IIE Transactions (2007) **39**, 191–201

ISSN: 0740-817X print / 1545-8830 online
DOI: 10.1080/07408170600733244

A Lagrangean heuristic for integrated production and transportation planning problems in a dynamic, multi-item, two-layer supply chain

SANDRA DUNI EKŞIOĞLU[1,*], BURAK EKŞIOĞLU[1] and H. EDWIN ROMEIJN[2]

[1] *Department of Industrial and Systems Engineering, Mississippi State University, P.O. Box 9542, Mississippi State, MS 39762, USA*
E-mail: sde47@msstate.edu
[2] *Department of Industrial and Systems Engineering, University of Florida, 303 Weil Hall, Gainesville, FL 32611, USA*

Received July 2005 and accepted February 2006

We present a Lagrangean-based decomposition that is used to generate solutions for an integrated production and transportation planning problem in a two-stage supply chain. This supply chain consists of a number of facilities, each capable of producing the final products, and a number of retailers. It is assumed that the retailers' demands are known and deterministic, and that there are production capacity constraints. The problem is formulated as a multi-commodity network flow problem with fixed charge costs which is a NP-hard problem. An alternative formulation is provided whose linear programming relaxation gives tighter lower bounds. The quality of the lower and upper bounds from the Lagrangean decomposition is tested on a large set of randomly generated problems.

Keywords: Supply chain, production planning, fixed-charge cost, multi-commodity network flow, Lagrangean decomposition

1. Introduction

Increasingly more demanding consumers and increased competition have resulted in individual companies becoming part of large and complex supply chains. Managing the production, inventory and transportation of commodities in complex supply chains is a challenging problem. In particular, high-tech industries, such as the electronics, semiconductor and telecommunications industries, which have capital intensive equipment and a short technology life cycle, struggle with the production and transportation planning problem in their complex and rapid changing environment (Wu and Golbasi, 2004). The difficulty of the combined problem is one of the reasons why for many years practitioners and academics have looked at these problems separately.

The supply chain we analyze is a two-stage, dynamic supply chain that consists of a number of facilities, which can be viewed as a plant with an associated warehouse, and a set of retailers. The goal is to find a minimum-cost production and transportation schedule. We propose a network flow formulation and use a Lagrangean decomposition procedure to provide solutions for this integrated production and transportation planning problem.

A special case of this problem is the multi-commodity lot sizing problem. A great deal of work has been devoted to the multi-commodity lot sizing problem since it is the core of "aggregated production planning" models. Solutions to lot sizing problems are often inputs to "master production schedules" and subsequently, to "materials requirements planning" systems in a "push" type manufacturing environment (Nahmias, 1997). Chen and Thizy (1990) show that the multi-commodity lot sizing problem is strongly NP-hard. Since the multi-commodity lot sizing problem is a special case of our problem, this means that our problem is also NP-hard.

Eppen and Martin (1987) provide a shortest-path formulation for the multi-commodity lot sizing problem. They show that the Linear Programming (LP) relaxation of the shortest-path formulation is very effective in generating lower bounds and that the bounds are equal to those that could be generated using Lagrangean relaxation or column generation methods. Most of the promising heuristic approaches to solve the multi-commodity lot sizing problem seem to be based on Lagrangean relaxation. Thizy and van Wassenhove (1985) and Trigeiro *et al.* (1989) propose a Lagrangean relaxation of the capacity constraints. They update the Lagrangean multipliers using a subgradient approach and propose a heuristic to obtain feasible solutions. Chen and Thizy (1990) analyze and compare the quality of different lower bounds using relaxation methods such

*Corresponding author

as Lagrangean relaxations with respect to different sets of constraints and the LP relaxation. Stadtler (1996) introduces novel formulations of the multi-commodity lot sizing problem and compares their suitability when used to solve problems with different characteristics.

The problem we study is related to the ones studied by Chandra and Fisher (1994), Wu and Golbasi (2004) and Ekşioğlu *et al*. (2005). Chandra and Fisher (1994) were among the first to study integrated production-distribution planning problems. They investigate the effect of coordinating production and distribution on a single-plant, multi-commodity, multi-period scenario. Their computational study shows that the coordinated approach can yield up to 20% in cost savings. Ekşioğlu *et al*. (2005) analyze the single-commodity case of the problem studied in this paper. They propose a primal-dual algorithm that gives tight lower and upper bounds. Wu and Golbasi (2004) study a similar multi-facility, single-retailer, multi-commodity, multi-period production and transportation planning problem. Their production model, unlike ours, considers a bill of materials for each end-item. They use a Lagrangean decomposition procedure to solve the problem.

2. Problem formulation

The problem studied here involves the coordination of production, inventory and transportation decisions of K commodities over a planning horizon of T periods. There are F facilities that produce and deliver the final product to R retailers. The facilities have identical production capabilities in the sense that each product can be produced in each facility. However, the unit production, transportation and inventory holding costs differ between facilities. The proposed models can be used for strategic and tactical decisions. Therefore, the time period we refer to in our model could be as long as a month or more. For this reason, it is assumed that the cost parameters vary between periods. Commodities share a common production resource with item-specific setup costs. The goal is to decide on a production schedule for each commodity, so that the sum of the production, transportation and inventory costs in all of the facilities is minimized, demand is satisfied and bundle capacity constraints are not violated. It is assumed that the initial and ending inventories are zero for all products, all costs are non-negative, and backlogging is not allowed. If the initial or ending inventories are not equal to zero then they can be set to zero by adjusting the demands for either the first or last period respectively. Also, the backlogging assumption can be easily relaxed by adding new arcs to the network.

The resulting problem is formulated as a network flow problem. The structure of this network is illustrated in Fig. 1 for a single-commodity problem. The network consists of a source node, T layers, and F facilities and R retailers in each layer. The arcs connecting each facility to each retailer represent the transportation network. The source node s supplies the total demand, and the arcs connecting facilities in one time period to the same facilities in the next time period are the inventory arcs.

