local search is applied to verify the local optimality.

In the update phase (Step 3) the before mentioned evaporation decreases the pheromone value by the constant factor ρ, and a number of ants with the best solution quality update the pheromone information. The MAX-MIN Ant System (MMAS) by Stützle and Hoos (1997) only allows the global best solution to update the pheromone information. The pheromone update rule is as follows:

$$\tau_{ij}(m+1) = \rho\tau_{ij}(m) + \Delta\tau_{ij}^* \tag{20}$$

Note that $\Delta\tau_{ij}^* = 1/f(s^*)$, where $f(s^*)$ represents the cost value, if city j is visited after i for the best ant, and 0 otherwise. The MMAS bounds the pheromone value by the maximum and minimum limits $[\tau_{\min}, \tau_{\max}]$ to avoid extreme differences in the pheromone amounts. For a convergence proof of the ACO algorithm see Stützle and Dorigo (2002) and Gutjahr (2003).

5.2 MAX-MIN Ant System for the MLCLS problem

In order to determine and subsequently improve the lot-sizing sequence described in the previous Section, a MMAS algorithm is applied.

The algorithm of Pitakaso et al. (2006) uses the ideas of Evolutionary Algorithms to find appropriate values for R and u which are used to calculate the modified setup costs in formula (9a). This concept is now used for the number of items I_s and number of periods P_s included in one subproblem (see Pitakaso et al., 2007). The visibility in the ant system framework η_j (or heuristic information) is based on the original setup costs s_j (see equation (21)). As Pitakaso et al. (2006) state in their paper, they tested various values for the heuristic information (combination of holding costs and setup costs or no heuristic information), but the use of the original setup costs turned out to deliver the best results. Normally, a local search would be applied to the solutions of the ants, but since the decomposition takes a lot of time, it is not included in the algorithm. The adapted MMAS to solve the MLCLS (called ASMLCLS) is illustrated by the pseudo-code (taken from Pitakaso et al., 2006) in Figure 2.

Procedure ASMLCLS
/* **Initialization Phase** */
Generate initial R_b, u_b, T_s^b, and I_s^b
(select best solution out of 20 randomly constructed ones)
Initialize pheromone information
while (termination condition not met) **do**
    for each ant **do**
        /* **Construction Phase (Step 1)** */
        Construct the production sequence according to decision rule (21)
        /* **Adaptation of** R **and** u **values for each ant** */
        Choose R randomly out of the set $\{R_b(1-\vartheta), R_b, R_b(1+\vartheta)\}$
        Choose u randomly from $\{\max\{-1, u_b(1-\vartheta)\}, u_b, \min\{1, u_b(1+\vartheta)\}\}$
        Calculate r_i according to (9c)
        Choose I_s randomly out of the set $\{I_s^b - 1, I_s^b, I_s^b + 1\}$
        Choose T_s randomly out of the set $\{T_s^b - 1, T_s^b, T_s^b + 1\}$
        (within the boundaries of Table 1)
        Perform the decomposition method from Section 3 to evaluate the sequence
    end
    /* **Pheromone update phase** */ **(Step 2)**
    Update the pheromone matrix according to (22a), update R_b, u_b, T_s^b, I_s^b
end

Figure 4: Pseudo-code of ASMLCS taken from Pitakaso et al. (2006).

The algorithm starts by randomly constructing twenty solutions and thereafter saves the values of R_b, u_b, T_s^b, and I_s^b from the best solution. Then, the pheromone value is initialized with the maximum pheromone value (see details below). In the next phase, every ant generates a product sequence on which the decomposition is applied afterwards.

The pheromone encoding scheme is taken from Pitakaso et al. (2006) which originates from Stützle (1998). In this scheme the intensity of pheromone trail ($= \tau_{pj}(\ell)$) represents the desirability of lot sizing item j on the p-th position. So the desirability of choosing item j as the p-th item depends on how preferable it was in the previous iteration. The probability that ant k selects product j on position p in iteration ℓ is calculated according to the decision rule (21). The descriptions about the indices, parameters and formulas in this Section are again taken from Pitakaso et al. (2006).

$p^k_{pj}(\ell)$	probability that ant k selects product j on position p in iteration ℓ
$\tau_{pj}(\ell)$	intensity of pheromone trail of product j in position p at iteration ℓ
α	parameter to regulate the influence of $\tau_{pj}(\ell)$
β	parameter to regulate the influence of s_j
N^k_p	set of selectable products in position p of ant k based on the bill of materials

$$p^k_{pj}(\ell) = \begin{cases} \dfrac{\left[\sum_{o=1}^{p} \tau_{oj}(\ell)\right]^\alpha [s_j]^\beta}{\sum_{l \in N^k_p} \left[\sum_{o=1}^{p} \tau_{ol}(\ell)\right]^\alpha [s_l]^\beta}, & \text{if } j \in N^k_p, \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

Note that the decision rule (21) not only considers the present pheromone value of lot sizing item j on the p-th position but also all the pheromone values for placing item j in all the predecessors positions of p. This so called summation decision rule was introduced by Merkle and Middendorf (1999).

$\rho \in [0, 1]$	trail persistence parameter to regulate the evaporation of τ_{pj}
$\Delta\tau_{pj}(\ell)$	total increase of trail level on edge (p, j) which is controlled by the maximum and minimum value along with the concept of MMAS
$f(s^{\text{opt}})$	global best solution value

$$\tau_{pj}(\ell + 1) = \max(\tau_{\min}, \min(\tau_{\max}, \rho\tau_{pj}(\ell) + \Delta\tau_{pj}(\ell))), \tag{22a}$$

$$\Delta\tau_{pj}(\ell) = \begin{cases} \dfrac{1}{f(s^{\text{opt}})}, & \text{if item } j \text{ is on position } p \text{ for the best ant,} \\ 0, & \text{otherwise.} \end{cases} \tag{22b}$$

Only the overall best ant updates the pheromone value (Step 2 in the pseudo code), but it is bounded by $[\tau_{\min} = 0.01, \tau_{\max} = 0.99]$. The evaporation rate ρ is set to 0.95.

As already stated before the values of R, u, T_s, and I_s are chosen by taking the ideas of Evolutionary Algorithms into account. After the initialization phase the corresponding values (R_b, u_b, T_s^b I_s^b) of the best objective are fixed. For every following iteration the best values from the initial phase or slightly changed ones (see pseduo-code) are taken. According to Pitakaso et al. (2007) this leads to better results than just taking the unperturbed values of the initialization phase. In addition, tests from Pitakaso et al. (2007) suggest a perturbation rate ϑ of 0.05.

6 Results for the MLCLS Problem

6.1 Computational Results

The ASMLCLS algorithm is implemented in C++ using CPLEX 11.0 to calculate the subproblems. All the instances were tested on a Pentium D 3.2GHz with 4 GB RAM and SUSE Linux 10.1.

The limits for the subproblem sizes are set so that a solution can be found within 2 seconds (see Table 1). Due to the improvement of computer speed these limits are higher than the bounds of Pitakaso et al. (2006). The overlapping for the items is set to 20% and the overlapping for periods is set to 60%. Tests from Pitakaso et al. (2006) have shown that this combination proved to be the best.

Table 1: The maximal subproblem size is set so that a solution can be found within 2 seconds. I_s denotes the number of items included in the subproblem while T_s represents the amount of period in that subproblem.

I_s	1	2	3	4	5	6	7	8	9	10	11	12
T_s	20	18	15	14	13	11	10	10	9	8	8	7

Two sets of test instances from Tempelmeier and Derstroff (1996) were taken to test the algorithm. The first group consists of 600 instances with 16 periods, 10 items and 4 resources. These instances are composed of assembly systems (A) and general systems (G). There is a distinction between cyclic cases (C) and non-cyclic cases (NC). Cyclic means that more than one resource is needed within the same production level. In addition, the demand patterns vary among the instances. All the test instances from the second group have a general system with 100 items, 16 periods, and 10 resources. Again, there are cyclic and non-cyclic cases, and five different capacity utilizations.

