

**R U T C O R
R E S E A R C H
R E P O R T**

**A VARIABLE NEIGHBORHOOD
DECOMPOSITION SEARCH
METHODOLOGY
FOR PRODUCTION-DISTRIBUTION
PLANNING IN
SUPPLY CHAIN MANAGEMENT**

M.A. Lejeune^a

RRR 25-2004, JULY 2004

RUTCOR
Rutgers Center for
Operations Research
Rutgers University
640 Bartholomew Road
Piscataway, New Jersey
08854-8003
Telephone: 732-445-3804
Telefax: 732-445-5472
Email: rrr@rutcor.rutgers.edu
<http://rutcor.rutgers.edu/~rrr>

^a mlejeune@andromeda.rutgers.edu

RUTCOR - Rutgers University Center for Operations Research, Piscataway, NJ, USA
Rutgers Business School, Rutgers University, Newark-New Brunswick, NJ, USA

RUTCOR RESEARCH REPORT

RRR 25-2004, JULY 2004

A VARIABLE NEIGHBORHOOD
DECOMPOSITION SEARCH METHODOLOGY
FOR PRODUCTION-DISTRIBUTION
PLANNING IN SUPPLY CHAIN
MANAGEMENT

M.A. Lejeune

Abstract. The planning and scheduling of the inventory, production and distribution functions remains an open research area, since few studies have developed models allowing their integrated management. In this paper, we consider a three-stage supply chain, for which a sustainable inventory-production-distribution plan over a multi-period horizon is constructed. The associated program takes the form of a general mixed-integer program, for which the sole reliance upon exact methods is shown to be insufficient. We propose a solving methodology based on the variable neighborhood decomposition search metaheuristic, that can be seen as a stagewise exploration of increasingly large neighborhoods. The stages are related to the decomposition scheme, i.e., the order on which integrality conditions are restored. Within each stage, a sequence of increasingly large neighborhoods is defined relying on the variable neighborhood search metaheuristic, while the exploration of the successive neighborhoods is performed using a branch-and-bound algorithm. The methodology is validated through its application to a problem faced by a North American supply chain. Empirical results show that (i) the methodology is most performing when the decomposition scheme accounts for the possibility of resources bottleneck, (ii) the primary source of savings is the distribution function, and (iii) congestion must be defined with respect to the availability of the distribution resources at the periods with high requirements.

1 Introduction

Despite the abundant literature on the analysis and optimization of production-distribution systems, this remains a widely open research area (Sarmiento and Nagi, 1999). Indeed, a great part of the research done so far has focused on a single component of the production-distribution network (Thomas and Griffin, 1996, Vidal and Goetschalckx, 1997). Instead of integrating the production and the distribution functions, production and distribution operations are often decoupled, and handled independently of each other, with little or no coordination. This results in increased inventory costs and longer lead times through the supply chain. Fierce competition made this situation non-sustainable for companies, and led them to invest aggressively to reduce inventory levels across the supply chain and be more responsive to customers (Chen, 2003). The two latter characteristics can only be reached by integrating the production and distribution operations.

However, in the extant literature, few studies have attempted to develop models allowing the simultaneous management of the inventory, production and distribution functions. Often this type of problems requires the solving of large dimensional mixed-integer programs, which solving is very challenging and is one of the main reasons for the current research gap (Thomas and Griffin, 1996) associated with integrated production-distribution models.

Exact methods such as branch-and-bound, branch-and-cut, or branch-and-prize algorithms, are generally used for solving mixed-integer programs. Within such approaches, some of the integer variables are fixed at an integer value -- each set of fixed integer values defining a node in the branch-and-bound tree --, while the remaining integer variables are left free to take real or integer values. The resulting problem, called the root-node linear programming relaxation, is then solved. Exact methods progress from node to node to implicitly exhaust all possible combinations of integer values. Even if it is theoretically possible to enumerate all possible combinations of values that the integer variables can take and to elect this that provides the most favorable outcome as the optimal solution, this approach is actually infeasible for problems of moderate- to high-dimension. The number of combinations of values taken by the integer variables indeed increases exponentially with the size of the problem, often precluding the use of exact algorithms. This is confirmed by Fourer and Gay (2002), saying that “branch-and-bound codes still have great difficulty solving many integer programs, particularly ones that are derived from highly combinatorial problems whose formulation involve great numbers of zero-one variables”. As a consequence, exact methods are often not applicable to solve large real-life problems, which require the development of ad-hoc algorithms or heuristics.

Discussing the challenges posed by mixed-integer programs, Bixby et al. (1999) strongly insist on the necessity to combine several approaches, calling this a “barrage of different but cooperating ideas”. Very often, heuristics are used in conjunction with exact methods, and have been proven very useful in the solving of combinatorial optimization problems. Their effectiveness resides in their ability to escape from a local optimum, and to exploit the intrinsic characteristics of the model.

While a few solving techniques combining exact and heuristics methods have been proposed for general mixed-integer programs (Savelsbergh and Nemhauser, 1996), most focus on 0-1 mixed-integer programs (Balas et al., 2001, Løkketangen and Glover, 1998). From this viewpoint, a new approach has been proposed by Lodi and Fischetti (2003), who enhance a mixed-integer programming solver through the introduction of linear invalid inequalities, called local branching cuts. The local branching cuts are introduced in the formulation of the mix-integer program and limit the size of the neighborhood explored at the successive nodes processed by the mixed-integer solver. However, the general formulation of the program associated with the construction of an inventory- production-distribution plan

takes the form of a mixed-integer program containing binary as well as general integer variables, further compounding the solving complexity, and preventing the use of the aforementioned techniques.

In this manuscript, we develop a new solving methodology enabling the solving of complex mixed-integer programs associated with the construction of a sustainable inventory-production-distribution plan for a multi-stage supply chain over a multi-period planning horizon. The proposed methodology jointly relies on the branch-and-bound technique and the variable search neighborhood metaheuristics (Mladenović and Hansen, 1997), more precisely the variable neighborhood decomposition search method (Hansen and Mladenović, 2001, Hansen et al., 2001). It is based on the stagewise exploration of increasingly large neighborhoods; the neighborhoods are defined using the variable neighborhood decomposition search method, and the search is conducted with a general branch-and-bound algorithm.