The following notation is used for problem formulation:

Problem data

p_{itk} = unit production cost for commodity k at facility i in period t;
s_{itk} = production set-up cost for commodity k at facility i in period t;
h_{itk} = inventory holding cost for commodity k at facility i in period t;

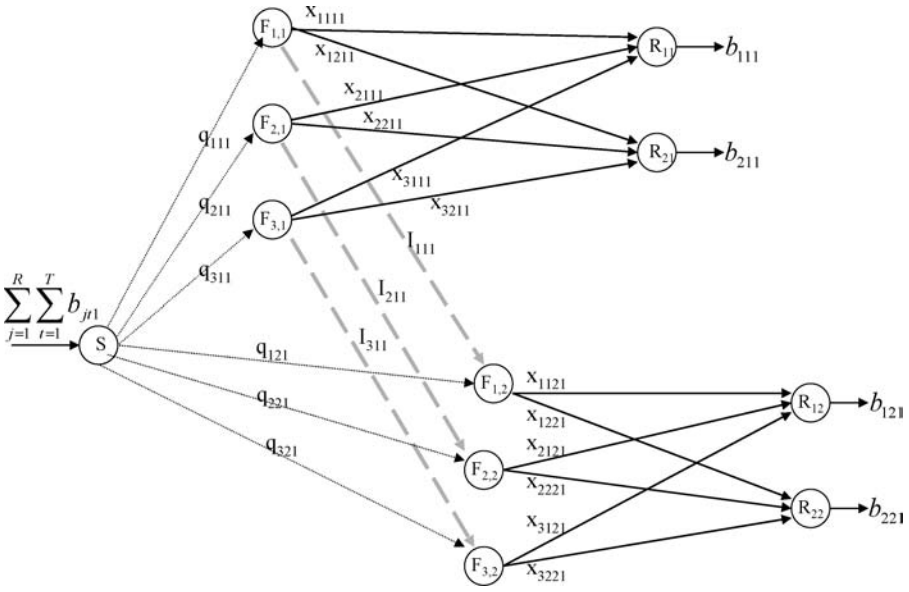

Fig. 1. Network representation of a single-commodity, two-period, two-retailer, three-facility problem.

c_{ijtk} = unit transportation cost for commodity k from facility i to retailer j in period t;
b_{jtk} = demand of retailer j for commodity k in period t;
v_{it} = production (bundle) capacity at facility i in period t.

Decision variables

q_{itk} = production quantity of commodity k at facility i in period t;
x_{ijtk} = amount of commodity k transported from facility i to retailer j in period t;
I_{itk} = amount of commodity k in inventory at facility i at the end of period t;
y_{itk} = a binary variable that is equal to one if production of commodity k occurs at facility i in period t, and is equal to zero otherwise.

The following is a Mixed-Integer Linear Programming (MILP) formulation of the problem:
(P):

$$\min \sum_{i=1}^{F}\sum_{t=1}^{T}\sum_{k=1}^{K}\left\{s_{itk}y_{itk}+p_{itk}q_{itk}+h_{itk}I_{itk}+\sum_{j=1}^{R}c_{ijtk}x_{ijtk}\right\},$$

subject to

$$I_{i,t-1,k}+q_{itk}=\sum_{j=1}^{R}x_{ijtk}+I_{itk},\quad i=1,\ldots,F,\ t=1,\ldots,T,\ k=1,\ldots,K, \tag{1}$$

$$\sum_{i=1}^{F}x_{ijtk}=b_{jtk},\quad j=1,\ldots,R,\ t=1,\ldots,T,\ k=1,\ldots,K, \tag{2}$$

$$\sum_{k=1}^{K}q_{itk}\le v_{it},\quad i=1,\ldots,F,\ t=1,\ldots,T, \tag{3}$$

$$q_{itk}\le\sum_{j=1}^{R}b_{jtTk}y_{itk},\quad i=1,\ldots,F,\ t=1,\ldots,T,\ k=1,\ldots,K, \tag{4}$$

$$y_{itk}\in\{0,1\},\quad i=1,\ldots,F,\ t=1,\ldots,T,\ k=1,\ldots,K, \tag{5}$$

where, $b_{jtTk}=\sum_{\tau=t}^{T}b_{j\tau k}$. Constraints (1) and (2) are the flow conservation constraints at the production and demand points, respectively. Constraint (3) is the production capacity constraint that reflects the multi-commodity nature of the problem. If it is relaxed then the problem can be decomposed into K single-commodity problems. Constraint (4) is the setup constraint and constraint (5) is the binary constraint. Standard solvers such as CPLEX can be used to solve formulation (P).

The above production and transportation planning problem is a generalization of the classical multi-commodity lot sizing problem (Eppen and Martin, 1987). When $F=1$, all demands are satisfied from the same facility. This is why we refer to problem (P) as the multi-facility lot sizing problem for the case where the demand can be satisfied through different facilities. In other words, we add a new dimension (facility selection) to the classical problem. The problem not only identifies the amount of each commodity to be produced and shipped every period to satisfy demand, but it also identifies the facility to be used to satisfy the demand.

The special case of problem (P), when there is a single commodity and a single retailer, has been studied by Wu and Golbasi (2004) and Ekşioğlu *et al.* (2005). Wu and Golbasi (2004) propose a shortest-path algorithm to solve the problem. Ekşioğlu *et al.* (2005) propose a primal-dual algorithm. They extend their algorithm to solve the single commodity and multi-retailer problem. The running time of the primal-dual algorithm is $O(FRT^2)$.

3. Extended problem formulation

In this section a new formulation for problem (P) is given that is referred to as the extended problem formulation. The LP relaxation of the extended formulation is used to generate lower bounds for problem (P). In the LP relaxation of problem (P), variable y_{itk} shows the fraction of demand for commodity k during periods t to T that is satisfied by production at facility i in period t (constraint (4)). Since the production in a period (q_{itk}) rarely equals $\sum_{j=1}^{R}b_{jtTk}$, the LP relaxation of problem (P) does not provide tight lower bounds. One way to tighten the formulation is to split the production variables q_{itk}, by destination, into variables $q_{ijt\tau k}$ ($\tau=t,\ldots,T$), where τ denotes the period for which production takes place. For the new variables, a trivial and tight upper bound is the demand for commodity k in period τ of retailer j (i.e., $b_{j\tau k}$). The split of the production variables leads to the following identities:

$$q_{itk}=\sum_{j=1}^{R}\sum_{\tau=t}^{T}q_{ijt\tau k},\quad x_{ijtk}=\sum_{s=1}^{t}q_{ijstk},$$
$$I_{itk}=\sum_{j=1}^{R}\sum_{s=1}^{t}\sum_{\tau=t+1}^{T}q_{ijs\tau k}, \tag{6}$$

where $q_{ijt\tau k}$ denotes the production quantity of commodity k at facility i in period t to satisfy the demand at retailer j in period τ. The extended formulation of problem (P) is as follows:
(Q):

$$\min \sum_{i=1}^{F}\sum_{t=1}^{T}\sum_{k=1}^{K}\left(s_{itk}y_{itk}+\sum_{j=1}^{R}\sum_{\tau=t}^{T}\bar{c}_{ijt\tau k}q_{ijt\tau k}\right),$$

subject to

$$\sum_{i=1}^{F}\sum_{t=1}^{\tau}q_{ijt\tau k}=b_{j\tau k},\quad j=1,\ldots,R,\ \tau=1,\ldots,T,\ k=1,\ldots,K, \tag{7}$$

$$\sum_{j=1}^{R}\sum_{k=1}^{K}\sum_{\tau=t}^{T} q_{ijt\tau k} \le v_{it}, \quad i = 1,\ldots,F, t = 1,\ldots,T, \tag{8}$$

$$q_{ijt\tau k} - b_{j\tau k} y_{itk} \le 0, \quad i = 1,\ldots,F, j = 1,\ldots,R, 0 \le t \le \tau \le T, \quad k = 1,\ldots,K, \tag{9}$$

$$q_{ijt\tau k} \ge 0, \quad i = 1,\ldots,F, j = 1,\ldots,R, \quad 0 \le t \le \tau \le T, k = 1,\ldots,K, \tag{10}$$

$$y_{itk} \in \{0,1\}, \quad i = 1,\ldots,F, t = 1,\ldots,T, k = 1,\ldots,K, \tag{11}$$

where $\bar{c}_{ijt\tau k} = p_{itk} + c_{ijt\tau k} + \sum_{s=t+1}^{\tau} h_{isk}$.