An open question is which mathematical formulation leads to better results for the ASMLCLS. For that purpose we randomly picked 200 instances from of the first group evenly distributed between G-C, G-NC, A-C, and A-NC. This version of the ASMLCLS does not include the demand shifting procedure. The results in Table 2 show that the standard formulation (X-Formulation) significantly outperforms the SPL formulation (Z-Formulation) for the ASMLCLS algorithm. A possible reason why the SPL formulation fails to work for the ASMLCLS could be the effects of the computational overhead.

Results for the first group (see Table 3) from Tempelmeier and Derstroff (1996) show that our ASMLCLS algorithm can beat the results from Tempelmeier and Derstroff (1996) and Pitakaso et al. (2006), but fails to reach the results from Almeder (2007). In

Table 2: 200 randomly chosen test instances from the first group of Tempelmeier and Derstroff (1996) to compare the standard formulation(X-Formulation) and the SPL representation (Z-Formulation). The number next to the name represents the run time (in minutes) of the algorithm.

	X-Formulation-10	Z-Formulation-10
Problem	Mean Cost	Mean Cost
G-NC	400 633	422 475
G-C	387 570	395 534
A-NC	48 370	49 871
A-C	274 923	339 972

fact, the hybrid approach of Almeder (2007) outperforms the other approaches by 86% (17.8% for the ALMLCS algorithm). Note that the lagrangean-based heuristic from Tempelmeier and Derstroff (1996) is very fast, but as they state in their paper, it was not possible to improve the solutions significantly if they had added more iterations to the heuristic. In addition, Tempelmeier and Derstroff (1996) used a computer that was 1000 times slower than the computer used here. The ASMLCLS algorithm and the approach of Almeder (2007) were tested on the same computer. Pitakaso et al. (2006) used a Pentium 4 2.4GHz personal computer with 1GB RAM and Microsoft Windows 2000 to test the instances. The rather big difference between our ASMLCLS algorithm and the one from Pitakaso et al. (2006) could be due to one of these reasons: (i) computer speed improvement, (ii) the increase of the maximal subproblem size, or (iii) a combination of both.

The results for the second group of instances (see Table 4) from Tempelmeier and Derstroff (1996) provide a different picture. Since not all the results are available in detail, only the basic results are provided in the Table 4. Our ASMLCLS algorithm yields to very poor results and is outperformed by every other approach. Again, the approach of Tempelmeier and Derstroff (1996) delivers fast results, but the solution quality is poor. In contrast, the algorithms from Pitakaso et al. (2006) and Stadtler (2003) are complex and time-consuming but the solution quality is superior to all the other approaches. The hybrid approach of Almeder (2007) is nearly as good as the approach of Stadtler (2003), but is unable to reach the results reported by Pitakaso et al. (2006). Due to the problems of our approach (described in detail in the next Subsection), it was also tested how the average of ten seeds changes the solution quality. The results are significantly better and can almost reach the results of Tempelmeier and

Table 3: Results for the first group of instances from Tempelmeier and Derstroff (1996). We compare our results (ASMLCLS) with the ones of Tempelmeier and Derstroff (1996), Pitakaso et al. (2006) and Almeder (2007). MAPD is the mean absolute percent deviation from the best solution and % represents the percentage number of best solutions found. The number next to the name represents the run time (in minutes) of the algorithm.

	T&D-0.02			Pitakaso-10			
Problem	MAPD	%	Cost	MAPD	%	Cost	Best Solution
G-NC	0.070	9.3	382 706	0.061	7.3	380 666	353 791
G-C	0.073	7.3	395 075	0.064	10.0	393 329	365 829
A-NC	0.064	6.0	47 134	0.041	10.7	46 081	44 066
A-C	0.057	4.0	45 700	0.036	11.3	44 650	43 052
Total	**0.066**	**6.7**	**217 654**	**0.051**	**9.8**	**216 181**	**201 684**
	Almeder-9			**ASMLCLS-10**			
Problem	MAPD	%	Cost	MAPD	%	Cost	Best Solution
G-NC	0.001	89.3	354 185	0.021	22.0	367 204	353 791
G-C	0.002	87.3	366 289	0.024	20.6	377 891	365 829
A-NC	0.004	87.3	44 234	0.045	12.6	46 705	44 066
A-C	0.003	82.7	43 201	0.020	16.0	44 021	43 052
Total	**0.003**	**86.7**	**201 977**	**0.027**	**17.8**	**208 955**	**201 684**

Derstroff (1996), which is still poor. The run time equals to 300 minutes.

6.2 Criticism

A main problem of the ASMLCLS algorithm is that the solution quality is very sensitive towards the starting solution. By randomly creating twenty solutions (see Section 5) it is not guaranteed that the initial best solution provides a good starting point for the MAX-MIN Ant System. An aggravating factor is the slowness of the whole approach (due to the time-consuming decomposition). For the large-scale instances our ASMLCLS algorithm could on average only reach ten iterations, which is very little for an ant-based algorithm. Therefore a bad starting solution makes it next to impossible for the ant algorithm to react and then to improve the solution considerably. Future improvements could involve a change of the decomposition by e.g. simplifying the process of modifying the setup costs to speed up the whole method. Furthermore, a revised method for searching a starting solution seems necessary to avert the above mentioned weakness.

Table 4: Results for the second group of instances from Tempelmeier and Derstroff (1996). We compare our results (ASML-CLS) with the ones of Tempelmeier and Derstroff (1996), Pitakaso et al. (2006), Almeder (2007) and Stadtler (2003). The number next to the name represents the run time (in minutes) of the algorithm.

Problem	T&D-2	Almeder-28	Pitakaso-30	Stadtler-20	ASMLCLS-30 (1 seed)	ASMLCLS-300 (10 seeds)
NC-1	364 186	355 335	353 401	354 709	367 356	360 011
NC-2	1 864 407	1 799 418	1 745 710	1 756 314	2 039 628	1 876 225
NC-3	4 035 239	3 823 610	3 731 525	3 750 211	4 515 288	4 144 182
NC-4	2 565 054	2 487 034	2 431 617	2 446 366	2 976 572	2 643 798
NC-5	1 838 860	1 676 321	1 642 021	1 648 295	1 790 900	1 736 391
C-1	388 112	360 063	359 441	359 723	399 776	368 518
C-2	2 056 183	1 954 444	1 892 534	1 920 455	2 141 281	2 086 952
C-3	4 550 056	4 300 901	4 211 022	4 265 387	5 334 872	4 784 399
C-4	2 830 900	2 683 988	2 573 459	2 643 123	3 237 338	3 022 704
C-5	1 914 168	1 794 764	1 736 597	1 778 768	2 003 604	1 920 407
Total	**2 240 717**	**2 123 588**	**2 067 733**	**2 092 335**	**2 480 662**	**2,294,359**

7 Solution Approach for the Linkage Property

7.1 Model Formulation

The formulation used in this thesis to solve the CLSPL was suggested by Stadtler and Suerie (2003), and it is based on the following assumptions:

- The fixed planning horizon T is divided into periods $(1 \dots T)$.
- The resource usage for any item i on a certain resource m and the assignment of items to resources is fixed.
- Setups are causing setup times and setup costs and therefore reducing the available capacity. Both are sequence independent.
- Only one setup state per resource can be linked from one period to the next.
- Single-item production is possible, which means that a setup state for an item can be preserved over two consecutive bucket boundaries.
- A setup state is preserved if there is no production in the following period.

Hence, the CLSPL is a big-bucket model with the characteristic of a small-bucket model to carry over setup states (see Section 2 for details). To implement the linkage property into our model we introduce two new variables:

w_{it}	is a binary variable (linkage variable) which equals to 1 if the setup state of item i is preserved from period $t-1$ to period t; 0 otherwise.
qq_{it}	is a product-dependent variable which equals to 1 if item i is only produced in period t and the setup state is linked to the preceding and the subsequent period, so $w_{it} = w_{it+1} = 1$; 0 otherwise.