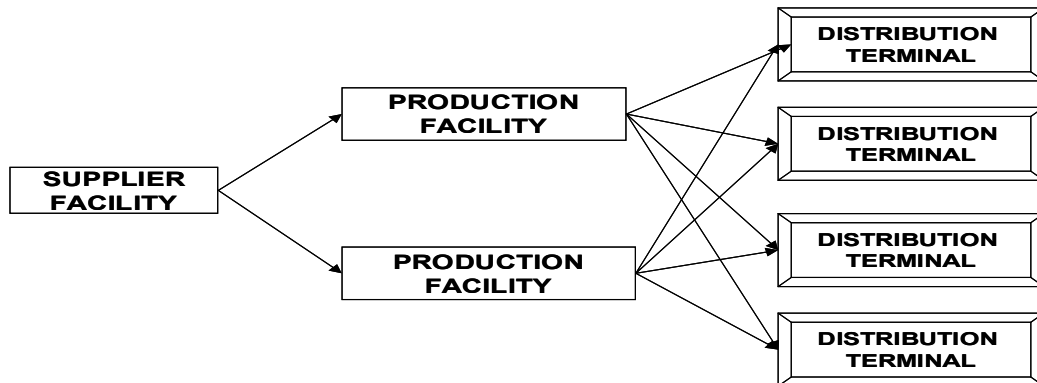
The remainder of the paper is organized as follows. In Section 2, we introduce the notation used, explain the hypotheses of the model, formulate the integrated inventory-production-distribution planning model, and provide its compact formulation. In section 3, we discuss the characteristics of the variable neighborhood search metaheuristic. In section 4, we describe the methodology proposed in this paper for designing sustainable inventory-production-distribution plans. Section 5 reports the numerical results of the methodology, which is applied on a real-life industrial problem faced by one of the major Northern American chemical supply chains. The tests have been conducted using the branch-and-bound algorithm of the CPLEX solver and the AMPL modeling language. Section 6 provides concluding remarks, and lists some possible extensions for the methodology proposed.

2 Model

2.1 Three-stage supply chain management

In this paper, we consider a three-stage supply chain represented in *Figure 1*, where arcs denote that an upstream stage supply a downstream stage. The first stage, i.e., supplier, assures the procurement of raw materials and/or components. The second stage, i.e., production, represents the manufacturing and/or assembly of the finished goods. The third stage, i.e., distribution, represents the transportation of the finished products to a distribution center (retailer, wholesaler) or to the end-customer.

The demand and lead times, defined as the sum of the traveling, loading and unloading times, are supposed to be known, and vary from period to period. There is no backlog; the demand that cannot be satisfied from on-hand inventory is lost. The supply chain has capacitated supply, production, storage and transportation capabilities. The distribution of the end products and raw materials is carried out by a fleet of heterogeneous carriers, differing in their maximal loading capacity, speed and availability. The possibility to recourse to external third-party logistics service providers is also allowed.

Figure 1: Three-stage supply chain

The objective pursued by the three-stage supply chain is to construct a sustainable inventory-production-distribution plan enabling it to minimize its costs while satisfying its customers' demand. More precisely, the program associated with the inventory-production-distribution plan to be constructed takes the form of a multi-dimensional mixed-integer program defining:

- the production scheme, i.e., the quantity of products and raw materials to be produced at each production facility at each period;
- the supply scheme, i.e., the quantity of products and raw materials to be sent from the supplier to the manufacturers, and from the manufacturers to the distributors at each time period;
- the ending inventory level required at each node in the supply chain and at each period;
- the scheduling and routing scheme for each carrier;
- the maintenance schedule for each carrier.

2.2 Model formulation and notation

2.2.1 Sets and indices

We define the following sets and indices:

$I = \{1, 2, \dots, i, \dots, l\}$: set of production facilities (manufacturers and suppliers),

$J = \{1, 2, \dots, j, \dots, \gamma\}$: set of distributors;

$K = I \cup J = \{1, 2, \dots, k, \dots, l + \gamma\}$: set of supply chain nodes (suppliers, manufacturers, and distributors);

$T = \{1, 2, \dots, t, \dots, \tau\}$: set of time periods over the planning horizon,

$V = \{1, 2, \dots, v, \dots, \nu\}$: set of distribution carriers.

2.3 From pseudo-observations to relative preferences

2.3.1 Parameters

The parameters used in the formulation of the model are listed below:

$c[v]$ = unit shipping cost incurred using carrier v ,

$C[v, t]$ = maximum loading capacity of carrier v at time t ,
 $l[k, v]$ = estimated loading time of carrier v at node k ,
 $u[k, v]$ = estimated unloading time of carrier v at node k ,
 $Z^{\max}[j]$ = maximum storage capacity for end products at distributor j ,
 $S^{\max}[i]$ = maximum storage capacity for end and semi-finished products at facility i ,
 $W^{\max}[i]$ = maximum storage capacity for raw materials at facility i ,
 $Z^{\min}[j, t]$ = safety stock requirement for end products at distributor j and time t ,
 $S^{\min}[i, t]$ = safety stock requirement for end and semi-finished products at facility i and time t ,
 $W^{\min}[i, t]$ = safety stock requirement for raw materials at facility i and time t ,
 $Tr[v, t]$ = time units available for carrier v at time t ,
 $ts[i, k, v]$ = traveling time from facility i to node k with carrier v ,
 $o[i, t]$ = demand for semi-finished product at node i and time t ,
 $r[i, t]$ = conversion rate of semi-finished product into end product at node i and time t ,
 $f[i, t]$ = conversion rate of raw material into end-product at node i and time t ,
 $g[i, t]$ = conversion rate of raw material into semi-finished product at node i and time t ,
 $h[k, t]$ = unit holding cost at node k and time t ,
 $p[i, t]$ = unit production cost at facility i and time t ,
 $b^{\max}[i, t]$ = maximum production capacity for raw material at facility i and time t .
 $p^{\max}[i, t]$ = maximum production capacity for end product at facility i and time t .
 $d[j, t]$ = demand for end product at distributor j and period t .

2.3.2 Decision Variables

The decision variables involved in the minimization of the costs of the three-stage supply chain are the following ones:

$z[j, t]$ = product inventory level at distributor j and time t ,
 $x[i, k, v, t]$ = number of direct shipments from facility i to node k made at time t with carrier v ,
 $P[i, t]$ = total quantity of end product provided by facility i at time t ,
 $s[i, t]$ = product inventory level at facility i at the end of period t ,
 $w[i, t]$ = raw material inventory level at facility i at the end of period t ,
 $b[i, t]$ = production of raw material at facility i and time t ,
 $\partial[v, t]$ = binary variable indicating the availability of carrier v at time t ;
 $q[i, k, v, t]$ = quantity of products or raw materials delivered to node k at time t with a carrier v leaving from facility i ,
 PC = total production cost,
 HC = total holding cost,
 TC = total transportation cost.