Formulation (Q) resembles the model for the uncapacitated facility location problem where $R \times T \times K$ customers can be served by $F \times T \times K$ facilities. In the facility location problem (our problem) the decision to be made is whether or not to open (produce at) facility itk. The cost of opening a facility (setting up the machines) is denoted by s_{itk}. The unit cost of serving customer $j\tau$ (satisfying demand for retailer j in period τ) is $\bar{c}_{ijt\tau k}$.

In the special case when $K = 1$ and $F = 1$, the LP relaxation of the extended formulation has an integer solution. The proof is given by Krarup and Bilde (1977). Although the same result is not true for problems with multiple facilities ($F > 1$), the lower bounds generated solving the LP relaxation of (Q) are tight.

Proposition 1. *The optimal cost of the LP relaxation of the extended formulation of the multi-commodity and multi-facility lot sizing problem* (*Q*) *is at least as high as the optimal cost of the LP relaxation of the original formulation* (*P*).

Proof. Every feasible solution to the LP relaxation of the extended formulation (Q) can be transformed to a solution to the LP relaxation of formulation (P) using Equation (6). Thus, an optimal solution for (Q) can be transformed to a feasible solution for (P). ■

4. Lagrangean decomposition heuristic

We apply the Lagrangean decomposition heuristic on the extended problem formulation (Q). The basic idea of our decomposition is to separate the multi-commodity problem into subproblems that are computationally easier to solve than the original problem. There are many ways one can do that; however, we aim to decompose the problem so that it also has interesting managerial implications.

By introducing new continuous variables $z_{ijt\tau k}$ that are simply "copies" of the production variables $q_{ijt\tau k}$ some of the constraints in formulation (*Q*) can be duplicated which leads to the following equivalent formulation:

$$\min \sum_{i=1}^{F}\sum_{t=1}^{T}\sum_{k=1}^{K}\left[s_{itk} y_{itk} + \sum_{j=1}^{R}\sum_{\tau=t}^{T} \bar{c}_{ijt\tau k} q_{ijt\tau k}\right],$$

subject to

Equations (7), (9), (10), and (11),

$$q_{ijt\tau k} - z_{ijt\tau k} = 0, \quad i = 1,\ldots,F, j = 1,\ldots,R, 0 \le t \le \tau \le T, \quad k = 1,\ldots,K, \tag{12}$$

$$\sum_{i=1}^{F}\sum_{t=1}^{\tau} z_{ijt\tau k} = b_{j\tau k}, \quad j = 1,\ldots,R, \tau = 1,\ldots,T, \quad k = 1,\ldots,K, \tag{13}$$

$$\sum_{j=1}^{R}\sum_{k=1}^{K}\sum_{\tau=t}^{T} z_{ijt\tau k} \le v_{it}, \quad i = 1,\ldots,F, t = 1,\ldots,T, \tag{14}$$

$$z_{ijt\tau k} \ge 0, \quad i = 1,\ldots,F, j = 1,\ldots,R, 0 \le t \le \tau \le T, \quad k = 1,\ldots,K. \tag{15}$$

Relaxing the "copy" constraints of Equation (12) and moving them to the objective function yields the following Lagrangean decomposition problem:
(LD (x, z, λ)):

$$\min \sum_{i=1}^{F}\sum_{t=1}^{T}\sum_{k=1}^{K}\left[s_{itk} y_{itk} + \sum_{j=1}^{R}\sum_{\tau=t}^{T}\{(\bar{c}_{ijt\tau k} + \lambda_{ijt\tau k}) q_{ijt\tau k} - \lambda_{ijt\tau k} z_{ijt\tau k}\}\right],$$

subject to

Equations (7), (9), (10), (11), (13), (14) and (15).

(LD(x, z, λ)) can now be separated into the following two subproblems:
(SP_1):

$$\min \sum_{i=1}^{F}\sum_{t=1}^{T}\sum_{k=1}^{K}\left[s_{itk} y_{itk} + \sum_{j=1}^{R}\sum_{\tau=t}^{T}(\bar{c}_{ijt\tau k} + \lambda_{ijt\tau k}) q_{ijt\tau k}\right],$$

subject to

Equations (7), (9), (10) and (11),

and
(SP_2):

$$\min \sum_{i=1}^{F}\sum_{j=1}^{R}\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{\tau=t}^{T} -\lambda_{ijt\tau k} z_{ijt\tau k},$$

subject to

Equations (13), (14) and (15).

The Lagrangean dual is:

(LD):

$$\max_{\lambda} v(\mathrm{LD}(x, z, \lambda))$$

Note that the constraints of Equation (7) are duplicated and added to (SP_2) as constraint (13). Constraint duplication improves the speed of the convergence (Guignard and Kim, 1987).

Subproblem (SP_1) can be further decomposed by commodity into k sub-subproblems (SP_{1k}). These sub-subproblems have a special structure and are relatively easier to solve. A detailed analysis of (SP_{1k}) can be found in Ekşioğlu *et al*. (2005) who propose a primal-dual algorithm that gives high-quality lower and upper bounds. In Section 4.1.1 we propose a dynamic programming algorithm to solve (SP_{1k}) and in Section 4.1.2, for the sake of completeness, we briefly describe the main steps of the primal-dual algorithm. On the other hand, subproblem (SP_2) is a linear program, and it is solved using CPLEX callable libraries. The solution of (SP_2) is feasible for problem (Q). However, since the set-up costs are not considered in the formulation, we use a simple procedure to calculate an upper bound (see Fig. 2).

Proposition 2. *Lagrangean decomposition of* (*Q*) *gives lower bounds that are at least as high as the objective function value of the corresponding LP relaxation.*

Proof. The Lagrangean decomposition procedure decomposes the problem into subproblems (SP_1) and (SP_2). Subproblem (SP_1) does not have the integrality property. Therefore, the objective function value we get by solving (LD) will be at least as high as the objective function value of the LP relaxation of (Q). ■

Proposition 3. *Lagrangean decomposition of* (*Q*) *gives lower bounds that are equal to the bounds from Lagrangean relaxation of* (*Q*) *with respect to the capacity constraints of Equation* (8).