The following constraints are added to the formulation described in Section 3:

$$x_{it} - G(y_{it} + w_{it}) \leq 0 \quad \forall i, t, \tag{23}$$

This alteration of the setup constraints is necessary since production is now possible by either producing item i in period t, or carrying over the setup state of item i from period $t-1$ to period t.

$$\sum_{i \in R_m} w_{it} \leq 1 \quad \forall m, t = 2, \ldots, T, \tag{24}$$

Constraints (24) guarantee that only one setup state is carried over on each resource. R_m is the set of item i produced on resource m. The next constraints (25) ensure that there can only be a setup for item i in period t ($y_{it} = 1$), a carry-over from period $t-1$ to period t ($w_{it} = 1$), single-item production for any item $k \neq j$ in period t ($qq_{kt} = 1$), or neither of them.

$$y_{it} + w_{it} + \sum_{\substack{j \in R_m \\ j \neq i}} qq_{jt} \leq 1 \quad \forall m, i \in R_m, t, \tag{25}$$

Linking the setup state for item i is only possible if either a setup activity is set in the previous period $t-1$, or the setup state is already preserved from period $t-2$ to $t-1$, which means single-item production of product i in period $t-1$. This is guaranteed by adding constraints (26) to the model.

$$w_{it} \leq y_{it-1} + qq_{it-1} \quad \forall i, t = 2, \ldots, T, \tag{26}$$

Constraints (27) limit the range of variables qq_{it} and constraints (28) are the usual non-negativity and binary constraints.

$$qq_{it} \leq w_{is} \quad \forall i, t = 2, \ldots, T-1, s = t, \ldots, t+1, \tag{27}$$

$$qq_{it} \geq 0 \;\; (qq_{i1} = 0, qq_{iT} = 0), w_{it} \in \{0, 1\} \;\; (w_{i1} = 0) \quad \forall i, t. \tag{28}$$

The complete MIP-formulation for the CLSPL is as follows:

$$\min \sum_{i=1}^{P} \sum_{t=1}^{T} (s_i y_{it} + h_i I_{it}) + \sum_{t=1}^{T} \sum_{m=1}^{M} c_m^o O_{mt},$$

subject to

$$I_{it} = I_{it-1} + x_{it} - \sum_{j \in \Gamma(i)} c_{ij} x_{jt} - E_{it}, \quad \forall i, t,$$

$$\sum_{i=1}^{P} (a_{mi} x_{it} + b_{mi} y_{it}) \leq L_{mt} + O_{mt}, \quad \forall m, t$$

$$x_{it} - G(y_{it} + w_{it}) \leq 0 \quad \forall i, t,$$

$$\sum_{i \in R_m} w_{it} \leq 1 \quad \forall m, t = 2, \ldots, T,$$

$$y_{it} + w_{it} + \sum_{\substack{j \in R_m \\ j \neq i}} qq_{jt} \leq 1 \quad \forall m, i \in R_m, t,$$

$$w_{it} \leq y_{it-1} + qq_{it-1} \quad \forall i, t = 2, \ldots, T,$$

$$qq_{it} \leq w_{is} \quad \forall i, t = 2, \ldots, T-1, s = t, \ldots, t+1,$$

$$I_{it} \geq 0, \quad O_{mt} \geq 0, \quad x_{it} \geq 0, \quad y_{it} \in \{0, 1\}, \quad \forall i, t,$$

$$qq_{it} \geq 0 \;\; (qq_{i1} = 0, qq_{iT} = 0), w_{it} \in \{0, 1\} \;\; (w_{i1} = 0) \quad \forall i, t.$$

7.2 Decomposition

This Subsection deals with the changes that have to be made if we solve the CLSPL with our decomposition approach. First, the constraints (29), (30) and (31) are adjusted so that they only hold for the items ($i = P_s^k, \ldots, P_e^k$) and periods ($t = T_s^k, \ldots, T_e^k$) included in the current subproblem. Since it is only possible to perform a setup in the first period of the planning horizon, the starting time period T_s^k of constraints (31) is restricted to values above 1.

$$x_{it} - G(y_{it} + w_{it}) \leq 0, \tag{29}$$

$$y_{it} + w_{it} + \sum_{\substack{j = P_s^k \\ j \neq i}}^{P_e^k} qq_{jt} \leq 1 \quad \forall m, i \in R_m, t, \tag{30}$$

$$w_{it} \leq y_{it-1} + qq_{it-1} \quad \text{if } T_s^k \neq 1, \tag{31}$$

In constraints (32) the original indices are exchanged by the indices of the current subproblem.

$$qq_{it} \leq w_{is} \quad \forall i = P_s^k, \ldots, P_e^k, t = T_s^k, \ldots, T_e^k - 1, s = t, \ldots, t+1, \tag{32}$$

Constraints (33) now have to consider linked setups that are made in previous subproblems. Therefore, the variable W_{it} stands for already determined linking decisions.

$$\sum_{\substack{i=P_s^k \\ i \in R_m}}^{P_e^k} w_{it} + \sum_{\substack{i=1 \\ i \in R_m}}^{P_s^k - 1} W_{it} \leq 1 \quad \forall m, t = T_s^k, \ldots, T_e^k, \text{ if } T_s^k \neq 1. \tag{33}$$

Now, when calculating a subproblem, it is not possible to forecast if a linking decision in any following subproblem might be more preferable than the current one. To circumvent this weakness we introduce a simple 'punishing scheme' to our model. More precisely, the objective function of a single subproblem is altered in the following way:

$$\min \sum_{i=P_s^k}^{P_e^k} \sum_{t=T_s^k}^{T_e^k} (S_{it}^k y_{it} + h_i I_{it}) + \sum_{t=T_s^k}^{T_e^k} \sum_{m=1}^{M} c_m^o O_{mt} + \sum_{\substack{i=P_s^k \\ i \in R_m}}^{P_e^k} \sum_{t=T_s^k}^{T_e^k} \sum_{m=1}^{M} w_{it} c_{it}^w \tag{34}$$

The new parameter c_{it}^w represents the maximum setup costs of any item that is located inside a subsequent subproblem. Thus, this extension to our objective function punishes every potential linking decision in the current subproblem. It has to make the decision if a linkage is more preferable in the present or in any following subproblem.

The complete mixed-integer problem for a single subproblem is as follows:

$$\min \sum_{i=P_s^k}^{P_e^k} \sum_{t=T_s^k}^{T_e^k} (S_{it}^k y_{it} + h_i I_{it}) + \sum_{t=T_s^k}^{T_e^k} \sum_{m=1}^{M} c_m^o O_{mt} + \sum_{\substack{i=P_s^k \\ i \in R_m}}^{P_e^k} \sum_{t=T_s^k}^{T_e^k} \sum_{m=1}^{M} w_{it} c_{it}^w,$$

subject to (if not stated otherwise each constraint must hold for all $i = P_s^k, \ldots, P_e^k$,

$t = T_s^k, \ldots, T_e^k$, and $m = 1, \ldots, M$)

$$I_{it} = I_{it-1} + x_{it} - \sum_{\substack{j \in \Gamma(i) \\ j < P_s^k}} c_{ij} X_{jt} - \sum_{\substack{j \in \Gamma(i) \\ j \geq P_s^k}} c_{ij} x_{jt} - E_{it},$$

$$\sum_{i=P_s^k}^{P_e^k} (a_{mi} x_{it} + b_{mi} y_{it}) \leq L_{mt} + O_{mt} - \sum_{i=1}^{P_s^k - 1} (a_{mi} X_{it} + b_{mi} Y_{it}),$$

$$\sum_{\tau=T_s^k}^{t} \sum_{i=P_s^k}^{P_e^k} (A_{mi}^k x_{i\tau} + B_{mi\tau}^k y_{i\tau}) \leq \sum_{\tau=T_s^k}^{t} (avC_{m\tau}^k + O_{m\tau}),$$

$$x_{it} - G(y_{it} + w_{it}) \leq 0,$$

$$\sum_{\substack{i=P_s^k \\ i \in R_m}}^{P_e^k} w_{it} + \sum_{\substack{i=1 \\ i \in R_m}}^{P_s^k - 1} W_{it} \leq 1 \quad \text{if } T_s^k \neq 1,$$

$$y_{it} + w_{it} + \sum_{\substack{j=P_s^k \\ j \neq i}}^{P_e^k} qq_{jt} \leq 1, \forall i \in R_m,$$

$$w_{it} \leq y_{it-1} + qq_{it-1} \quad \text{if } T_s^k \neq 1,$$

$$qq_{it} \leq w_{is} \quad \forall t = T_s^k, \ldots, T_e^k - 1, s = t, \ldots, t+1,$$

$$I_{it} \geq 0, \quad x_{it} \geq 0, \quad O_{mt} \geq 0, \quad y_{it} \in \{0, 1\},$$

$$qq_{it} \geq 0 \ \ (qq_{i1} = 0, qq_{iT} = 0), w_{it} \in \{0, 1\} \ \ (w_{i1} = 0).$$

8 Results for the CLSPL

As for the MLCLS problem, the CLSPL is implemented in C++ using CPLEX 11.0 to calculate the subproblems. All the instances were tested on a Pentium D 3.2GHz with 4 GB RAM and SUSE Linux 10.1. No changes were made concerning the subproblem sizes.