2.3.3 Objective function

The objective function is the minimization of the distribution, production and holding costs of the supply chain. The distribution costs are defined with respect to the carrier used. More precisely, they depend on the cost per time unit, the lead time, i.e., the sum of the loading, traveling and unloading times, and the number of shipments performed. The

production costs vary with the production level at the facilities. The holding costs are proportional to the stock levels of products and raw material at all nodes in the supply chain. The objective function is formulated as follows:

$$\begin{aligned} & \min PC + HC + TC \\ & \text{with } PC = \sum_{i \in I} \sum_{t \in T} p[i, t] * P[i, t] \\ & HC = \sum_{i \in I} \sum_{t \in T} h[i, t] * (p[i, t] / 2 + s[i, t]) + \sum_{i \in I} \sum_{j \in J} h[j, t] * s[j, t] \\ & TC = \sum_{i \in I} \sum_{t \in T} \sum_{k \in K, k \neq i} \sum_{v \in V} c[v] * (l[i, v] + 2 * ts[i, k, v] + u[v, k]) * x[i, k, v, t] \end{aligned}$$

2.3.4 Constraints

The following constraints must be accounted for:

Flow balance for raw materials at suppliers' facilities:

$$w[i, t] = w[i, t - 1] + b[i, t] - \sum_{v \in V} \sum_{i' \in I, i' \neq i} q[i, i', v, t] - f[i, t] * P[i, t] - g[i, t] * o[i, t], i \in I, t \in T \quad (1)$$

The raw material is consumed at the suppliers' and manufacturers' facilities. All the raw material needed by the manufacturers' facilities comes from the suppliers' facilities. Denoting by $b[i, t]$ the amount of raw material available in time t , by $q[i, i', v, t]$ the amount of raw material moved from the supplier's facility i to the manufacturer's facility i' using vessel v at time t , by $p[i, t]$ the production at the supplier's facility at time t , by $f[i, t]$ the amount of raw material for a unit of production, by $o[i, t]$ the demand for the semi-finished product at time t , and by $g[i, t]$ the appropriate conversion coefficient, we obtain (1).

Flow balance for raw materials at manufacturers' facilities:

$$w[i, t] = w[i, t - 1] + \sum_{v \in V} \sum_{i' \in I, i' \neq i} q[i', i, v, t] - f[i, t] * P[i, t] - g[i, t] * o[i, t], i \in I, t \in T \quad (2)$$

Flow balance for products at facilities:

$$s[i, t] = s[i, t - 1] * r[i, t - 1] / r[i, t] + p[i, t] - \sum_{j \in J} \sum_{v \in V} q[i, j, v, t], i \in I, t \in T \quad (3)$$

Constraints (1), (2) and (3) enforce that the current raw material and product demand at facilities be met from current production and on-hand inventory.

Flow balance for products at distributors:

$$z[j, t] = z[j, t - 1] + \sum_{i \in I} \sum_{v \in V} r[i, t] * q[i, j, v, t] - d[j, t], j \in J, t \in T \quad (4)$$

Constraint (4) enforces that the current demand be met from the on-hand inventory and the quantity received at the current period.

Production capacity of raw material:

$$0 \leq b[i, t] \leq b^{\max} [i, t], i \in I, t \in T \quad (5)$$

Production capacity of products:

$$0 \leq P[i, t] \leq p^{\max}[i], i \in I, t \in T \quad (6)$$

Constraints (5) and (6) are the capacity restriction constraints, limiting the production of raw materials and products below a specified maximum level.

Storage capacity of raw materials at facilities:

$$0 \leq w[i, t] < w^{\max}[i], i \in I, t \in T \quad (7)$$

Storage capacity of products at distributors:

$$0 \leq z[j, t] < z^{\max}[j], j \in J, t \in T \quad (8)$$

Storage capacity of products at facilities:

$$0 < s[i, t] < s^{\max}[i], i \in I, t \in T \quad (9)$$

Constraints (7), (8) and (9) limit from above the inventories of raw materials and products. The non-negativity restrictions ensure that no backlog occurs.

Shipment indivisibility:

$$x[i, j, v, t] \in \mathbb{Z}_+, i \in I, j \in J, v \in V, t \in T \quad (10)$$

Denoting by \mathbb{Z}_+ the set of the positive integers, the constraint (10) enforces the carriers' indivisibility requirement. This constraint imposes to deal with a great number of integer variables and renders the computing process very complicated. The attempt of constructing a complex distribution scheme with carriers delivering diverse destinations within a single trip would lead to a computationally intractable problem. As a result, the distribution scheme constructed in this manuscript allows direct shipments only; complex shipment routes involving deliveries at several supply chain nodes within the same shipment are not considered.

Transportation capacity:

$$0 \leq q[i, k, v, t] = C[v, t] * x[i, k, v, t], i \in I, v \in V, k \in K, t \in T \quad (11)$$

Constraint (11) is related to the limited capacity of the transportation carriers, and enforces that "full-truck load" shipments be carried out.

Time availability of carriers:

$$\sum_{i \in I} \sum_{k \in K} (l[i, v] + 2 * ts[i, k, v] + u[k, v]) * x[i, k, v, t] \leq Tr[v, t], v \in V, t \in T \quad (12)$$

Constraint (12) represents the limited time availability of the carriers, and accounts for the lead time for a shipment made with the carrier v between the facility i and the node j .

Maintenance of carriers:

$$x[i, j, v, t] \leq M * \delta[v, t], v \in V, t \in T \quad (13)$$

$$\sum_{t=1}^{12} \partial[v, t] = 1, v \in V, \partial[v, t] \in \{0, 1\} \quad (14)$$

Constraint (13), in which the parameter M is a large positive number, introduces a binary variable, $\partial[v, t]$, which is equal to 1 if the carrier v is set up at time t , and is equal to 0 otherwise. Constraint (14) enforces that carriers be not used for at least one period in the planning horizon, this allowing the maintenance of the carriers.

Operationalization of the distribution scheme:

$$x[i, j, v, t] = 0, i \in I, v \in V \quad (15)$$

Constraint (15) indicates the impossibility to deliver to a certain supply chain node j , at a given time period t , this resulting, for example, from bad weather conditions or the closing of facilities for some time.

Non-negativity:

$$z[j, t], P[i, t], s[i, t], w[i, t], b[i, t], q[i, j, v, t], CS[j, t] \geq 0, i \in I, j \in J, t \in T, v \in V \quad (16)$$

The set of constraints in (16) enforces the non-negativity restrictions on the decision variables.

