Computers & Industrial Engineering 58 (2010) 411–422

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

A genetic approach to two-phase optimization of dynamic supply chain scheduling

Alebachew D. Yimer *, Kudret Demirli

Fuzzy Systems Research Lab., Department of Mechanical and Industrial Engineering, Concordia University, 1515 St-Catherine W., Montreal, QC, Canada

ARTICLE INFO

Article history:
Available online 22 January 2009

Keywords:
Genetic algorithm
Build-to-order
Supply chain scheduling
Agile manufacturing
Mixed integer programming

ABSTRACT

In today's competitive environment, *agility* and *leanness* have become two crucial strategic concerns for many manufacturing firms in their efforts to broaden market share. Recently, the build-to-order (BTO) manufacturing strategy is becoming a popular operation strategy to achieve both in a mass-scale customization process. BTO system combines the characteristics of make-to-order strategy with a forecast driven make-to-stock strategy. As a means to improve customer responsiveness, customized products are assembled according to specific orders while standard components are pre-manufactured based on short-term forecasts. Planning of the two subsystems using a two-phase sequential approach offers both operational and modeling incentives. In this paper, we formulate a two-phase mixed integer linear programming (MILP) model for material procurement, components fabrication, product assembly and distribution scheduling of a BTO supply chain system. In the proposed approach, the entire problem is first decomposed into two subsystems and evaluated sequentially. The first phase deals with assembling and distribution scheduling of customizable products, while the second phase addresses fabrication and procurement planning of components and raw-materials. The objective of both models is to minimize the aggregate costs associated with each subsystem, while meeting customer service requirements. The search space for the first phase problem involves a complex landscape with too many candidate solutions. A genetic algorithm based solution procedure is proposed to solve the sub-problem efficiently.

1. Introduction

The build-to-order (BTO) manufacturing system is a pull system in which materials are pulled downstream of the supply chain driven by customer orders. It basically incorporates the characteristics of both lean and agile manufacturing strategies. Unlike the traditional make to stock supply chain, BTO strategy reduces the dependency of the system on demand forecasts, hence diminishing the requirement of high inventory buffers in the supply chain as pointed out in Gunasekaran and Ngai (2005). BTO systems combine the characteristics of both make-to-stock (forecast driven) and make-to-order (demand driven) strategies. Standard component parts and non-customizable subassemblies are acquired or build in-house based on short-term forecasts, while schedules for the few customizable parts and the final assembly are executed after detailed product specifications have been derived from booked customer orders, see Demirli and Yimer (2008).

Customization of products can only be achieved if there is some form of postponent strategy either in the assembly state, assembly area, delivery or at the design phase. As described by Li, Cheng, and Wang (2007), postponent refers to delaying some product differentiation or process as late as possible until the supply chain becomes cost effective. Customer's input in BTO manufacturing environment would involve postponent in downstream decisions with some speculation on the upstream manufacturing and supplies, see Prasad, Tata, and Madan (2005). Manufacturing plants operating under BTO supply chain use one of the three form postponent strategies in their functions: finished goods, work-in-process parts and purchased items or raw-materials as shown in Krajewski, Wei, and Tang (2005). Sharma and LaPlaca (2005) study the long-term impact of adopting a BTO manufacturing system on the marketing function and identify the marketing strategies used by successful BTO companies. A BTO strategy positively affects market performance through its influence on the supply chain application knowledge downstream with customers, while a JIT strategy does the upstream application with suppliers, see Christensen, Germain, and Birou (2005).

If we consider the upholstered furniture business, it is characterized by a wide range of product styles and a diversified customer demand. A variety of basic frame styles, fabrics, colors and other special options would generate a wide range of custom-built products. Therefore, a lean production system along with an agile strategy must be implemented to keep the units moving through the plant and to the customer smoothly as shown in Lyons, Coronado-Mondragon, and Kehoe (2004). As a result, firms such as Pella, Herman Miller and Norwalk have shifted to a BTO manufacturing strategy and assemble different customized products, see Gunasekaran and Ngai (2005), Sharma and LaPlaca (2005), Yao and

* Corresponding author. Tel.: +1 702 353 6038.
E-mail addresses: alebachew@ie.mie.conocrdia.ca (A.D. Yimer), demirli@encs.conocrdia.ca (K. Demirli).

0360-8352/$ - see front matter © 2010 Published by Elsevier Ltd.
doi:10.1016/j.cie.2009.01.010

Nomenclature

Sets and indices

- Ω_f set of component fabricating plants
- Ω_a set of assembling plants
- Ω_d set of distribution centers
- Ω_r set of product retailers (dealers)
- i product type index, $i = 1, 2, \cdots, I$,
- j component part or subassembly index, $j = 1, 2, \cdots, J$,
- l raw-material index, $l = 1, 2, \cdots, L$,
- t period index, $t = 1, 2, \cdots, T$,
- k fabrication plant index, $k \in \Omega_f$,
- p assembly plant index, $p \in \Omega_a$,
- q distribution center index, $q \in \Omega_d$,
- r retailer index, $r \in \Omega_r$,

Input parameters

- ψ_{kpj} 1 if k supplies p with component j; or 0 otherwise
- χ_{pq} 1 if p can supply to q with products; or 0 otherwise
- χ_{qr} 1 if q can deliver products to r; or 0 otherwise
- ρ_{kl} holding cost of raw-material l by fabricator k
- s_{kl} order setup cost of raw-material l by fabricator k
- η_{kl} unit purchasing cost of r.m. l by fabricator k
- σ_{kpj} fixed cost of p to acquire j from fabricator k
- ϑ_{kpj} unit variable cost of p to procure j from k
- λ_{pj} holding cost of plant p per unit of part j
- c_{pij} unit customization cost of j in assembling i by p
- γ_{pi} fixed cost of plant p to assemble i
- β_{pi} unit regular time assembling cost of i at plant p
- ω_{pi} unit overtime assembling cost of i at plant p
- h_{qi} inventory holding cost per unit of i at distributor q
- τ_{pqi} unit transport cost of i from plant p to distributor q
- τ_{qri} unit transport cost of i from distributor q to dealer r
- a_{ri} setup cost of dealer r per order of product i
- π_{ri} penalty cost of r per unit backordered of i
- δ_{lj} proportion of r.m. l required per unit of part j
- u_{ij} units of j required per unit of product type i
- ℓ_k expected raw-material procurement lead-time at k
- ℓ_{pq} transportation lead-time from p to q
- ℓ_{qr} delivery lead-time from q to r
- ℓ_r expected production–distribution lead-time at r
- T planning horizon
- M^∞ very big positive integer
- D_{rit} demand volume of i at r in period t; equal to O_{rit} if $t \leqslant \ell_r$; or $max(O_{rit}, F_{rit})$ otherwise,
- SL_{min} min. customer service level requirement in % demand
- MDC_{kpj} capacity of k to supply p with component j per period
- MRC_{pi} regular time capacity of p to assemble product i per period
- MOC_{pi} overtime capacity of p to assemble product i per period
- MLC_{kl} capacity of k to stock r.m. l per period
- MJC_{pj} inventory capacity of plant p to carry part j
- MIC_{qi} storage capacity of q to carry i per period
- MTC_{pqi} capacity of p to deliver q with product i
- MTC_{qri} capacity limit to ship i from q to r

Decision variables

- DM_{klt} demand volume of r.m. l by fabricator k in period t
- DC_{pjt} demand of component j by assembler p in period t
- F_{pjt} anticipated demand for component j by p in period t
- L_{klt} inventory level of r.m. l at the end of period t
- J_{pjt} inventory status of j at the beginning of period t
- I_{qit} on hand balance of product i at q in period t
- Qin_{qit} quantity of i delivered to q by all plants in period t
- $Qout_{qit}$ quantity of i shipped from q to all retailers in period t
- QL_{klt} scheduled receipt of r.m. l by k in period t
- QC_{kpjt} quantity of j procured by p from k in period t
- QR_{pit} regular time assembled volume of i in period t
- QO_{pit} overtime assembled volume of i in period t
- QA_{pit} total volume of product i assembled in period t
- QT_{pqit} volume of i shipped from p to q in period t
- QT_{qrit} volume of i transported from q to r in period t
- SL_{rit} demand satisfaction level of i at retailer r in period t
- QN_{rit} quantity of i backordered by r in period t
- Q_{rit} quantity of i delivered to r in period t
- Z_{RM} aggregate raw-materials cost
- Z_{CF} aggregate components fabrication cost
- Z_{AS} aggregate products assembling cost
- Z_{DC} aggregate distribution cost
- Z_{RT} aggregate retailers cost
- $\mathbf{Z_{PD}}$ total cost of production and distribution (*phase-1*)
- $\mathbf{Z_{CR}}$ total cost of components and raw-materials(*phase-2*)
- θ_{klt} 1 if k places order to procure l in period t, or 0 otherwise;
- ϕ_{kpjt} 1 if p procures part j from k in period t, or 0 otherwise;
- α_{pit} 1 if p is setup to assemble i in period t, or 0 otherwise;
- φ_{rit} 1 if r places assembly order of i in period t, or 0 otherwise;

Carlson (2003). Agile manufacturing facility can cope with changes in customer requirements including price, quality, customization, and promised delivery dates as indicated by Christian and Zimmers (1999). In most cases, furniture products consume large amounts of space during production, storage and shipment. A lean production system is thus important to curb large space requirements. A lean furniture production system uses its skilled work force and flexible handling equipment to quickly move small batch of material units from one workstation to the next thereby minimizing WIP. To enhance both agility and leanness, constructing a recommended cluster of fabrics available in different styles and colors would help limit the degree of customization.