In total, the ant system for the capacitated lot-sizing problem with linked lot sizes (ASCLPL) was tested with two groups of instances. Note that it is not necessary to make any modifications to the ant system of Section 5 when solving the CLSPL. The first group of single-level instances from Trigeiro et al. (1989) was modified by Stadtler and Suerie (2003) by aggregating some of the items. The reason behind this modification is that the original instances proved not to be appropriate for the CLSPL. Tests from Stadtler and Suerie (2003) for example showed that the possibility of linking a setup state over two consecutive bucket boundaries was never used. The modified set is divided into three different classes:

Table 5: Classification of the first group of instances from Trigeiro et al. (1989) which was modified by Stadtler and Suerie (2003).

Class	#Items	#Periods	#Instances
1	4	20	180
2	6	20	180
3	8	20	180

Table 6: Results for the first group of instances. We compare our results (ASCLSPL) with the best known solutions (BKS) and the lower bound (LB) of the best known solutions. The number next to the class represents the number of instances for which the ASCLSPL could only find solutions with an extensive use of overtime. They are excluded from the results.

Class	BKS	LB	ASCLSPL	Gap to BKS	Gap to LB
1 (-2)	25 236	23 136	27 779	6.13%	12.46%
2 (-9)	48 754	44 788	53 151	5.31%	13.90%
3 (-10)	76 798	70 406	79 922	2.56%	8.42%

Table 6 shows our results for the single-level instances compared to the best solutions known to Stadtler and Suerie (2003). Since we were not able to verify if the lower bounds of the best solutions known are equal to those from Stadtler and Suerie (2003), a direct comparison between our approaches is not possible. For the sake of completeness we will state the best results from their time-oriented decomposition heuristic and their Branch and Cut (B&C) approach with valid inequalities in Table 8 at the end of the Section. Note that for some instances (Class 1: 1; Class 2: 9; Class 3: 10) the ASCLSPL could only find solutions with an extensive use of overtime, and they are therefore excluded from the results. The run time for the first and the second class equals to 200 seconds, while the third class runs for 300 seconds. Although the ASCLSPL algorithm can not reach the best known solutions it still provides good results for the first test group. Again, the expressed criticism in Section 6.2 also holds for the ASCLSPL.

Table 7: Results for the second group of instances. We compare our results (ASCLSPL) with the best known solutions (BKS) and the lower bound (LB) of the best known solutions.

Class	BKS	LB	ASCLSPL	Gap to BKS	Gap to LB
B+	82 220	65 493	87 431	6.23%	33.38%

The second group of instances was taken from Stadtler (2003) and it is called B+. It consists of 60 instances with 10 items on 3 resources over 24 periods. Results are provided in Table 7. Due to the multilevel case and the bigger problem size the gap to the lower bound is higher than in the first group. Again, the ASCLSPL with a run time of 400 seconds delivers good results, but it is unable to reach the best known solutions.

Table 8: Gap to lower bound from Stadtler and Suerie (2003) for the first and second group of instances. The number next to the class represents the number of instances for which the heuristic could only find infeasible solutions. They are excluded from the results. The run time for the heuristic equals to maximal 15 seconds for the first group, and to maximal 60 seconds for the second group. The Branch and Cut (B&C) approach has a run time of maximal 60 seconds for the first group and of maximal 600 seconds for the second group.

Class	Heuristic - Gap to LB	B&C - Gap to LB
1 (-20)	9.5%	8.7%
2 (-32)	10.3%	10.2%
3 (-32)	7.1%	7.1%
B+	29.1%	37.5%

9 Conclusion

In this thesis, a metaheuristic that uses the ideas of MMAS and Evolutionary Strategies combined with exact solvers for mixed-integer problems has been applied to solve multi-level and single-level capacitated lot-sizing problems with linked lot sizes.

Two different mathematical formulations have been presented and tested for effectiveness, whereas the standard formulation proved to be significantly better than the SPL formulation. A possible explanation for the large gap between those formulations could be the computational overhead when using the SPL formulation. After selecting the formulation, the ASMLCLS algorithm has been tested on middle-sized and large-sized multi-level test instances. While the results for the smaller test instances are among the best, the results for the larger instances are poor.

The results reported in Section 8 for the CLSPL do not outperform the best known solutions but nevertheless provide good results. In addition, the results show that the bigger the problem gets, the better is the gap to the best known solutions. Since the decomposition approach is very complex, the computational overhead causes the algorithm to need more time to find good solutions.

As explained in Section 6.2, the algorithm is very slow and furthermore, the solution quality is very sensitive towards the starting solution. For this reason, further research will be necessary to improve or exchange the method of finding a starting solution, and also reducing the complexity to speed up the algorithm.

References

Almada-Lobo, B., Klabjan, D., Carravilla, M., and Oliveira, J. (2007). Single machine multi-product capacitated lot sizing with sequence-dependent setups. *International Journal of Production Research*, 45:4873–4894.

Almeder, C. (2007). A hybrid optimization approach for multi-level capacitated lot-sizing problems. Working paper, submitted.

Berretta, R. and Rodrigues, L. (2004). A memetic algorithm for a multistage capacitated lot-sizing problem. *International Journal of Production Economics*, 87:67–81.

Bitran, G. and Yanasse, H. (1982). Computational complexity of the capacitated lot size problem. *Management Science*, 28(10):1174–1186.

Dellaert, N. and Jeunet, J. (2000). Solving large unconstrained multilevel lot-sizing problems using a hybrid genetic algorithm. *International Journal of Production Research*, 38(5).

Dellaert, N. and Jeunet, J. (2003). Randomized multi-level lot sizing heuristics for general product structures. *European Journal of Operational Research*, 148:211–228.

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy.

Drexl, A. and Haase, K. (1995). Proportional lotsizing and scheduling. *International Journal of Production Economics*, 40:73–87.

Fleischmann, B. (1990). The discrete lot-sizing and scheduling problem. *European Journal of Operational Research*, 44:337–348.

Fleischmann, B. and Meyr, H. (1997). The general lotsizing and scheduling problem. *OR Spektrum*, 19:11–21.

Florian, M., Lenstra, J., and Kan, R. (1980). Deterministic production planning: Algorithms and complexity. *Management Science*, 26(7):669–679.

Franca, P., Armentano, V., Berretta, R., and Clark, A. (1994). A heuristic method for lot sizing in multi-stage systems. *Computers & Operations Research*, 24:861–874.

Gopalakrishnan, M., Ding, K., and Bourjolly, J.-M. (2001). A tabu-search heuristic for the capacitated lot-sizing problem with set-up carryover. *Management Science*, 47:851–863.

Gopalakrishnan, M., Miller, D. M., and Schmidt, C. P. (1995). A framework for modelling setup carryover in the capacitated lotsizing problem. *Journal of Production Research*, 33:1973–1988.

Gutjahr, W. (2003). A generalized convergence result for the graph-based ant system metaheuristic. *Probability in the Engineering and Informational Sciences*, 17:545–569.

Haase, K. (1994). Lotsizing and scheduling for production planning. Springer:Berlin.

Harrison, T. and Lewis, H. (1996). Lot sizing in serial assembly systems with multiple constrained resources. *Management Science*, 42(1):19–36.

Karmarkar, U. and Schrage, L. (1985). The deterministic dynamic product cycling problem. *Operational Research*, 33:326–345.

Kimms, A. (1996). Competitive methods for multi-level lot sizing and scheduling: Tabu search and randomized regrets. *International Journal of Production Research*, 34(8):2279–2298.