Sustainability:

$$z[j, \tau] \geq z[j, 0], j \in J \quad (17)$$

$$s[i, \tau] \geq s[i, 0], i \in I \quad (18)$$

$$w[i, \tau] \geq w[i, 0], i \in I \quad (19)$$

The construction of a sustainable plan requires the inclusion of the constraints (17), (18) and (19), which enforce that the end inventory levels of products and raw materials will be at least equal to the initial ones. The introduction of constraints related to the end inventory levels aim at making the plan reproducible for the next planning horizon (see, for example, Chand and Morton, 1986, McClain and Thomas, 1977, Nahmias, 1993). The omission of this type of constraints would result in the construction of a myopic plan (see, e.g., Heyman and Sobel, 1984, and Zipkin, 2000), that may be characterized by unacceptably low, possibly equal to 0, end inventory levels, and leave the supply chain in an undesirable state for the future.

2.4 Compact formulation

The problem to be solved for constructing a sustainable inventory-production-distribution plan for a three-stage supply chain takes the form of a general mixed-integer program. Using matrix notation, we give below its compact formulation:

$$\min c^T x$$

$$s.to Ax \geq b$$

$$x = [\underline{x} \ x] \in \mathbb{R}_+ \times \mathbb{Z}_+$$

where A is the sparse $[m \times n]$ -dimensional coefficient matrix. The vector b is m -dimensional, while c and x are n -dimensional. The n -dimensional vector x can be partitioned into its real positive components, \underline{x} and its integer positive components, x . We use \mathbb{Z}_+ and \mathbb{R}_+ to denote

the set of positive integers and positive real numbers respectively. The inequality \geq must be understood coordinatewise.

3 Variable Neighborhood Search

The modeling of complex industrial issues often leads to complex (mixed-)integer programs that often cannot be efficiently solved by exact algorithms. As a substitute or a complement for exact algorithms, heuristics are used, and often allow substantial improvements as compared to the solution found by relying on an exact algorithm only.

The main family of heuristics, i.e., local search heuristics, aims at uncovering the best solution in a defined neighborhood. Considering a current incumbent solution, local search methods involve a sequence of changes of the initial solution, triggering improvements in the value of the objective solution, until a local optimum is found. The main issue related to the local search strategies is that they end up trapped in the first local optimum found. To avoid or at least alleviate this issue, the variable neighborhood search (VNS) metaheuristic involves a change of neighborhood during the search process (Mladenović and Hansen, 1997, Hansen and Mladenović, 2001). Starting from an incumbent solution \tilde{z} , the VNS metaheuristic involves the definition of a finite sequence of neighborhoods,

$$\mathcal{N}_1(\tilde{z}) \subseteq \mathcal{N}_2(\tilde{z}), \dots, \mathcal{N}_{k_{\max}-1}(\tilde{z}) \subseteq \mathcal{N}_{k_{\max}}(\tilde{z}),$$

that become increasingly large and distant to the incumbent solution. The successive neighborhoods are explored using a local search algorithm.

The best solution found, z_k^* , in the currently explored neighborhood, $\mathcal{N}_k(\tilde{z})$, becomes the incumbent one if and only if an improvement in the objective value results. In that case, a new sequence of neighborhood around the updated incumbent solution \tilde{z} is defined and explored. If no improvement can be reached in the neighborhood, $\mathcal{N}_k(\tilde{z})$, a more distant neighborhood, $\mathcal{N}_{k+1}(\tilde{z})$, from the incumbent solution is explored.

The key advantage of proceeding along this line is that the favorable characteristics of the incumbent solution, i.e., the fact that many variables are already at their optimal value, are kept when exploring more distant neighborhoods. The algorithmic process stops when the stopping criterion is met. The stopping criterion can be defined with respect to the CPU time used, the maximum number of iterations, the minimal magnitude of the improvement over a predefined number of iterations, the maximum number of iterations between two improvements, etc.

Instead of conditioning the replacement of the incumbent solution, \tilde{z} , by a tentative one, z_k^* , at the only condition that an improvement can be reached, a simulated annealing approach of the VNS may allow the substitution of the tentative solution, z_k^* , for the incumbent one, \tilde{z} , with a probability p , even if the tentative solution is not as good as the incumbent one. The probability p is a decreasing function of the magnitude of the degradation of the solution: $p = f(z_k^* - \tilde{z})$.

3.1 Variable Neighborhood Decomposition Search

The possibility to find an optimal or near-optimal solution with the variable search neighborhood depends largely on the size of the successive neighborhoods considered. For small- to moderate-size neighborhoods, the variable search neighborhood is very fast and efficient. However, for very large problems, the time- and computing-resources may be

excessive, thus making attractive or necessary the use of parallel computing or decomposition methods (Hansen and Mladenović, 1999).

The variable neighborhood decomposition search (VNDS) follows the basic variable neighborhood search within a decomposition method. The solution z of the problem to be solved is characterized by a number (q) of attributes or decision variables, x . The VNDS metaheuristic involves the fixing of a number (q'') of the decision variables, x'' , while the other q' ($q'=q-q''$) variables, x' , are unfixed, the optimization process being focused on finding their optimal value. By contrast to the VNS approach which is applied to a q -dimensional problem (i.e., to find the optimal solution z defined by q decision variables), the VNDS approach is applied to a lower, q' -dimensional problem consisting of finding the optimal solution $z^{(k)}$ characterized by the q' unfixed decision variables.

The sequence of neighborhoods is defined with respect to the incumbent solution, \tilde{z} , of the q' -dimensional problem, i.e., with respect to the q' unfixed decision variables x' :

$$\begin{cases} \mathcal{N}_k^{q'}(\tilde{z}), & q' = \alpha, \alpha > 0, k = 1, \dots, k_{\max}, \\ \dots \\ \mathcal{N}_k^{q'}(\tilde{z}), & q' = \alpha + \beta, \beta > 0, \alpha + \beta \leq q, k = 1, \dots, k_{\max}, \end{cases} \quad (20)$$

The formula (20) shows that the number q' of unfixed variables is iteratively increased by a number β as one moves forward in the computing process. For example, one may first consider the uni-dimensional problem containing a single ($q' = \alpha = 1$) unfixed variable x' , which solving provides a supposedly (near-)optimal value for the unfixed variable x' . For each successive problems, i.e., for each value of q' , a variable search neighborhood search is conducting, in which a number k_{\max} of neighborhoods are considered.

