This article was downloaded by: [National Chiao Tung University 國立交通大學]
On: 26 April 2014, At: 00:27
Publisher: Taylor & Francis
Informa Ltd Registered in England and Wales Registered Number: 1072954 Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK

International Journal of Production Research
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tprs20

Integrated scheduling of production and delivery with time windows

Chinyao Low [a], Rong-Kwei Li [b] & Chien-Min Chang [b]
[a] Institute of Industrial Engineering and Management, National Yunlin University of Science and Technology, Taiwan
[b] Department of Industrial Engineering and Management, National Chiao Tung University, Taiwan
Published online: 14 May 2012.

To cite this article: Chinyao Low, Rong-Kwei Li & Chien-Min Chang (2013) Integrated scheduling of production and delivery with time windows, International Journal of Production Research, 51:3, 897-909, DOI: 10.1080/00207543.2012.677071

To link to this article: http://dx.doi.org/10.1080/00207543.2012.677071

PLEASE SCROLL DOWN FOR ARTICLE

Taylor & Francis makes every effort to ensure the accuracy of all the information (the "Content") contained in the publications on our platform. However, Taylor & Francis, our agents, and our licensors make no representations or warranties whatsoever as to the accuracy, completeness, or suitability for any purpose of the Content. Any opinions and views expressed in this publication are the opinions and views of the authors, and are not the views of or endorsed by Taylor & Francis. The accuracy of the Content should not be relied upon and should be independently verified with primary sources of information. Taylor and Francis shall not be liable for any losses, actions, claims, proceedings, demands, costs, expenses, damages, and other liabilities whatsoever or howsoever caused arising directly or indirectly in connection with, in relation to or arising out of the use of the Content.

This article may be used for research, teaching, and private study purposes. Any substantial or systematic reproduction, redistribution, reselling, loan, sub-licensing, systematic supply, or distribution in any form to anyone is expressly forbidden. Terms & Conditions of access and use can be found at http://www.tandfonline.com/page/terms-and-conditions

International Journal of Production Research
Vol. 51, No. 3, 1 February 2013, 897–909

Taylor & Francis
Taylor & Francis Group

Integrated scheduling of production and delivery with time windows

Chinyao Low[a]*, Rong-Kwei Li[b] and Chien-Min Chang[b]

[a]*Institute of Industrial Engineering and Management, National Yunlin University of Science and Technology, Taiwan;*
[b]*Department of Industrial Engineering and Management, National Chiao Tung University, Taiwan*

(*Received 16 October 2011; final version received 13 March 2012*)

This paper deals with an integrated scheduling problem in which orders have been processed by a distribution centre and then delivered to retailers within time windows. We propose a nonlinear mathematical model to minimise the time required to complete producing the product, delivering it to retailers and returning to the distribution centre. The optimal schedule and vehicle routes can be determined simultaneously in the model. In addition, two kinds of genetic-algorithm-based heuristics are designed to solve the large-scale problems. The conventional genetic algorithm provides the search with a high transition probability in the beginning of the search and with a low probability toward the end of the search. The adaptive genetic algorithm provides an adaptive operation rate control scheme that changes rate based on the fitness of the parents. The experimental results have shown that the solution quality of these two algorithms is not significant but that the adaptive genetic algorithm can save more time in finding the best parameter values of the genetic algorithm.

Keywords: integrated scheduling; vehicle route; time windows; genetic algorithm; adaptive operation

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

1. Introduction

Introducing the zero-inventory concept into production and delivery problems has been widely discussed in many industries in which a product cannot be inventoried due to its short lifespan. The production factory or distribution centre has to deliver the product within a strictly limited time to fulfil their orders from customers in order to be more competitive. In such an integrated system, an important element of achieving the just-in-time concept, reducing the retailer's inventory, is incorporated between order scheduling and the delivery of finished goods. Traditional research has separately and sequentially investigated machine scheduling and order delivery without effective coordination. However, making two individual but uncoordinated decisions will not produce a global optimal solution.

This study investigates a practical scheduling problem of a distribution centre (logistics plant) addressing the needs of various demand points. In-factory, there are two types of inventory groups divided by product size, weight, demand cycle and product value. High-value commodities are produced by a predetermined conveyor belt system. The products in the system are produced by high mix low volume (HMLV) and picked into order totes in order to facilitate loading and unloading. The number of high-volume commodities are packed in a 'dozen' or 'box' as a unit stacked shelves. In the order-picking region, there are several picking personnel driving electric pallet trucks shuttling between the shelves to fill such orders. We are interested in the scheduling problem related to serving the demand from customers who are high-value commodity users. Because the retailer's warehouse capacity is quite limited, it prefers to receive merchandise on time to increase turnover of internal products. Hence, each order requires processing in the distribution centre and delivery to a predetermined location within a time window.

We assume that all requests are known in advance and each order is produced immediately in one pack containing several goods. In other words, each production order is considered as a continuous process according to each retailer's requests. Consequently, the manufacturing stage of an order can be considered as a single process without interruption. The decision maker has to decide when the production orders should start to be picked up and when they can be assigned to a vehicle so that the orders can be optimised into a delivery route. The operational decision making is to determine the minimum makespan.