Krarup, J. and Bilde, . (1977). *Plant location, set covering and economic lot sizing: An O(mn) algorithm for structured problems*. Birkhäuser:Basel.

Lambrecht, M. and Vander Eecken, J. (1978). A facilities in series capacity constrained dynamic lot-size model. *European Journal of Operational Research*, 2:42–49.

Maes, J. and McClain, J. (1991). Multilevel capacitated lot sizing complexity and LP-based heuristics. *European Journal of Operation Research*, 53:131–148.

McClain, J. and Thomas, L. (1989). A facilities in series capacity constrained dynamic lot-size model. *IIE Transactions*, 21(2):144–152.

McLaren, B. (1977). *A study of multiple level lot sizing procedures for material requirements planning systems*. PhD thesis, Purdue University.

Merkle, D. and Middendorf, M. (1999). An ant algorithm with a new pheromone evaluation rule for total tardiness problems. In *Applications of Evolutionary Computing: EvoWorkshops*, pages 287–296. Springer:Berlin.

Oezdamar, L. and Barbarosoglu, G. (2000). An integrated langrangean relaxation-simulated annealing approach to the multi-level mulit-item capacitated lot sizing problem. *International Journal of Production Research*, 68(3):319–331.

Pitakaso, R., Almeder, C., Doerner, K., and Hartl, R. (2006). Combining population-based and exact methods for multi-level capacitated lot-sizing problems. *International Journal of Production Research*, 44 (22):4755–4771.

Pitakaso, R., Almeder, C., Doerner, K., and Hartl, R. (2007). A MAX-MIN ant system for the uncapacitated multi-level lot sizing problem. *Computers & Operations Research*, 34:2533–2552.

Pochet, Y. and Wolsey, L. (1987). Lot-size models with backlogging: Strong reformulations and cutting planes. *Mathematical Programming*, 40:317–335.

Rosling, K. (1986). *Optimal lot-sizing for dynamic assembly systems*. Springer:Berlin.

Salomon, M. (1986). Deterministic lotsizing models for production planning.

Silver, E. and Meal, H. (1973). A heuristic for selecting lot size requirements for the case of a deterministic time-varying demand rate and discrete opportunities for replenishment. *Production and Inventory Management*, 14:64–74.

Sox, C. and Gao, Y. (1999). The capacitated lot sizing problem with setup carry-over. *IIE Transactions*, 31:173–181.

Stadtler, H. (1996). Mixed integer programming model formulations for dynamic multi-item multi-level capacitated lotsizing. *European Journal of Operation Research*, 94:561–581.

Stadtler, H. (2003). Multilevel lot sizing with set up times and multiple constrained resources: internally rolling schedules with lot-sizing windows. *Operations Research*, 51(3):487–502.

Stadtler, H. and Suerie, C. (2003). The capacitated lot-sizing problem with linked lot sizes. *Management Science*, 49:1039–1054.

Stützle, T. (1998). An ant approach to the flow shop problem. In *Proceedings of EU-FIT'98*, pages 1560–1564.

Stützle, T. and Dorigo, M. (2002). A short convergence proof for a class of ACO algorithms. *IEEE Transactions on Evolutionary Computation*, 6:358–365.

Stützle, T. and Hoos, H. (1997). The MAX-MIN ant system and local search for the traveling salesman problem. In *Proceedings of ICEC'97 IEEE 4th International Conference on Evolutionary Computation*, pages 308–313.

Tempelmeier, H. and Derstroff (1996). A lagrangean-based heuristic for dynamic multi-level multi-item constrained lotsizing with setup times. *Management Science*, 16:738–757.

Tempelmeier, H. and Helber, S. (1994). A heuristic for dynamic multi-item multi-level capacitated lotsizing for general product structures. *European Journal of Operational Research*, 75:296–311.

Trigeiro, W., Thomas, L., and McClain, J. (1989). Capacitated lot sizing with set-up times. *Management Science*, 35:738–757.

Wagner, H. and Whitin, T. (1958). Dynamic version of the economic lot size model. *Management Science*, 5:89–96.

Wolsey, L. (1989). Uncapacitated lot-sizing problems with start-up costs. *Operations Research*, 37(5):741–747.

Xie, J. and Dong, J. (2002). Heuristic genetic algorithms for general capacitated lot-sizing problems. *Computers and Mathematics with Applications*, 44:263–276.

Zangwill, W. (1966). A deterministic multiproduct, multifacility production and inventory model. *Operations Research*, 14(3):486–507.

Zangwill, W. (1969). A backlogging model and a multi-echelon model of a dynamic economic lot size production system - a network approach. *Management Science*, 15(9):506–527.

A Detailed Results

Solutions for the medium-sized MLCLS problems					
Instance	Result	Instance	Result	Instance	Result
g0061111	56016	g8065111	75312	k0029111	16492
g0061112	56197	g8065112	73752	k0029112	15877
g0061121	358345	g8065121	353643	k0029121	49555
g0061122	355334	g8065122	355594	k0029122	51316
g0061131	929494	g8065131	896371	k0029131	143289
g0061132	887565	g8065132	837173	k0029132	139801
g0061141	651433	g8065141	651855	k0029141	58367
g0061142	675723	g8065142	643590	k0029142	55623
g0061151	211240	g8065151	229990	k0029151	51197
g0061152	211987	g8065152	218631	k0029152	55996
g0061211	56000	g8065211	57040	k0029211	9750
g0061212	56000	g8065212	58095	k0029212	8495
g0061221	314808	g8065221	335125	k0029221	38470
g0061222	317937	g8065222	327465	k0029222	38403
g0061231	688459	g8065231	675894	k0029231	77118
g0061232	657348	g8065232	737856	k0029232	75106
g0061241	497584	g8065241	514437	k0029241	42757
g0061242	495777	g8065242	506407	k0029242	40673
g0061251	197171	g8065251	204601	k0029251	40826
g0061252	197283	g8065252	204897	k0029252	41065
g0061311	56000	g8065311	55990	k0029311	6323
g0061312	56000	g8065312	55970	k0029312	5782
g0061321	308103	g8065321	314409	k0029321	35114
g0061322	306028	g8065322	312609	k0029322	33695
g0061331	599267	g8065331	612550	k0029331	70537
g0061332	600874	g8065332	616927	k0029332	67067
g0061341	460938	g8065341	468022	k0029341	37984
g0061342	463021	g8065342	469746	k0029342	36175
g0061351	185605	g8065351	183596	k0029351	36518
g0061352	187611	g8065352	183877	k0029352	34760
g0061411	56016	g8065411	64984	k0029411	14188

g0061412	56197	g8065412	64358	k0029412	12750
g0061421	310608	g8065421	329483	k0029421	41675
g0061422	314748	g8065422	329139	k0029422	40081
g0061431	616723	g8065431	666012	k0029431	78477
g0061432	614033	g8065432	655462	k0029432	81100
g0061441	463542	g8065441	494477	k0029441	44831
g0061442	465374	g8065442	490651	k0029442	42517
g0061451	199812	g8065451	210250	k0029451	45744
g0061452	200235	g8065452	211801	k0029452	42941
g0061511	56000	g8065511	61826	k0029511	11667
g0061512	56000	g8065512	63050	k0029512	11293
g0061521	353205	g8065521	349368	k0029521	42107
g0061522	344064	g8065522	342316	k0029522	40095
g0061531	900403	g8065531	743019	k0029531	86793
g0061532	877132	g8065532	815083	k0029532	84232
g0061541	637243	g8065541	587267	k0029541	46957
g0061542	627959	g8065542	582392	k0029542	43968
g0061551	203655	g8065551	209548	k0029551	44328
g0061552	202591	g8065552	205802	k0029552	43242
g0065111	77990	g8069111	77875	k8021111	7071
g0065112	74805	g8069112	102088	k8021112	7076
g0065121	361439	g8069121	390217	k8021121	46373
g0065122	360856	g8069122	396506	k8021122	46342
g0065131	842831	g8069131	884296	k8021131	115497
g0065132	826349	g8069132	887234	k8021132	115923
g0065141	672087	g8069141	637212	k8021141	51444
g0065142	613032	g8069142	663971	k8021142	52088
g0065151	227761	g8069151	234307	k8021151	53480
g0065152	219121	g8069152	239207	k8021152	51028
g0065211	58105	g8069211	58551	k8021211	7040
g0065212	57952	g8069212	69655	k8021212	7040
g0065221	317390	g8069221	325571	k8021221	40091
g0065222	318491	g8069222	331241	k8021222	40401
g0065231	642581	g8069231	645728	k8021231	86168
g0065232	622065	g8069232	659852	k8021232	85283