Proof. Let Φ_1 be the set of all feasible solutions corresponding to the LP relaxation of (SP_1) and Φ_3 be the set of solutions satisfying the constraints of Equations (8) and (10). Let Φ^* be the intersection of Φ_3 with the convex hull of Φ_1 (co$\{\Phi_1\}$). Solving the Lagrangean relaxation with respect to the capacity constraints of Equation (8) is equivalent to minimizing the objective function of (Q) over Φ^* (Geoffrion 1974). On the other hand, solving the Lagrangean decomposition of (Q) is equivalent to minimizing the objective function over the intersection of $\Phi_2 = \Phi_3 \cap$ Equation (7) with the co(Φ_1) (Proposition 2). Note that $\Phi_2 \cap \mathrm{co}\{\Phi_1\} = \Phi^* \cap$ Equation (7) $= \Phi^*$. ■

The lower bounds generated from the Lagrangean decomposition heuristic are the same as the bounds from Lagrangean relaxation with respect to the capacity constraints. However, we implement the Lagrangean decomposition approach because the decomposition scheme provides feasible solutions to (Q) at every iteration.

4.1. *Solving the uncapacitated single-commodity problem*

An optimal solution to the uncapacitated single-commodity subproblem (SP_{1k}) has the following properties (Ekşioğlu *et al*., 2005): (i) a facility in a time period t ($t = 2, \ldots, T$) either produces or carries inventory from period $t-1$, but not both; (ii) demand in a time period is satisfied from a single facility; (iii) every facility in a given time period either produces the demand for a number of periods (the periods do not need to be successive) or does not produce. These properties show that a production, inventory and transportation plan is optimal if and only if the corresponding arcs with positive flow form an arborescence (rooted tree) in the network. An extreme flow solution is not necessarily sequential. In a sequential extreme flow the demand in a number of successive periods is satisfied by the same facility in a given period. An example would be a solution in which demands in periods 1 and 3 are satisfied from production at facility 1 in period 1, and the demands in period 2 are satisfied from production at facility 2 in period 2.

The single-commodity subproblem (SP_{1k}) is shown to be NP-complete (Wu and Golbasi, 2004) even in the following special cases when $R = 1$: (i) transportation costs c_{it} are non-negative; (ii) transportation costs are ignored and (a) the holding costs h_{it} are not restricted in sign, or (b) the

Let z^{*s} be the optimal solution to subproblem (SP_2) in iteration s.

Initialize $\mathrm{UB}^s = 0$; $\mathrm{sum}_{itk} = 0$ for $i = 1, \ldots, F$, $t = 1, \ldots, T$, $k = 1, \ldots, K$.

for $i = 1, \ldots, F$, $t = 1, \ldots, T$, $k = 1, \ldots, K$

  for $\tau = t, \ldots, T$ **do**

    if $z^{*s}_{it\tau k} \geq 0$ **then** $UB^s = UB^s + \bar{c}_{it\tau k} z^{*s}_{it\tau k}$

    $\mathrm{sum}_{itk} = \mathrm{sum}_{itk} + z^{*s}_{it\tau k}$

  if $\mathrm{sum}_{itk} > 0$ **then** $UB^s = UB^s + s_{itk}$.

Fig. 2. Upper bound procedure.

period demands b_t are constant over periods $t = 1, \ldots, T$, or (c) the setup costs s_{it} are constant across facilities $i = 1, \ldots, F$ and over periods $t = 1, \ldots, T$.

4.1.1. *Dynamic programming algorithm*

Let Υ_{it} be the set of all periods covered by production at facility i in period t in an optimal arborescence. Then the optimal production plan is:

$$q_{it} = \sum_{\tau=t}^{T} q_{it\tau} = \sum_{j=1}^{R} \sum_{\tau \in \Upsilon_{it}} b_{j\tau}, \quad i = 1, \ldots, F, \ t = 1, \ldots, T.$$

Using this information we can now simplify the multi-facility lot sizing problem to a shortest-path problem in an acyclic network. Let G' represent this network. G' is built in the following way: let the total number of nodes in G' be equal to $(T + 1)$, one for each time period along with a dummy node $T + 1$. Traversing arc $(\tau, \tau') \in G'$ represents the choice of producing in a single facility in a single time period $t = 1, \ldots, \tau$ to satisfy the demand in periods $\tau, \tau + 1, \ldots, \tau' - 1$. The cost of arc (τ, τ') is calculated using the following cost function:

$$g_{\tau,\tau'} = \min_{i=1,\ldots,F; 1 \le t \le \tau} s_{it} + \sum_{j=1}^{R} \sum_{l=\tau}^{\tau'-1} \bar{c}_{ijtl} b_{jl}. \tag{16}$$

Let v_τ be the minimum cost of a solution for period $\tau = 1, \ldots, T - 1$ and $\tau' \ge \tau$. The recursive function for the multi-facility lot sizing problem is:

$$v_\tau \left(\sum_{i=1}^{F} I_{i,\tau-1} \right) = \min_{\tau'} \left\{ g_{\tau,\tau'} + v_{\tau'} \left(\sum_{i=1}^{F} I_{i,\tau'-1} \right) \right\}$$

$$\text{and} \quad v_T \left(\sum_{i=1}^{F} I_{i,T-1} \right) = g_{T,T+1}. \tag{17}$$

The total number of arcs in G' is equal to $T(T + 1)/2$. Given the costs $g_{\tau,\tau'}$ for every arc $(\tau, \tau') \in G'$, the recursive functions of Equation (17) will provide the optimal solution in $O(T^2)$. The running time of the dynamic programming algorithm is $O(FRT^4)$. The network G' for a four-period problem is presented in Fig. 3.

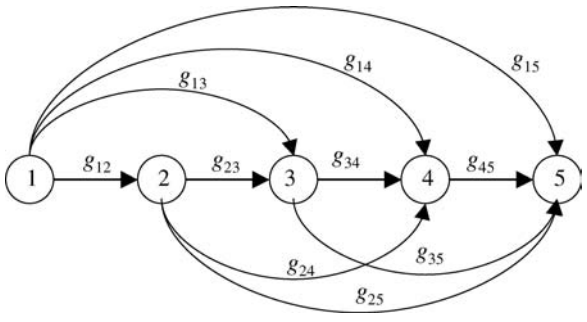

Fig. 3. Network representation of the multi-facility lot sizing problem with $T = 4$.