4 Solving Methodology

The construction of an inventory-production-distribution plan for the three-stage supply chain considered involves the solving of a complex mixed-integer program, whose number of integer variables exponentially increases with the number of nodes in the supply chain. The solving of such a mixed-integer program by uniquely relying on a branch-and-bound (cut) algorithm is unlikely to be sufficient to obtain a (near-) optimal solution as confirmed by the results reported in Section 5.

To solve the general mixed-integer program associated with the construction of a sustainable inventory-production-distribution plan, we propose a variable neighborhood search solving methodology that can be seen as the juxtaposition of the two following ideas:

- the definition of a stagewise decomposition scheme (Section 4.1), and
- the use of a variable neighborhood search approach (Section 4.2) that involves, at each stage s ,
 - ◆ the definition of a finite sequence of increasingly large neighborhoods, and
 - ◆ the exploration of the successive neighborhoods with a branch-and-bound algorithm.

These two ideas are explained in details in the next sections.

4.1 Decomposition scheme

The decomposition scheme is related to the relaxation of the integrality constraints on the integer decision variables. Initially, the integrality constraints are removed for the majority of the integer variables, redefined as continuous variables, and are maintained for a subset of integer variables. Hence, in the first stage of the algorithm, a mixed-integer problem of reduced complexity is solved, in which the number of integer variables, i.e., the subset of

integer variables on which the integrality conditions are maintained, is lower than this in the original problem. The probability of finding the optimal (or near-optimal) values of the integer variables for which the integrality conditions are maintained is high. The solution obtained is used as an incumbent solution for the subsequent stages of the algorithmic process, in which the integrality constraints are progressively restored on an increasing number of the integer variables. The integer variables for which the near-optimal value has been found are allowed to vary within an interval of reduced size in the next stages.

4.1.1 Dynamic assignment of integer variables to clusters

The decomposition scheme involves the *dynamic* assignment of the integer variables to three *clusters*, C_1 , C_2 , C_3 . The cluster assignment of the integer variables does not concern a single variable x_i at a time, but rather involves a subset of variables: $\{x_i : i \in X_s\}$. The index set $\{i, \dots, q\}$ is indeed partitioned into a number of subsets, $X_s, s = 1, \dots, S$. The subset X_s stands for the index set of integer variables for which integrality restrictions are restored at stage s .

Denoting by q ($q = q' + q'' + q'''$) the number of integer variables x_i , the clusters C_1 , C_2 , C_3 are at the current stage, s' , of the decomposition scheme composed as follows:

- (1) the cluster C_1 contains a number q' of variables, x_i , on which the integrality restrictions have been removed, and are to be restored in the next stages ($s = s' + 1, \dots, S$),

$$C_1 = \{x_i, i \in X_s, s = s' + 1, \dots, S\},$$

where the q' variables x_i are defined as continuous variables:

$$x_i \in C_1 \rightarrow x_i \in [L_i, U_i] \subseteq \mathbb{R}_+^1, i = 1, \dots, q';$$

- (2) the cluster C_2 contains a number q'' of integer variables on which the integrality conditions are restored at the current stage, s' ,

$$C_2 = \{x_i, i \in X_{s'}\},$$

where the q'' variables x_i , which optimal values have yet to be found, are defined on their original definition interval, I_i :

$$x_i \in C_2 \rightarrow x_i \in I_i = \{L_i, \dots, U_i\} \subseteq \mathbb{Z}_+^1, i = 1, \dots, q'';$$

- (3) the cluster C_3 contains a number q''' of integer variables on which the integrality conditions have been restored in the earlier stages ($s = 1, 2, \dots, s' - 1$), at which their (near-) optimal values have been found,

$$C_3 = \{x_i, i \in X_s, s = 1, \dots, s' - 1\},$$

where the q''' variables x_i are defined on a *narrow* definition interval, $I_i^{(n)}$:

$$x_i \in C_3 \rightarrow x_i \in I_i^{(n)} = \{l_i, \dots, u_i\} \subseteq I_i \subseteq \mathbb{Z}_+^1, i = 1, \dots, q'''.$$

To illustrate the cluster assignment of the integer variables, we report below the cluster assignment of integer at the s^{th} stage of the decomposition scheme:

- the integer variables x_i such that $i \in X_s$ are assigned to C_2 ;
- the integer variables x_i such that $i \in X_{s'}, s' > s$ are assigned to C_1 ;
- the integer variables x_i such that $i \in X_{s'}, s' < s$ are assigned to C_3 ;

It is worth underlining that integer variables are not definitely assigned to one of the clusters introduced above. Each integer variable is successively assigned to C_1 , C_2 , and, finally, C_3 . The cluster C_2 is a transitional cluster, i.e., the variables are assigned to it for a single stage: a subset of the variables previously ($s = 1, \dots, s' - 1$) assigned to C_1 are assigned to

C_2 at stage s' , before being assigned to C_3 for the remaining stages ($s=s'+1, \dots, S$) of the decomposition scheme. The cluster C_3 is *absorbing*: once assigned to C_3 , the variables remain in it for the subsequent stages of the algorithmic process. The dynamic assignment of integer variables continues until all integer variables are assigned to C_3 .

To each stage of the decomposition scheme, i.e., to each of the successive assignment, corresponds a specific mixed-integer program to be solved. Below, we provide the compact formulation of the mixed-integer program to be solved at stage s :

$$\begin{aligned} & \min c^T x \\ & \text{s.to } Ax \geq b \\ & \underline{x} \in \mathbb{R}_+ \\ & x_i \in \mathbb{R}_+, i \in X_{s+1}, X_{s+2}, \dots, X_S \\ & x_i \in I_i \subseteq \mathbb{Z}_+, i \in X_s \\ & x_i \in I_i^{(n)} \subseteq \mathbb{Z}_+, i \in X_1, X_2, \dots, X_{s-1} \end{aligned}$$

The last row of *Table 1* shows how the dimensions of the successive mixed-integer programs varies.