In recent years, for the two-echelon supply chain problems, most articles have focused on production and distribution planning such as those by Liang (2008), Safaei *et al.* (2010), Samaranayake *et al.* (2011), and

*Corresponding author. Email: lowcy@yuntech.edu.tw

ISSN 0020–7543 print/ISSN 1366–588X online
© 2013 Taylor & Francis
http://dx.doi.org/10.1080/00207543.2012.677071
http://www.tandfonline.com

Steinrücke (2011). However, this study is related to research concerned with production scheduling and delivery (Zdrzalka 1995, Chang and Lee 2004, Zhong *et al*. 2007, Lu *et al*. 2008, Li and Yuan 2009, Wang and Cheng 2009). Chang and Lee (2004) studied machine scheduling and product delivery, in which jobs require different amounts of storage space during delivery. Their objective was to find a schedule for processing jobs such that the time required for all jobs to be processed and delivered to the corresponding customers was minimised. Zhong *et al*. (2007) dealt with two production environments, a single machine and two parallel machines, and delivery to a single customer set. Their objective was to minimise the time when the vehicle finished delivering the last batch to the customer and returns to the machine(s). Lu *et al*. (2008) developed a single-machine-scheduling problem that incorporated the routing decisions of a delivery vehicle with at most c jobs at a shipment. When preemption is not allowed, they showed that this problem is strongly NP-hard for each fixed $c \geq 1$. The objective was to minimise the maximum delivery completion time of the jobs. Although the joint production and delivery scheduling problem has received significant attention, the previous research has focused on delivering to customers who are located in a few areas ($n < 3$) or where few vehicles are available to deliver the products to specified customers. It may not occur in every industrial environment in practice. Therefore, the detailed planning between each customer's ($n \geq 3$) delivery will be proposed in this study.

Although the production scheduling with delivery problem has received significant attention, few papers apply it to a practical production environment. The literature on practical issues has been discussed by the following: Naso *et al*. (2007), Bredström and Ronnqvist (2008), Geismar *et al*. (2008), Chen *et al*. (2009), Day *et al*. (2009). In their paper, Naso *et al*. (2007) focused their work on ready-mixed concrete delivery. The authors integrated production planning and distribution routing. The strict time constraints forbid both earliness and lateness of the supply to be taken into account as well. The objective was to minimise total cost, including transportation costs, loading and unloading waiting costs and additional costs related to outsourced production, hired trucks and overtime work by some truck drivers. Chen *et al*. (2009) addressed the integrated production and distribution problem for perishable food products. The demands at retailers are assumed stochastic and they will be a random variable with a probability density function. Thus, the revenue of the supplier is uncertain and the objective of this study is to maximise the expected total profit of the supplier.

The integrated scheduling problem that we study extends from these recent papers. In the next section, we present a formulation mix with production scheduling and vehicle routing problems with a time window into an integer nonlinear programming model. We then describe two evolutionary algorithms for making improvements in Section 3, and in Section 4 numerical results are provided. This paper is concluded in Section 5.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

2. Model formulation

2.1 *Description of problem*

We consider a distribution centre with one manufacturing process but that produces several kinds of products. The products that each retailer requires are put into one pack immediately. Our objective is to minimise the time, $t_{\max}$, required to manufacture and deliver goods to satisfy all demands. We implement this expiration time by requiring that all batches must be delivered within a time window $[a_i, b_i]$. There is a set of retailers, $N = \{1, 2, \ldots, n\}$; each has a geographic location and a demand, d_i, $i \in N$, to be satisfied. Production and distribution can occur continuously throughout the period, so this problem can be envisioned as planning a day's operation for a product that has a production cycle that is less than one day.

Trucks with capacity Q, $\max_{i \in N} d_i \leq Q \leq \sum_{i=1}^{n} d_i$, are used to deliver the products to retailers. Since each truck has a limited capacity, many vehicles are needed and the number of vehicles is given, $K = \{1, 2, \ldots, k\}$. We assume that as long as the total physical space of the products loaded into the vehicle does not exceed Q, they can be arranged to fit in the physical space provided by the vehicle. Each trip starts at the distribution centre (location 0), travels to a sequence of retailer locations and returns to the distribution centre. We assume that each retailer is visited only once and that all demands must be satisfied.

Other assumptions in formulating are given as follows:

(1) The demand quantity of retailers is known and given.
(2) The vehicle fleet size is consistent.
(3) All products of the same retailer are produced continuously as the same batch.
(4) The setup time for producing different batches is negligible.

(5) The truck loading time is negligible.
(6) The travel distances of delivery networks are symmetric.

2.2 *Notation*

The notations which are used to develop a mathematical model of the problem are defined and interpreted as follows:

Decision variables:

C_{ik}	makespan of the demand of retailer i in the distribution centre for vehicle k
Z_{ij}	a binary decision variable indicating if product i is performed before product j
y_{ik}	a binary decision variable indicating if the order from retailer i is delivered by vehicle k
x_{ijk}	a binary decision variable indicating if the arc (i,j) is part of a vehicle route k
t_i	time departing from retailer i

Non-decision variables and parameters:

p_i	unit processing time of retailer i
d_i	demand of retailer i
f_{ij}	travel time between retailers i and j
s_i	service time at retailer i
r_i	flow variables of retailer i
N	retailer set
N_0	retailer set including location 0
M	very large positive number

2.3 *Mathematical models*

The addressed problem with minimisation of the total complete time objective can be formulated as follows.

$$\min \ t_{\max} \tag{1}$$

The constraints in the model are presented below in three sets, each representing one type of system constraint.
Production scheduling constraints:

$$\sum_{k\in K} C_{ik} - p_i d_i \geq 0, \quad i \in N, \tag{2}$$

$$\sum_{k\in K} C_{jk} - p_j d_j \geq \sum_{k\in K} C_{ik} + M(Z_{ij} - 1), \quad i \times j \in N^2, \tag{3}$$

$$Z_{ij} + Z_{ji} = 1, \quad i \times j \in N^2, \tag{4}$$

$$C_{ik} \leq M \times y_{ik}, \quad i \in N; k \in K. \tag{5}$$

Flow conservation constraints:

$$\sum_{k\in K} y_{ik} = \begin{cases} K, & i = 0 \\ 1, & i \in N \end{cases}, \tag{6}$$

$$\sum_{i\in N}\sum_{k\in K} x_{ijk} = \begin{cases} K, & j = 0 \\ 1, & j \in N \end{cases}, \tag{7}$$

$$\sum_{j\in N_0} x_{ijk} = \sum_{j\in N_0} x_{jik} = y_{ik}, \quad i \in N_0; k \in K, \tag{8}$$