g0065241	487209	g8069241	504909	k8021241	44065
g0065242	483061	g8069242	510375	k8021242	44152
g0065251	201099	g8069251	200833	k8021251	42384
g0065252	201148	g8069252	210186	k8021252	42209
g0065311	55990	g8069311	55635	k8021311	7040
g0065312	55970	g8069312	61050	k8021312	7040
g0065321	304097	g8069321	301869	k8021321	38358
g0065322	303099	g8069322	305351	k8021322	38728
g0065331	600543	g8069331	618215	k8021331	76097
g0065332	593652	g8069332	625891	k8021332	75979
g0065341	461050	g8069341	466052	k8021341	41496
g0065342	461286	g8069342	468359	k8021342	41425
g0065351	186620	g8069351	183689	k8021351	39779
g0065352	181992	g8069352	187979	k8021352	40167
g0065411	73170	g8069411	67792	k8021411	7040
g0065412	72080	g8069412	80545	k8021412	7065
g0065421	327404	g8069421	323571	k8021421	40583
g0065422	322753	g8069422	336125	k8021422	40417
g0065431	631868	g8069431	642676	k8021431	85295
g0065432	628055	g8069432	664044	k8021432	85906
g0065441	482488	g8069441	493781	k8021441	42386
g0065442	476864	g8069442	506052	k8021442	42855
g0065451	214163	g8069451	210655	k8021451	44568
g0065452	211270	g8069452	221400	k8021452	45206
g0065511	63085	g8069511	63906	k8021511	7071
g0065512	64519	g8069512	79429	k8021512	7054
g0065521	336254	g8069521	341362	k8021521	42670
g0065522	331293	g8069522	334279	k8021522	43413
g0065531	700812	g8069531	747324	k8021531	95842
g0065532	768917	g8069532	779253	k8021532	99815
g0065541	567002	g8069541	578269	k8021541	48522
g0065542	572860	g8069542	553197	k8021542	49313
g0065551	206286	g8069551	202786	k8021551	45866
g0065552	205766	g8069552	213928	k8021552	46512
g0069111	82001	k0021111	7107	k8025111	9703

g0069112	102936	k0021112	7086	k8025112	9640
g0069121	357255	k0021121	46665	k8025121	42696
g0069122	377377	k0021122	47262	k8025122	44526
g0069131	849953	k0021131	122450	k8025131	103775
g0069132	884729	k0021132	123924	k8025132	106798
g0069141	620626	k0021141	53876	k8025141	48598
g0069142	621865	k0021142	56271	k8025142	51630
g0069151	225849	k0021151	53299	k8025151	47120
g0069152	240100	k0021152	52425	k8025152	49679
g0069211	60272	k0021211	7040	k8025211	7489
g0069212	71883	k0021212	7040	k8025212	7054
g0069221	313989	k0021221	41156	k8025221	39648
g0069222	322235	k0021222	41799	k8025222	39358
g0069231	627630	k0021231	86652	k8025231	81008
g0069232	632527	k0021232	87716	k8025232	81647
g0069241	483477	k0021241	44705	k8025241	42536
g0069242	512781	k0021242	44814	k8025242	43007
g0069251	200361	k0021251	44251	k8025251	36582
g0069252	209687	k0021252	44278	k8025252	40955
g0069311	55635	k0021311	7040	k8025311	6520
g0069312	60634	k0021312	7040	k8025312	6377
g0069321	298726	k0021321	38888	k8025321	36720
g0069322	300177	k0021322	39114	k8025322	36146
g0069331	588030	k0021331	75960	k8025331	72315
g0069332	588152	k0021332	75779	k8025332	73830
g0069341	455962	k0021341	41646	k8025341	40329
g0069342	463137	k0021342	42164	k8025342	39357
g0069351	183041	k0021351	40585	k8025351	37278
g0069352	189122	k0021352	41071	k8025352	36585
g0069411	78081	k0021411	7040	k8025411	8869
g0069412	93272	k0021412	7059	k8025412	8695
g0069421	326564	k0021421	40831	k8025421	39063
g0069422	343274	k0021422	41409	k8025422	40033
g0069431	625100	k0021431	86746	k8025431	80507
g0069432	633381	k0021432	87515	k8025432	87325

g0069441	487314	k0021441	42389	k8025441	42700
g0069442	500051	k0021442	42765	k8025442	42979
g0069451	218009	k0021451	45331	k8025451	42725
g0069452	233051	k0021452	46236	k8025452	43133
g0069511	66166	k0021511	7079	k8025511	7866
g0069512	83135	k0021512	7058	k8025512	7609
g0069521	330792	k0021521	44977	k8025521	40691
g0069522	329270	k0021522	45081	k8025522	40769
g0069531	744915	k0021531	106075	k8025531	86967
g0069532	736678	k0021532	108836	k8025532	86917
g0069541	541660	k0021541	51316	k8025541	44523
g0069542	540127	k0021542	51683	k8025542	44748
g0069551	202728	k0021551	47385	k8025551	42768
g0069552	209626	k0021552	46796	k8025552	42451
g8061111	56000	k0025111	9936	k8029111	15032
g8061112	56058	k0025112	10006	k8029112	13858
g8061121	377577	k0025121	48700	k8029121	45232
g8061122	366866	k0025122	48451	k8029122	45020
g8061131	931253	k0025131	104751	k8029131	93003
g8061132	924958	k0025132	120663	k8029132	88451
g8061141	681869	k0025141	56135	k8029141	48444
g8061142	681629	k0025142	51951	k8029142	48900
g8061151	222524	k0025151	50786	k8029151	48571
g8061152	217949	k0025152	52868	k8029152	47227
g8061211	56000	k0025211	7536	k8029211	9310
g8061212	56000	k0025212	7120	k8029212	7939
g8061221	329324	k0025221	40828	k8029221	37587
g8061222	323329	k0025222	39047	k8029222	35679
g8061231	727329	k0025231	83185	k8029231	76086
g8061232	721501	k0025232	82221	k8029232	72683
g8061241	514507	k0025241	43393	k8029241	40791
g8061242	550984	k0025242	42197	k8029242	38756
g8061251	207422	k0025251	43056	k8029251	40072
g8061252	209877	k0025252	42816	k8029252	38298
g8061311	56000	k0025311	6557	k8029311	6300

g8061312	56000	k0025312	6377	k8029312	5646
g8061321	309471	k0025321	37332	k8029321	33833
g8061322	309011	k0025322	37028	k8029322	31912
g8061331	608922	k0025331	73149	k8029331	67974
g8061332	622967	k0025332	74050	k8029332	65552
g8061341	469313	k0025341	40832	k8029341	38115
g8061342	469756	k0025342	40610	k8029342	35113
g8061351	188826	k0025351	38823	k8029351	35183
g8061352	187190	k0025352	38177	k8029352	33683
g8061411	56000	k0025411	9131	k8029411	12222
g8061412	56058	k0025412	8946	k8029412	11096
g8061421	320323	k0025421	40809	k8029421	39626
g8061422	319293	k0025422	40653	k8029422	38858
g8061431	666976	k0025431	84432	k8029431	77625
g8061432	662555	k0025432	85067	k8029432	75283
g8061441	486222	k0025441	42629	k8029441	42585
g8061442	491753	k0025442	43085	k8029442	41384
g8061451	207940	k0025451	43455	k8029451	43111
g8061452	207255	k0025452	45243	k8029452	41616
g8061511	56000	k0025511	8153	k8029511	10742
g8061512	56000	k0025512	7774	k8029512	9502
g8061521	361358	k0025521	42769	k8029521	39266
g8061522	361124	k0025522	41726	k8029522	37283
g8061531	859612	k0025531	93363	k8029531	80673
g8061532	845562	k0025532	97097	k8029532	74538
g8061541	658710	k0025541	48452	k8029541	42508
g8061542	642994	k0025542	48513	k8029542	40539
g8061551	213995	k0025551	45529	k8029551	42001
g8061552	211557	k0025552	44510	k8029552	39522