The overwhelming majority of the literature in the area of supply chain modeling consider the traditional make-to-stock demand satisfying strategy. Production–distribution planning and scheduling is one important issue in multi-plant supply chain modeling. Scheduling models in multi-stage supply chains usually involve trade-offs among different conflicting objectives such as minimization of overall operating cost and safe inventory levels, while maximizing customer service performance and total profit with fair distribution among all partners, see Aghezzaf, Raa, and Landeghem (2006), Ertogral, Darwish, and Ben-Daya (2006), Guillen, Badell, and Puigjaner (2006), Neiro and Pinto (2004), Selvarajah and Steiner (2005). LP models to minimize total tardiness or total operation costs and considering capacity constraints, alternative machines sequences, sequence-dependent setup, and distinct due dates are also proposed in Ertogral et al. (2006), Liang (2006), Moon, Kim, and Hur (2002), Spitter, Hurkens, Kok, Lenstra, and Negenman (2005). Lakhal, Martel, Kettani, and Oral (2001) Perea-Lopèz, Ydstie, and Grossmann (2003) formulate a mixed integer linear programming (MILP) model to optimize strategic networking issues in multi-echelon supply chains. Multi-objective approaches for production and distribution scheduling scheme in multi-echelon supply chain networks are shown in Chen and Lee (2004), Sabri and Beamon (2000), Sakawa, Kato, and Nishizaki (2003). Talluri and Baker (2002) develop a multi-phase mathematical programming model with a combination of multi-criteria efficiency measures based on game theory concepts, and mixed integer linear programming methods. Amiri (2006), Ding, Benyoucef, and Xie (2005), Jayaraman and Prkul (2001) and Ross

(2000) addressed performance planning through resource allocation in supply networks by developing a profit maximizing model for distribution planning for the traditional make-to-stock supply chains.

Demirli and Yimer (2008) propose a mixed integer fuzzy programming (MIFP) approach to the production–distribution scheduling problem in BTO manufacturing supply chains with uncertain cost parameters. The effect of price incentives on demand characteristics of customizable BTO products is modeled as a stochastic dynamic programming problem in Weng and Parlar (2005). The use of modularity in the BTO product design as a solution to optimal return policy in internet marketing is presented in Mukhopadhyay and Setoputro (2005).

Genetic algorithm (GA) based approaches have been applied to different production, distribution and inventory related problems in supply chain operations. A GA based approach is proposed by Moon et al. (2002) to determine optimal schedule of machine assignments and operations sequences in a make to stock supply chain, so that the total tardiness will be minimized. In two-echelon single vendor–multiple buyers supply chain model under vendor managed inventory (VMI) mode of operation, Nachiappan and Jawahar (2007) formulates a nonlinear integer programming problem (NIP) with a GA based heuristic in order to find out the optimal sales quantity for each buyer. A genetic algorithm solution procedure for a mixed integer nonlinear programming model of a dynamic integrated distribution network of third party logistic providers (3PLs) is discussed in Ko and Evans (2007). Naso, Surico, Turchiano, and Kaymak (2007) present a scheduling algorithm that combines a GA and a set of constructive heuristics for the just-in-time production and delivery of ready-mixed concrete on a set of distributed and coordinated production centers. A hybrid genetic algorithm (HGA) is implemented in Torabi, Ghomi, and Karimi (2006), to solve the economic lot sizing and delivery scheduling problem in a simple supply chain where the production system is considered to be a flexible flow line. Lee, Jeong, and Moon (2002) suggested a genetic algorithm based heuristic to solve an advanced planning and scheduling (APS) model of a manufacturing supply chain with outsourcing and due date requirements.

In this paper, a multi-product and multi-plant BTO supply chain with manifold supply and distribution channels is considered for analysis. We proposed a two-phase MILP model for procurement, production, and distribution of customized products in a BTO manufacturing system. As opposed to formulating the problem as a fully integrated single model, the sequential two-phase approach helps to develop a robust solution procedure by decoupling the customized manufacturing operation from the standard component fabrication process. The proposed modeling is comprehensive in nature as it incorporates crucial pragmatic constraints resulting from capacity limitations, material flow equations, product customization and customer service requirements. The rest of the paper is organized as follows. Brief description of the problem considered for analysis and its mathematical formulation is presented in Section 2. In Section 3, we put forward a genetic algorithm solution procedure adopted in solving the problem. Results of numerical experimentation used to demonstrate the proposed approach are illustrated in Section 4. Summary and concluding remarks are given in Section 5. Finally, list of notations used to formulate the mathematical models are set out in the nomenclature part.

2. Description and formulation of the problem

In general, the build-to-order supply chains operate as a pull system driven by customer orders at the downstream end of the network. The finished products manufactured by assembly plants reach to end user consumers through channels of a distribution system. Efficient supply and distribution systems are thus essential entities for manufacturing firms to meet the demand of customers for good quality products at reasonably low cost. The BTO supply chain is essentially a hybrid of make-to-order (MTO) and assemble-to-order (ATO) strategies. Standard parts and subassemblies are acquired or manufactured in-house according to short-term forecasts, while schedules for few customized components and the final assembly of products are not executed until detailed product specifications have been derived from booked customer orders, see Sen, Pokharel, and YuLei (2004). As a strategy, the objective of BTO is to provide custom-made products in a mass-scale. Therefore, the customer order decoupling point (CODP) in BTO system falls between the CODP's of MTO and ATO systems.

As illustrated in Fig. 1, the BTO supply chain network consists of two major subsystems: a production-subsystem and a distribution-subsystem. The production-subsystem includes raw-material suppliers, component fabricators and product assemblers. The distribution-subsystem while consists finished product warehouses, intermediate distribution centers (DCs), retailers and downstream customers. Therefore, the supply chain scheduling problem can logically be decomposed into two sub-problems:

(i) *Phase-1:* Developing a dynamic model for assembling and distribution planning of final products as per customer order specifications and
(ii) *Phase-2:* Formulating a planning model for components fabrication and raw-materials procurement, based on the outputs of the previous model.

The material acquisition, production and distribution planning and scheduling problem should be approached in an integrated manner. The two sub-problems are interrelated to each other and should be dealt sequentially. The issue of integrated approach in planning and scheduling of BTO supply chains is addressed by Demirli and Yimer (2008). Splitting the problem into two-phases, however, offers a twofold advantage: both operational and modeling. From operational management perspective, this approach simplifies planning and control of materials acquisition, processing and distribution of products. From modeling perspective, the two-phase approach gives an opportunity to develop more efficient solution techniques without compromising the optimality of a fully integrated problem.

The entire production–distribution plan operates in a rolling horizon to allow changes in later periods as new plans are constructed. A plan is drawn for all periods in the horizon, but only the first few periods that fall within the current delivery lead-time will actually be implemented. When the plan for the first period is frozen, a new plan is redone for periods from the second up to the last period plus one. Due to latest information introduced in the new run, the updated plan discloses more accurate results in the short-term. The proposed sequential mathematical models are presented in the next two subsections. The list of notations used in developing the models are given in nomenclature.

2.1. Phase-1: assembling and distribution plan of products (Model-I)

The production–distribution schedule in BTO systems is driven by actual orders received from customers. Customers pick their preferred product styles from retailer catalogue and sign order requisitions. Retailers are the market outlets from which final products are delivered to customers while new orders are passed to assemblers. They accumulate and make job orders to the assembly plants ahead of the projected production and delivery lead-time. Efficient communication channel between dealers and finished product assemblers is therefore a critical factor to ensure product availability and improve customer service level. The distribution centers are middle agents, which receive and temporarily stock the finished products until they are delivered to the retailers. Inventories of final products at the distribution centers are used to decouple the unequal flow rate of incoming and outgoing finished products. The assembly plants, which operate in a build-to-order (BTO) environment, classify the specific order requisitions made by downstream retailers into families of product modules and commit resources to satisfy the demand within the specified due dates.

Given the demand volume for various styles of products in each period, the objective is to propose a capacity and resource feasible economic plan of assembling and distribution of products over a planning horizon that minimizes the overall operating cost while maintaining the desired customer service levels. Taking the volume of assembly job orders in queue and the capacity limitations into account, two options of assembling schedules-regular time and overtime – are drawn for each period. The objective cost function and set of constraints involved are set out in Eqs. (1)–(19).

Minimize :

$$\mathbf{Z_{PD}} = Z_{AS} + Z_{DC} + Z_{RT} \tag{1}$$

Subject to :

$$Z_{AS} = \sum_{t=1}^{T}\sum_{p\in\Omega_a}\sum_{i=1}^{I}\left\{\gamma_{pi}\cdot\alpha_{pit} + \omega_{pi}\cdot QO_{pit} + \beta_{pi}\cdot QR_{pit} + \sum_{j=1}^{J} c_{pij}\cdot u_{ij}(QO_{pit} + QR_{pit})\right\} \tag{2}$$

$$Z_{DC} = \sum_{t=1}^{T}\sum_{q\in\Omega_d}\sum_{i=1}^{I}\left\{h_{qi}\cdot I_{qit} + \sum_{p\in\Omega_a}\tau_{pqi}\cdot QT_{pqit}\right\} \tag{3}$$

$$Z_{RT} = \sum_{t=1}^{T}\sum_{r\in\Omega_r}\sum_{i=1}^{I}\left\{a_{ri}\cdot\varphi_{rit} + \pi_{ri}\cdot QN_{rit} + \sum_{q\in\Omega_d}\tau_{qri}\cdot QT_{qrit}\right\} \tag{4}$$

$$QN_{rit} = (1 - SL_{rit})\cdot D_{rit} \quad \forall r\in\Omega_r \tag{5}$$

$$QN_{rit} = QN_{ri,t-1} + D_{rit} - Q_{rit} \quad \forall r\in\Omega_r \tag{6}$$

$$Q_{rit} = \sum_{q\in\Omega_d} QT_{qri,t-\ell_{qr}} \quad \forall r\in\Omega_r,\ t > \ell_{qr} \tag{7}$$

$$I_{qit} = I_{qi,t-1} + \sum_{p\in\Omega_a} QT_{pqi,t-\ell_{pq}} - \sum_{r\in\Omega_r} QT_{qrit} \quad \forall q\in\Omega_d \tag{8}$$

$$\sum_{q\in\Omega_d} QT_{pqit} = (QR_{pit} + QO_{pit}) \quad \forall p\in\Omega_a \tag{9}$$

$$SL_{rit} \geqslant SL_{min} \quad \forall r\in\Omega_r \tag{10}$$

$$QT_{qrit} \leqslant \chi_{qr}\cdot MTC_{qri} \quad \forall q\in\Omega_d, \quad \forall r\in\Omega_r \tag{11}$$

$$QT_{pqit} \leqslant \chi_{pq}\cdot MTC_{pqi} \quad \forall p\in\Omega_a, \quad \forall q\in\Omega_d \tag{12}$$

$$I_{qit} \leqslant MIC_{qi} \quad \forall q\in\Omega_d \tag{13}$$

$$QR_{pit} \leqslant MRC_{pi} \quad \forall p\in\Omega_a \tag{14}$$

$$QO_{pit} \leqslant MOC_{pi} \quad \forall p\in\Omega_a \tag{15}$$

$$QR_{pit} \leqslant \alpha_{pit}\cdot M^{\infty} \quad \forall p\in\Omega_a \tag{16}$$

$$D_{rit} \leqslant \varphi_{ri,t-\ell_r}\cdot M^{\infty} \quad \forall r\in\Omega_r, \quad t\geqslant \ell_r \tag{17}$$

$$QR_{pit}, QO_{pit}, QN_{rit}, Q_{rit}, QT_{pqit}, QT_{qrit} \in \mathbf{N} \tag{18}$$

$$\alpha_{pit}, \varphi_{rit} \in \{\mathbf{0},\mathbf{1}\} \tag{19}$$

where, $\forall p\in\Omega_a, \quad \forall q\in\Omega_d, \quad \forall r\in\Omega_r, \quad i = 1,\ldots,I, \quad$ and $\forall t$