Theorem 1. *For every path on G' that connects node 1 to $T + 1$, there exists a corresponding extreme point solution in (Q), and every sequential extreme point of (Q) is represented by a path on G'.*

Proof. First, we show that for every path in G' that connects node 1 to node $T + 1$, there is a corresponding extreme point solution in (Q). Let Π be the set of all paths in G' that connect node 1 to $T + 1$, and $p \in \Pi$. Since p is a path in G', there will be exactly one arc in p starting at node 1 and exactly one arc ending at node $T + 1$. Thus, demand in periods 1 and T will be satisfied. Let $(\tau, \tau') \in p$. By construction we know that an arc $(\tau, \tau') \in p$ implies that demand in periods $\tau, \tau + 1, \ldots, \tau' - 1$ is satisfied from production in a single facility in a single time period t $(t = 1, \ldots, \tau)$. Since $(\tau, \tau') \in p$, then there will be no other arc in the path that starts at one of the nodes $\tau, \tau + 1, \ldots, \tau' - 1$ and no other arc that ends at nodes $\tau + 1, \ldots, \tau'$. This implies that demand in periods $\tau, \tau + 1, \ldots, \tau'$ is satisfied from a single facility in a single time period prior to or beginning at τ.

By definition, there is exactly one arc entering and exactly one arc leaving every node in a path. Therefore, there will be exactly one arc entering node τ and exactly one arc leaving τ', which implies that demands in time periods before τ and after τ' are satisfied by production in only one period at exactly one facility. We conclude that every path in G' that connects nodes 1 to $T + 1$ represents a production, inventory and transportation schedule that satisfies demand in every period. This schedule is such that demands are satisfied from production in a single facility in a single time period. Such a schedule fits the requirements of an extreme point solution of (Q).

Next, we show that for every sequential extreme point solution of (Q) there is a corresponding path in G' that connects node 1 with $T + 1$. We define a sequential extreme flow to be such that: if facility i produces in time period t, then production satisfies demands (of all retailers) in a sequence of time periods $\tau, \ldots, \tau'$, where $\tau \ge t$ and $\tau' \le T$. Consider the following cases:

Case 1. Demands in periods 1 to T are satisfied from production in different time periods. This case can be generalized to all the tree solutions that are such that the demand in every period is satisfied from different combinations of facilities and production periods. These solutions are represented by a path $p \in G'$ such that all the nodes are part of the path (p consists of nodes $1, \ldots, T + 1$ and arcs $(1, 2), (2, 3), \ldots, (T, T + 1)$).

Case 2. Demands in periods $\tau, \ldots, \tau'$ are satisfied from production in the same facility in the same time period. All extreme flow solutions that satisfy this condition have a single arc $(\tau, \tau') \in G'$ and no other arc that enters nodes $\tau + 1, \ldots, \tau'$ or leaves nodes $\tau, \ldots, \tau' - 1$.

Case 3. Demands in periods $\tau, \ldots, \tau'$ are satisfied from production in different facilities in the same time period or from production in the same facility in different time periods. All extreme flow solutions in (Q) that satisfy this condition are represented by a sequence of arcs in G' such that there will be an arc entering node $\tau \in G'$ followed by a sequence of arcs $(\tau, \tau+1), (\tau+1, \tau+2), \ldots, (\tau'-1, \tau')$.

This shows that every sequential extreme flow solution to (Q) is represented by a path in G'.

Based on Theorem 1, the dynamic programming algorithm gives the optimal solution to subproblem (SP_{1k}) if the optimal solution is sequential and therefore satisfies the following conditions: (i) no simultaneous production over more than one facility can take place in a given period i.e., $q_{it} \times q_{lt} = 0$ for $i, l = 1, \ldots, F, i \neq l; t = 1, \ldots, T$; (ii) no production will be scheduled at all if there is inventory carried over from a previous period in one of the facilities i.e., $q_{it} \times I_{l,t-1} = 0$ for $i, l = 1, \ldots, F, i \neq l, t = 1, \ldots, T$.

4.1.2. *Primal-dual algorithm*

The dual of the LP relaxation of the extended formulation of (SP_{1k}) has a simple structure. This special structure can be exploited to obtain near-optimal feasible solutions and lower bounds by inspection.

A dual feasible solution is obtained by calculating the value of the dual variables $v_{j\tau}$ ($j = 1, \ldots, R, \tau = 1, \ldots, T$) sequentially using the following equation:

$$v_{j\tau} = \min_{i=1,\ldots,F;\tau \geq t} \left\{ \bar{c}_{ijt\tau} + \frac{M_{ijt,\tau-1}}{b_{j\tau}} \right\}, \tag{18}$$

where, $M_{ijt,\tau-1} = s_{it} - \sum_{l=1}^{R} \sum_{s=t}^{\tau-1} b_{ls} \max(0, v_{ls} - \bar{c}_{ilts}) - \sum_{l=1}^{j-1} b_{l\tau} \max(0, v_{l\tau} - \bar{c}_{ilt\tau})$.

A backward construction algorithm is then used to generate primal feasible solutions. The algorithm sets $q_{ijt\tau} = b_{j\tau}$ when $\bar{c}_{ijt\tau} - v_{j\tau} - \max(0, v_{j\tau} - \bar{c}_{ijt\tau}) = 0$ and sets $y_{it} = 1$ if $s_{it} = \sum_{j=1}^{R} \sum_{\tau=t}^{T} b_{j\tau} \max(0, v_{j\tau} - \bar{c}_{ijt\tau})$.

4.2. *Subgradient optimization algorithm*

Subgradient optimization (Nemhauser and Wolsey, 1988) is an iterative method in which steps are taken along the negative of the subgradient of the Lagrangean function $z(\lambda)$ ($z(\lambda) = v(LD(x, z, \lambda))$. At each iteration s, we calculate the Lagrangean multipliers $\lambda_{it\tau k}$ using the following equation:

$$\lambda_{it\tau k}^{s+1} = \lambda_{it\tau k}^{s} + u^s (q_{it\tau k} - z_{it\tau k}), \tag{19}$$

where

$$u^s = \frac{\gamma^s (\min_{UB} - \max_{LB})}{\sum_{i=1}^{F} \sum_{t=1}^{T} \sum_{\tau=t}^{T} \sum_{k=1}^{K} (q_{it\tau k} - z_{it\tau k})^2},$$

$\max_{LB}$ ($\min_{UB}$) is the best lower (upper) bound found up to iteration s, and $\gamma^s \in (0, 2]$. γ^s is reduced if the lower bound fails to improve after a fixed number of iterations. The algorithm is terminated if one of the following happens: (i) the best lower bound is equal to the best upper bound (the optimal solution is found); (ii) the number of iterations reaches a prespecified bound; or (iii) the scalar γ^s is less than or equal to ϵ (a predefined number close to zero). Figure 4 presents the steps of the Lagrangean decomposition algorithm. This subgradient optimization algorithm is one of many subgradient "recipes" proposed over the years. For a short recent account see Crainic *et al.* (2001).

Step 1. Initialize λ, $\min_{UB}$, $\max_{LB}$, s, u, γ, ϵ, *count*.