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

$$\sum_{j\in N} x_{0jk} = 1, \quad k \in K, \tag{9}$$

$$r_0 = 0, \tag{10}$$

$$r_j - r_i \geq (d_j + Q)\sum_{k\in K} x_{ijk} - Q, \quad i \in N_0; j \in N, \tag{11}$$

$$r_j \leq \sum_{k\in K}\sum_{i\in N_0} Q x_{ijk}, \quad i \in N. \tag{12}$$

Definitional constraints:

$$t_j \geq \max\{t_i + f_{ij}, a_j\} + s_j - M(1 - x_{ijk}), \quad i \times j \in N^2; k \in K, \tag{13}$$

$$t_j \geq \max\{\max C_{ik} y_{ik} + f_{0j}, a_j\} + s_j - M(1 - x_{0jk}), \quad i \times j \in N^2; k \in K, \tag{14}$$

$$a_j \leq t_j - s_j \leq b_j, \quad j \in N, \tag{15}$$

$$t_{\max} \geq t_j + f_{j0}, \quad j \in N. \tag{16}$$

Nonnegative constraints:

$$t_j \geq 0, \quad j \in N, \tag{17}$$

$$C_{ik} \geq 0, \quad i \in N; k \in K. \tag{18}$$

Equation (1) minimises the time required to manufacture and deliver goods to satisfy all production demands by all vehicles. The constraints in Equations (2)–(5) are production scheduling constraints. The constraint in Equation (2) ensures that the batch at the manufacturing stage can be completed only after it has been on the conveyor belt for the necessary processing time. The constraint in Equation (3) imposes that the one batch at the manufacturing stage cannot start before the previous batch on the same machine has been completed for different batches. The constraint in Equation (4) defines that a batch can only be performed either before or after the other for any pair of batch i and j. The constraint in Equation (5) assures the y variables.

The constraints in Equations (6)–(12) are flow conservation constraints of vehicle route problems. The constraints in Equations (6) and (7) enforce that one vehicle arrives at retailer j exactly once and every route starts and ends at the depot. The constraint in Equation (8) allows that for each retailer i, the entering vehicle must eventually leave this node. The constraint in Equation (9) restricts each vehicle to only leave the supplier once. The constraints in Equations (10)–(12) are the subtour elimination constraints. The constraint in Equation (11) also guarantees the vehicle's capacity. The constraints in Equations (13)–(16) are definitional constraints. The constraints in Equations (13) and (14) define the arrival time at retailer j. The constraint in Equation (15) denotes that the service at retailer i begins in the intervals $[a_i, b_i]$. The constraint in Equation (16) ensures that the ending time at the delivery stage can be completed only after all vehicles return to the depot. The constraints in Equations (17) and (18) are nonnegative constraints.

The model formulated above is an integer nonlinear programming model embedding a vehicle routing problem, which is known to be NP-hard. Therefore, genetic algorithm (GA)-based heuristics are proposed for solving the addressed problem.

3. Developing evolutionary algorithms

So far, the genetic algorithm is most commonly applied to global optimisation, especially in complex nonlinear problems and in situations when an analytical solution cannot be obtained. Hence, in this article two GA based heuristics are developed to obtain a near optimal schedule in a reasonable computation time.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

In the proposed encoding scheme is integer coding combined with order (or permutation)-based encoding for the priority assignment. Although a binary coding has some advantages in terms of implicit parallelism of the search, we apply an integer coding to keep the chromosome simple and reduce the overhead of coding/decoding (see Michalewicz 1996, Ishibuchi *et al*. 2003, Iyer and Saxena 2004).

The decoding scheme is another key component for establishing an effective search subject to a given permutation by the chromosome. This algorithm is an example of route first cluster second heuristics. It uses the backward method to avoid a large number of infeasible solution spaces and to ensure the feasibility of chromosomes in decoding. After decoding the permutation, we can obtain the vehicle assignment and route planning of vehicles. The detailed process of the decoding is described in Subsection 3.3.

3.1 *Conventional genetic algorithm*

GA is inspired by the natural capability of living beings to evolve in order to adapt to their environment. The driving force in this algorithm is the selection of chromosomes based on their fitness. Chromosomes with better fitness have a higher probability of being chosen as members of the next iteration population. During each generation, we have operators recombine two or more chromosomes to produce new individuals called crossover operators. Furthermore, we have operators which cause the self-adaptation of individuals and create more offspring eugenics called mutation operators. These two operators rates are p_c and p_m respectively. The basic structure of the algorithm is illustrated as follows.

Input. Randomly produce an initial population.
Output. The best chromosome from the final population after the evolution procedure.

While the termination condition is not yet achieved:

Evaluate fitness of the population,
Select chromosomes for the candidate population,
Perform crossover and mutation,
Reproduce new population.

When the permutations produced by the GA procedure are complete, the solving process of each individual chromosome can be treated as a two-stage flowshop scheduling. A GA for flowshop scheduling was proposed by Reeves (1995), which was then tested on several categories of problems with time gradients and job correlations and some hard test problems proposed by Taillard (1993). Murata *et al*. (1996) compared various crossover and mutation operators and showed that the two-point crossover and shift mutation operators are effective for this problem. In the past, research by Cheng and Chang (2007) showed complete optimal parameter combination design in GA for general flowshop problems. The operators which are used to develop the heuristic are interpreted as follows:

(1) Select new population

Selection strategy plays an important role in GA, as a bad selection strategy will lead to premature convergence. In this article, roulette wheel selection is adopted to determine the candidates for chromosomes. The higher the adaptation function value, the greater the percentage of being the candidate. The adaptation value of each chromosome is the fitness over the whole to design the occupied percentage on the roulette wheel. The object of this article is to minimise the total completion time, and this value is used to express the fitness function.

