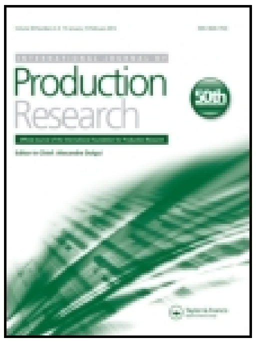

Taylor & Francis
Taylor & Francis Group

International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: https://www.tandfonline.com/loi/tprs20

Multi-objective analysis of an integrated supply chain scheduling problem

Eray Cakici , Scott J. Mason & Mary E. Kurz

To cite this article: Eray Cakici , Scott J. Mason & Mary E. Kurz (2012) Multi-objective analysis of an integrated supply chain scheduling problem, International Journal of Production Research, 50:10, 2624-2638, DOI: 10.1080/00207543.2011.578162

To link to this article: https://doi.org/10.1080/00207543.2011.578162

Published online: 08 Jul 2011.

Submit your article to this journal

Article views: 617

View related articles

Citing articles: 28 View citing articles

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=tprs20

International Journal of Production Research
Vol. 50, No. 10, 15 May 2012, 2624–2638

Multi-objective analysis of an integrated supply chain scheduling problem

Eray Cakici[a]*, Scott J. Mason[b] and Mary E. Kurz[b]

[a]*IBM, Yapi Kredi Plaza, Levent, Istanbul 34330, Turkey;* [b]*Department of Industrial Engineering, Clemson University, 110 Freeman Hall, Clemson, SC 29634, USA*

(*Received 29 August 2010; final version received 25 March 2011*)

We study the problem of minimising the total weighted tardiness and total distribution costs in an integrated production and distribution environment. Orders are received by a manufacturer, processed on a single production line, and delivered to customers by capacitated vehicles. Each order (job) is associated with a customer, weight (priority), processing time, due time, and size (volume or storage space required in the transportation unit). A mathematical model is presented in which a number of weighted linear combinations of the objectives are used to aggregate both objectives into a single objective. Because even the single objective problem is NP-hard, different heuristics based on a genetic algorithm (GA) are developed to further approximate a Pareto-optimal set of solutions for our multi-objective problem.

Keywords: supply chain management; heuristics; multi-criteria decision making; genetic algorithms; math programming

1. Introduction

Problems with two or more objectives are common in real-world applications and are called ‘multi-objective’ or ‘multi-criteria’ optimisation problems. Scheduling problems often require multiple objective analyses. Many problems faced by decision makers involve making a selection from different alternative solutions while satisfying several criteria that are usually in conflict with each other (for example, the cost and service level). Industries – such as aircraft, electronics, semiconductor manufacturing, etc. – have trade-offs in their scheduling problems in which multiple objectives are considered in the process of optimising the overall performance of the system (Cochran *et al.* 2003). A schedule minimising one objective can lead to a poorly satisfied objective elsewhere. Therefore, any methodology applied in such multi-objective scheduling problems has to find a compromise between these conflicting objectives. The aim is to achieve a ‘best-compromise’ solution of all of the objectives. Generally, no such single solution exists, and the decision maker’s preference affects the selection of the best compromise among the set of efficient solutions. Such solutions are also called Pareto-optimal solutions; for those non-dominated solutions, no feasible solution exists that can better satisfy one objective without negatively affecting at least one other criterion. Therefore, many multi-objective methods try to reduce the solution space to the set of efficient solutions (Jaszkiewicz 2002). Special algorithms should be used in order to generate several non-dominated solutions. This is the fundamental difference between a single-objective and multi-objective optimisation problem (Deb 2001).

Costs and service levels are two main objectives of interest in a typical supply chain. Both objectives can be better optimised by collaborative decision-making models. Lack of integration between production and distribution schedules yields substantial inefficiencies and, consequently, poor total system performance. Especially in make-to-order industries, lead times are short and limited inventory is held in the supply chain. Therefore, coordinating production and distribution operations becomes more crucial for satisfying on-time delivery requirements without intermediate storage. In such a situation, delivery times to meet service considerations and consolidation opportunities to reduce transportation costs are highly affected by production schedules. Motivated by the fact that an increasing number of companies are now adopting make-to-order business models, we study the problem of optimising customer service levels and total distribution costs in an integrated production and distribution environment. Because, generally, there is very little or no inventory and production costs are typically independent of the processing sequence, transportation costs are the main driver for minimising total system cost. Customer service levels are measured by weighted tardiness, where tardiness is the positive difference between a job’s delivery

*Corresponding author. Email: eray.cakici@gmail.com

ISSN 0020–7543 print/ISSN 1366–588X online
© 2012 Taylor & Francis
http://dx.doi.org/10.1080/00207543.2011.578162
http://www.tandfonline.com

time and due time. Orders are received by a manufacturer, processed on a single production line, and delivered directly to customers by capacitated vehicles without visiting any other locations. Customers are positioned in different locations of the network and a fixed customer-dependent transportation cost is incurred for each delivery. Every order (job) is associated with a customer, weight (priority), processing time, due time, and size (volume or storage space required in the transportation unit). A mathematical model is presented in which weighted linear combinations of the objectives are used to aggregate objectives into a single objective. By changing the weights of objective functions, different solutions can be achieved through the proposed mathematical model. For example, suppose T and C are two objectives of interest. Then, a single objective of the problem to be solved via mathematical programming is $\alpha T + (1 - \alpha)C$, where α is a constant value (weight). Because even the single objective problem is NP-hard, several genetic algorithm-based approaches are developed to further approximate a Pareto-optimal set of solutions without aggregating objectives for this multi-objective problem.

The rest of the paper is organised as follows. In the next section, we review the previous work on multi-objective supply chain scheduling. In Section 3, we present a mixed-integer programming formulation of the problem under study. Section 4 describes different heuristic algorithms to further approximate the Pareto-optimal set of solutions. Section 5 presents the results of computational experiments conducted to evaluate the performance of the proposed algorithms. We conclude the paper with a summary and suggestions for future research directions in Section 6.

2. Literature review

A recent review of the integrated production and distribution scheduling models in the literature can be found in Chen (2010). The objective functions studied in the supply chain scheduling literature are summarised in Table 1.