- (1) is the objective function of the model. It refers to the total cost equation for the products assembly and distribution-subsystem. Eqs. (2)–(19) describe the list of constraints introduced into the model as a result of capacity limitations, material balance equations, and service level requirements,

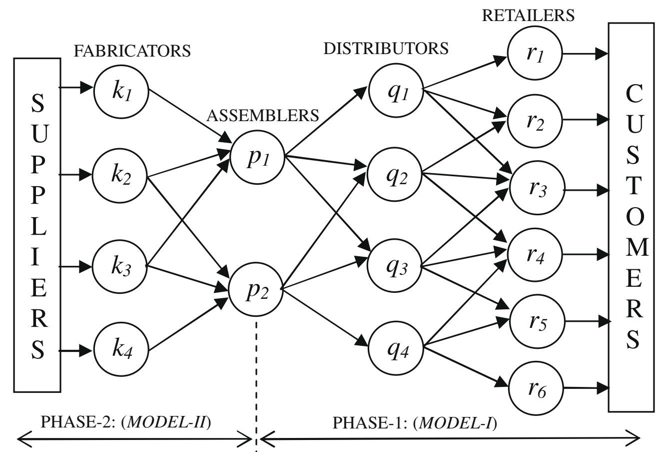

Fig. 1. A BTO supply chain network structure.

- (2) describes the aggregate cost of assembling products by all plants. It consists of overhead and setup costs, overtime and regular time assembling costs, and other costs incurred due to customization,
- (3) defines the total inventory and transportation costs of products by all distribution centers,
- (4) determines the aggregate order setup, shipment and shortage penalty costs of the retail dealers,
- (5) and (6) define the quantity of backorders for product i in each period t,
- (7) allows that all the products shipped from distribution centers to each retail node r should be delivered to the end user customers in the same time period t in order to improve responsiveness,
- (8) provides the inventory status balance of product i at distribution center q as the sum of the previous period stock balance and the quantity procured in the current period minus the volume of product shipped to upstream customers,
- (9) ensures that whole products assembled by plant p are delivered to the distributors in the same period t so as to eliminate inventory of finished products at the assembly plants,
- (10) restricts customer service level at each retailer r in period t is higher than the allowable minimum requirement.
- (11) and (12) define the transportation capacity constraints from assembly plant p to distributor q and from distributor q to dealer r, respectively,
- (13) describes the inventory capacity limitation of product i at each distribution center q,
- (14) and (15) limit the volume of regular time and overtime assembled products to be under the allowable production capacities,
- (16) sets the value of α_{pit} to 1 if product i is produced by p in period t or 0 otherwise, and
- (17) sets the value of φ_{rit} to 1 if dealer r places an assembly order in period t or 0 otherwise,
- (18) and (19) restrict the values of the specified decision variables to be non-negative integers and binary integers, respectively.

2.2. Components fabrication and raw-materials procurement (Model-II)

Each assembly job requires different components and sub-assembled units, which have to be either manufactured in-house or purchased from external sources. Once the quantity of products to be assembled in each period are known from Model-I, the next step is determine the volume of component parts and raw-materials required to meet the production plan. Taking the assembly tree structure shown in Fig. 2 into account, demand for component j is given as the maximum of the forecasted value or the actual demand determined from the bill of quantities (BOQ):

$$DC_{pjt} = Max\left(F_{pjt}, \sum_{i=1}^{I} u_{ij}(QR_{pit} + QO_{pit})\right) \quad \forall p \in \Omega_a, \quad \forall t, \text{ and } \quad j = 1, \ldots, J \tag{20}$$

where the regular and overtime assembling quantities, QR_{pit} and QO_{pit}, are obtained from the results of the first phase sub-problem (Model-I).

The demand for component j can be fulfilled either from inventory or by making a replenishment order right away to the upstream component fabricators. Therefore, the due date for each assembly job is decided based on the volume of waiting jobs and quantity of required components on hand. Given demand volume of components in each period, the objective in this case is to obtain a capacity and resource feasible economic plan fabrication of components and requisition of materials. The unit variable cost associated with each component part represents the material and labor cost per item if it is manufactured in-house, or the unit price and transport cost if it is purchased from an external source. The complete listing of the mathematical model for this sub-problem is organized as follows:

Minimize :

$$\mathbf{Z_{CM}} = Z_{CF} + Z_{RM} \tag{21}$$

Subject to :

$$Z_{CF} = \sum_{t=1}^{T} \sum_{p\in\Omega_a} \sum_{j=1}^{J} \left\{ \lambda_{pj} \cdot J_{pjt} + \sum_{k\in\Omega_f} (\sigma_{kpj} \cdot \phi_{kpjt} + \vartheta_{kpj} \cdot QC_{kpjt}) \right\} \tag{22}$$

$$Z_{RM} = \sum_{t=1}^{T} \sum_{k\in\Omega_f} \sum_{l=1}^{L} \left\{ \rho_{kl} \cdot L_{klt} + s_{kl} \cdot \theta_{klt} + \eta_{kl} \cdot QL_{klt} \right\} \tag{23}$$

$$DM_{klt} = \sum_{p\in\Omega_a} \sum_{j=1}^{J} \delta_{lj} \cdot QC_{kpjt} \quad \forall k \in \Omega_f \tag{24}$$

$$J_{pjt} = J_{pj,t-1} + \sum_{k\in\Omega_f} QC_{kpjt} - DC_{pjt} \quad \forall p \in \Omega_a \tag{25}$$

$$L_{klt} = L_{kl,t-1} + QL_{klt} - DM_{klt} \quad \forall k \in \Omega_f \tag{26}$$

$$QC_{kpjt} \leqslant \psi_{kpj} \cdot MDC_{kpj} \quad \forall k \in \Omega_f, \quad \forall p \in \Omega_a \tag{27}$$

$$J_{pjt} \leqslant MJC_{pj} \quad \forall p \in \Omega_a \tag{28}$$

$$L_{klt} \leqslant MLC_{kl} \quad \forall k \in \Omega_f \tag{29}$$

$$QC_{kpjt} \leqslant \phi_{kpjt} \cdot M^{\infty} \quad \forall k \in \Omega_f, \quad \forall p \in \Omega_a \tag{30}$$

$$QL_{klt} \leqslant \theta_{klt} \cdot M^{\infty} \quad \forall k \in \Omega_f \tag{31}$$

$$J_{pjt}, QC_{kpjt} \in \mathbb{N} \quad \forall k \in \Omega_f, \quad \forall p \in \Omega_a \tag{32}$$

$$L_{klt}, QL_{klt} \geqslant \mathbf{0} \quad \forall k \in \Omega_f \tag{33}$$

$$\phi_{kpjt}, \theta_{klt} \in \{\mathbf{0}, \mathbf{1}\} \quad \forall k \in \Omega_f, \ \forall p \in \Omega_a \tag{34}$$

where, $j = 1, \ldots, J, \quad l = 1, \ldots, L$, and $\forall t$

- (21) is the objective function of the current model. It represents the total cost equation for the components fabrication and raw-materials replenishment subsystem. Eqs. (22)–(34) represent the various capacity and material balance constraints of the model,
- (22) determines the aggregate cost of components procurement and stock keeping by all assembly plants,
- (23) describes the aggregate cost of replenishment and warehousing of raw-materials by all fabrication plants,

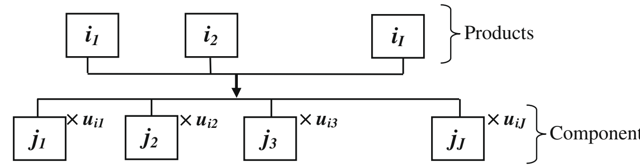

Fig. 2. Assembling tree structure.

- (24) calculates the volume of raw-material l to be consumed by fabricator k in each period t,
- (25) and (26) show the inventory level of component j at assembly plant p, and raw-material l at fabrication plant k in each period t, respectively. The inventory status balance is calculated as the sum of the previous period stock balance and the quantity procured in the current period minus the quantity utilized in period t,
- (27) restricts the quantity of component j procured by plant p from k in period t to be within the delivery capacity of the fabricator,
- (28) and (29) define the inventory capacity limitations of components at each assembly plant p and raw-materials at each fabrication plant k, respectively,
- (30) sets the value of ϕ_{kpjt} to 1 if p procures component j from fabricator k in period t or 0 otherwise,
- (31) sets the value of θ_{klt} to 1 if k places an order to purchase raw-material l in period t or 0 otherwise,
- (32)–(34) restrict the values of the specified decision variables to be non-negative continuous, general integers or binary integers, respectively.

3. A genetic algorithm based solution procedure

The product assembling and distribution model formulated in Phase-1 involves a complex shape of search space with too many candidate solutions. When the underlying solution space has a complex landscape, general search methods such as genetic algorithm (GA) are applicable for fast exploration. Different genetic operators and selection mechanisms can be implemented to protect the GA from being trapped at a local optimum area. In this section, a GA based solution methodology developed to solve the first phase problem efficiently will be discussed.