(2) Crossover

The chromosomes in the candidate population go through crossover and mutation processes to produce offspring for the next generation. We use the longest common subsequence (Cormen *et al*. 1990) to operate in the mating system. Suppose that A and B are the individuals chosen for crossover $A=(8\,5\,6\,2\,7\,3\,9\,1\,4)$ and $B=(8\,6\,1\,2\,9\,4\,5\,3\,7)$. First, we must locate the longest common subsequence, $LCS=(8\,6\,2\,9\,4)$, of two candidate individuals. This allows us to search over a larger space, while preserving the good part of the parent chromosomes. Then, we copy common subsequence positions to each offspring and make exchanges for the remaining genes according to their original sequence. The resulting individuals are $A'=(\underline{8}1\underline{6}\underline{2}\underline{5}3\underline{9}\underline{7}\underline{4})$ and $B'=(\underline{8}\underline{6}\underline{5}\underline{2}\underline{9}\underline{4}731)$. The advantage of this mating system is that it will not produce infeasible solutions and requires no additional time to adjust the structure of individuals.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

(3) Mutation

After crossover, individuals go through a mutation operation with certain mutation rates. The displacement mutation is implemented in this search. For instance, if the segment (6 2 5 3) of individual A' were chosen to be mutated, this segment could be randomly inserted into any hierology placement. Hence, the new string could become $A'' = (819\underline{62534})$.

3.2 *Adaptive genetic algorithm*

When GA is used to solve different problems, the parameter adjustment has a great impact on effectiveness. In the standard form, the designer is often bothered by the parameter design of a specific problem that can be obtained through trial and error or some algorithmic search. Due to the aforementioned facts, adaptive genetic algorithm (AGA) forms an alternative: it will start with initial parameter values and dynamically modify its parameter set. Lee and Fan (2000) indicated that there are two kinds of processes to adjust the parameters in AGA. The first one changes these parameters according to the deterministic rules, e.g. the number of generations. The second one is adjusting the parameters according to the feedback information, e.g. the fitness value or the diversity of the population.

In AGA, we focus on balancing the exploitation and exploration on solution space in a different manner. The solution structure of this algorithm is the same as the congenital genetic algorithm referred to in Subsection 3.1. However p_c and p_m are increased when the population falls into the local optimum and decreased when the population is scattered in the solution space.

When GA is implemented, whether the solution is converged to an optimum is the concern of most researchers. One possible way of detecting convergence is to observe the average fitness value $\bar{f}$ of the population in relation to the best value at present $f_{\min}$. $\bar{f} - f_{\min}$ is likely to be less for a population that has converged to an optimum solution than for a population scattered in the solution space. However the value of p_c also depends on the fitness of both of the present solutions, that is to say p_c should be directly affected by $f' - f_{\min}$, where f' is the better fitness of the solutions to be the crossover. Similarly, the value of p_m depends on the fitness, f, of the solution. The closer f is to $f_{\min}$, the smaller p_m should be. The expressions for p_c and p_m take the forms

$$p_c = k_1(f' - f_{\min})/(\bar{f} - f_{\min}), \quad f' \leq \bar{f}, \tag{19}$$

$$p_c = k_3, \quad f' - \bar{f} \tag{20}$$

and

$$p_m = k_2(f - f_{\min})/(\bar{f} - f_{\min}), \quad f \leq \bar{f}, \tag{21}$$

$$p_m = k_4, \quad f > \bar{f} \tag{22}$$

Where $k_1, k_2, k_3, k_4 \leq 1.0$.

From the preliminary analysis, the values for the operators k_1, k_2, k_3 and k_4 are determined and described as follows. Typical values of p_c are in the range 0.5–1.0. The moderately large value of p_c promotes the extensive recombination of the schemata. We assign k_1 and k_3 a value of 1.0. This setting is to determine the implementation of the crossover when all of the parents' fitness values are inferior compared to the average fitness value. The crossover rate decreases as the fitness value of the adapter of parents tends to $f_{\min}$. Typically, p_m is chosen in the range 0.001–0.05. As p_m increases, the solutions have the opportunity to escape the local optimum. When the solution with inferior average fitness is obtained, this could be completely disruptive and we use a value of 0.5 for k_4. We assign a value of 0.5 to k_2 because a solution with a fitness value of $\bar{f}$ should also be able to escape the local region to act in a diversified manner.

3.3 *Decoding*

3.3.1 *Finding a set of feasible trips*

In this subsection, we show the decoding process in the genetic algorithm for a chromosome. For any chromosome σ of N retailers, this algorithm starts with a tour that begins at the central depot, visits each retailer exactly once and

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

then returns to the depot. Suppose the travelling salesman tour (TSP) can be written as $0-\sigma(1)-\sigma(2)-\cdots-\sigma(n)-0$. The subtour between retailer $\sigma(i)$ and $\sigma(j)$, denoted by $\pi(i,j)$, is the total distance of having a vehicle service retailers 0, $\sigma(i)$, $\sigma(i+1),\ldots,\sigma(k)$ for $i<k$, in that order. Hence, the total demand of retailers $\sigma(i)$, $\sigma(i+1),\ldots,\sigma(k)$ has to be less than the largest vehicle's capacity. We use the notation δ_j and $\delta_j(m)$ to stand for the jth feasible solution and the trip generated by the vehicle m in the jth feasible solution, respectively. If the number of arcs in the first path is sequential until $\sigma(i)$ in which i is determined by a vehicle's capacity and time window, the sequence of trips generated by $\delta(1)=\pi(0,\sigma(i))$. Similarly we can define the paths of other vehicles to be

$$\delta(2)=\pi(\sigma(i+1),\sigma(j)),$$

$$\delta(3)=\pi(\sigma(j+1),\sigma(k)),$$

and the trips of the last vehicle is

$$\delta(K)=\pi(\sigma(K))\backslash\delta(1)\cup\delta(2)\cup\cdots\cup\delta(K-1).$$

Example 1: Consider a vehicle routing problem (VRP) with a manufacturing stage. There are four retailers being considered and the capacity of a vehicle $Q=10$. The required information for each retailer is shown in Table 1.