Solutions for the large-sized MLCLS problems

Instance	Result - 1 seed	Result - 10 seeds	Instance	Result - 1 seed	Result - 10 seeds
g0151111	324640	324640	g8151111	324728	324728
g0151121	2412057	2112241	g8151121	2623737	2393311
g0151131	6221087	5580657	g8151131	7112950	6853750

g0151141	3973785	3426972	g8151141	4288072	3688653
g0151151	2198830	1956240	g8151151	2454455	2360971
g0151211	324640	324640	g8151211	324640	324640
g0151221	1845576	1821422	g8151221	1990441	1988476
g0151231	4162749	3915774	g8151231	5094379	4521949
g0151241	2587728	2491228	g8151241	2728997	2711466
g0151251	1727935	1714571	g8151251	1962508	1826923
g0151311	324640	324640	g8151311	324640	324640
g0151321	1798430	1757360	g8151321	1815550	1800421
g0151331	3498660	3457858	g8151331	3994728	3677158
g0151341	2326431	2311537	g8151341	2410700	2348338
g0151351	1683282	1636765	g8151351	1703724	1652322
g0151411	324640	324640	g8151411	324712	324712
g0151421	1905389	1855504	g8151421	2037115	1984026
g0151431	4268227	3953455	g8151431	4621724	4295122
g0151441	2423131	2357335	g8151441	2746450	2551180
g0151451	1833967	1760779	g8151451	1927846	1900818
g0151511	324640	324640	g8151511	324640	324640
g0151521	2036511	2036511	g8151521	2137330	2137330
g0151531	5411441	4910869	g8151531	5870806	5390320
g0151541	3534862	3301939	g8151541	3614200	3400968
g0151551	1976125	1876230	g8151551	1891916	1891916
g0155111	440258	420798	g8155111	578946	460070
g0155121	2975168	2056174	g8155121	3163066	2865117
g0155131	6439242	5232336	g8155131	8331306	8057254
g0155141	4906887	3108412	g8155141	5275715	4643210
g0155151	1913182	1860650	g8155151	2760916	2365123
g0155211	332847	331467	g8155211	335281	332360
g0155221	1810422	1758545	g8155221	1965129	1929176
g0155231	3838712	3624095	g8155231	4746236	4418319
g0155241	2489065	2382061	g8155241	2653123	2574014
g0155251	1670599	1661314	g8155251	1805575	1778032
g0155311	322302	322272	g8155311	322308	322272
g0155321	1714699	1684618	g8155321	1764061	1755831
g0155331	3438410	3416240	g8155331	3654253	3566698

g0155341	2270238	2247645	g8155341	2322293	2310437
g0155351	1621035	1597776	g8155351	1649381	1645582
g0155411	419974	410117	g8155411	453866	414569
g0155421	1895421	1823657	g8155421	2033212	1963730
g0155431	3984437	3701499	g8155431	4577730	4188083
g0155441	2423125	2345143	g8155441	2593739	2535122
g0155451	1820397	1767308	g8155451	1933526	1887419
g0155511	364480	354425	g8155511	353918	348383
g0155521	2003944	1872251	g8155521	2070969	2043104
g0155531	4470729	4353071	g8155531	5877676	4912112
g0155541	3006449	2801734	g8155541	3306884	3126712
g0155551	1756011	1734951	g8155551	2005063	1844747
g0159111	498479	460720	g8159111	753720	539627
g0159121	2921029	2194042	g8159121	2909327	2909327
g0159131	6754113	5061799	g8159131	7139585	7139585
g0159141	4641876	3194321	g8159141	5329602	5137444
g0159151	1907314	1874292	g8159151	2681703	2679682
g0159211	348770	342592	g8159211	362938	349271
g0159221	1772907	1753132	g8159221	1958940	1903479
g0159231	3769484	3564766	g8159231	5362546	4188755
g0159241	2472312	2352394	g8159241	2659635	2605615
g0159251	1633130	1624604	g8159251	1791632	1735199
g0159311	310726	309869	g8159311	310726	309878
g0159321	1702832	1670933	g8159321	1728067	1728067
g0159331	3325102	3286648	g8159331	3577401	3482397
g0159341	2270055	2210731	g8159341	2338245	2312451
g0159351	1572171	1554541	g8159351	1623878	1600777
g0159411	461119	447999	g8159411	503131	441539
g0159421	1884554	1858711	g8159421	1957757	1938377
g0159431	3711604	3678157	g8159431	4010344	4010344
g0159441	2456334	2370705	g8159441	3334374	2490987
g0159451	1812072	1751374	g8159451	1883744	1846743
g0159511	388190	376701	g8159511	398450	386439
g0159521	1915484	1888278	g8159521	1964510	1964510
g0159531	4435322	4425501	g8159531	6051418	3064139

g0159541	2866301	2754809	g8159541	2958048	2903970
g0159551	1737448	1674467	g8159551	1978193	1789845

Solutions for the single-level CLSPL problems					
Class 1	Result	Class 2	Result	Class 3	Result
401	5553	581	11976	761	19469
402	5151	582	11284	762	19465
403	5963	583	10585	763	17361
404	5410	584	13407	764	16469
405	5597	585	10885	765	17911
406	5140	586	11842	766	18478
407	5550	587	11544	767	17062
408	3991	588	11767	768	18104
409	6247	589	11739	769	18024
410	6568	590	12733	770	20239
411	5597	591	12906	771	23638
412	8238	592	13332	772	21838
413	7557	593	12351	773	25973
414	5410	594	14111	774	20506
415	5576	595	13034	775	24197
416	5462	596	12073	776	16206
417	6037	597	11525	777	21970
418	6140	598	13012	778	21947
419	5413	599	10521	779	19832
420	5991	600	10981	780	17350
421	5194	601	10851	781	19564
422	6377	602	12059	782	19664
423	6425	603	11111	783	18747
424	5517	604	12208	784	18444
425	6841	605	10202	785	20131
426	7035	606	11648	786	20870
427	3939	607	10453	787	18850
428	5525	608	10650	788	18298
429	5447	609	14068	789	18868
430	5935	610	10387	790	19707

431	20246	611	34126	791	55791
432	18937	612	38806	792	59000
433	19303	613	39398	793	58811
434	18720	614	36136	794	56946
435	20993	615	33240	795	60074
436	19302	616	42007	796	58244
437	18190	617	38388	797	68280
438	17385	618	40629	798	67601
439	17131	619	35338	799	58075
440	17505	620	37747	800	62987
441	19896	621	49451	801	73725
442	25557	622	49104	802	70615
443	25904	623	52496	803	73898
444	22043	624	51023	804	72340
445	23699	625	46403	805	71261
446	20427	626	38185	806	54744
447	20446	627	40182	807	61489
448	21425	628	35281	808	62623
449	17758	629	35337	809	68483
450	20768	630	38645	810	60475
451	20261	631	41670	811	62354
452	23423	632	39953	812	61792
453	17507	633	35845	813	60468
454	20164	634	37794	814	63395
455	22513	635	40904	815	64882
456	22513	636	49853	816	76940
457	21530	637	47355	817	71524
458	21530	638	56926	818	64816
459	24794	639	46693	819	69725
460	24575	640	56043	820	75480
461	51739	641	113944	821	143838
462	45013	642	100873	822	163709
463	48463	643	105876	823	163244
464	48463	644	96765	824	141974
465	51022	645	95858	825	152048

466	52291	646	107274	826	162832
467	51208	647	117194	827	167635
468	51208	648	105084	828	166340
469	51969	649	99244	829	170826
470	57221	650	114509	830	169258
471	49807	651	134465	831	187127
472	49807	652	141187	832	189779
473	70701	653	124305	833	207870
474	70701	654	122728	834	194353
475	55818	655	140315	835	206333
476	52725	656	105053	836	168130
477	52936	657	105638	837	160634
478	60335	658	105675	838	173731
479	52751	659	104645	839	158588
480	47381	660	110040	840	158415
481	58811	661	111651	841	178314
482	71463	662	125840	842	192706
483	54746	663	108783	843	172891
484	69984	664	108816	844	197262
485	58626	665	155020	845	192313
486	54695	666	6252870	846	176004
487	76858	667	8808953	847	21474460
488	2225676	668	133360	848	25077588
489	80050	669	4278749	849	11236196
490	69360	670	8413667	850	10995349
491	5145	671	10483	851	19455
492	5877	672	10926	852	21115
493	5000	673	11004	853	17520
494	5337	674	10354	854	18933
495	5080	675	10892	855	19591
496	6683	676	12564	856	18592
497	5867	677	11840	857	18018
498	4871	678	10548	858	17613
499	6035	679	12422	859	19150
500	6062	680	12627	860	19534