Table 1: Dynamic assignment of integer variables to clusters

Stage	C_1	C_2	C_3
1	X_2, \dots, X_{S-1}, X_S	X_1	\emptyset
2	X_3, \dots, X_{S-1}, X_S	X_2	X_1
\vdots	\vdots	\vdots	\vdots
S	$X_{s+1}, \dots, X_{S-1}, X_S$	X_s	X_1, \dots, X_{s-1}
\vdots	\vdots	\vdots	\vdots
S	\emptyset	X_S	X_1, \dots, X_{S-1}
Defined on:	$[L_i, U_i] \subseteq \mathbb{R}_+$	$I_i = [L_i, U_i] \subseteq \mathbb{Z}$	$I_i^{(n)} = [l_i, u_i] \subseteq I_i \subseteq \mathbb{Z}_+$

After having processed the S stages of the decomposition scheme, the final mixed-integer program defined below is to be solved:

$$\begin{aligned} & \min c^T x \\ & \text{s.to } Ax \geq b \\ & \underline{x} \in \mathbb{R}_+ \\ & x_i \in I_i^{(n)}, i \in X_1, X_2, \dots, X_S \end{aligned}$$

4.1.2 Subsets of integer variables

In the context of the construction of an inventory-production-distribution plan, it is worth recalling that integer variables represent the number of shipments between nodes of the supply chain at a time t with a certain carrier v . Several alternatives for the composition of the subsets and the restoration of the integrality conditions are presented below and tested in Section 5:

- the *chronological order* first restores the integrality conditions on the variables associated with the first period(s) in the planning horizon, i.e., the shipments performed in the first period(s) in the planning horizon;

- the *demand-based order on the supply chain nodes* consists of constructing an ordering of the supply chain nodes, k , based on their cumulative demand, $CD[k]$, over the whole planning horizon:

$$CD[k] = \sum_{t \in T} d[k, t]$$

The integrality conditions are first restored on the variables associated with the supply chain node, which have the largest cumulative demand, $CD[k]$.

- the *congestion-based order on the periods in the planning horizon* consists of constructing an ordering of the periods in the planning horizon based on the total demand, $TD[t]$, arising at each period t :

$$TD[t] = \sum_{k \in J} d[k, t]$$

The integrality conditions are first restored on the variables associated with the period at which the total demand is the highest. The underlying idea is that the period(s) which are characterized by a large demand are the most constraining ones, i.e., those at which the use of the production, distribution and inventory resources of the supply chain will be the greatest. The decisions taken at these periods are more likely to cause significant additional costs if they are not properly managed.

In case of high seasonality, a selection based on *the Pareto rule*, also called *80-20 rule*, may be applicable. This would mean that 80% of the total demand over the whole planning horizon arises from over 20% of the periods (20% of the supply chain nodes). The corresponding integer variables would then be prioritized.

4.2 Variable neighborhood search within the decomposition scheme

At each stage s of the decomposition scheme, a variable neighborhood search is conducted, involving the definition of a finite set of neighborhoods, the local exploration of these latter, the substitution procedure of the incumbent solution, and the stopping criterion. We discuss these aspects below.

4.2.1 Neighborhood definition

The sequence of neighborhoods is defined with respect to the definition interval of the integer variables x_i assigned to C_2 , i.e., the integer variables on which the integrality conditions are restored at the current stage s .

Denoting by \tilde{x}_i the incumbent solution (value) for $x_i, i \in X_s$, the finite sequence of neighborhoods is given by:

$$\begin{cases} \mathcal{N}_0(x_i) = x_i \in I_i, \\ \mathcal{N}_k(x_i) = x_i \in I_i^{(n)} = [\max(L_i, \tilde{x}_i - k), \min(U_i, \tilde{x}_i + k)], k = 1, 2, \dots, k_{\max}, \end{cases} \quad x_i \in X_s \quad (22)$$

The formula (22) shows that the size of the neighborhood is progressively increased as one move forward in the exploration of the neighborhoods associated with the current stage s . The progressive increase of the parameter k is an illustration of the diversification mechanism of the variable neighborhood search metaheuristics. Within a stage s , we progressively enlarge the size of the neighborhood by increasing the value of k . The underlying idea for determining the value of k_{\max} , i.e., the maximum allowed enlargement for the neighborhood, is to keep the explored neighborhood sufficiently small to be optimized with reasonable computing resources, and also large enough so that improvements of the incumbent solution are attainable.

4.2.2 Local search with branch-and-bound

In this paper, we use the branch-and-bound algorithm of the CPLEX solver to conduct a local search on the successive neighborhoods defined in (22). The branch-and-bound algorithm operates by fixing some of the integer variables at an integer value, each set of fixed integer variables defining a node in the search tree. The remaining integer variables are left free to take integer or real values, the resulting problem being called the relaxation of the node. The branch-and-bound algorithm uses the best integer feasible solution found so far to bound out some combinations, and explores, if not otherwise indicated, all non-excluded, possible combinations of integer values.

Two additional remarks relative to the use of the branch-and-bound algorithm are to be underlined:

- for each neighborhood, the number of nodes to be processed is limited from above. This is advised by Hansen and Mladenović (2001), since the exhaustion of all integer feasible solutions at each neighborhood would be extremely time-consuming;
- when starting the exploration of a new neighborhood, the starting incumbent solution is the best solution found so far in the preceding neighborhoods explored.

4.2.3 Substitution procedure

The general substitution procedure is adopted, in which the best solution, z_k^* , found in the k^{th} neighborhood $\mathcal{N}_k(x)$ becomes the incumbent one if and only if it is better than the best solution found so far: $z_k^* \succ \tilde{z}$. In that case, we proceed to the substitution:

- $\tilde{z} \leftarrow z_k^*, \tilde{x}_i \leftarrow x_{ik}^*$, and
- $k \leftarrow 1$,

while, otherwise, we set: $k \leftarrow k + 1$.

4.2.4 Stopping criterion

At every stage s , the diversification mechanism stops if a better solution than the incumbent one cannot be found over a number $K, K \leq k_{\max}$, of consecutive neighborhoods. In that case, we move to the next stage. The computing process is definitively stopped after having solved the final mixed-integer program (21).

4.3 Overview

The global scheme of the solving methodology proposed is given below:

(1) Initialization:

- (1.1) definition of the *decomposition scheme*,
- (1.2) definition of the *finite sequence of neighborhood* $\mathcal{N}_k(x), k = 1, 2, \dots, k, \dots, k_{\max}$,
- (1.3) finding of an *initial solution* z .

(2) **Recursive process**: each stage involves the following steps:

- (2.1) setting $k \leftarrow 0$,
- (2.2) *local search* of the k^{th} neighborhood with a branch-and-bound algorithm using as incumbent solution the best solution found so far, and finding of a local optimum, z_k^* , on the k^{th} neighborhood,
- (2.3) *substitution procedure*:
 - ◆ if $z_k^* > \tilde{z}$, then set: $\tilde{z} \leftarrow z_k^*$, $\tilde{x}_i \leftarrow x_{ik}^*$, $k \leftarrow k + 1$, and move to step (1);
 - ◆ otherwise, if $z_k^* \not> \tilde{z}$, set $k \leftarrow k + 1$, and move to step (1).