The symmetric travel times are the following.

$$f_{ij}=\begin{bmatrix} - & 5 & 6 & 3 & 2\\ 5 & - & 4 & 3 & 7\\ 6 & 4 & - & 6 & 4\\ 3 & 3 & 6 & - & 6\\ 2 & 7 & 4 & 6 & - \end{bmatrix}$$

If a chromosome $\sigma=(3,2,4,1)$, all the feasible solutions are generated and described in Table 2. In Table 2, the trips proceed from the distribution centre and the complete time is used to verify the time window of the ith retailer. Additionally, the completion time has to consider the manufacturing time, delivery time and service time, in other words the manufacturing time sum of the processing time from the demand until the ith retailer, $\sigma(i)$; delivery time and service time are calculated by general VRP problems. Additionally, the completion time of the addressed problem is the sum of the manufacturing time, delivery time and service time. The completion time required is calculated as

$$T_{\sigma(i)}=\sum_{j=1}^{i}p_{\sigma(j)}d_{\sigma(j)}+f_{0\sigma(1)}+\sum_{j=1}^{i-1}f_{\sigma(j)\sigma(j+1)}+\sum_{j=1}^{t}s_{\sigma(j)}.$$

Table 1. Retailer information for Example 1.

i	p_i	d_i	s_i	$[a_i,b_i]$
0	0	0	0	[0, 100]
1	5	5	6	[60, 80]
2	7	4	3	[50, 60]
3	3	5	5	[40, 60]
4	2	6	4	[60, 70]

Table 2. Potential arcs for Example 1.

Trip sequence until ith retailer	Demand	Complete time on ith retailer
$0\rightarrow\sigma_1$	5	45
$0\rightarrow\sigma_1\rightarrow\sigma_2$	9	60
$0\rightarrow\sigma_1\rightarrow\sigma_2\rightarrow\sigma_3$	15*	63*
$0\rightarrow\sigma_2$	4	53
$0\rightarrow\sigma_2\rightarrow\sigma_3$	10	64
$0\rightarrow\sigma_2\rightarrow\sigma_3\rightarrow\sigma_4$	15*	74*
$0\rightarrow\sigma_3$	6	64
$0\rightarrow\sigma_3\rightarrow\sigma_4$	11*	77
$0\rightarrow\sigma_4$	5	66

Note: the asterisks indicate values that disqualify the potential arc.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

When the completion time is calculated, it is not illegal to violate the time window of each retailer until the ith retailer has been violated. According to Table 2, all feasible solutions could be solved as follows.

$$\delta_1 = \begin{cases} \delta_1(1) = \pi(0, \sigma_1) \\ \delta_1(2) = \pi(\sigma_2) \\ \delta_1(3) = \pi(\sigma_3) \\ \delta_1(4) = \pi(\sigma_4) \end{cases}, \quad \delta_2 = \begin{cases} \delta_2(1) = \pi(0, \sigma_1) \\ \delta_2(2) = \pi(\sigma_2, \sigma_3), \\ \delta_2(3) = \pi(\sigma_4) \end{cases} \quad \delta_3 = \begin{cases} \delta_3(1) = \pi(0, \sigma_2) \\ \delta_3(2) = \pi(\sigma_3) \\ \delta_3(3) = \pi(\sigma_4) \end{cases}.$$

This method expands when the number of vehicles is unknown. When the number of vehicles is determined, the suitable solutions will be found from the group of feasible solutions. For example, it will be deduced by three vehicles that δ_2 and δ_3 are the expected solutions. In the third feasible solution, the route of each vehicle is 0-3-2-0, 0-4-0, 0-1-0, respectively.

3.3.2 *Minimising the total completion time for a given set of trips*

We now discuss a situation in which the retailers should be served by the known vehicles and the corresponding trips could satisfy all time limit requirements. For specific vehicle k, the processing time can be divided into the production time in the distribution centre $M^{(1)}_{\delta(k)}$ and delivery time outside the plant $M^{(2)}_{\delta(k)}$. Thus, the problem is equivalent to a two-stage flowshop system and an optimal schedule can be found by using Johnson's algorithm. Johnson's algorithm can be implemented as follows.

Johnson's algorithm: divide K-trips into two disjoint subsets S_1 and S_2, where $S_1 = \{K_k | M^{(2)}_{\delta(k)} \geq M^{(1)}_{\delta(k)}\}$ and $S_2 = \{K_k | M^{(2)}_{\delta(k)} < M^{(1)}_{\delta(k)}\}$. Order the jobs in S_1 in non-decreasing order of $M^{(1)}_{\delta(k)}$ and those jobs in S_2 in non-increasing order of $M^{(2)}_{\delta(k)}$. Sequence jobs in S_1 first, followed by those in S_2.

Using the Johnson's algorithm framework, we have to define $M^{(1)}_{\delta(k)}$ and $M^{(2)}_{\delta(k)}$ in our situation. While retailers of one trip $\delta(k)$ are known at the distribution centre stage, the production time $M^{(1)}_{\delta(k)}$ is the sum of the processing times of the demands of each retailer on trip $\delta(k)$. However, we must take into account explicitly the connection between delivery time and the time window outside the plant. These observations lead to the following formula for $M^{(2)}_{\delta(k)}$:

$$\min M^{(2)}_{\delta(k)} \tag{23}$$

$$M^{(2)}_{\delta(k)} \leq \sum_{i \in \delta(k)} \max\{t_i + f_{ij}, a_i\} + s_i - M^{(1)}_{\delta(k)} + M(1 - y), \quad i \times j \in \delta(k)^2, \tag{24}$$

$$t_0 \leq M^{(1)}_{\delta(k)}, \tag{25}$$

$$(t_i + f_{ij} - b_i) \times y \leq 0, \quad i \times j \in \delta(k)^2, \tag{26}$$

$$y \in \{0, 1\} \tag{27}$$

Equation (23) minimises the time at the second stage for when vehicle k completes the assigned travel trip and returns to the depot. The constraint in Equation (24) ensures that the trip for the delivery stage can be completed only after all necessary retailers have been served within their time window. The constraint in Equation (25) imposes the start time on trip $\delta(k)$, while the constraint in Equation (26) defines the arrival time on retailer i to be earlier than its time limit.