Introduced by Holland (1992), Genetic algorithms (GAs) belong to a class of intelligent stochastic search techniques inspired from the principle of 'survival-of-the-fittest' in natural evolution and genetics. GAs are known to search efficiently in a large search space, without explicitly requiring additional information (such as convexity, or differentiability) about the objective function to be optimized. As a result, in the last decade, GAs have been applied successfully for a wide variety of combinatorial optimization problems to find (near-) optimal solutions. Genetic algorithms work iteratively on a population of candidate solutions of the problem (chromosomes), performing a search guided by genetic operators (selection, crossover and mutation) based on a'fitness value' assigned to each individual according to a problem-specific objective function. GA explore solutions with increasing fitness, i.e., the higher the fitness, the more likely the genes of a chromosome are propagated to the next generations (Naso et al., 2007, Torabi et al., 2006).

3.1. Chromosomal encoding of solution

Prior to the application of GA, it is important to define an encoding strategy to transform a generic solution of the problem into a string of symbols suitable to the application of genetic operators. In GA literature, an encoded solution is generally referred to as chromosome, and a single parameter of the solution vector is called a gene. Designing a more suitable chromosomal representation of a solution is a key issue for successful implementation of GA (Naso et al., 2007, Ko & Evans, 2007). For the problem under study, the chromosome structure shown in Fig. 3 is selected.

The first member, $ID \in \{1,..,I\}$ refers to the index number of a product considered in the current loop of GA application. The second member Q_{pt} is the total volume of product assembled by plant p in period t both at regular time and overtime. The third member I_{qt} is the inventory level of the same product in period t at distribution center q. Given the quantity of products delivered to q in period t, the fourth member $\kappa_{pqt} \in (0,1)$ is the proportion of those assembled by plant p. Similarly given the volume of products reached at retailer r in period t, the fifth member $\eta_{qrt} \in (0,1)$ refers to the fraction that comes from distributor q. The customer satisfaction level (i.e., ratio of actual to promised delivery volume) at retailer r in period t is given by the sixth member SL_{rt}. The second member from the last, f, represents the fitness value assigned to the particular chromosome based on its objective value-aggregate production–distribution cost. The last member of the solution structure, $status \in \{0,1\}$ is a binary variable representing the feasibility of the chromosome. When the solution is decoded into its phenotype space, if an individual chromosome satisfies all of the constraints, then $status = 1$; or $status = 0$ otherwise. Note that the given solution structure is composed of a mixture of binary, integer, and floating type members.

3.2. Decoding and fitness evaluation

Chromosome decoding is the process of transforming a genotype solution representation into a corresponding phenotype version. It generates a candidate solution to a set of decision variables and the associated objective function value. In our case, the components of the original objective function include the aggregate costs of assembling, distribution and retailing of a specific product at all nodes of the supply chain.

Given a genotype representation of an individual solution: $[ID|\mathbf{Q}_{pt}|\mathbf{I}_{qt}|\kappa_{pqt}|\eta_{qrt}|\mathbf{SL}_{rt}|f|status]$, the values corresponding to the phenotype variables in the original problem are determined as follows:

Step-1: Demand satisfaction level, quantity backordered and quantity delivered to retailer r in period t are first determined by

$$SL_{rit} = \mathbf{SL}_{rt} \quad \forall r \in \Omega_r,\ \forall t \text{ and } i = ID \tag{35}$$

$$QN_{rit} = \text{round}\{(1 - \mathbf{SL}_{rt}) \cdot D_{rit}\} \quad \forall r \in \Omega_r,\ \forall t \text{ and } i = ID \tag{36}$$

$$Q_{rit} = D_{rit} + QN_{ri,t-1} - QN_{rit} \quad \forall r \in \Omega_r,\ \forall t \text{ and } i = ID \tag{37}$$

$$\varphi_{rit} = 1 \text{ if } (D_{rit} > 0);\ \text{or } 0 \text{ otherwise},\ \forall r \in \Omega_r,\ \forall t,\ \text{and } i = ID \tag{38}$$

Step-2: Fig. 4 illustrates the product flow in, through and out of a distribution node q.

Therefore, the inventory and quantity transport parameters at distributor q in period t are given by

$$I_{qit} = \mathbf{I}_{qt} \quad \forall q \in \Omega_d,\ \forall t \text{ and } i = ID \tag{39}$$

$$QT_{qrit} = \text{round}(Q_{rit} \cdot \eta_{qrt}) \quad \forall q \in \Omega_d,\ \forall t \text{ and } i = ID \tag{40}$$

$$Qout_{qit} = \sum_{r \in \Omega_r} QT_{qrit} \quad \forall q \in \Omega_d,\ \forall t \text{ and } i = ID \tag{41}$$

$$Qin_{qit} = Qout_{qit} + I_{qit} - I_{qi,t-1} \quad \forall q \in \Omega_d,\ \forall t \text{ and } i = ID \tag{42}$$

$$QT_{pqit} = \text{round}(Qin_{qit} \cdot \kappa_{pqt}) \quad \forall q \in \Omega_d,\ \forall t \text{ and } i = ID \tag{43}$$

Step-3: The volume of a product assembled by plant p in period t on regular time and overtime basis are given by

$$QA_{pit} = \mathbf{Q}_{pt} = \sum_{q \in \Omega_d} QT_{pqit} \quad \forall p \in \Omega_a,\ \forall t \text{ and } i = ID \tag{44}$$

$$QR_{pit} = \min\{\mathbf{Q}_{pt}, MRC_{pi}\} \quad \forall p \in \Omega_a,\ \forall t \text{ and } i = ID \tag{45}$$

$$QO_{pit} = \max\{0, \mathbf{Q}_{pt} - MRC_{pi}\} \quad \forall p \in \Omega_a,\ \forall t \text{ and } i = ID \tag{46}$$

$$\alpha_{pit} = 1 \text{ if } (\mathbf{Q}_{pt} > 0);\ \text{or } 0 \quad \text{otherwise},\ \forall p \in \Omega_a, \quad \forall t \text{ and } i = ID \tag{47}$$

Step-4: Finally, the aggregate production and distribution cost of the particular product ($i = ID$) can be determined as:

$$Z_{PD,i} = \sum_{t=1}^{T} \sum_{p \in \Omega_a} \left\{ \gamma_{pi} \cdot \alpha_{pit} + \omega_{pi} \cdot QO_{pit} + \beta_{pi} \cdot QR_{pit} + \sum_{j=1}^{J} c_{pij} \cdot u_{ij}(QO_{pit} + QR_{pit}) \right\} + \sum_{t=1}^{T} \sum_{q \in \Omega_d} \left\{ h_{qi} \cdot I_{qit} + \sum_{p \in \Omega_a} \tau_{pqi} \cdot QT_{pqit} \right\} + \sum_{t=1}^{T} \sum_{r \in \Omega_r} \left\{ a_{ri} \cdot \varphi_{rit} + \pi_{ri} \cdot QN_{rit} + \sum_{q \in \Omega_d} \tau_{qri} \cdot QT_{qrit} \right\}, \tag{48}$$

If a candidate solution is found to be infeasible (i.e., fails to satisfy all constraints of the model), then it is first treated by a repair heuristic as shown in Fig. 5. The chromosome repair heuristic facilitates searching around the boundaries between the feasible and infeasible region. If the repair operation does not succeed to cure the infeasible candidate, then a penalty term is introduced to its objective value in order to undermine its chance of surviving in the subsequent generations. The penalty value is considerably larger than any possible objective value corresponding to the current population of individuals as described in Ko and Evans (2007).

The fitness value is the measure of goodness of a solution with respect to the original objective function and the degree of infeasibility. For the cost minimization problem we have considered, candidate solutions with lower costs imply better solutions and vice versa (Torabi et al., 2006). Therefore, for each chromosome i, its fitness value f_i can be evaluated by taking a proportional factor K_f times the reciprocal of the objective function value $Z_{PD,i}$.

$$f_i = \frac{K_f}{Z_{PD,i}} \quad \text{for } i = 1, \ldots, N \quad \text{and} \quad K_f = constant \tag{49}$$

3.3. Initial population

The quality and size of the initial population can largely affect the efficiency of a genetic algorithm. As a result, a carefully crafted heuristic is required to generate random chromosomes within the

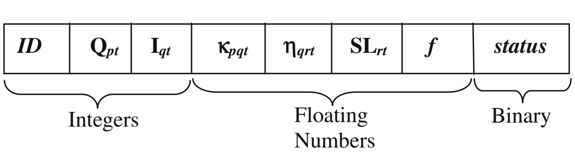

Fig. 3. Chromosomal representation of solution.

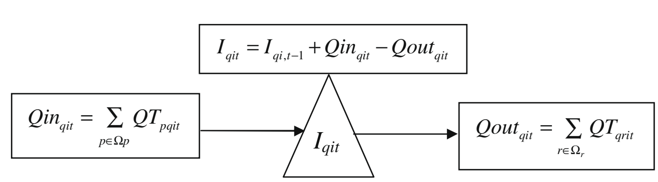

Fig. 4. Product flow in, through and out of node q.

constrained solution pace. The population size (N) is chosen by measuring the convergence time of the GA through trial and error approach. Fig. 6 shows flow chart of a constructive heuristic, which is used to randomly generate new chromosomes that can easily satisfy most of the constraints. The steps can be summarized as follows:

(i) Generate random values of service level achievements (SL_{rt}) at each node r in period t within the allowable ranges. The quantities delivered (Q_{rit}) and backordered (QN_{rit}) can easily be decoded from the given values as shown in the flow chart.
(ii) For a given node r, identify the arcs that stretch back to the upstream distributors and generate set of random numbers. Sum up the values over q, and assign the ratio of each random number to the sum as the genotype value of the proportion matrix η_{qrt}. The quantities transported from node q to r in each period (QT_{qrit}) and the volume of product shipped from each distributor ($Qout_{qit}$) are decoded as illustrated.
(iii) Randomly generate values of inventory level at node q in each period (I_{qt}) within the allowable bounds as indicated. Decode the phenotype variable for quantity of products entering to each distributor (Qin_{qit}) as shown in the flow chart.
(iv) Considering the links from the upstream nodes to each node q, generate a set of random numbers and get their sum over p. Take the ratio of each random number to the sum as the genotype value of the proportion matrix κ_{pqt}. Quantity of products transported from each node p to q are then decoded as shown.
(v) Sum up the phenotype values of QT_{pqit} over q to get the genotype values for quantities of products assembled in each period (Q_{pt}).
(vi) Calculate the objective function and fitness value f from the phenotype solution. Check all the values in the phenotype space against the constraints. If all constraints are satisfied, assign a value of '1' to the genotype member *status* or assign '0' otherwise.