In this paper, optimising the trade-off between total weighted tardiness and transportation costs is studied. Relatively few papers exist in the literature studying integrated decisions at a detailed scheduling level in which multiple objectives involve tardiness and transportation costs. Synchronisation of assembly operations with air transportation is investigated by Li *et al.* (2005) for a make-to-order-based computer manufacturer. The objective is to minimise the overall total cost including the total delivery tardiness cost. The problem is divided into two sub-problems. The air transportation allocation is formulated and solved as an integer linear program, and two heuristic approaches are presented for the assembly-scheduling problem. Pundoor and Chen (2005) study minimising the total distribution costs plus maximum tardiness. A set of orders with equal sizes is received at the beginning of the planning horizon, processed by the supplier, and delivered to the customers by capacitated vehicles. Both single and multiple customer cases are studied along with a special case. Either an efficient algorithm or proof of intractability is proposed for different cases of the problem. A heuristic approach incorporating a dynamic programming algorithm is developed for the general case. The authors show that an integrated approach yields significantly better results when compared with a sequential approach in which scheduling decisions are first made for order processing then followed by delivery scheduling decisions. In our study, we assume order sizes are unequal, which is more difficult to solve than problems with equal sizes (Chen 2010). The batching problem in the distribution part involves bin packing, and that problem itself is strongly NP-hard (Garey and Johnson 1979). Hall and Potts (2005) consider different integrated production and distribution problems. The objective is to minimise the sum of total transportation costs and total scheduling cost. A variant of scheduling costs is considered involving total weighted completion time, the maximum lateness, total weighted number of late jobs, and the total tardiness. The production side is modelled as either a single or parallel machine environment. A fixed transportation cost is incurred for each delivery and it is assumed that vehicles have an infinite capacity. The authors present several algorithms and intractability results. Chen and Hall (2007) examine various supply chain configurations in which suppliers provide parts to a manufacturer. Conflict issues are discussed when each party has its own objective, such as minimising total completion times for suppliers and minimising maximum lateness for the manufacturer. Cooperation opportunities and ways to resolve conflicts are also shown under various assumptions of the relative bargaining powers of the suppliers and the manufacturer.

In summary, generally the objective function examining the trade-off between the tardiness-related objective and transportation cost is defined as $\alpha T + (1 - \alpha)C$, where T can be the total weighted tardiness and C is the total distribution cost. It can be seen that when α is close to 0, more emphasis is given to the total distribution cost and when α is close to 1, more emphasis is given to total weighted tardiness. This α value can also be used to normalise the total weighted tardiness and total distribution cost representations, since the objectives are represented by different units (i.e., 'dollars' for the total distribution cost and 'minutes' for tardiness). Another approach often

Table 1. Objective functions studied in the supply chain scheduling literature.

Objective	Reference
Total flow time + total distribution cost	Hall and Potts (2003)
Maximum lateness + total distribution cost	
Number of late jobs + total distribution cost	
Total weighted flow time + total distribution cost	Ji *et al.* (2007)
Inventory holding cost + total distribution cost	Selvarajah and Steiner (2006)
Total flow time + total distribution cost	Averbakh and Zhihui (2007)
Makespan	Zhong *et al.* (2007)
Total flow time + total distribution cost	Mazdeh *et al.* (2007)
Makespan	Wang and Cheng (2007)
Total flow time + setup costs	Qi (2005)
Total setup cost	Cheng (2001)
Total completion time	Lee and Chen (2001)
Makespan	
Inventory holding cost + total distribution cost	Anily and Tzur (2005)
Mean delivery time + total distribution cost	Chen and Vairaktarakis (2005)
Maximum delivery time + total distribution cost	
Total delivery time + total distribution cost	Chen and Lee (2008)
Makespan + lateness	Dawande *et al.* (2006)
Makespan + total inventory holding cost	
Total tardiness + total earliness + total distribution cost	Li *et al.* (2005)
Total delivery time	Li *et al.* (2005)
Total cost	Lei *et al.* (2006)
Weighted sum of total delivery time + total cost	Chen and Pundoor (2006)
Weighted sum of maximum delivery time + total cost	
Total cost	
Sum of completion times + total distribution cost	Hall and Potts (2005)
Maximum tardiness + total distribution cost	
Total tardiness + total distribution cost	
Number of late jobs + total distribution cost	
Maximum tardiness + total distribution cost	Pundoor and Chen (2005)
Total flow time + total distribution cost	Wang and Cheng (2000)
Maximum tardiness + total completion times	Chen and Hall (2007)
Makespan	Geismar *et al.* (2008)

suggested is to minimise one of the criteria as the objective with an additional constraint so that the second constraint is limited by a specified value. In this case, dual variables can be used to evaluate the change in the objective value due to a unit increase on the constraint's right-hand side. To our knowledge, there is no previous research in the supply chain scheduling literature applying multi-objective analysis to tackle this problem in which a set of non-dominated solutions are generated instead of a single solution.

3. Mathematical formulation

3.1 *Problem description*

The problem considered in this study consists of scheduling n orders in an integrated production and distribution system. There exist an infinite number of vehicles, which is a valid assumption for many companies where transportation is outsourced and a fixed transportation cost incurs for each delivery. The objective is to optimise the trade-off between total weighted tardiness and total distribution costs. Orders are received by a manufacturer, processed on a single production line, and delivered to customers by capacitated vehicles. Each order (job) is associated with a customer, weight (priority), processing time, due time, and size (volume or storage space required in the transportation unit). The vehicle capacity is defined as the maximum total size of the jobs that can be delivered together. Only direct deliveries without any intermediate stops are allowed (one customer per trip) as in the case of the full truckload industry. Transportation times are also considered in addition to the processing times. It is assumed that all jobs are available at the beginning of the planning horizon and no preemption is allowed.

3.2 *The model*

This problem is formulated as a mixed-integer programming (MIP) problem. First, the pertinent notation for mathematical formulation is introduced:

J	set of jobs such that $J \in \{1, 2, \ldots, n\}$
B	set of vehicle trips such that $B \in \{1, 2, \ldots, n\}$

Each job is ordered by a specific customer and it has to be delivered directly to that customer. Every vehicle trip is pre-assigned to a specific customer regardless of whether it is performed or not in the optimal solution. The maximum number of trips for each customer is equal to the number of jobs ordered by that particular customer. Therefore, there are a total of n trips introduced and only jobs of the associated customer can be assigned to a particular trip (batch). Trips are numbered the same as the jobs of the corresponding customer. For example, if a particular customer ordered jobs 3 and 5, then batches 3 and 5 are associated with that customer and can include jobs 3 and 5. Λ denotes the set of job-to-trip pairs, where $(i, k) \in \Lambda$ indicates that job i and trip k are destined to same customer. By this definition the problem size is reduced compared with the case introducing all job-to-trip combinations. Instead, only feasible combinations are introduced because jobs of the same customer can only form a delivery together. The vehicle has capacity δ and τ_b is the time required to perform trip $b \in B$. Each delivery incurs a distribution cost, γ_b.

A dummy job 0 is introduced whose processing time, ready time, and weight are each set equal to 0. In the network formulation of the machine scheduling problem, job 0 is required to be both the first and the last job processed on each machine in order to indicate both the starting and finishing of job processing on each machine. Binary decision variable x_{ij} is defined to assist with job sequencing such that $x_{ij} = 1$ if job $i \in J$ immediately precedes job $j \in J$; otherwise, $x_{ij} = 0$. In the distribution part, z_b is introduced to track the delivery trips that are performed such that $z_b = 1$ if trip $b \in B$ is performed; otherwise, $z_b = 0$. The time at which job $j \in J$ finishes its required processing is denoted by C_j. Each trip $b \in B$ starts its delivery at S_b and the time job $j \in J$ is delivered is Γ_j. In order to handle both batch assignments and incompatible job families at the same time, binary decision variable y_{jk} is introduced, where $y_{jk} = 1$ if job $j \in J$ is assigned to trip $k \in B$ where $(j, k) \in \Lambda$; otherwise, $y_{jk} = 0$.