3.3.3 *Enhancing the algorithm*

In the decoding process, a mechanism that converts N retailers into K trips to form a K-job two-stage flowshop scheduling problem is designed. We need to estimate the processing time within each stage. In the first stage, we have determined which retailers should be served on which trips and then intuitively measure of the pickup time by customer demands and unit processing times. Because of a lack of the optimal service sequences of retailers through $\delta_j(m)$ in the second stage, we attempted to improve each trip's delivery time by using 2-OPT on each pair of retailers within each trip $\delta_j(m)$. We now proceed to formally state the solution algorithm as follows.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

Input. A chromosome $\sigma = \{\sigma(1), \sigma(2), \ldots, \sigma(N)\}$.
Output. Feasible solution for all N retailers.

For each feasible solution set $\delta_j = \{\delta(1), \delta(2), \ldots, \delta(K)\}$
Evaluate each trip's production time $M^{(1)}_{\delta(k)}$.
Evaluate each trip's delivery time $M^{(2)}_{\delta(k)}$.
Exchange each pair of retailers on this trip using 2-OPT.

When the time window condition is valid and requires less travel time than the original trip then:
Update this trip and $M^{(2)}_{\delta(k)}$.
Use Johnson's algorithm to schedule the solution set δ_j.

Example 2: To facilitate the understanding of our decoding with an example, consider that 12 retailers and then one chromosome $\sigma = (1, 4, 7, \ldots, 10, 12, 2)$. First we find a set of feasible trips and the shortest path base for five trips is given in Table 3. Because we consider this problem as a two-stage flowshop, we use the notation $M^{(1)}$ for the plant and $M^{(2)}$ for outside the plant. Note that $M^{(2)}$ is calculated by Equation (19) and retailers of each trip are ordered by the 2-OPT, that is to say Trip B, $\delta(B) = \{(0, 3, 7, 5, 0), (0, 7, 3, 5, 0), (0, 7, 5, 3, 0)\}$, contains three kinds of travel paths because of the symmetric network. Figure 1 depicts the production schedule and the vehicle routing plan of a feasible solution.

We found the best possible trip set and the mandated delivery time window that can be met with certainty for a given permutation of retailers. Johnson's algorithm obtains the optimum total complete time for the sequence of trips generated by δ_j, so the enhanced algorithm could find an efficient schedule under this scenario. However, this framework can be implemented only after establishing one sequence involving all retailers. Hence, we will next use evolutionary algorithms to find efficient sequences.

4. Computational results

4.1 *Data set generation*

To implement a comparison of the findings from the proposed GA-based heuristics, some test problems are generated and described as follows. The retailers' information including locations, demands and service times are directly adopted from Solomon's problem set (1987). The macro structure of the data set is preserved, but the other data are randomly generated. The unit processing time of a product follows a discrete uniform distribution $U(1, 10)$, and the manufacturing time of each retailer is obtained by multiplying demands by the unit processing time. The time window $[a_i, b_i]$ for each retailer i is presented as a_i: Using the same a_i from Solomon's problem set b_i: Original b_i + group number $\times 50\% \times \sum_{j \in N} p_i d_i$. Where all retailers in the problem set are divided into four groups, each group has a group number should be 1, 2, 3, or 4. Each retailer in the problem set is randomly assigned into a group. For example, b_i of the first group is the original b_i plus 50% of the total processing time and is plus 100% of the second group. Computations were performed in Visual C++ 6.0 on a personal computer with an Intel Pentium D CPU at 3.40 GHz with 1.00 GB of memory.

4.2 *Performance of the proposed GA-based heuristics*

To assess the solution quality and efficiency of the proposed GA based heuristic, some experiments were conducted and analysed. The initial populations were randomly chosen for each heuristic, and each generation had 60 members and the process was continued through 300 generations. And set values of p_c and p_m are 0.95 and 0.05 respectively in the GA after preliminary tests. For the AGA, the values for k_1, k_2, k_3 and k_4 are set to be 1.0, 0.5, 1.0 and 0.5.

Table 3. Trip information for Example 1.

Trip ($\delta(k)$)	$M^{(1)}$	$M^{(2)}$
A (0, 1, 4, 0)	16	17
B (0, 7, 3, 5, 0)	25	Min (36, M, 48) = 36
C (0, 9, 6, 8, 0)	23	Min (51, 22, M) = 22
D (0, 11, 10, 0)	27	20
E (0, 12, 2, 0)	11	21

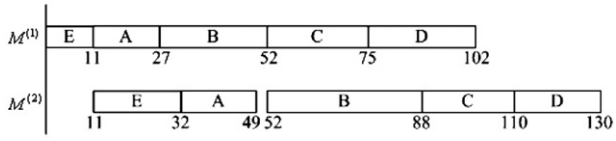

Figure 1. Schedule for Table 3.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

Table 4. Comparison of three search methods.

		Lingo		GA		AGA	
N	K	Solution	Time (s)	Solution	Time (s)	Solution	Time (s)
5	2	88	1	88	0.053	88	0.028
	2	244	12	244	0.044	244	0.025
	2	228	39	228	0.042	228	0.028
	3	228	1	228	0.040	228	0.034
	3	172	18	172	0.030	172	0.036
	3	235	5	235	0.034	235	0.038
6	2	145	27	145	0.061	145	0.058
	2	130	127	130	0.045	130	0.038
	2	171	58	171	0.044	171	0.042
	3	–	–	187	0.050	187	0.045
	3	–	–	174	0.042	174	0.041
	3	–	–	240	0.045	240	0.038

Table 5. CPU time (in seconds) comparison between the GA and the AGA.