Step 2. Solve the subproblems (SP_1) and (SP_2). Compute the lower bound:

$$LB^s = \sum_{k=1}^{K} \omega(SP_{1k}(\lambda^s)) + v(SP_2(\lambda^s))$$

for the current iteration s

count = count + 1

If $LB^s > \max_{LB}$, **then** $\max_{LB} = LB^s$

Step 3. Compute an upper bound UB^s (see Fig. 2)

If $UB^s < \min_{UB}$, **then** $\min_{UB} = UB^s$

If $\min_{UB} = \max_{LB}$, **then STOP**

If $\gamma \leq \epsilon$, **then STOP**

Step 4. Update the multipliers using Equation (19)

Step 5. Stop if the number of iterations reach the prespecified limit (*count*)

Otherwise go to **Step 2**

Fig. 4. Lagrangean decomposition algorithm.

4.3. *Managerial interpretation of the decomposition*

The choice of the Lagrangean decomposition scheme we develop is motivated not only by its computational capability but also by its interesting managerial implications. Several studies (including those of Burton and Obel (1984) and Jörnsten and Leisten (1994)) have recognized that mathematical decomposition often leads to insights for general modeling strategies and new decision structures. In this discussion we refer to subproblem (SP_1) as the product (commodity) subproblem and (SP_2) as the resource subproblem.

The proposed decomposition helps in understanding and solving managerial issues that arise in multi-facility manufacturing planning. The resource subproblem can be viewed as a decision problem for a production manager who supervises multiple facilities, and each product subproblem can be viewed as a decision problem for a product manager. Therefore, the decomposition can be viewed as a decision system where product managers compete for the resource capacity available from manufacturing facilities. The production manager, on the other side, is interested in efficiently allocating resources from multiple facilities to the products. Often the solutions proposed by the production manager will disagree with the individual solutions of product managers. A search based on the Lagrangean multipliers basically penalizes their differences, while adjusting the penalty vector iteratively. This process stops when the degree of disagreement (the duality gap) is acceptably low or when further improvement is unlikely. Wu and Golbasi (2004) present a similar discussion on a related problem.

5. Computational results

The objective of our computational analysis is to test the performance of the Lagrangean decomposition algorithm and identify factors that affect its performance. For this purpose, two sets (small and large sized) of random problems are generated. The algorithms were written in the C-programming language and were then compiled and executed on a HP personal computer with a 2.0 GHz Intel Pentium 4 processor and a 512 MB RAM.

The tightness of the upper bounds on production arcs is one of the factors that affects the problem complexity. If the arc capacities are very tight then there are only a few feasible solutions, which makes the search for the optimal solution easy. On the other hand, if the arc capacities are so loose that the capacity constraints can be removed from the formulation then the problem loses its multi-commodity nature. In this case the problem can be decomposed into K single-commodity problems. In order to create challenging test problems the upper bounds were calculated in the following way. A necessary condition for feasibility of the problem is:

$$\sum_{i=1}^{F}\sum_{\tau=1}^{t} v_{i\tau} \geq \sum_{j=1}^{R}\sum_{\tau=1}^{t}\sum_{k=1}^{K} b_{j\tau k}, \quad \forall t = 1, \ldots, T. \tag{20}$$

Under the assumption that all facilities in any time period have the same production capacity $\bar{v}$:

$$\sum_{i=1}^{F}\sum_{\tau=1}^{t} v_{i\tau} = Ft\bar{v},$$

thus

$$\bar{v} \geq \frac{1}{Ft}\sum_{j=1}^{R}\sum_{\tau=1}^{t}\sum_{k=1}^{K} b_{j\tau k}, \quad \forall t = 1, \ldots, T.$$

In fact $\bar{v}$ should be such that the following is satisfied:

$$\bar{v} = \frac{\delta}{F}\max_{t}\frac{\sum_{j=1}^{R}\sum_{\tau=1}^{t}\sum_{k=1}^{K} b_{j\tau k}}{t}.$$

Where $\delta(\geq 1)$ is the capacity tightness coefficient. Using this procedure we generated challenging (but feasible) test problems with tight capacity constraints.

5.1. *Small-sized problems*

Table 1 presents the characteristics of the small-sized problems. The supply chain for these problems has three facilities, six retailers and seven periods. There are three commodities produced and shipped to customers. The tightness coefficient for the production arcs is 1.3. Production variable costs and holding costs are uniformly distributed in [5, 15]. To generate meaningful transportation variable costs, we uniformly generated points on a [0, 10] square corresponding to the facilities and retailers. The Euclidean distances between the facilities and the retailers were used as unit transportation costs.

The small-sized problems have the same number of variables. Problems differ from one another in terms of the value of the set-up costs and demands. Demand and setup costs are uniformly distributed. For each problem 20 instances were generated. The presented results are averages over the 20 instances. The above parameters are the same as those used in Chen *et al*. (1994), Wu and Golbasi (2004) and Ekşioğlu *et al*. (2005) for related problems. In implementing the Lagrangean decomposition algorithm, for all problem instances $\gamma = 1.8$ initially. γ was reduced by 20% if there was no improvement in the last five iterations. The Lagrangean decomposition algorithm stops after 100 iterations.

Table 1. Characteristics of the small-sized problems

	Demand			
Set-up costs	*100–200*	*50–200*	*50–100*	*40–60*
200–300	Problem 1	Problem 6	Problem 11	Problem 16
200–900	Problem 2	Problem 7	Problem 12	Problem 17
600–900	Problem 3	Problem 8	Problem 13	Problem 18
900–1200	Problem 4	Problem 9	Problem 14	Problem 19
1200–1500	Problem 5	Problem 10	Problem 15	Problem 20

Table 2. Duality gap (%) for small-sized, single-commodity problems

Problem	*Primal*	*Dynamic prog.*	*Problem*	*Primal*	*Dynamic prog.*
1	0.30	2.42	11	0.96	1.05
2	1.00	2.09	12	2.12	1.58
3	1.87	1.73	13	5.66	1.48
4	3.27	1.14	14	6.89	0.68
5	4.53	1.26	15	8.71	0.80
6	0.58	2.05	16	1.98	1.81
7	1.06	1.51	17	3.74	1.11
8	2.95	1.55	18	5.50	0.97
9	4.06	1.25	19	9.42	0.73
10	5.89	0.99	20	10.34	0.70

Table 4. Quality of upper bounds (% gap) for small-sized problems

Problem	*Sch 1*	*Sch 2*	*Sch 3*	*Problem*	*Sch 1*	*Sch 2*	*Sch 3*
1	0.13	0.13	0.28	11	0.37	0.35	0.89
2	0.47	0.48	1.02	12	1.20	1.14	2.10*
3	0.60	0.58	1.80	13	2.08	2.04	2.39*
4	1.41	1.47	2.03*	14	3.56	3.51	3.65*
5	1.51	1.58	1.91*	15	4.55	4.53	4.91*
6	0.20	0.22	0.47	16	0.51	0.48	1.25
7	0.55	0.51	1.43	17	1.39	1.43	2.14*
8	1.12	1.08	2.33*	18	2.91	2.72	2.81*
9	2.13	2.08	2.43*	19	4.74	4.80	4.83*
10	2.10	1.96	2.48*	20	5.18	5.30	5.38*

*Refers to problems where subproblems (SP_{1k}) are solved using the dynamic programming algorithm.