Job $j \in J$ has the following parameters associated with it:

w_j	the weight (priority) of job $j \in J$
v_j	the size (volume) of job $j \in J$
d_j	the due time of job $j \in J$

The tardiness of job $j \in J$, T_j, is calculated as $\max(0, \Gamma_j - d_j)$. The objective functions of interest are to minimise the sum of the total weighted tardiness (TWT) of all jobs, in which $TWT = \sum_{j:j \in J} w_j T_j$, and to minimise the total transportation cost (TC) of all deliveries, in which $TC = \sum_{b:b \in B} z_b \gamma_b$. Jobs are assigned to a single production line (machine) with a unique predecessor and a unique successor, subject to the machine starting and ending its schedule with job 0:

$$\sum_{j \in J: j \neq i} x_{ji} = 1, \quad \forall i \in J, \tag{1}$$

$$\sum_{j \in J: j \neq i} x_{ij} = 1, \quad \forall i \in J. \tag{2}$$

In order to process job j immediately after job i, job $i \in J$ completes p_j time units before job $j \in J$:

$$C_i - C_j + p_j \leq (1 - x_{ij})M, \quad \forall i \in J, \quad \forall j \in J : j \neq 0, i \neq j. \tag{3}$$

Constraints (4)–(9) address the distribution part of the problem. Jobs are assigned to one of the available trips that are associated with the same customer

$$\sum_{k=1}^{n} y_{jk} = 1, \quad \forall j \in J : (j, k) \in \Lambda, \tag{4}$$

and the machine (vehicle) capacity cannot be exceeded:

$$\sum_{j=1}^{n} v_j y_{jk} \leq \delta, \quad \forall k \in B : (j, k) \in \Lambda. \tag{5}$$

A vehicle cannot start its delivery until all jobs to be delivered in the corresponding batch have finished their processing:

$$S_b \geq C_i - (1 - y_{ib})M, \quad \forall i \in J, \quad \forall b \in B : i \neq 0, (i, b) \in \Lambda. \tag{6}$$

Γ_i is the delivery start time plus the delivery time:

$$\Gamma_i \geq S_b + \tau_b y_{ib} - (1 - y_{ib})M, \quad \forall i \in J, \quad \forall b \in B : i \neq 0, (i, b) \in \Lambda. \tag{7}$$

In constraints (3), (6), and (7), $M \geq \sum_{j \in J} p_j$. Each possible trip should be performed if any job is assigned to it:

$$n_b z_b \geq \sum_{i:(i,b) \in \Lambda}^{n} y_{ib}, \quad \forall b \in B, \tag{8}$$

where n_b is the total number of jobs ordered by the customer that trip b is associated with. Finally, the tardiness of the jobs is calculated using the following relationship:

$$T_i \geq \Gamma_i - d_i, \quad \forall i \in J. \tag{9}$$

In the model presented above, different weights are given to each objective function in order to produce diverse efficient solutions. Subsequently, both objectives are aggregated into a single objective by summing up the weighted objectives in which well-suited weights for different objectives are required to obtain useful non-dominated points on the Pareto front.

3.3 *Problem complexity*

Considering the production stage alone by assuming transportation time and costs are equal to 0, our scheduling problem can be described in the $\alpha|\beta|\gamma$ notation of Graham *et al.* (1979) as $1| \quad |\sum w_j T_j$. This reduced problem is NP-hard in the strong sense (Lawler 1977). Due to the greater complexity of our problem, heuristic approaches are needed to produce good solutions. A mathematical model is used to assess the heuristic solution quality for small-sized instances. In this study, different genetic algorithm-based approaches are developed to solve our challenging, practically motivated multi-objective problem.

4. Heuristics

Genetic algorithms have been shown to be a promising technique by many researchers for solving multi-objective optimisation problems (Arroyo and Armentona 2005). A GA is applied to the distribution part of our problem in which an infinite number of capacitated vehicles exists and only direct deliveries (one customer per trip) are allowed. We first describe our chromosomal decoding scheme, infeasibility evaluation methodology, crossover, and mutation operations as follows.

Coding: To decode trip assignments, each job is associated with a random trip number where the maximum number of delivery trips is equal to the number of jobs (see Figure 1 for an example representation).

The existence of capacity limitations and incompatibility (only jobs of the same customer can be delivered together) lead to infeasible solutions while producing new chromosomes. Consider three customers with orders $\{J_1, J_2, J_3, J_4\}$, $\{J_5, J_6, J_7, J_8\}$ and $\{J_9, J_{10}\}$, respectively. In the above example, assignment of jobs 3 and 9 is infeasible because they are assigned to the same trip but ordered by different customers. To overcome the infeasibility issues caused by direct delivery restrictions (when jobs of different customers are randomly assigned to the same trip), trips that are eligible to deliver the jobs of each customer are pre-defined as being similar to the proposed mathematical model, and trip assignments are performed by randomly selecting a trip from the set of pre-determined trips of the associated customer. For each customer, possible trips are pre-determined, one for each job ordered by that customer, and none of the possible trips overlap between customers. For example, jobs of customer 1 can only be

Job	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
Trip	8	6	4	8	5	7	1	3	4	9

Figure 1. Example chromosome representation.

	Customer 1				Customer 2				Customer	
Job	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
Trip	2	3	4	3	6	7	8	7	10	10

Figure 2. Example chromosome modified representation.

	Customer 1				Customer 2				Customer 3	
Parent 1	2	3	4	3	6	**7**	**8**	**7**	**10**	**10**
Parent 2	**1**	**2**	**4**	**2**	**7**	8	6	6	10	9
Child 1	2	3	4	3	6	8	6	6	10	9
Child 2	1	2	4	2	7	7	8	7	10	10

Figure 3. Example of crossover operation.

	Customer 1				Customer 2				Customer 3	
Parent	2	**3**	4	3	6	7	8	7	10	10
Child	2	**1**	4	3	6	7	8	7	10	10

Figure 4. Example of mutation operation.

assigned to a trip randomly chosen between 1 and 4, any trips numbered from 5 to 8 can deliver jobs to customer 2, and jobs of customer 3 can only be assigned to either trip 9 or 10 (see Figure 2 for the modified example representation).

Crossover: Two parent chromosomes and a crossover point are selected randomly. A one-point crossover is applied with a probability of p_c such that a random number is drawn from U(0,1) and crossover at a random point is applied if the number is less than p_c. Job assignments before the crossover point are copied from the first parent, and the rest are copied from the second parent. If crossover is not applied to the parent chromosomes, they are copied directly to the offspring. In both cases, two resulting chromosomes are obtained (Figure 3).

Mutation: From each chromosome of the offspring, a job is randomly selected and assigned to a random trip (batch) of the associated customer according to a pre-defined mutation probability, p_m (Figure 4).