BEGIN

$$IF\left(Q_{rit} > \sum_{q\in\Omega_q} MTC_{qri}\right) \quad \forall r, \forall t, \textit{ and } i = ID$$

$$Q_{rit} = \sum_{q\in\Omega_q} MTC_{qri}$$

$$QN_{rit} = D_{rit} + QN_{ri,t-1} - Q_{rit}$$

$$SL_{rt} = 1 - \frac{QN_{rit}}{Q_{rit}} \quad \{\text{genotype variable}\}$$

$$\eta_{qrt} = \frac{MTC_{qri}}{Q_{rit}} \quad \{\text{genotype variable}\}$$

END IF

$$IF\left(Qin_{qit} > \sum_{p\in\Omega_p} MTC_{pqi}\right) \quad \forall q, \forall t, \textit{ and } i = ID$$

$$Qin_{qit} = \sum_{p\in\Omega_p} MTC_{pqi}$$

$$I_{qt} = I_{qt-1} + Qin_{qit} - Qout_{qit} \quad \{\text{genotype variable}\}$$

$$\kappa_{pqt} = \frac{MTC_{pqi}}{Qin_{qit}} \quad \{\text{genotype variable}\}$$

END IF

END

Fig. 5. Chromosome repair heuristic.

3.4. Offspring population

A set of genetic operators are usually used to perturb the current individuals and create new chromosomes from the old ones. In the corresponding phenotype space, this is equivalent to searching for new candidate solution (Eiben & Smith, 2003). The newly generated individuals in the current generation will constitute the offspring population. A set of problem-specific variation operators is implemented in our proposed GA. The adapted genetic operators are described in the next subsections.

3.4.1. Crossover operators

The main purpose of crossover operation is producing better offspring(s) by combining the genetic alleles of two randomly selected parents from the mating pool. A crossover probability p_c indicates how often crossover will be performed. This parameter is tuned by a trial and error approach. In our implementation of GA, three types of problem-specific crossover operators have been applied:

(i) *Averaging crossover (AVEXO):* As shown in Fig. 7, this operator takes the average of some genes in the two parents and copy the result to the new offspring. At the end of the recombination process, only one offspring is produced. Its operation can be summarized as follows:
(i) Select two parent chromosomes from the mating poll.
(ii) Transfer the genetic information of product *ID* from either of the two parents into the offspring.
(iii) Calculate the simple average values of $I_{qt}, \kappa_{pqt}, \eta_{qrt}$ and SL_{rt} for the two parents.
(iv) Copy the results to the corresponding alleles of the offspring.

$$\text{Offspring} := \begin{cases} I^{av}_{qt} = \frac{1}{2}(I^{p1}_{qt} + I^{p2}_{qt}), & \forall q \in \Omega_d,\ \forall t \\ \kappa^{av}_{pqt} = \frac{1}{2}(\kappa^{p1}_{pqt} + \kappa^{p2}_{pqt}), & \forall p \in \Omega_a,\ \forall q \in \Omega_d,\ \forall t \\ \eta^{av}_{qrt} = \frac{1}{2}(\eta^{p1}_{qrt} + \eta^{p2}_{qrt}), & \forall q \in \Omega_d,\ \forall r \in \Omega_r,\ \forall t \\ SL^{av}_{rt} = \frac{1}{2}(SL^{p1}_{rt} + SL^{p2}_{rt}), & \forall r \in \Omega_r,\ \forall t \end{cases}$$

(v) Decode the genetic values and update the remaining information for Q_{pt}, f and *status* in the offspring chromosome.

(ii) *Convex crossover (CONXO):* This operator has one random parameter α called the weighting factor. As illustrated in Fig. 8, it calculates the convex combinations of some genes inside the two parents and put the results into the corresponding alleles of the new offsprings. Its operation can be summarized as follows:
(i) Repeat *steps* (i) and (ii) of AVEXO.
(ii) Randomly select a value of the parameter α, where α is uniformly distributed over (0.05,0.45).
(iii) Giving more weight to the second parent, compute the weighted average of $I_{qt}, \kappa_{pqt}, \eta_{qrt}$ and SL_{rt} for the two parents,and copy the results to the corresponding alleles of offspring-1.

$$\text{Offspring-1} := \begin{cases} I^{c1}_{qt} = \alpha \cdot I^{p1}_{qt} + (1-\alpha) \cdot I^{p2}_{qt} & \forall q \in \Omega_d, \forall t \\ \kappa^{c1}_{pqt} = \alpha \cdot \kappa^{p1}_{pqt} + (1-\alpha) \cdot \kappa^{p2}_{pqt} & \forall p \in \Omega_a, \forall q \in \Omega_d, \forall t \\ \eta^{c1}_{qrt} = \alpha \cdot \eta^{p1}_{qrt} + (1-\alpha) \cdot \eta^{p2}_{qrt} & \forall q \in \Omega_d, \forall r \in \Omega_r, \forall t \\ SL^{c1}_{rt} = \alpha \cdot SL^{p1}_{rt} + (1-\alpha) \cdot SL^{p2}_{rt} & \forall r \in \Omega_r, \forall t \end{cases}$$

(iv) Decode and update the genetic values of Q_{pt}, f and *status* in Offspring-1.
(v) To define Offspring-2, repeat the *steps* (1) to (4) by reversing the roles of the two parents.

$$\text{Offspring-2} := \begin{cases} I^{c2}_{qt} = (1-\alpha)\cdot I^{p1}_{qt} + \alpha\cdot I^{p2}_{qt} & \forall q\in\Omega_d, \forall t \\ \kappa^{c2}_{pqt} = (1-\alpha)\cdot\kappa^{p1}_{pqt} + \alpha\cdot\kappa^{p2}_{pqt} & \forall p\in\Omega_a, \forall q\in\Omega_d, \forall t \\ \eta^{c2}_{qrt} = (1-\alpha)\cdot\eta^{p1}_{qrt} + \alpha\cdot\eta^{p2}_{qrt} & \forall q\in\Omega_d, \forall r\in\Omega_r, \forall t \\ SL^{c2}_{rt} = (1-\alpha)\cdot SL^{p1}_{rt} + \alpha\cdot SL^{p2}_{rt} & \forall r\in\Omega_r, \forall t \end{cases}$$

(iii) *Uniform crossover (UNIXO):* This operator basically copies partial genetic information from both parents to the two offsprings without making any modification. The coping procedure depends on either of the two possible outcomes of a random tossing experiment. Its general operation is illustrated in Fig. 9, and summarized next:

(i) Repeat *steps* (i) and (ii) of AVEXO.
(ii) Perform a tossing experiment and record the outcome (i.e., $Toss = RANDOM\{0,1\}$).
(iii) Make copies of the genes according to the following rule:

$IF \quad (Toss = 1)$

$$\begin{array}{ll} \text{Offspring-1}: & \text{Offspring-2}: \\ I^{c1}_{qt} = I^{p2}_{qt} & I^{c2}_{qt} = I^{p1}_{qt} \quad \forall q\in\Omega_d, \forall t \\ SL^{c1}_{rt} = SL^{p2}_{rt} & SL^{c2}_{rt} = SL^{p1}_{rt} \quad \forall r\in\Omega_r, \forall t \\ \kappa^{c1}_{pqt} = \kappa^{p1}_{pqt} & \kappa^{c2}_{pqt} = \kappa^{p2}_{pqt} \quad \forall p\in\Omega_a, \forall q\in\Omega_d, \forall t \\ \eta^{c1}_{qrt} = \eta^{p1}_{qrt} & \eta^{c2}_{qrt} = \eta^{p2}_{qrt} \quad \forall q\in\Omega_d, \forall r\in\Omega_r, \forall t \end{array}$$

$ELSE \quad \{Toss = 0\}$

$$\begin{array}{ll} \text{Offspring-1}: & \text{Offspring-2}: \\ I^{c1}_{qt} = I^{p1}_{qt} & I^{c2}_{qt} = I^{p2}_{qt} \quad \forall q\in\Omega_d, \forall t \\ SL^{c1}_{rt} = SL^{p1}_{rt} & SL^{c2}_{rt} = SL^{p2}_{rt} \quad \forall r\in\Omega_r, \forall t \\ \kappa^{c1}_{pqt} = \kappa^{p2}_{pqt} & \kappa^{c2}_{pqt} = \kappa^{p1}_{pqt} \quad \forall p\in\Omega_a, \forall q\in\Omega_d, \forall t \\ \eta^{c1}_{qrt} = \eta^{p2}_{qrt} & \eta^{c2}_{qrt} = \eta^{p1}_{qrt} \quad \forall q\in\Omega_d, \forall r\in\Omega_r, \forall t \end{array}$$

$END\ IF$

(iv) Decode and update the remaining genes inside the two offspring chromosomes.

3.4.2. Mutation operators

Mutation slightly alters the genetic composition of a randomly selected chromosome. The intention here is to provide a small amount of randomness, and to prevent solutions from being trapped at a local optimum. Mutation occurs with some probability p_m smaller than a crossover probability. Depending on the encoding scheme of the problem, different mutation operators can be utilized. In our GA application to the problem under study, the following five problem-specific mutation operators are implemented.

(i) *FLIP mutation:* This operator first clones a randomly selected parent chromosome into a new offspring chromosome. Then, target indices (q^*, r^*, t^*) are set by random selection, where $q^*\in\Omega_q$, $r^*\in\Omega_r$, and $t^*\in\{1,\ldots,T\}$. Based on the outcome of a random tossing experiment, apply the following rule to alter the alleles of the cloned offspring chromosome.