For these problems, the quality of the lower and upper bounds generated using the Lagrangean decomposition algorithm were compared to the optimal solutions (these results will be presented in Tables 3 and 4). The optimal solutions were found by solving the MILP formulation (Q) using CPLEX 9.0 callable libraries. The gap is measured as follows:

$$Gap\ (\%) = \frac{CPLEX\ -\ Lagrangean}{CPLEX} \times 100.$$

Subproblem (SP_{1k}) can be solved to optimality using CPLEX. However, we tested the effectiveness of the primal-dual and dynamic programming algorithms as heuristics in the context of the subgradient search algorithm. The potential computational savings from using these heuristics (instead of using CPLEX) is significant since the subgradient search requires solving K single-commodity subproblems in each iteration. If a problem instance considers ten commodities for example, this results in up to 1000 calls to the subproblem (note that the maximum number of iterations is 100).

Three schemes were developed to run the Lagrangean decomposition algorithm: (i) subproblem (SP_{1k}) for $k = 1, \ldots, K$) was solved optimally using CPLEX (scheme 1); (ii) a lower bound was generated for subproblem (SP_{1k}) by solving the LP relaxation of the corresponding extended formulation using CPLEX (scheme 2); and (iii) a lower bound was generated for subproblem (SP_{1k}) using the dual of the primal-dual algorithm (scheme 3).

Note that, in order to find a subgradient direction at each step of the Lagrangean decomposition procedure, we need to find a feasible solution to subproblem (SP_1). Feasible solutions are found using the primal of the primal-dual algorithm and the dynamic programming algorithm.

In order to compare the performance of these algorithms, problems 1 to 20 were solved for the case when the number of commodities is equal to one and there are no bundle capacities. The results presented in Table 2 show that for problems with large set-up costs and small demands, the dynamic programming algorithm performs better.

Because the LP relaxation of the extended formulation gives near-optimal solutions, the difference in the quality of the lower bounds generated using schemes 1 and 2 was no more than 0.01% (Table 3). However, the time it takes to solve the problem using scheme 2 is considerably shorter. The gap (for the lower bounds) among the three schemes is no more than 3%. Scheme 3, however, gives the shortest running times.

The quality of upper bounds from the three schemes is comparable (Table 4). The algorithms seem to give the worst performance for problems with a small demand levels and high set-up costs. The reason for such a performance is because our problem is an MILP with binary set-up variables, and as the set-up costs increase the problem behaves closer to a combinatorial problem than to a LP problem. The running time of the Lagrangean algorithm using schemes 2 and 3 were less than 1 CPU second whereas scheme 1 took up to 19 CPU seconds.

Table 3. Quality of lower bounds (% gap) for small-sized problems

Problem	*Sch 1*	*Sch 2*	*Sch 3*	*Problem*	*Sch 1*	*Sch 2*	*Sch 3*
1	0.15	0.15	0.61	11	0.34	0.34	1.32
2	0.34	0.34	1.27	12	0.76	0.77	2.35
3	0.43	0.43	1.90	13	1.09	1.09	3.37
4	0.92	0.92	2.89	14	1.26	1.27	3.30
5	0.98	0.98	3.19	15	1.49	1.49	3.75
6	0.18	0.18	0.84	16	0.42	0.43	2.01
7	0.39	0.40	1.41	17	0.88	0.88	3.16
8	0.57	0.58	2.37	18	1.27	1.28	3.83
9	1.10	1.10	3.26	19	1.84	1.85	4.49
10	1.22	1.22	3.41	20	1.62	1.64	4.34

5.2. *Large-sized problems*

The group of large-sized problems consists of 35 problems. We first created a nominal case problem and then altered the nominal case by changing one characteristic at

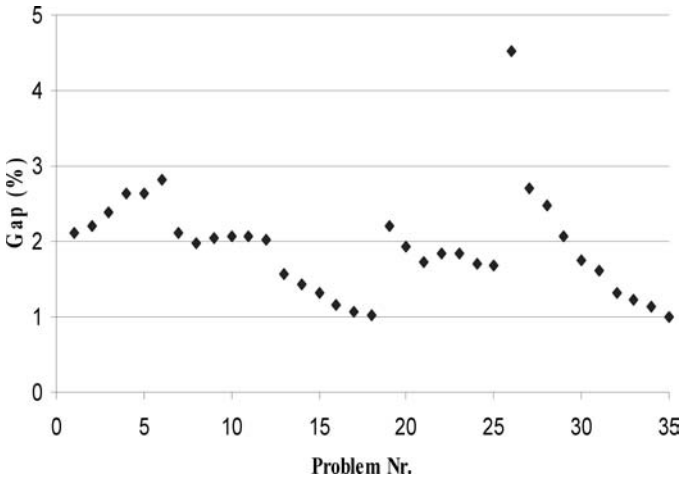

Fig. 5. Summary of results (gap in %) for large-sized problems.

a time to generate additional problems. For each problem class 20 instances were generated, and the averages over the 20 instances are presented. The nominal problem has the following characteristics: $s_{itk} \sim U[200, 300]$, $p_{itk} \sim U[5, 15], h_{itk} \sim U[5, 15], b_{jtk} \sim U[100, 200], F = 10$, $T = 10$, $K = 20$, $R = 30$, $\delta = 1.3$. First, the number of facilities was changed from ten to $11, \ldots, 15$ generating problem classes 1 to 6 (problem class 1 is the nominal case). In problem classes 7 to 12 the number of periods is changed from ten to $15, \ldots, 35$. Then, the number of commodities was changed from 20 to $25, \ldots, 50$ (problem classes 13 to 18). In problem classes 19 to 25 the number of retailers was changed from 30 to $35, \ldots, 60$. Finally, the capacity tightness coefficient was changed (δ) to 1.10, 1.20, 1.25, 1.30, 1.35, 1.40, 1.45, 1.50, 1.55 and 1.60 (problem classes 26 to 35). For the large-sized problems, we were unable to find optimal solutions because CPLEX ran out of memory. Figure 5 presents the gap between the lower and upper bounds generated using Lagrangean decomposition procedure. For this set of problems, subproblems (SP_{1k}) were solved using the primal-dual algorithm. The running times vary from 4 to 87 CPU seconds.