Penalty Function: Feasibility is not guaranteed when a crossover or mutation is performed to produce the next generations. Each infeasible solution is penalised by multiplying both objective values by an exponential function of the number of infeasible trips (deliveries in which capacity is exceeded) to maintain feasibility in the next generations. For example, if a particular solution i has a total weighted tardiness of $TWTi$ and includes two infeasible trips, then the updated total weighted tardiness is $TWT_i \times e^2$.

Our two main goals in generating solutions for this multi-objective problem are: (1) superior convergence to the Pareto-optimal front and (2) diversified non-dominated solutions. The fast and elitist Non-Dominated Sorting Genetic Algorithm II (NSGA-II) is used to obtain widely distributed Pareto-optimal solutions in an effective manner. Deb *et al.* (2002) show that, in many problems, NSGA-II is able to perform better than other multi-objective evolutionary algorithms with respect to fitness (quality) and spread (diversity) of the solutions. The algorithm uses an explicit diversity-preserving mechanism and runs in $O(MN^2)$ time where M is the number of objectives and N is the population size.

The algorithm starts with randomly generating N solutions as the initial population P_0. Then, an offspring population Q_0 of size N is generated from the initial population by genetic operations such as crossover and mutation. After generating the offspring population, P_0 and Q_0 are combined into mating pool R_0. Each solution in R_0 is evaluated based on the non-dominated sorting scheme (Figure 5) and a crowding distance measure (Figure 6). Solutions are first sorted in non-decreasing order of fronts and then non-increasing order of crowding distance. The next generation P_1 is created by adding the individuals from the sorted list until the size of P_1 exceeds N. A non-dominated sorting approach helps to classify individuals in a fast manner into different fronts based on their fitness. N solutions are selected from the mating pool by transferring solutions of sorted fronts. When the population size N is exceeded by transferring a particular front, selection is performed based on the crowding distance. The crowding

```
1. For each individual solution $i$, identify the number of solutions that dominates $i$, $n_i$, and create $S_i$, the set of solutions that $i$ dominates such that solution $i$ dominates solution $j$ if $i$ is better in both objectives. And set $k = 1$.
2. Each unplaced solution $i$ with $n_i = 0$ is placed in the $k$th front, $F_k$.
3. For each solution $i$ in $F_k$, find each individual $j$ that is dominated by $i$ and reduce $n_j$ by one.
4. If all solutions are placed in a front, stop. Otherwise, $k \leftarrow k+1$ and go to Step 2
```

Figure 5. Fast non-dominated sorting.

1. For each objective m, sort all n_k solutions in a particular front k, F_k, in non-decreasing order of the mth objective function value and calculate the crowding distance CD_{im} of each individual i with respect to objective m as follows:

$$CD_{im} = \frac{f_m(i+1) - f_m(i-1)}{f_m^{\max} - f_m^{\min}}, \quad i = 2, \ldots, n_k - 1$$

$f_m^{\max}$ and $f_m^{\min}$ are the maximum and minimum values of the mth objective in the population

$$CD_{im} = E, \quad i = 1 \text{ and } i = n_k$$

A very large number E is assigned to the boundary solutions in every front.

2. Compute the total crowding distance of each solution i in which M is the number of objectives

$$CD_i = \sum_{m=1}^{M} CD_{im}$$

Figure 6. Crowding distance.

Step 1. Initialization. Generate N feasible starting solutions. Set these solutions as P_k where k=0

Step 2. Offspring. Using crossover and mutation operators, generate offspring population Q_k from parent population P_k, both of size N.

Step 3. Mating Pool. Combine parent and offspring populations to create $R_k = P_k \cup Q_k$. Sort R_k based on non-domination (identify different fronts: $F_1, F_2, F_3 \ldots$)

Step 4. Next Generation. From 2N solutions in R_k, select N best solutions to form P_{k+1} as follows:

-Select N solutions from R_k by transferring solutions of sorted fronts. Only for the last front, selection is performed based on the crowding distance.

Step 5. Generation update. $k \leftarrow k+1$. If $k \geq k_{\max} + 1$, stop. Otherwise, go to Step 2.

Figure 7. Steps of NSGA-II.

distance measure preserves the diversity when selecting solutions from the same front, and as non-dominated individuals (lowest rank front) are selected every time to transfer individuals, elitism is also attained in the search. The algorithm continues to generate offspring populations and update the next generations until the stopping criterion is met. The stopping criteria can be defined by a limit on the total computational time or the total number of generations. Figure 7 illustrates the implementation of NSGA-II in this paper.

To further improve the convergence to the Pareto-optimal front and diversify the solution space, we introduce immigration, which is a simple move of solutions from one generation to the next generation without any changes and propose two variants of NSGA (NSGA-II without immigration is NSGA-II-Type0). Instead of copying all N sorted solutions, we add new randomly generated chromosomes to the next mating pool by immigration. In the first variant, NSGA-II-Type1, a constant percentage of the next generation is created by immigration, where 10% is chosen in this study. The second variant, NSGA-II-Type2, transfers only the solutions that are non-dominated over all solutions in the mating pool (first front) to the next generation. The remaining solutions are generated through immigration. A detailed example of a NSGA-II iteration is given in Appendix A.

Genetic algorithms used to form delivery trips are incorporated with dispatching rules to schedule jobs on the production side. Given delivery trips of orders, there exists an optimal schedule to minimise the TWT in which there is no idle time between the processing of orders at the manufacturer, and orders delivered together are also

Table 2. Heuristic descriptions.

Heuristic	Trip assignment (distribution)	Batch sequencing (production)
MO1	NSGA-II-Type0	BSR1
MO2	NSGA-II-Type0	BSR2
MO3	NSGA-II-Type1	BSR1
MO4	NSGA-II-Type1	BSR2
MO5	NSGA-II-Type2	BSR1
MO6	NSGA-II-Type2	BSR2

processed consecutively (Pundoor and Chen 2005). Therefore, once the delivery trips (batches) are formed via the genetic algorithm, the problem reduces to a single machine scheduling problem with the objective of minimising the total weighted tardiness in which each batch can be viewed as a single job while sequencing the batches.

4.1 *Batch sequencing*

Because every order will be delivered immediately after the corresponding batch completes its processing, a modified due time d'_j is introduced such that $d'_j = d_j - t$ for job $j \in J$. A composite dispatching rule, ATC (Vepsalainen and Morton 1987), is applied to sort the batches in two ways.

BSR1: Similar to Perez *et al.* (2005), batches are sorted in non-increasing order of $\sum_{j:y_{jb}=1}^{v} I_j(t)$, where

$$I_j(t) = \frac{w_j}{p_j}\exp\left(\frac{-\max(d'_j - p_j - t, 0)}{k\bar{p}}\right).$$

k is the look-ahead parameter and $\bar{p}$ is the average processing time of all the jobs. Look-ahead parameter k should be determined based on the problem characteristics. As k increases, the ATC rule reduces to the weighted shortest processing time (WSPT) and smaller values of k can help to prioritise late jobs or jobs with minimum slack time. Every time a machine becomes idle at time t, a new sorting index is calculated and the batch with the highest index is chosen.