$IF \quad (Toss = 1)$

$$SL^c_{r^*,t} = \begin{cases} SL^p_{r^*,T-t^*+t} & \text{if } t \leqslant t^* \\ SL^p_{r^*,t-t^*} & \text{otherwise} \end{cases}$$

$ELSE \quad \{Toss = 0\}$

$$I^c_{q^*,t} = \begin{cases} I^p_{q^*,T-t^*+t} & \text{if } t \leqslant t^* \\ I^p_{q^*,t-t^*} & \text{otherwise} \end{cases}$$

$END\ IF$

(ii) *SWAP mutation:* Similarly, this operator clones a parent chromosome into an offspring chromosome. Target indices (q^*, r^*, t^*_1, t^*_2) are then chosen randomly from their respective domain. Observing the outcome of a random tossing experiment, the alleles pointed by t^*_1 and t^*_2 are switched one another according to the following rule:

$IF \quad (Toss = 1)$

$$\begin{cases} SL^c_{r^*,t^*_1} = SL^p_{r^*,t^*_2} \\ SL^c_{r^*,t^*_2} = SL^p_{r^*,t^*_1} \end{cases}$$

$ELSE \quad \{Toss = 0\}$

$$\begin{cases} I^c_{q^*,t^*_1} = I^p_{q^*,t^*_2} \\ I^c_{q^*,t^*_2} = I^p_{q^*,t^*_1} \end{cases}$$

$END\ IF$

(iii) *COMBINE mutation:* In this operator too, we apply the same principle to fix the target indices(q^*, r^*, t^*_1, t^*_2). A weighting factor $\alpha\in(0.05, 0.45)$ is first randomly chosen. Performing a random tossing experiment, some alleles of the cloned chromosome are modified by taking the convex combinations of the values pointed by t^*_1 and t^*_2 according to the following rule:

$IF \quad (Toss_1 = 1)$

$$\begin{cases} SL^c_{r^*,t^*_1} = \alpha\cdot SL^p_{r^*,t^{ast}_1} + (1-\alpha)\cdot SL^p_{r^*,t^*_2} \\ SL^c_{r^*,t^*_2} = \alpha\cdot SL^p_{r^*,t^*_2} + (1-\alpha)\cdot SL^p_{r^*,t^*_1} \end{cases}$$

$ELSE \quad \{Toss_1 = 0\}$

$$\begin{cases} I^c_{q^*,t^*_1} = \alpha\cdot I^p_{q^*,t^*_1} + (1-\alpha)\cdot I^p_{q^*,t^*_2} \\ I^c_{q^*,t^*_2} = \alpha\cdot I^p_{q^*,t^*_2} + (1-\alpha)\cdot I^p_{q^*,t^*_1} \end{cases}$$

$END\ IF$

Repeating the tossing experiment one more times, other genes within the new chromosome are also modified based on the following rule:

$IF \quad (Toss_2 = 1)$

$$\begin{cases} \eta^c_{qr^*,t^*_1} = \alpha\cdot\eta^p_{qr^*,t^*_1} + (1-\alpha)\cdot\eta^p_{qr^*,t^*_2}, & \forall q\in\Omega_d \\ \eta^c_{qr^*,t^*_2} = \alpha\cdot\eta^p_{qr^*,t^*_2} + (1-\alpha)\cdot\eta^p_{qr^*,t^*_1}, & \forall q\in\Omega_d \end{cases}$$

$ELSE \quad \{Toss_2 = 0\}$

$$\begin{cases} \kappa^c_{pq^*,t^*_1} = \alpha\cdot\kappa^p_{pq^*,t^*_1} + (1-\alpha)\cdot\kappa^p_{pq^*,t^*_2} & \forall p\in\Omega_a \\ \kappa^c_{pq^*,t^*_2} = \alpha\cdot\kappa^p_{pq^*,t^*_2} + (1-\alpha)\cdot\kappa^p_{pq^*,t^*_1} & \forall p\in\Omega_a \end{cases}$$

$END\ IF$

(iv) *BORDERVAL mutation:* This operator basically helps to direct the genetic search process around the border line between feasible and infeasible regions of the solution space. It alters the values of some target alleles pointed by randomly selected indices (q^*, r^*, t^*) to their extreme lower or extreme higher values.After performing a first tossing experiment, the values of the genotype member $SL_{r^*t^*}$ in the cloned offspring chromosome are adjusted as follows:

$$SL^c_{r^*,t^*} = \begin{cases} SL_{min}, & \text{If } (Toss_1 = 1) \\ 1.0, & \text{If } (Toss_1 = 0) \end{cases}$$

Repeating the tossing exercise one more times, the values of member $I_{q^*t^*}$ are also modified in the same fashion:

$$I^c_{q^*,t^*} = \begin{cases} I^*_{min} = \max\{0, I^p_{q^*,t^*-1} - Qin_{q^*i,t^*}\},\ i = ID & \text{If } (Toss_2 = 1) \\ I^*_{max} = \min\{MIC_{q^*i}, \sum_p MTCpq^*i - Qout_{q^*,t^*}\}, & \text{If } (Toss_2 = 0)\cdot \end{cases}$$

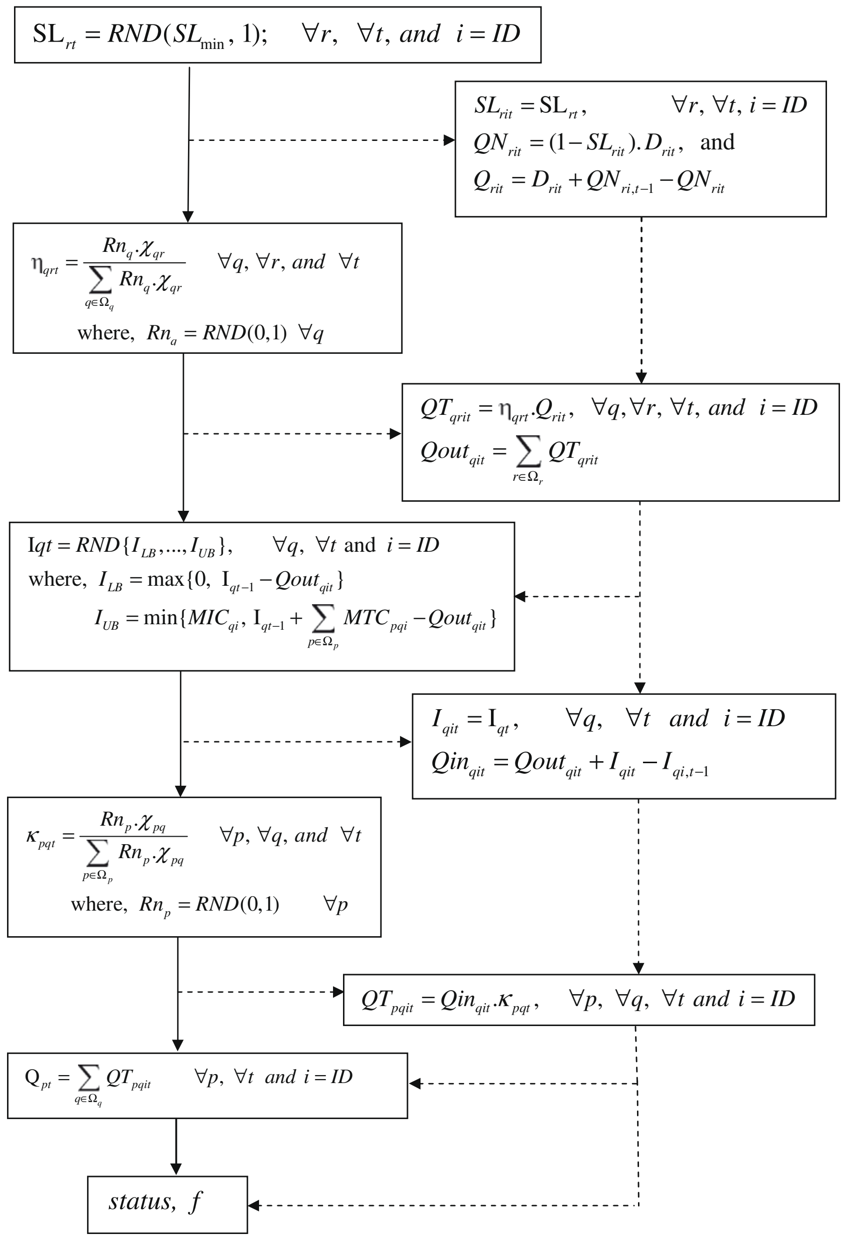

Fig. 6. Flow chart of random chromosome generating heuristic.

(v) *RANDOMVAL mutation:* This operator intends to introduce new values of some genes inside a cloned chromosome from the current population. Some alleles pointed by target indices (q^*, r^*, t^*) are replaced by new values within their allowable ranges. The following rule is applied to make the necessary modifications on some members of the cloned individual.

$$
\begin{aligned}
&IF \quad (Toss = 1)\\
&SL^{c}_{r^*,t^*} = RANDOM(SL_{min}, 1.0)\\
&ELSE \quad \{Toss = 0\}\\
&I^{c}_{q^*,t^*} = RANDOM\{I^*_{min}, \ldots, I^*_{max}\}\\
&END\ IF
\end{aligned}
$$

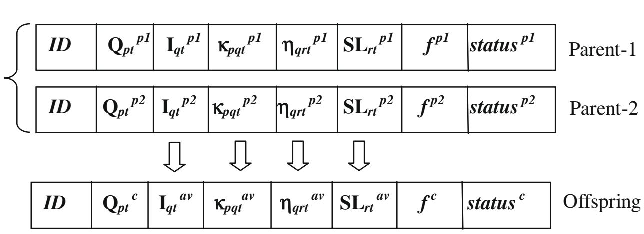

Fig. 7. Averaging crossover operation.