(3) **Stopping criterion**: if no improvement has been detected over a number K of consecutive neighborhoods, set $s \leftarrow s + 1$; stop after solving of final mixed-integer program.

5 Numerical result

5.1 Test laboratory

As a test bed for the solving methodology proposed, we use the data provided by one of the three largest chemical North American companies. This company produces soda ash and calcium chloride, and is part of a three-stage supply chain, including suppliers, manufacturers, and distribution centers. The distribution is carried out with a heterogeneous fleet of ships and barges differing in their speed, availability and loading capacity. A one-year planning horizon with monthly time-periods is considered.

For solving the general mixed-integer program – its dimensions are given in *Table 2* below -- inherent to the construction of a sustainable inventory-production-distribution plan, we use the AMPL modeling language and the CPLEX solver. A customized script file has been written, allowing the testing of the variable neighborhood decomposition search algorithm proposed in this paper.

Table2 : Dimension of the problem

Variables and constraints	Dimension
Number of variables	1341
Number of continuous variables	801
Number of general integer variables	484
Number of binary variables	56
Number of constraints	1741

5.2 Results

In the experiences discussed below, we deliberately decide to not fine-tune the parameters (K , number of nodes processed for each k) of the variable neighborhood decomposition search, but to pay special attention to the ordering in which integrality conditions are restored.

To evaluate the solving methodology, we compare the solutions obtained for the solving of the inventory-production-distribution model by relying:

- on the branch-and-bound algorithm (*B-B*) embedded in the CPLEX solver only,
- on the variable neighborhood decomposition search solving methodology, for which three variants are implemented, differing in the order in which integrality restrictions are restored on the integer variables. As explained above, the three variants considered are the chronological order (*C-VNDS*), the demand-based order of supply chain nodes (*DN-VNDS*), and the congestion-based order of periods (*CP-VNDS*).

More precisely, for each solution obtained, we compute the optimality gap, which is used as a measure of the quality of the integer solution found. Denoting by *UB* the best feasible integer solution found with the considered methodology, and *LB* the optimal continuous solution for the problem on-hand, the optimality gap, *G*, is measured as follows:

$$G = \frac{UB - LB}{LB}$$

The value of the objective function, i.e., total costs, and the optimality gap obtained with each methodology are reported in *Table 3* below.

Table 3 : Evaluation of the solving methodologies: objective function and optimality gap

	<i>B-B</i>	<i>C-VNDS</i>	<i>DN-VNDS</i>	<i>CP-VNDS</i>
<i>G</i>	23.81%	7.81%	6.54%	4.54%
Objective value	\$6889150	\$5998720	\$5928370	\$5816940

Different comments result from *Tables 2* and *3*. The first is related to the high value, i.e., 23.81%, of the optimality gap when solely relying on the standard branch-and-bound algorithm of a commercial solver. This underlines the need to supplement the standard branch-and-bound algorithm with a suitable solving technique when confronted with such a complex mixed-integer program.

The second one is that the methodology based on the *VNDS* allows the finding of a significantly better integer solution. The conclusion is valid for the three tested variants of the variable neighborhood decomposition search. Indeed, using the three variants of the *VNDS* and comparing them to the methodology relying on the sole branch-and-bound algorithm, the optimality gap is reduced by at least 16%, and can amount up to 19.27%. The corresponding reduction in the total costs of the supply chain is huge, ranging between \$900000 and \$1100000.

The third one is that the three variants of the variable neighborhood decomposition search do not result in the same savings for the supply chain; this highlighting the essential role of the ordering in which the integrality conditions are restored. The *VNDS* method relying on the congestion-based order outperforms its two counterparts. The *CP-VNDS* restores the integrality conditions first on the integer variables associated with the periods at which the consumption of resources in the supply chain are the highest, i.e., the *most congested* periods. The underlying idea of the *CP-VNDS* is that it is at the most congested periods that inappropriate planning or scheduling decisions have the most harmful consequences, possibly leading to a bottleneck of resources, and higher costs for the supply chain. The results displayed in *Table 2* show the prevalence of the *CP-VNDS* variant, that prioritizes the periods at which congestion in the supply chain is the most likely to occur.

The fourth comment is derived from *Table 4*, which shows that the savings associated with the *CP-VNDS* approach are predominantly due to a reduction in the distribution costs. Denoting by S^{CP-C} (S^{CP-DN}) the savings of the *CP-VNDS* approach relatively to the *C-VNDS*

(*DN-VNDS*) approach, it can be seen that 71.62% (80.84%) of these savings originate from the decrease in the distribution costs. This shows that the prioritization of the distribution decisions related to the most congested periods allows the construction of a significantly more economical distribution schedule. More precisely, the prioritization of the most congested periods results in a distribution schedule in which ahead of time shipments with the most economical carriers are carried out. Proceeding to ahead-of-congested periods shipments reduces the distribution requirements at the congested periods, and facilitates the minimization of the use of less economical carriers.

Since the outcome of the *CP-VNDS* is a major reduction in the distribution costs allowed by an extended use of the savings-prone carriers, it turns out that the congestion concept has to be defined with respect to the use of the distribution resources at the highly demanding periods in the planning horizon. It therefore can be induced that the risk of a bottleneck in the resources of the supply chain can be reduced by increasing the loading capacity or the speed of the economical carriers, or by extending the fleet of such carriers.

Table 4 : Costs and savings distribution

	COSTS			SAVINGS			
	<i>C-VNDS</i>	<i>DN-VNDS</i>	<i>CP-VNDS</i>	<i>(CP-VNDS - D-VNDS)</i>		<i>(CP-VNDS - DN-VNDS)</i>	
				\$	%	\$	%
Total costs	\$5975110	\$5928370	\$5816940	\$111430	100%	\$158170	100%
Production costs	\$1944930	\$1935550	\$1906130	\$29420	26.40%	\$38800	24.53%
Inventory costs	\$990390	\$1001090	\$988890	\$12200	10.95%	\$1500	0.95%
Distribution costs	\$3039790	\$2991730	\$2911920	\$79810	71.62%	\$127870	80.84%

6 Conclusions

Considering the problem of constructing a multi-period, sustainable inventory-production-distribution plan for a multi-stage supply chain, we propose a new variable neighborhood decomposition search methodology, which can be seen as a stagewise exploration of increasingly large neighborhoods. The stages are related to the decomposition scheme, i.e., to the order on which integrality conditions are restored. Within each stage of the decomposition scheme, a sequence of increasingly large neighborhoods is defined relying on the variable neighborhood search metaheuristic, while the exploration of the successive neighborhoods is performed using a general branch-and-bound algorithm embedded in a commercial MIP solver.