BSR2: An aggregated batch due time D_b, weight W_b, and processing time P_b are calculated for each batch $b \in B$ as follows:

$$D_b = \frac{\sum_{j:y_{jb}=1}^{n} w_j d'_j}{\sum_{j:y_{jb}=1}^{n} w_j}, \quad W_b = \sum_{j:y_{jb}=1}^{n} w_j, \quad P_b = \sum_{j:y_{jb}=1}^{n} p_j.$$

The batch due time D_b is the weighted average of the modified due times of the jobs included in the batch and the batch processing time P_b is the sum of the processing times of the jobs in the batch. Every time a machine becomes available, the batch sorting index $BI_b(t)$ is used to select the batch for processing. The $BI_b(t)$ index for batch $b \in B$ at time t is

$$BI_b(t) = \frac{W_b}{P_b}\exp\left(\frac{-\max(D_b - P_b - t, 0)}{k\bar{P}}\right).$$

Similar to BSR1, the sorting index requires the use of the look-ahead parameter, k, and $\bar{P}$ is calculated as the average of all batch processing times.

We employ both sequencing rules combined with the three variants of NSGA-II discussed above. As a result, we examine six different heuristics approaches for our multi-objective supply chain scheduling problem (Table 2).

5. Computational results

5.1 *Test problems*

An extensive set of problem instances was used to test the performance of our proposed approach (Table 3). We consider three different sets of jobs (8, 20, and 50). The weights of the jobs are randomly generated integer

Table 3. Experimental design.

Factor	Level	Level description
Number of customers	2	2, 4
Job due times	3	DU$[p_{\min}+t_{\min}, \lambda/2((p_{\min}+p_{\max})n+(t_{\min}+t_{\max}))]$, $\lambda = 0.5, 1, 1.5$
Job weights	2	DU[1,5], DU[1,10]
Job sizes	2	DU[1,25], DU[1,50]
Job processing times	1	DU[1,10]
Transportation times	1	DU[10,100]
Number of jobs	3	8, 20, and 50
Total	72	

values from [1, 5] and [1, 10]. Two different levels of job sizes are generated from a discrete uniform distribution in the ranges [1, 25] and [1, 50] where the vehicle capacity is 50. The number of customers, job due times, processing times, transportation times, and transportation costs are generated similar to Pundoor and Chen (2005). We examine two levels for the number of customers, 2 and 4. The transportation times required to travel to each customer are discrete random numbers between 10 and 100. Transportation costs are equal to transportation times. A random integer is generated from [1, 10] for each job's processing time. Job due times are generated from a discrete uniform distribution DU$[p_{\min}+t_{\min}, \lambda/2((p_{\min}+p_{\max})n+(t_{\min}+t_{\max}))]$, where $p_{\min}$ and $p_{\max}$ are the minimum and maximum processing times, and likewise $t_{\min}$ and $t_{\max}$ are the minimum and maximum transportation times. Three different levels of the due time tightness factor λ were investigated (0.5, 1, and 1.5). Due times are also characterised by processing and transportation times.

For each of the 72 test combinations ($3 \times 2 \times 2 \times 2 \times 3$), 10 random instances were generated. Therefore, a total of 720 problem instances were used to examine the performance of our proposed algorithms.

5.2 *Mathematical model solutions*

The optimisation model is implemented in AMPL and solved by CPLEX 11.1 to generate Pareto-optimal solutions for small-sized (eight-job) problems. Both objectives are aggregated into a single objective by summing the weighted objectives. A non-negative scaling parameter α is used to assign different weights to objectives such that α for TC and $(1-\alpha)$ for TWT. Because it is not possible to investigate every combination in a reasonable time (within 2 hours), we employ α in declines of 0.1 starting from 1 and resulting in 11 different objective functions for the same problem instance. For eight-job instances, these 11 points may not be representative of the entire solution space. Therefore, optimal solutions for the given objectives are placed in the set of partial Pareto-optimal solutions, PPO, and compared with the non-dominated solutions found by each heuristic.

5.3 *Heuristic solutions*

Our heuristic algorithms are implemented in Visual Basic for Applications (Excel 2007). All tests are performed on a PC with an Intel Pentium Dual-Core Processor (3.39 GHz CPU speed) with 3 GB RAM. Look-ahead parameter k is selected as 1.5 in the batch sequencing rules. To employ GAs, the population size is set as 100. Algorithms are iterated 100 times, in other words 100 generations are created. The crossover probability, p_c, is 0.8, and 0.1 is used for the mutation probability, p_m. Many researchers use Pareto front solutions as the performance measure for algorithms developed for multi-objective problems (Cochran *et al.* 2003). Such an approach helps to compare the performances of each heuristic with respect to all objectives at the same time. We combine all Pareto front solutions achieved from different heuristics in a new set of non-dominated solutions, the Pareto best front ($PBF\_MO1-6(1rep)$). Because there can be overlapping solutions between heuristics or a solution can be dominated by another solution of a different heuristic, the number of solutions in $PBF\_MO1-6(1rep)$ is less than or equal to the aggregate of the Pareto front solutions of all heuristics. We define the performance of the heuristics based on the number of Pareto front solutions that have been contributed to the set of solutions in $PBF\_MO1-6(1rep)$ by each heuristic. Let $ND(H)$ be the Pareto front solutions in $PBF\_MO1-6(1rep)$ that are obtained by heuristic H and $ND(B)$ be the total number of non-dominated solutions in $PBF\_MO1-6(1rep)$.

Table 4. Heuristic results (%).

	Level	MO1	MO2	MO3	MO4	MO5	MO6
No. of jobs	8	40.9	85.7	42.6	90.4	42.2	88.5
	20	8.0	34.9	6.7	40.9	4.6	29.0
	50	17.9	47.6	11.5	38.1	2.8	5.9
No. of customers	2	22.6	49.8	20.6	53.4	16.5	37.4
	4	20.9	61.5	19.5	60.4	17.3	52.1
Job due time Tightness factor	0.5	34.2	53.7	32.0	51.9	28.1	40.6
	1	15.3	54.8	14.4	59.8	11.2	45.0
	1.5	15.8	57.4	13.6	58.6	11.1	46.6
Job sizes	DU[1,25]	21.6	51.9	20.9	51.3	17.0	40.9
	DU[1,50]	22.0	58.8	19.2	62.4	16.7	47.3
Job weights	DU[1,5]	21.4	55.1	19.5	57.0	16.3	44.3
	DU[1,10]	22.2	55.5	20.6	56.6	17.4	43.8
Averages		21.8	55.3	20.1	56.8	16.8	44.0

The performance ratio of a particular heuristic H, $PR(H) = ND(H)/ND(B)$, is computed to assess the solution quality of the different heuristics in all instances. The performance ratios of each heuristic over all problem instances are given in Table 4. Heuristics employing BSR2 in the production scheduling stage (aggregated batch due times, processing times, and weights are utilised for the jobs of the same trip) outperform the heuristics with BSR1. (An index is computed for each job and the total index of a particular trip's jobs is considered for sequencing.) The best two heuristics are highly competitive. On average, 56.8% of $PBF\_MO1 - 6(1rep)$ is obtained by MO4 (10% of the next generation is created by immigration and BSR2 is used as the batch sequencing rule), whereas 55.3% of $PBF\_MO1 - 6(1rep)$ is obtained by MO2 (BSR2 is applied with no immigration). As the number of jobs increases, heuristics without immigration perform better. MO2 produces the most Pareto best front solutions when the number of job sizes is large, due times are tight, and job sizes are small. MO2 also produces more Pareto best front solutions than MO4 when more customers are positioned in the supply chain. There is also a slight increase in solution times when immigration is employed. (For example, in 50-job instances, MO2 takes 121 seconds on average, whereas MO4 averages 130 seconds.)