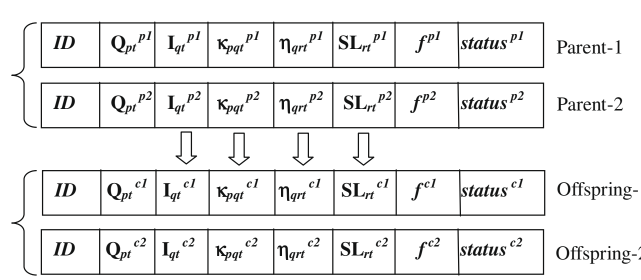

Fig. 8. Convex crossover operation.

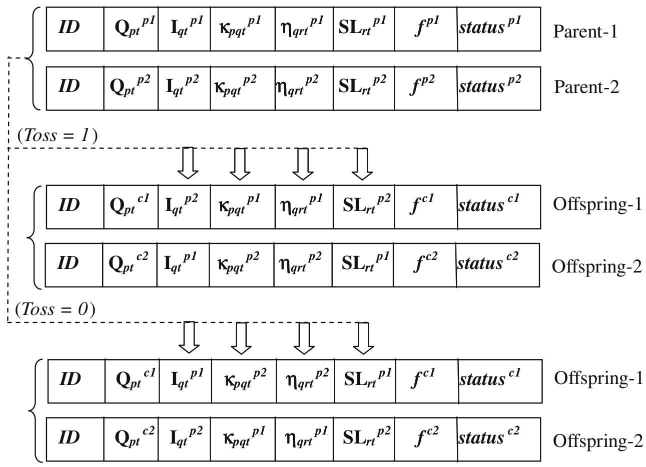

Fig. 9. Uniform crossover operation.

3.5. Parent selection

In genetic algorithm, the selection operator is used to guide the search process towards more promising regions in a search space. Several selection methods, such as roulette wheel selection, tournament selection, ranking selection, elitist selection are discussed in Eiben and Smith (2003) and Michalewicz (1996). In the proposed GA approach, a combination of roulette wheel and elitism are applied with threshold point being at P_E%. In this strategy, P_E% of the population in the new generation are selected by elitist mechanism, while the rest are selected by fitness proportional means. The degree of elitism, P_E%, is chosen by repeated execution of the algorithm for a given problem size.

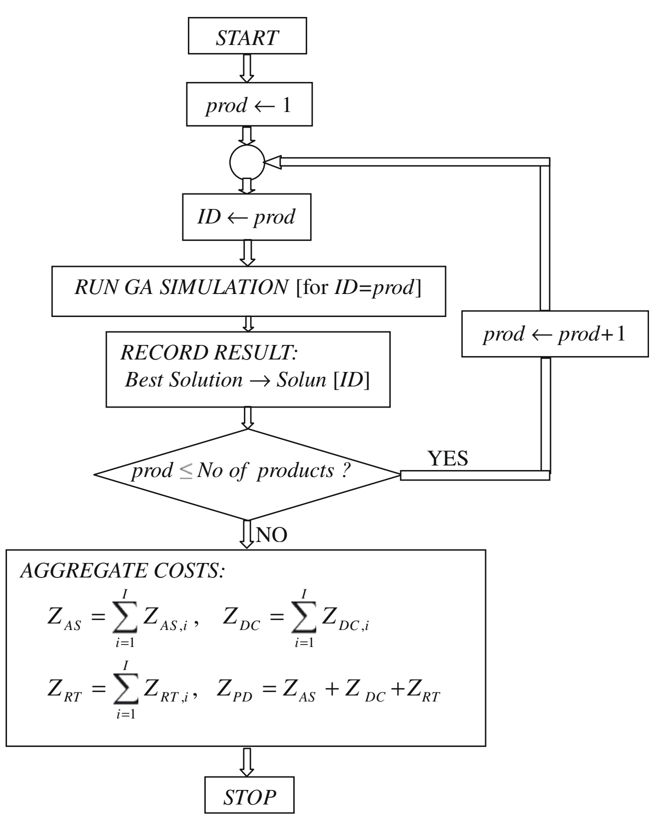

Fig. 10. Execution of GA in a loop.

3.6. Termination criterion and looping

In the proposed GA solution method, for a fixed product *ID*, the algorithm is executed until it converges to an incumbent solution according to a preset stopping criteria. The algorithm terminates when at least one of the following conditions are satisfied:

(i) Starting from its last improvement, if GA fails to provide better solution after additional generations of '*Max-Interval*', or
(ii) If the maximum generation reaches to a preset value of' *Max-Gen*'.

The best solution obtained is recorded before the algorithm re-executes itself with the next product *ID*. The loop will continue until all product indexes are considered in the analysis. The flow chart shown in Fig. 10 shows the general framework of the proposed GA operating in a loop.

4. Numerical results

The prototype of the proposed GA algorithm is developed in a C++ programming platform and executed on a Pentium(R)-4, 300 MHz PC with 1GB of RAM. To illustrate the approach, a supply chain structure composed of four component fabricators, two assembly plants, four product distributors and six retailers are considered (Ref. Fig. 1). Each of the component fabrication factory procures the required raw-materials from different sources. Factories k_1 and k_4 can produce all sorts of components and supply to either of downstream assembler p_1 or p_2. Factory k_2 produce components represented by odd indices only while k_3 produce those represented by even indices; and both can supply to p_1 and p_2 as the same time. The assembly plants in turn stock the component parts and assemble different styles of products according to order specifications. Once customized finishing is done, the final products are transported to the downstream distribution centers. Only three retail outlets are served by each distribution center q as shown in Fig. 1.

The proposed algorithm is tested for different problem instances with varying complexity and size. The minimum target service level at each retail point is set to be 75% of the demand in each period t. The input values for the cost and resource parameters of the models are randomly generated from uniform distributions over certain range of intervals. The range of intervals selected for each input parameter are tabulated in Table 1.

For better performance of the algorithm, the GA parameters are selected through numerical experimentation. In one complete loop of GA simulation, the population size is 25 while the degree of elitism P_E equals 75% of the population. The maximum total generations and allowable interval generations for improvement before termination are set to 10^4 and 10^3, respectively. The proportionality coefficient for fitness evaluation $K = 10^8$, and the penalty factor for infeasible solution $K_p = 10^{14}$. The probability values for crossover and mutation operators are set to be 0.1 and 0.05, respectively. Table 2 summarizes the results of the production-distribution scheduling model (Model-I) obtained by repeated execution of the GA for different problem instances.

An execution of the proposed GA for the simplest problem instance (P_1) with only one product item exhibits the convergence history shown in Fig. 11. The simulation result depicts that the solution approaches the lower bound very rapidly in the first few 100 generations, and improves slowly in higher generations.

The same set of problem instances are also solved with a commercial software called LINGO 8.0. The software gives exact solutions for the first three problem instances (P_1–P_3). For higher problems instances (P_4–P_9), it cannot converge to an optimal solu-

Table 1
Selected range of values for input data of parameters in the test problems.

Parameter	Range of values	Parameter	Range of values	Parameter	Range of values
ρ_{kl}	$(1, 20) \times 10^{-2}$	c_{pij}	$(100, 200)$	u_{ij}	$(0, 3)$
s_{kl}	$(1, 10)$	τ_{pqi}	$(10, 20)$	MTC_{pqi}	$(10, 15) \times 10^2$
η_{kl}	$(10, 100)$	τ_{qri}	$(10, 20)$	MTC_{qri}	$(15, 25) \times 10^2$
σ_{kpj}	$(100, 200)$	γ_{pi}	$(100, 200)$	MDC_{kpj}	$(25, 40) \times 10^4$
ϑ_{kpj}	$(100, 200)$	β_{pi}	$(100, 200)$	MRC_{pi}	$(20, 35) \times 10^2$
λ_{pj}	$(5, 10) \times 10^{-1}$	ω_{pi}	$(200, 300)$	MOC_{pi}	$(7, 150) \times 10^2$
h_{qi}	$(1, 2)$	$I_{qi,0}$	$(5, 10) \times 10^2$	MIC_{qi}	$(2, 3) \times 10^3$
a_{ri}	$(10, 25)$	$J_{pj,0}$	$(2, 5) \times 10^3$	MJC_{pj}	$(20, 25) \times 10^4$
π_{ri}	(200, 300)	$L_{kl,0}$	$(5, 10) \times 10^3$	MLC_{kl}	$(5, 10) \times 10^4$
δ_{lj}	$(1, 25) \times 10^{-2}$	$QN_{ri,0}$	$(10, 100)$	D_{rit}	$(500, 750)$

Table 2
Summary of GA results for Model-I.

Prob. code	Problem size				Aggregate costs in millions of dollars				Runtime (min)
	I	J	L	T	Z_{AS}	Z_{DC}	Z_{RT}	Total Z_{PD}	
P_1	1	5	3	5	6.97	0.23	0.51	7.71	0.61
P_2	5	10	3	5	109.04	1.15	2.65	112.84	3.75
P_3	10	15	4	5	262.76	2.39	5.25	270.40	9.02
P_4	15	25	4	5	699.26	3.46	7.72	710.44	12.35
P_5	20	30	5	5	1,770.15	4.66	10.74	1,785.55	18.04
P_6	25	40	5	5	2,229.56	5.65	13.38	2,248.59	20.67
P_7	30	45	5	5	3,181.07	6.84	15.88	3,203.79	25.41
P_8	40	60	5	5	5,709.43	9.43	21.60	5,740.45	38.70
P_9	50	75	6	5	6,903.99	11.75	27.07	6,942.81	49.51
P_{10}	100	150	10	5	25,672.12	23.82	55.68	25,751.62	124.26
P_{11}	200	300	10	5	101,539.45	46.76	115.92	101,702.13	375.87

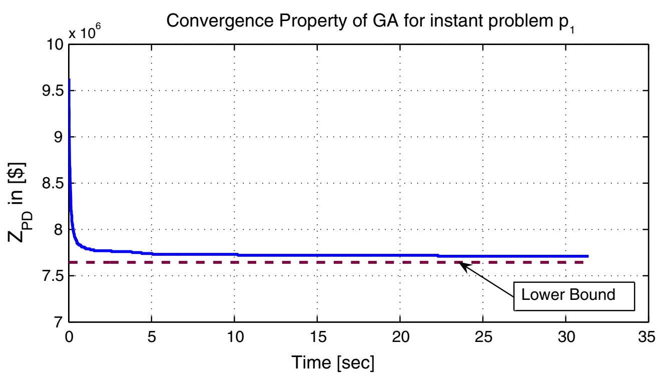

Fig. 11. Convergence of GA for test problem P_1.

tion in the first 48 h, but bears a lower bound solution of the LP relaxation. For the last two problems (P_{10} and P_{11}), the number of constraints and integer variables involved exceed the allowable limitations of the software to produce any feasible solution. Comparing the results obtained by GA with that of the lower bound from LINGO for identical problem instance, the closeness rating factor is computed by:

$$C.R. = \left(1 - \frac{Z_{PD,GA} - Z_{PD,LB}}{Z_{PD,LB}}\right) \times 100\% \tag{50}$$

For the second sub-problem, however, the software gives optimal solutions for a wide range of problem sizes in less than two hours of execution time. Therefore, the GA solution approach is not extended beyond Phase-1 to include the second model. A summary of results obtained by LINGO, and a comparison with the GA solutions are summarized in Table 3.