Number of retailers		10	30	50	80	100
GA	Average	0.193	1.806	3.372	7.354	12.532
	Standard deviation	0.013	0.091	0.128	0.142	0.486
AGA	Average	0.312	2.325	5.150	11.398	19.019
	Standard deviation	0.099	0.077	0.089	0.313	0.603

These experiments are divided into two parts. The first part provides the optimal solutions for small-sized test problems as the basis for providing the feasibility and the adaptability of the proposed GA-based heuristic. It is noted that the optimal solutions are obtained by running the commercial mathematical programming software Lingo 8.0 (the mathematical model is provided in Subsection 2.3). Twelve small-sized test problems are evaluated with 10 runs for each test problem. The results are summarised and shown in Table 4. The results shown in Table 4 reveal that, in each small-sized case, GA and AGA have the same ability of search for the solution space and efficiency disparities in finding a solution are not discernable. For the case of six retailers with three vehicles, the optimum solution cannot be obtained in 24 running hours. It is noted that, for all the small-sized test problems evaluated, the solutions are the same in 10 runs of each test problem. This signifies the robustness of the designed GA-based heuristics.

To demonstrate the efficiency of the proposed-GA based heuristics, we randomly generated 10 samples for each problem size. The results for each problem size are shown in Table 5. The average time for 100 retailers of the GA was 12.532 seconds. The average time of the AGA was 19.019 seconds under the same data sets. The results shown in Table 5 reveal that the difference in efficiency between the two heuristics become increasingly obvious.

From the above test results, we found that the proposed algorithms can solve the production scheduling problem with delivery efficiently and return a reliable solution. The efficiency of the algorithm makes it suitable for solving real cases, which are normally large in scale.

4.3 *Comparison with other heuristics*

In this section, the performance of the proposed heuristics is compared with the H3 heuristic reported by Chang and Lee (2004). The scenario of this problem is similar as the jobs are processed on a single machine and then delivered to two areas. Additionally, they defined one customer area as a location where a group of customers are located in

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

Table 6. Comparison with other heuristics.

N	Q	H3	GA	AGA	GA/ H3	AGA/H3	GA < H3	AGA < H3
30	20	640.00	547.00	547.00	0.85	0.85	0.028	0.028
50	30	957.17	819.50	818.17	0.86	0.85	0.013	0.013
50	60	991.00	848.33	848.33	0.86	0.86	0.009	0.009
80	60	1592.33	1356.17	1354.83	0.85	0.85	0.013	0.013
80	100	1673.83	1433.50	1433.50	0.86	0.86	<0.001	<0.001
100	40	1922.67	1629.50	1628.83	0.85	0.85	<0.001	<0.001
100	60	1898.67	1614.00	1610.00	0.85	0.85	0.005	0.005

Note: Columns GA < H3 and AGA < H3 lists *p*-values for the comparison.

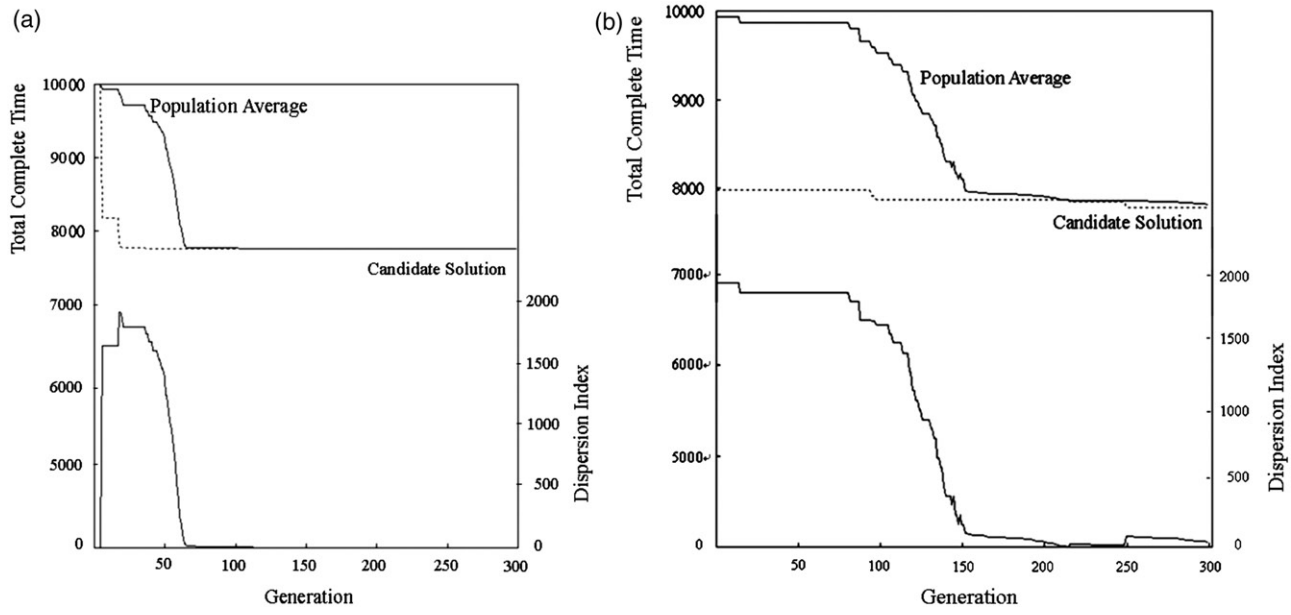

Figure 2. Statistics of GA and AGA convergence over 100 retailers.

close proximity to each other. When we relaxed the time windows, seven different data sets of customer numbers and vehicle capacity were randomly generated.

We compared the results from using the three different algorithms. Table 6 presents all the completion times for GA, AGA and the corresponding heuristic H3; these values are averages taken over six examples. GA and AGA have a lower average completion time than H3 for each parameter combination. On average, GA and AGA are more efficient than H3 by 14.79% and 14.86% respectively. Paired *t*-tests revealed that our heuristics are statistically better than H3 according to the *p*-value in the last two columns at the 95% significance level. With these problem sets, we also found that the effectiveness between GA and AGA not discernible.

4.4 *Convergence*

To illustrate how GA works differently than AGA, we provide variations of the population averages, the best solution and dispersion index values for GA and AGA as shown in Figure 2. This index measures how chromosomes spread around the global best of the population, and the formula for the dispersion index is evaluated as

$$\bar{\delta} = \frac{\sum_{i \in G} \sum_{j \in C} |f_{ij} - f_i^*|}{G \times C} \tag{28}$$

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

where

G number of generations,
C number of chromosomes,
f_{ij} fitness of the jth chromosome for the ith generation,
f_i^* global best of the ith generation.