5.4 *Comparison of PPO and PBF_MO1 – 6(1rep)*

We now examine the performance of heuristics relative to the mathematical modelling approach. To obtain a better understanding of the solution quality, a Pareto super front (PSF) is formed by combining PPO and $PBF\_MO1 - 6(1rep)$. Similar to computing individual heuristic performances, the quality of the solutions in $PBF\_MO1 - 6(1rep)$ and PPO is evaluated by the number of solutions contributed to PSF. For example, the performance ratio of $PBF\_MO1 - 6(1rep)$, $PR(H*)$, is $ND(H*)/ND(S)$, where the number of Pareto front solutions achieved by heuristics that are also non-dominated in PSF is defined as $ND(H*)$ and $ND(S)$ is the total number of non-dominated solutions in the Pareto super front. Table 5 shows the results for each level of the experimental design factors when the approaches are employed in eight-job instances. Both approaches produce a significant number of non-dominated solutions for the super front. The heuristics are able to find new non-dominated solutions when compared with solutions of the mathematical modelling approach. On average, 16.6% of the Pareto super front is new solutions produced by the heuristics in addition to those obtained from the optimisation model using different objective functions. On the other hand, heuristics can produce solutions in a much faster manner; on average, GAs run in 24 seconds, whereas mathematical models take over 1.5 hours to achieve solutions for each eight-job instance.

5.5 *An example problem*

We present both a mathematical model and heuristic solutions for an example eight-job problem instance in order to obtain a better understanding of our solution approaches and performance evaluation. Figure 8 shows a plot of the non-dominated solutions of the mathematical models (PPO), along with solutions achieved with the heuristics

Table 5. Mathematical models versus heuristics (%).

		PPO	*PBF_MO*1 − 6(1*rep*)
No. of customers	2	82.9	55.5
	4	83.9	45.3
Job due time Tightness factor	0.5	76.0	56.7
	1	84.2	38.5
	1.5	90.0	56.0
Job sizes	DU[1,25]	77.0	56.7
	DU[1,50]	89.8	44.1
Job weights	DU[1,5]	81.4	58.4
	DU[1,10]	85.4	42.5
Averages		83.4	50.4

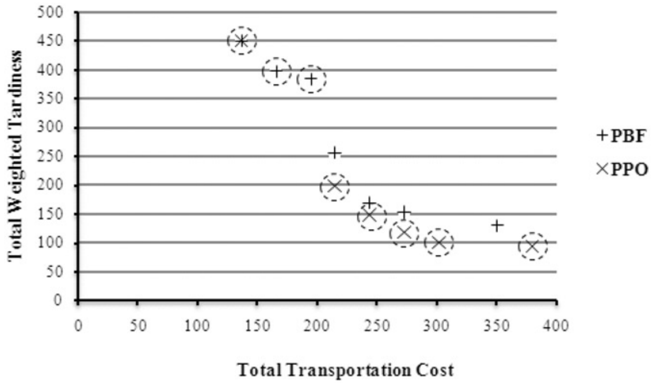

Figure 8. Solution plot for an example problem.

Table 6. Heuristic performances (%) compared with *PBF_MO*2, 4(30*rep*).

		One replication		10 replications		20 replications		30 replications	
	Level	MO2	MO4	MO2	MO4	MO2	MO4	MO2	MO4
No. of jobs	8	94.16	99.42	99.57	100.00	99.71	100.00	99.79	100.00
	20	59.65	62.68	73.76	78.10	78.75	80.78	81.45	81.60
	50	49.33	42.10	58.90	42.15	59.26	42.85	61.71	43.11
No. of customers	2	65.27	67.03	75.45	72.17	77.38	73.40	78.93	73.83
	4	70.16	69.13	79.35	74.63	81.12	75.73	83.04	75.96
Job due time Tightness factor	0.5	68.63	69.74	76.90	73.45	78.49	74.39	79.44	74.63
	1	65.84	67.39	74.06	74.20	78.13	74.42	80.26	74.65
	1.5	68.67	67.12	81.23	72.62	81.24	74.90	83.26	75.49
Job sizes	DU[1,25]	68.39	66.45	75.51	70.56	78.71	71.12	79.22	71.58
	DU[1,50]	67.03	69.72	79.29	76.28	79.80	77.99	82.75	78.23
Job weights	DU[1,5]	69.59	66.13	77.08	74.08	80.10	74.44	81.91	74.91
	DU[1,10]	65.84	70.04	77.72	72.75	78.40	74.66	80.06	74.88
Averages		67.71	68.08	77.40	73.41	79.25	74.55	80.98	74.90

(*PBF_MO*1 − 6(1*rep*)) for the same example problem instance. *PSF* solutions, which are non-dominated solutions over all solutions obtained by both approaches, are circled. Even if the heuristics initially generated the same number of non-dominated solutions as the mathematical model, four of them were dominated by mathematical model solutions and one solution (136, 450) is obtained by both. In total, *PSF* includes eight solutions – five of them obtained by only mathematical models, and heuristics provide two new non-dominated solutions where one solution is obtained by both approaches. Therefore, the performance ratio of the mathematical models is 75% (6/8), and 37.5% (3/8) is the heuristics' performance ratio. Given the super front, a decision-maker can evaluate the trade-off between different solutions. For example, with a slight increase in the total transportation cost, one may prefer solution (214, 199) instead of solution (194, 384) to achieve a significant reduction in the total weighted tardiness.

5.6 *Heuristic solutions after 30 replications*

In the above computational studies, each instance is replicated only once in order to make a fair comparison with mathematical model solutions. We further evaluate the performance of the heuristics with respect to replication times by replicating each instance 30 times for the best two heuristics (MO2 and MO4). Another set of experiments was run and *PBF_MO*2, 4(30*rep*) was formed by only considering the solutions achieved by the two heuristics after 30 replications. The performance ratios of each heuristic over all problem instances and all replications are given in Table 6.