Validated on a real-industrial life problem faced by one of the main North American chemical supply chains, the solving methodology proposed is very effective, resulting in a much better solution than this used by solely relying on the commercial MIP solvers, and allowing substantial savings for the supply chain, primarily of a distribution nature.

The benefits of the methodology are maximized by defining the decomposition scheme with regards to the possibility of congestion in the supply chain. This conclusion results from the testing of various decomposition schemes. Empirical results show that they have unequal potential, and that the ordering, according to which the integrality conditions are restored is essential. The benefits generated by the congestion-based order decomposition scheme predominantly originate from a reduction of the distribution costs, which indicates that the risk of a bottleneck is to be defined with respect to the availability of the distribution resources at the congested periods in the planning horizon. This must be contrasted to the bottleneck approach of Glasserman and Wang (1999), which consists of identifying the

facility at which the allocation of additional resources would have the greatest impact on service. In this paper, the bottlenecks are not linked with the capacitated production and supply facilities, but with the limited transportation capacity of the carriers at the time-periods at which the demand is the greatest.

The algorithm proposed in this paper is related, but has also to be contrasted to this of Lodi and Fischetti (2003). While the former aims at finding an optimal (partial) solution as soon as possible, and is applied for a general mixed-integer program, the latter aims at finding a reasonable feasible solution as soon as possible, and is primarily designed for 0-1 mixed-integer programs. It must also be noted that the variable neighborhood decomposition search proposed in this paper differs from the “traditional” variable neighborhood decomposition search approach (Hansen and Mladenović, 2001). Indeed, the former allows a subset of integer variables to move within a narrow interval, while the latter approach fixes a single or a number of integer variables.

Although designed for the construction of a sustainable replenishment plan in the supply chain management context, the range of application of the proposed methodology can be easily extended to the solving of other types of mixed-integer programs. One of its key characteristics is its applicability to MIP containing both binary and general integer variables, while most of the extant literature has been devoted to the solving of 0-1 mixed-integer programs.

References

- Ahuja R.K., Ergun O., Orlin J.B. and Punnen A.P. 2002. A Survey of Very Large-Scale Neighborhood Search Techniques. *Discrete Applied Mathematics* 123 (1-3), 75-102.
- Balas E., Ceria S., Dawande M., Margot F. and Pataki G. 2001. A New Heuristic Approach for Pure 0-1 Programs. *Operations Research* 49 (2), 207-225.
- Bixby R.E., Felton M., Gu Z., Rothberg E. and Wunderling R. 1999. MIP: Theory and Practice: Closing the Gap. *System Modeling and Optimization*, in *System Modelling and Optimization: Methods, Theory and Applications*, Kluwer, (eds.) M. J. D. Powell and S. Scholtes, 19-50.
- Chand S. and Morton T.E. 1986. Minimal Forecast Horizon Procedures for Machine Replacement Models with Several Possible Replacement Alternatives. *Naval Research Logistics Quarterly* 29, 483-493.
- Chen Z. L. 2003. Integrated Production and Distribution Operations: Taxonomy, Models, and Review. *Working paper*, Robert H. Smith School of Business, University of Maryland.
- Erengüç S.S., Simpson N.C. and Vakharia A.J. 1999. Integrated Production/Distribution Planning in Supply Chains: An Invited Review. *European Journal of Operational Research* 115, 219-236.
- Fischetti M. and Lodi A. 2004. Local Branching. *Mathematical Programming*, forthcoming.
- Fourer R. and Gay D.M. 2002. Extending an Algebraic Modeling Language to Support Constraint Programming. *Informatics Journal of Computing* 14 (4), 322-344.
- Fourer R., Gay D.M. and Kernighan B.W. 1993. *AMPL: A Modeling Language for Mathematical Programming*, Boyd and Fraser, Danvers, Massachusetts.
- Glasserman P. and Wang Y. 1999. Fill-Rate Bottlenecks in Production-Inventory Networks. *Manufacturing and Service Operations Management* 1 (1), 62-76.
- Hansen O. and Mladenović N. 1999. An Introduction to Variable Neighborhood Search, in: Voss S., Martello S., Osman I.H. and Roucairol C.(Eds.), *Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Dordrecht, 433-458.

- Hansen P. and Mladenović N. 2001. Variable Neighborhood Search : Principles and Applications. *European Journal of Operational Research* **130**, 449-467.
- Hansen P., Mladenović N., and Perez-Brito D. 2001. Variable Neighborhood Decomposition Search. *Journal of Heuristics* **4**, 335-350.
- Heyman D.P. and Sobel M.J. 1984. *Stochastic Models in Operations Research Volume II: Stochastic Optimization*. McGraw Hill Book Company, New York.
- Løkketangen A. and Glover F. 1998. Solving Zero/One Mixed-Integer Programming Problems Using Tabu Search. *European Journal of Operational Research* **106**, 624-658.
- McClain J. and Thomas J. 1977. Horizon Effects in Aggregate Production Planning with Seasonal Demands. *Management Science* **23**, 728-736.
- Mladenović N. and Hansen P. 1997. Variable Neighborhood Search. *Computers Operations Research* **24** (11), 1097-1100.
- Nahmias S. 1993. *Production and Operations Analysis*. 2nd Edition, Irwin.
- Nemhauser G.L. and Wolsey L.A. 1988. *Integer and Combinatorial Optimization*. John Wiley & Sons, New York.
- Sarmiento A.M. and Nagi R. 1999. A Review of Integrated Analysis of Production-Distribution Systems. *IIE Transactions* **31** (11), 1061-1074.
- Hall N.G. and Potts C.N. 2003. Supply Chain Scheduling: Batching and Delivery. *Operations Research* **51** (4), 566-584.
- Savelsbergh M.W.P. and Nemhauser G.L. 1996. Functional Description of MINTO, a Mixed-Integer Optimizer, Version 2.3, ISYE, Georgia Institute of Technology, Atlanta, GA, 30332, USA.
- Thomas, D.J. and Griffin P.M.. 1996. Coordinated Supply Chain Management. *European Journal of Operational Research* **94** 1-15.
- Vidal C.J. and Goetschalckx M. 1997. Strategic Production-Distribution Models: A Critical Review with Emphasis on Global Supply Chain Models. *European Journal of Operational Research* **98**, 1-18.
- Voss S., Martello S., Osman I.H. and Roucairol C. 1999. *Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization*. Kluwer Academic Publishers, Dordrecht.
- Zipkin P.H. 2000. *Foundations of Inventory Management*. McGraw Hill Company, New York.