5. Conclusion

This paper addresses the dynamic scheduling of materials replenishment, component fabrication, customized assembly and distribution of products in a multi-stage BTO supply chain manu-

Table 3
Summary of LINGO results and comparison with GA.

Prob. code	LB solution for Phase-1 aggregate costs in millions				Comp. with GA ($C.R.$(%))	Optimal solutions for Phase-2 aggregate costs in millions		
	Z_{AS}	Z_{DC}	Z_{RT}	Total Z_{PD}		Z_{CF}	Z_{RM}	Total Z_{CM}
P_1	6.94	0.19	0.52	7.64	99.17	2.52	0.05	2.57
P_2	108.79	1.10	2.58	112.47	99.67	99.68	10.48	110.16
P_3	262.16	2.25	5.14	269.55	99.69	249.78	53.43	303.21
P_4	698.20	3.29	7.56	709.05	99.80	1153.44	224.08	1377.52
P_5	1769.10	4.37	10.35	1783.81	99.90	1838.56	318.25	2156.81
P_6	2227.42	5.24	12.89	2245.56	99.87	2200.65	489.73	2690.38
P_7	3177.85	6.40	15.36	3199.61	99.87	3099.46	662.02	3761.47
P_8	5706.28	8.73	20.62	5735.63	99.92	5524.14	1546.08	7070.22
P_9	6892.00	11.00	32.83	6935.83	99.90	6645.94	1685.71	8331.64
P_{10}	x	x	x	x	x	x	x	x
P_{11}	x	x	x	x	x	x	x	x

x = Problem oversized to be solved by LINGO.

facturing system. For the sake of efficient modeling performance, the entire problem is first decomposed into two sub-problems: products assembling and distribution plan, and component fabrication and materials requisition plan. The sub-problems are then formulated as mixed integer linear programming (MILP) models with the objective of minimizing the associated aggregate costs while improving customer's satisfaction. A genetic algorithm based heuristic solution approach is proposed to the production-distribution planning sub-problem. Using some instances of test problems, the best solutions obtained from GA are compared with their lower bounds obtained from LINGO. The GA results indicate that the range of gaps with respect to solution quality is in the order of 99.17–99.92% of its lower bounds. In addition, the proposed GA approach solved all the test problems within a very short period of computational time, mostly in less than 3 h. However, the exact solution approach using LINGO could not provide a solution to the bigger test problems within two days, owing to the complexity of the problem structure.

References

Aghezzaf, E., Raa, B., & Landeghem, H. (2006). Modeling inventory routing problems in supply chains of high consumption products. *European Journal of Operational Research, 169*, 1048–1063.

Amiri, A. (2006). Designing a distribution network in a supply chain system: Formulation and efficient solution procedure. *European Journal of Operational Research, 171*, 567–576.

Chen, C., & Lee, W. (2004). Multi-objective optimization of multi-echelon supply chain networks with uncertain product demands and prices. *Computers and Chemical Engineering, 28*, 1131–1144.

Christensen, W., Germain, R., & Birou, L. (2005). Build-to-order and just-in-time as predictors of applied supply chain knowledge and market performance. *Journal of Operations Management, 23*, 470–481.

Christian, P., & Zimmers, E. Jr., (1999). Age of agile manufacturing puts quality to the test. *Quality Progress, 32*(5).

Demirli, K., & Yimer, A. (2008). Fuzzy scheduling of a build-to-order supply chain. *International Journal of Production Research, 46*(14), 3931–3958.

Ding, H., Benyoucef, L., Xie, X. (2005). A simulation-based multi-objective genetic algorithm approach for networked enterprises optimization. Engineering Applications of Artificial Intelligence.

Eiben, A., & Smith, J. (2003). *Introduction to evolutionary computing* (1st ed.). Amsterdam: Springer.

Ertogral, K., Darwish, M., & Ben-Daya, M. (2006). Production and shipment lot sizing in a vendor–buyer supply chain with transportation cost. *European Journal of operations Research*.

Guillen, G., Badell, M. E. A., & Puigjaner, L. (2006). Simultaneous optimization of process operations and financial decisions to enhance the integrated planning/scheduling of chemical supply chains. *Computers and Chemical Engineering, 30*, 421–436.

Gunasekaran, A., & Ngai, E. (2005). Build-to-order supply chain management: A literature review and framework for development. *Journal of Operations Management, 23*, 423–451.

Holland, J. (1992). *Adaptation in natural and artificial systems* (2nd ed.). University of Michigan: MIT Press.

Jayaraman, V., & Prkul, H. (2001). Planning and coordination of production and distribution facilities for multiple commodities. *European Journal of operations Research, 133*, 394–408.

Ko, H., & Evans, G. (2007). A genetic algorithm-based heuristic for the dynamic integrated forward/reverse logistics network for 3pls. *Computers and Operations Research, 34*, 346–366.

Krajewski, L., Wei, J., & Tang, L. (2005). Responding to schedule changes in build-to-order supply chains. *Journal of Operations Management, 23*, 452–469.

Lakhal, S., Martel, A., Kettani, O., & Oral, M. (2001). On the optimization of supply chain networking decisions. *European Journal of Operational Research, 129*, 259–270.

Lee, Y., Jeong, C., & Moon, C. (2002). Advanced planning and scheduling with outsourcing in manufacturing supply chain. *Computers and Industrial Engineering, 43*, 351–374.

Li, J., Cheng, T., & Wang, S. (2007). Analysis of postponent strategy for perishable items by eoq-based models. *International Journal of Production Economics, 107*, 31–38.

Liang, T. (2006). Distribution planning decisions using interactive fuzzy multi-objective linear programming. *Fuzzy Sets and Systems*.

Lyons, A., Coronado-Mondragon, A., & Kehoe, D. (2004). Factors influencing the design of build-to-order supply chains. *IEEE International Engineering Management Conference*, 1269–1272.

Michalewicz, Z. (1996). Genetic algorithms + data structures = evolution programs. *3rd ed.*. Berlin: Springer-Verlag.

Moon, C., Kim, J., & Hur, S. (2002). Integrated process planning and scheduling with minimizing total tardiness in multi-plants supply chain. *Computers and Industrial Engineering, 43*, 331–349.

Mukhopadhyay, S., & Setoputro, R. (2005). Optimal return policy and modular design for build-to-order products. *Journal of Operations Management, 23*, 496–506.

Nachiappan, S., & Jawahar, N. (2007). A genetic algorithm for optimal operating parameters of vmi system in a two-echelon supply chain. *European Journal of Operational Research*.

Naso, D., Surico, M., Turchiano, B., & Kaymak, U. (2007). Genetic algorithms for supply–chain scheduling: A case study in the distribution of ready-mixed concrete. *European Journal of Operational Research, 177*, 2069–2099.

Neiro, S., & Pinto, J. (2004). A general modeling framework for the operational planning of petroleum supply chains. *Computers and Chemical Engineering, 28*, 871–896.

Perea-Lopèz, E., Ydstie, B., & Grossmann, I. (2003). A model predictive control strategy for supply chain optimization. *Computers and Chemical Engineering, 27*, 1201–1218.

Prasad, S., Tata, J., & Madan, M. (2005). Build to order supply chains in developed and developing countries. *Journal of Operations Management, 23*, 551–568.

Ross, A. D. (2000). Performance-based strategic resource allocation in supply networks. *International Journal of Production Economics, 63*, 255–266.

Sabri, E., & Beamon, B. (2000). A multi-objective approach to simultaneous strategic and operational planning in supply chain design. *International Journal of Management Science, Omega, 28*, 581–598.

Sakawa, M., Kato, K., & Nishizaki, I. (2003). An interactive fuzzy satisfying method for multi-objective stochastic linear programming problems through an expectation model. *European Journal of Operations Research, 145*, 665–672.

Selvarajah, E., & Steiner, G. (2005). Batch scheduling in a two-level supply chain: A focus on the supplier. *European Journal of Operations Research*.

Sen, W., Pokharel, S., & YuLei, W. (2004). Supply chain positioning strategy integration, evaluation, simulation, and optimization. *Computers and Industrial Engineering, 46*, 781–792.

Sharma, A., & LaPlaca, P. (2005). Marketing in the emerging era of build-to-order manufacturing. *Industrial Marketing Management, 34*, 476–486.

Spitter, J., Hurkens, C., Kok, A., Lenstra, J., & Negenman, E. (2005). Linear programming models with planned lead times for supply chain operations planning. *European Journal of Operational Research, 163*, 706–720.

Talluri, S., & Baker, R. (2002). A multi-phase mathematical programming approach for effective supply chain design. *European Journal of Operational Research, 141*, 544–558.

Torabi, S., Ghomi, S., & Karimi, B. (2006). A hybrid genetic algorithm for the finite horizon economic lot and delivery scheduling in supply chains. *European Journal of Operational Research, 173*, 173–189.

Weng, Z., & Parlar, M. (2005). Managing build-to-order short life-cycle products: Benefits of pre-season price incentives with standardization. *Journal of Operations Management, 23*, 482–495.

Yao, A., & Carlson, J. (2003). Agility and mixed-model furniture production. *International Journal of Production Economics, 81*(82), 95–102.