As we replicate the same instance multiple times, we continue to get similar results. Both heuristics are still quite competitive and MO2 (BSR2 is used in the batch sequencing phase with no immigration) constantly produces more Pareto best front solutions than MO4 (10% of the next generation is created by immigration where BSR2 is used) for large-sized instances. For small-sized problems, 10 replications with MO4 are sufficient to achieve all non-dominated solutions found by replicating both heuristics 30 times. Running both of the heuristics in parallel may be preferred for large-sized instances because each heuristic can find different solutions of $PBF\_MO2, 4(30rep)$. For example, in 50-job problem instances, only 61.71% of $PBF\_MO2, 4(30rep)$ is found by MO2 after 30 replications. The conclusions concerning other levels are more indistinct. Performance ratios are either close to each other, or one heuristic's dominance over another changes after a certain number of replications (i.e. when the due time tightness factor is 1). Overall, we can state that both heuristics are competitive and able to contribute to the Pareto best front with different solutions. Thus, a decision-maker may prefer employing both of them together to obtain a better Pareto front.

6. Conclusions and future research

In this paper, we investigate integrated production and distribution planning decisions in a supply chain at a detailed scheduling level. Our problem involves multiple objectives: minimising the weighted tardiness and minimising the total distribution costs. We develop different genetic algorithms combined with dispatching rules to handle both objectives of our supply chain scheduling problem and find an approximation of the Pareto-optimal set. The algorithms use the fast and elitist non-dominated sorting scheme of Deb *et al*. (2002). The solutions obtained by each heuristic are compared with the Pareto best front, a combination of Pareto front solutions from all heuristics. To further assess the solution quality of the heuristics, another set of solutions for each small-sized instance is generated using different objective functions in the proposed mathematical model.

Algorithms employing BSR2 on the production side (aggregated batch due dates, processing times, and weights are utilised for jobs of the same trip) produce more Pareto best front solutions than those with BSR1 (where an index is computed for each job and a total index of a particular trip's jobs is considered for sequencing). Heuristics also find more Pareto best front solutions without immigration in large-sized problem instances. Computational tests have shown that both the mathematical modelling and heuristics approaches are competitive and can produce a significant number of non-dominated solutions for the super front. Heuristics are able to find new non-dominated solutions. On average, a significant fraction of the Pareto super front is composed of new solutions produced by heuristics in addition to those obtained from the optimisation model using different objective functions for small-sized instances. These results indicate that heuristics are not only able to produce the same Pareto front solutions, but also new solutions because it is not practical to run the mathematical model for every combination of the weighted objectives.

To our knowledge, this research is the first study tackling a multi-objective supply chain scheduling problem by generating an approximate set of Pareto-optimal solutions. There is still a vast area of research problems (Table 1) in which multi-objective evolutionary algorithms can be applied. The problem studied in this paper can be extended by considering multiple machines in the production stage and involving routing decisions in order deliveries. Another interesting research topic would be to examine the different chromosome encodings for NSGA-II in which production schedules could also be involved.

References

Anily, S. and Tzur, M., 2005. Shipping multiple items by capacitated vehicles: An optimal dynamic programming approach. *Transportation Science*, 39, 233–248.

Arroyo, J.E.C. and Armentano, V.A., 2005. Genetic local search for multi-objective flowshop scheduling problems. *European Journal of Operational Research*, 167, 717–738.

Averbakh, I. and Zhihui, X., 2007. On-line supply chain scheduling problems with preemption. *European Journal of Operational Research*, 173, 500–504.

Chen, B. and Lee, C., 2008. Logistics scheduling with batching and transportation. *European Journal of Operations Research*, 189, 871–876.

Chen, Z.L. and Vairaktarakis, G.L., 2005. Integrated scheduling of production and distribution operations. *Management Science*, 51 (4), 614–628.

Chen, Z.L. and Pundoor, G., 2006. Order assignment and scheduling in a supply chain. *Operations Research*, 54 (3), 555–572.

Chen, Z.L. and Hall, N.G., 2007. Supply chain scheduling: Conflict and cooperation in assembly systems. *Operations Research*, 55, 1072–1089.

Chen, Z.L., 2010. Integrated production and outbound distribution scheduling: Review and extensions. *Operations Research*, 58, 130–148.

Cheng, T.C.E., 2001. Single supplier scheduling for multiple deliveries. *Annals of Operations Research*, 107, 51–63.

Cochran, J., Horng, S., and Fowler, J., 2003. A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines. *Computers & Operations Research*, 30, 1087–1102.

Dawande, M., *et al.*, 2006. Supply chain scheduling: Distribution systems. *Productions and Operations Management*, 15, 243–261.

Deb, K., 2001. *Multi-objective optimization using evolutionary algorithms*.Chichester, UK: Wiley.

Deb, K., *et al.*, 2002. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6, 182–197.

Garey, M.R. and Johnson, D.S., 1979. *Computers and intractability: A guide to the theory of NP-completeness*. New York: W.H. Freeman.

Geismar, H.N., *et al.*, 2008. The integrated production and transportation scheduling problem for a product with a short lifespan. *INFORMS Journal on Computing*, 20, 21–33.

Graham, R., *et al.*, 1979. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5, 237–287.

Hall, N.G. and Potts, C.N., 2003. Supply chain scheduling: Batching and delivery. *Operations Research*, 51, 566–584.

Hall, N.G. and Potts, C.N., 2005. The coordination of scheduling and batch deliveries. *Annals of Operations Research*, 135, 41–64.

Hoogeveen, H., 2005. Multi-criteria scheduling – Invited review. *European Journal of Operational Research*, 167, 592–623.

Jaszkiewicz, A., 2002. Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137, 50–71.

Ji, M., He, Y., and Cheng, T.C.E., 2007. Batch delivery scheduling with batch delivery cost on a single machine. *European Journal of Operational Research*, 176, 745–755.

Lawler, E., 1977. A pseudopolynomial time algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, 1, 331–342.

Lee, C.Y. and Chen, Z.L., 2001. Machine scheduling with transportation considerations. *Journal of Scheduling*, 4, 3–24.

Lei, L., *et al.*, 2006. On the integrated production, inventory, and distribution routing problem. *IIE Transactions*, 38, 955–970.

Lenstra, J.K., Rinnooy-Kan, A.H.G., and Brucker, P., 1977. Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, 1, 343–362.

Li, C.L., Vairaktarakis, G.L., and Lee, C.Y., 2005. Machine scheduling with deliveries to multiple customer locations. *European Journal of Operational Research*, 164 (1), 39–51.

Li, K.P., Ganesan, V.K., and Sivakumar, A.I., 2005. Synchronizing scheduling of assembly and multi-destination air-transportation in a customer electronics supply chain. *International Journal of Production Research*, 43 (13), 2671–2685.

Mazdeh, M.M., Sarhadi, M., and Hindi, K.H., 2007. A branch-and-bound algorithm for single-machine scheduling with batch delivery minimizing flow times and delivery costs. *European Journal of Operational Research*, 183, 74–86.

Perez, I.C., Fowler, J.W., and Carlyle, W.M., 2005. Minimizing total weighted tardiness on a single batch process machine with incompatible job families. *Computers and Operations Research*, 32, 327–341.

Pinedo, M., 2002. *Scheduling: Theory, algorithms, and systems*. Engelwood Cliffs, NJ: Prentice Hall.