The results in Fig. 5 indicate that increasing the number of facilities increases the duality gap (problems 1 to 6). Adding the facility selection dimension to the classical lot sizing problem makes the problem complex. Increasing the number of commodities that flow on the network improves the quality of the lower bounds (problems 13 to 18). Increasing the number of retailers (problems 19 to 25) and the capacity tightness coefficient (problems 26 to 35) improves the quality of the lower bounds from the Lagrangean decomposition algorithm. Increasing the number of retailers, increases the total demand and as a result increases the flow in the network. This in turn, increases the total variable costs which makes the problems easier to solve. Increasing the tightness coefficient increases the available capacity and leads to easier problems.

The results in Fig. 6 indicate that increasing the length of the planning period increases the running time of the algorithm (problems 7 to 12). Also, increasing the number of commodities increases the running time of the algorithm because of the increase in the number of subproblems (SP_{1k}) to be solved at every iteration. The other factors do not seem to affect the running time of the algorithm.

6. Conclusions

In this paper we have presented an integrated production and transportation planning problem in a two-stage supply chain. The problem was modelled as a multi-commodity network flow problem with fixed-charge arc costs, and a new formulation for this problem has been presented that is referred to as the extended formulation.

A Lagrangean decomposition procedure is proposed to generate lower and upper bounds for this problem. The

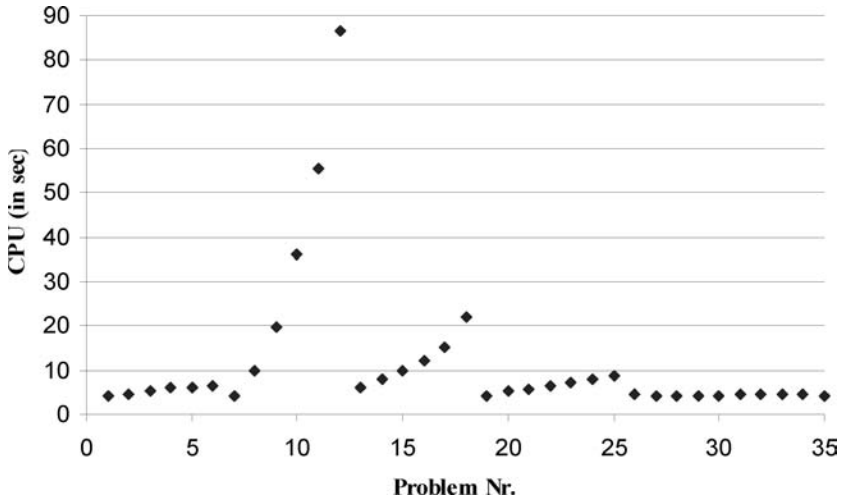

Fig. 6. Summary of running times for large-sized problems.

performance of the algorithm has been tested on a large set of randomly generated data. We present our computational experience with both small and large-sized problem sets. For the small-sized problems, a lower and an upper bound were generated using the proposed algorithm and the bounds were compared to the optimal solution (found solving the MILP formulation of the problem using CPLEX). For the large-sized problems, the gap between the lower bounds and the corresponding upper bounds are presented. CPLEX fails to solve these problems because of their size. The computational results show that the performance of the algorithm is affected by the ratio of total variable to fixed costs, demand and number of facilities. The length of the time horizon affects the running time of the algorithm.

References

Burton, R.M. and Obel, B. (1984) *Designing Efficient Organizations*, North Holland, Amsterdam, The Netherlands.

Chandra, P. and Fisher, M.L. (1994) Coordination of production and distribution planning. *European Journal of Operational Research*, **72**, 503–517.

Chen, H.-D., Hearn, D.W. and Lee, C.-L. (1994) A new dynamic programming algorithm for the single item dynamic lot size model. *Journal of Global Optimization*, **4**, 285–300.

Chen, W.H. and Thizy, J.M. (1990) Analysis of relaxations for the multi-item capacitated lotsizing problem. *Annals of Operations Research*, **26**, 29–72.

Crainic, T.G., Frangioni, A. and Gendron, B. (2006) Bundle-based relaxation methods for multi-commodity capacitated fixed charge network design. *Discrete Applied Mathematics*, **112**, 73–99.

Ekşioğlu, S.D., Romeijn, H.E. and Pardalos, P.M. (2006) Cross-facility management of production and transportation planning problem. *Computers & Operations Research*, **33**(11), 3231–3251.

Eppen, G.D. and Martin, R.K. (1987) Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, **35**(6), 832–848.

Geoffrion, A.M. (1974) Lagrangean relaxation for integer programming. *Mathematical Programming Study*, **2**, 82–114.

Guignard, M. and Kim, S. (1987) Lagrangean decomposition: a model yielding stronger Lagrangean bounds. *Mathematical Programming*, **39**, 215–228.

Jörnsten, K. and Leisten, R. (1994) Aggregation and decomposition for multi-divisional linear programs. *European Journal of Operational Research*, **72**, 175–191.

Krarup, J. and Bilde, O. (1977) Plant location, set covering, an economic lot-size: an $o(mn)$ algorithm for structured problems, in *Numerische Methoden bei Optimierungsaufgaben. Band 3: Optimierung bei Graphentheorietischen und Ganzzahlieng Problemen*, Collatz et al. (eds.). Birkhauser, Basel, Switzerland, pp. 155–186.

Nahmias, S. (1997) *Production and Operations Analysis*, 3rd edn., Irwin, New York.

Nemhauser, G.L. and Wolsey, L.A. (1988) *Integer and Combinatorial Optimization*, Wiley, New York.

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

Thizy, J.-M. and Van Wassenhove, L.N. (1985) Lagrangean relaxation for the multi-item capacitated lot-sizing problem. *IIE Transactions*, **17**, 308–313.

Trigeiro, W., Thomas, L.J. and McLain, J.O. (1989) Capacitated lotsizing with setup times. *Management Science*, **35**, 353–366.

Wu, S.D. and Golbasi, H. (2004) Multi item, multi facility supply chain planning: modeling, analysis, and algorithms. *Computational Optimization & Applications*, **28**(3), 325–356.

Biographies

Sandra Duni Ekşioğlu is an Assistant Professor in the Department of Industrial and Systems Engineering at Mississippi State University. Her research is focused on modeling, analyzing and solving large-scale logistics and dynamic supply chain management problems. She is also conducting research in other areas such as transportation scheduling, network optimization, and project management. She is a member of INFORMS, IIE, and ASEE.

Burak Ekşioğlu is an Assistant Professor in the Department of Industrial and Systems Engineering at Mississippi State University. His research interests include heuristic optimization, supply chain optimization, and analysis of large-scale transportation systems. He is a member of INFORMS, IIE, ASEE, and TBP-Engineering Honor Society.

H. Edwin Romeijn is an Associate Professor in the Department of Industrial and Systems Engineering at the University of Florida. His research focuses on the development of new models and solution techniques for optimizing the design and control of complex systems. Current activities mainly deal with systems arising in radiation therapy treatment planning, supply chain management and analysis, and financial engineering.