501	11629	681	16494	861	41629
502	14893	682	21785	862	28323
503	9654	683	17119	863	24241
504	8531	684	17858	864	24346
505	7085	685	10516	865	25905
506	4955	686	10882	866	18047
507	5921	687	11279	867	16774
508	6086	688	11437	868	18803
509	4793	689	12112	869	18307
510	6151	690	9976	870	19201
511	5823	691	9493	871	17947
512	5007	692	10227	872	19113
513	5907	693	11553	873	16387
514	5660	694	11817	874	18417
515	4774	695	9329	875	17674
516	5541	696	12072	876	17889
517	6175	697	11875	877	18651
518	6133	698	9309	878	17931
519	4981	699	15323	879	20813
520	5439	700	12274	880	16644
521	17862	701	33153	881	52969
522	17402	702	35610	882	59666
523	21034	703	35254	883	59318
524	19662	704	39057	884	60291
525	15283	705	34196	885	62682
526	20935	706	41282	886	59118
527	17747	707	43531	887	62772
528	17550	708	38449	888	61995
529	20171	709	38045	889	57556
530	17715	710	42121	890	59516
531	31085	711	45416	891	67861
532	22235	712	52200	892	72494
533	24209	713	47948	893	70162
534	22735	714	49952	894	78976
535	22172	715	51270	895	81132

536	16449	716	40697	896	57691
537	21093	717	38300	897	58837
538	20347	718	34593	898	58076
539	19839	719	37875	899	59992
540	16970	720	33153	900	53334
541	21225	721	34846	901	62097
542	19731	722	42312	902	60034
543	21962	723	38820	903	64989
544	17106	724	40614	904	61426
545	23218	725	38697	905	58493
546	19061	726	55558	906	64688
547	23819	727	45144	907	68354
548	2749964	728	49822	908	65048
549	30541	729	53016	909	74289
550	23127	730	51310	910	65485
551	51915	731	99038	911	148586
552	50309	732	99323	912	136091
553	46610	733	102413	913	147606
554	45969	734	108151	914	149512
555	49227	735	93064	915	165429
556	43314	736	99272	916	154602
557	51115	737	119997	917	148927
558	53042	738	114889	918	167060
559	41053	739	109881	919	160428
560	40751	740	102789	920	158435
561	58434	741	147631	921	4858481
562	92947	742	142582	922	223830
563	58814	743	139591	923	188493
564	58458	744	124326	924	201320
565	64145	745	121958	925	180148
566	50760	746	97275	926	165107
567	54543	747	115533	927	179454
568	51247	748	101814	928	145998
569	42554	749	94679	929	155435
570	49530	750	103136	930	156446

571	58697	751	139479	931	180393
572	62169	752	118627	932	178010
573	64606	753	133150	933	184888
574	65292	754	113840	934	174386
575	60557	755	106952	935	181496
576	63436	756	15290283	936	18051286
577	64807	757	5392641	937	7209169
578	73927	758	5176258	938	27760829
579	68717	759	13104901	939	9706015
580	61884	760	4756803	940	23618397

Solutions for the multi-level CLSPL problems

Instance	Result	Instance	Result	Instance	Result
1	95242	21	77930	41	81050
2	103729	22	74335	42	89080
3	95346	23	72282	43	93785
4	78429	24	73154	44	96497
5	79024	25	77471	45	100594
6	81647	26	81623	46	100889
7	74410	27	97198	47	100356
8	72748	28	88914	48	102106
9	73455	29	95306	49	84627
10	79411	30	97984	50	82219
11	81480	31	102677	51	87299
12	91575	32	103217	52	76365
13	90698	33	118521	53	73834
14	89535	34	85434	54	73177
15	101148	35	84048	55	81917
16	92420	36	84135	56	80902
17	107240	37	75647	57	89103
18	87505	38	74074	58	95704
19	77997	39	73904	59	102558
20	79428	40	82937	60	100562

Abstract

In this thesis I focus my attention on the lot-sizing problem, which is part of the material requirements planning (MRP). A lot-sizing problem intends to minimize the inventory, setup, and production costs while meeting the required demand. Since producing firms in the globalized economy more and more pay attention to production decisions and costs the lot sizing problem is of prime importance. After a theoretical overview of the lot-sizing problems two different types of the lot-sizing problem are covered in this thesis: Firstly, the multi-level capacitated lot-sizing problem (MLCLS), and secondly the capacitated lot-sizing problem with linked lot sizes (CLSPL). The CLSPL is a big-bucket model that allows to carry over setup states from one period to the next. Furthermore, I test two different mathematical formulations for effectiveness. The solution approach I use is a hybrid algorithm which decomposes the given problem into multiple smaller subproblems. These subproblems are then solved by CPLEX. An Ant Colony Optimization (ACO) algorithm is then applied to determine the lot-sizing sequence and to improve the decomposition. My approach for the MLCLS problem works very well with medium-sized instances, but has difficulties with respect to solution quality when solving large-sized test instances. Good results are obtained for the CLSPL problem.

Zusammenfassung

In dieser Diplomarbeit lege ich meinen Fokus auf das Losgrößenproblem, das ein Teil der Materialbedarfsplanung ist. Das Losgrößenproblem versucht die Lagerhaltungs-, Rüst-, und Produktionskosten zu minimieren und dabei die notwendige Nachfrage zu bedienen. Da jede produzierende Firma in einer globalisierten Welt immer mehr auf Produktionsentscheidungen und Kosten achtet, ist das Losgrößenproblem von größtmöglicher Bedeutung. Nach einem theoretischen Überblick über das Losgrößenproblem werden zwei verschiedene Varianten des Losgrößenproblems abgedeckt: Erstens das sogenannte Multi-Level Capacitated Lot-Sizing (MLCLS) Problem und zweitens das sogennante Capacitated Lot-Sizing Problem with Linked Lot Sizes (CLSPL). Das CLSPL ist ein Big-Bucket-Modell, das es erlaubt Rüstzustände in die nächste Periode mitzunehmen. Desweiteren werden zwei mathematische Formulierungen auf ihre Effektivität getestet. Der von mir benutzte Lösungsansatz ist ein hybrider Algorithmus der das gegebene Problem in mehrere kleinere Subprobleme zerlegt. Dies Subprobleme werden dann mit CPLEX gelöst. Eine Ant Colony Optimization(ACO)-Metaheuristik wird dann angewendet um die Reihung der Losgrößen zu bestimmen und die Zerlegung in Subprobleme zu verbessern. Mein Lösungsansatz funktioniert sehr gut mit mittelgroßen Instanzen, hat aber Schwierigkeiten bezüglich der Lösungsgüte bei großen Instanzen. Gute Resultate werden für das CLSPL erreicht.

Lebenslauf

Zur Person

Wolfgang Summerauer

geboren am **23. Jänner 1983** in Gmunden, OÖ

Nationalität: **Österreich**

Religion: **o.r.B.**

Familienstand: **Ledig**

Ausbildung

2002-2008	Studium der Internationalen Betriebswirtschaftslehre, Universität Wien
2001 -2002	Studium der Ethnologie und Afrikanistik, Universität Wien
1993-2001	Realgymnasium Höhere Internatsschule des Bundes Schloss Traunsee
1989-1993	Volksschule Laakirchen

Anstellungen

Juli 2005 - Sept. 2005 Praktikum bei der EA Generali, Abteilung für Controlling

Wissenschaftliche Anstellungen

-

Wissenschaftliche Arbeiten

-

Sprachen

Deutsch (Muttersprache), Englisch (fließend), Französisch (sehr gut),

Spanisch/Italienisch (Grundkenntnisse)