Pundoor, G. and Chen, Z.L., 2005. Scheduling a production-distribution system to optimize the tradeoff between delivery tardiness and distribution cost. *Naval Research Logistics*, 52, 571–589.

Qi, X.A., 2005. Logistics scheduling model: Inventory cost reduction by batching. *Naval Research Logistics*, 52, 312–320.

Selvarajah, E. and Steiner, G., 2006. Batch scheduling in a two-level supply chain – a focus on the supplier. *European Journal of Operational Research*, 173, 226–240.

Vepsalainen, A. and Morton, T., 1987. Priority rules for job shops with weighted tardiness costs. *Management Science*, 33, 1035–1047.

Wang, G. and Cheng, T.C.E., 2000. Parallel machine scheduling with batch delivery costs. *International Journal of Production Economics*, 68, 177–183.

Wang, X. and Cheng, T.C.E., 2007. Machine scheduling with an availability constraint and job delivery coordination. *Naval Research Logistics*, 54, 11–20.

Zhong, W., Dosa, G., and Tan, Z., 2007. On the machine scheduling problem with job delivery coordination. *European Journal of Operational Research*, 182, 1057–1072.

Appendix A: An example iteration of NSGA-II

Suppose that there are three customers with orders $\{J_1, J_2, J_3, J_4\}$, $\{J_5, J_6, J_7, J_8\}$, and $\{J_9, J_{10}\}$, respectively, as in the previous example and we set the initial population size as 10. Our aim is to show how an iteration of NSGA-II-type0 (without immigration) takes place.

(1) A population of strings is generated randomly or transferred from the previous generation (Figure A1).

(2) An offspring population is generated through crossover and mutation operations (Figure A2). In our study, crossover and mutation are not applied deterministically. Therefore, there can be solutions that are tranferred directly to the offspring population, such as C3–C4 (parent solutions P3–P4 but crossover was not applied). Moreover, infeasibilities are penalised. For example, solutions C7 and C8 are obtained from parent solutions P8 and P9 where the first five job assignments of the parent solutions are exchanged. Since for C8 all jobs of customer 2 are assigned to the same trip (trip 8), the vehicle capacity is exceeded and the objective values are multiplied by an exponential function of the number of infeasible trips ($1279 \times e^1 = 3477$ and $6596 \times e^1 = 17{,}390$). Finally, an example of mutation can be found in solution C6. Crossover is applied to parent solutions P3 and P8 and offspring solutions C5 and C6 are formed. Then, offspring solution C6 is selected for mutation in which a random trip (trip 1 instead of trip 2) is assigned to job 1.

(3) All parent and offspring solutions are combined. The fronts for the combined populations are determined as well as the crowding distance in each front. The combined populations are sorted based on the front and the crowding distance in each front (Figure A3). The first 10 individuals (above the dashed line) are selected to be the parent population of the next generation. Since the number of individuals in the first front exceeds 10, crowding distances are used to identify those to be transferred. A very large number *E*, which is assigned to the boundary solutions of each objective in every front, is set as 9999 in the crowding distance calculation of the above example. Note that although some solutions may have the same objective values, they may correspond to different solution configurations, and the crowding distances differ depending on how the neighbouring solutions are located in objective space.

	Solution	**Customer 1**				**Customer 2**				**Customer 3**		**Objective Values**	
	#	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	*TC*	*TWT*
Assigned Trips	**P1**	2	3	4	3	6	7	8	7	10	10	1373	6522
	P2	3	4	1	4	5	8	5	8	10	10	1349	6623
	P3	4	1	1	2	6	5	6	6	9	10	1396	7340
	P4	1	2	2	1	8	7	8	8	9	9	1317	6974
	P5	4	2	3	3	8	7	6	5	10	9	1519	7657
	P6	2	3	4	2	5	7	7	5	10	10	1349	6623
	P7	4	2	2	1	6	5	6	5	9	10	1469	7168
	P8	2	2	1	1	7	8	8	8	9	9	1317	6974
	P9	4	3	4	2	8	5	6	5	10	9	1469	7168
	P10	1	2	1	4	8	8	5	7	9	10	1469	7168

Figure A1. A population of 10 trip scenarios.

	Solution	**Customer 1**				**Customer 2**				**Customer 3**		**Objective Values**	
	#	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	*TC*	*TWT*
Assigned Trips	**C1**	3	4	1	4	5	8	8	8	9	9	1349	6623
	C2	2	2	1	1	5	8	5	8	10	10	1317	6974
	C3	4	1	1	2	6	5	6	6	9	10	1396	7287
	C4	1	2	2	1	8	7	8	8	9	9	1317	6974
	C5	4	1	1	2	6	5	6	6	9	9	1349	6623
	C6	1	2	1	1	7	8	8	8	9	10	1358	6558
	C7	2	2	1	1	7	5	6	5	10	9	1421	7287
	C8	4	3	4	2	8	8	8	8	9	9	3477	17930
	C9	2	3	2	1	8	7	8	8	9	9	1349	6623
	C10	1	2	4	3	6	7	8	7	10	10	1524	7627

Figure A2. Offspring population.

	Solution	**Customer 1**				**Customer 2**				**Customer 3**		**Objective Values**		**Front**	**Crowding**
	#	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	*TC*	*TWT*		**Distance**
	P1	2	3	4	3	6	7	8	7	10	10	1373	6522	1	19998
	P4	1	2	2	1	8	7	8	8	9	9	1317	6974	1	9999.7765
	C4	1	2	2	1	8	7	8	8	9	9	1317	6974	1	9999.5714
	C9	2	3	2	1	8	7	8	8	9	9	1349	6623	1	0.937263
	P2	3	4	1	4	5	8	5	8	10	10	1349	6623	1	0.7152339
	C6	1	2	1	1	7	8	8	8	9	10	1358	6558	1	0.6520228
	P6	2	3	4	2	5	7	7	5	10	10	1349	6623	1	0
	C1	3	4	1	4	5	8	8	8	9	9	1349	6623	1	0
	C5	4	1	1	2	6	5	6	6	9	9	1349	6623	1	0
Assigned	**P8**	2	2	1	1	7	8	8	8	9	9	1317	6974	1	0
Trips	**C2**	2	2	1	1	5	8	5	8	10	10	1317	6974	1	0
	P3	4	1	1	2	6	5	6	6	9	10	1396	7340	2	19998
	P10	1	2	1	4	8	8	5	7	9	10	1469	7168	2	9999.6919
	P7	4	2	2	1	6	5	6	5	9	10	1469	7168	2	9999.6575
	C7	2	2	1	1	7	5	6	5	10	9	1421	7287	2	1.3081395
	C3	4	1	1	2	6	5	6	6	9	10	1396	7287	2	1.0343262
	P9	4	3	4	2	8	5	6	5	10	9	1469	7168	2	0
	C10	1	2	4	3	6	7	8	7	10	10	1524	7627	3	19998
	P5	4	2	3	3	8	7	6	5	10	9	1519	7657	3	19998
	C8	4	3	4	2	8	8	8	8	9	9	3476.682	17929.79	4	19998

Figure A3. Mating pool.