This index explains the coverage searching area of the population. A population with a high dispersion index has relatively wider coverage of the searching area than one with a lower dispersion index.

When comparing the two plots in Figure 2, we observe that the average completion time of the population decreases gradually for AGA while it decreases rapidly for GA. For the GA, as shown in Figure 2, within 50 generations, the candidate solution decreased rapidly and the last improvement was seen at 37 generations while the AGA continued improving for 254 generations. The best candidate solution in the GA was reported to be 7756 and the optimal total complete time for AGA was seen at 7784.

Another feature that stands out was related to the population search ability for GA. The candidate solution stayed at the 7772 level until it found a better solution at the 7756 level and stayed there for the rest of the search. In the AGA, however, the population solutions moved continuously down and therefore searched for a variety of solutions. This was mainly due to the difference in the rates of the operators' control mechanism between the two methods. In the class GA, the rates of operators were constant from the beginning of the search. Therefore, by the time the search hit a new solution, it had lost most of its energy and was hardly able to avoid being trapped in the local optimal solution. The AGA provides a mechanism to disturb the rates of operations. Therefore, the latter one was a high probability to escape the local optimum.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014

5. Conclusions

This paper considers production scheduling with delivery in order to minimise the total completion time while satisfying the request time windows of each retailer. The integer nonlinear programming model has been presented first. Although the mathematical model provides the benchmark solution, the variables and constraints increase drastically, even for a small-sized problem. Therefore, two heuristics based on genetic algorithms have also been proposed for solving medium/large-sized problems. We have developed the encoding/decoding method of characteristics for this problem. The quality and efficiency of these heuristics have been evaluated through randomly generated test problems in various environments. The results indicate that two heuristics can obtain the benchmark solution for small-sized test problem. As the problem size increases, a genetic algorithm has better efficiency.

In quasi-experimental evaluations which Chang and Lee (2004) have proposed to ensure that our heuristics have sufficient solving ability, the results indicate that the genetic algorithm and adaptive genetic algorithm are both better than the existing H3 method for solving problems with multiple retailers when not considering time windows and when concentrating on two regions. With these problem sets, there is no discernable difference between the effectiveness of the genetic algorithm and the adaptive genetic algorithm.

In this paper, the comparisons of the proposed GAs clearly demonstrate that the genetic algorithm is more efficient in solving production and delivery problems but the differences in quality solution are not discernable. In our experiment, we have demonstrated that during the search, when a local minimum is visited, the genetic algorithm might not be able to improve the solution quality due to low population dispersion, whereas, there is an adaptive operator rate avoiding being trapped in the local minimum. In the genetic algorithm, such a shortcoming could be alleviated if proper search parameters are set, such as selection, crossover and mutation, and if the rates of the aforementioned operators are selected through a pre-search analysis. While, for the adaptive genetic algorithm, the advantage is that less pre-search analysis may be required.

References

Bredström, D. and Ronnqvist, M., 2008. Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. *European Journal of Operational Research*, 191 (1), 19–31.

Chang, Y.C. and Lee, C.Y., 2004. Machine scheduling with job delivery coordination. *European Journal of Operational Research*, 158 (2), 470–487.

Chen, H.K., Hsueh, C.F., and Chang, M.S., 2009. Production scheduling and vehicle routing with time windows for perishable food products. *Computers and Operations Research*, 36 (7), 2311–2319.

Cheng, B.W. and Chang, C.L., 2007. A study on flowshop scheduling problem combining Taguchi experimental design and genetic algorithm. *Expert Systems with Applications*, 32 (2), 415–421.

Cormen, T.H., Leiserson, C.E., and Rivest, R.L., 1990. *Introduction to algorithms*. New York: MIT Press/McGraw-Hill.

Day, J.M., Wright, P.D., and Schoenherr, T., 2009. Improving routing and scheduling decisions at a distributor of industrial gasses. *Omega*, 37 (1), 227–237.

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 (1), 21–33.

Ishibuchi, H., Yoshida, T., and Murata, T., 2003. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7 (2), 204–223.

Iyer, S.K. and Saxena, B., 2004. Improved genetic algorithm for the permutation flowshop scheduling problem. *Computers and Operations Research*, 31 (2), 593–607.

Lee, L.H. and Fan, Y., 2000. Developing a self-learning adaptive genetic algorithm. *Intelligent Control and Automation*, 1 (1), 619–624.

Li, S. and Yuan, J., 2009. Scheduling with families of jobs and delivery coordination under job availability. *Theoretical Computer Science*, 410 (47–49), 4856–4863.

Liang, T.F., 2008. Integrating production–transportation planning decision with fuzzy multiple goals in supply chains. *International Journal of Production Research*, 46 (6), 1477–1494.

Lu, L., Yuan, J., and Zhang, L., 2008. Single machine scheduling with release dates and job delivery to minimize the makespan. *Theoretical Computer Science*, 393 (1–3), 102–108.

Michalewicz, Z., 1996. *Genetic algorithms + data structures = evolution programs*. New York: Springer-Verlag.

Murata, T., Ishibuchi, H., and Tanaka, H., 1996. Genetic algorithms for flowshop scheduling problems. *Computers and Industrial Engineering*, 30 (4), 1061–1071.

Naso, D., *et al*., 2007. Genetic algorithms for supply-chain scheduling: A case study in the distribution of ready-mixed concrete. *European Journal of Operational Research*, 177 (3), 2069–2099.

Reeves, C.R., 1995. A genetic algorithm for flowshop sequencing. *Computers and Operations Research*, 22 (1), 5–13.

Safaei, A.S., *et al*., 2010. Integrated multi-site production-distribution planning in supply chain by hybrid modeling. *International Journal of Production Research*, 48 (14), 4043–4069.

Samaranayake, P., Laosirihongthong, T., and Chan, T.S., 2011. Integration of manufacturing and distribution networks in a global car company – Network models and numerical simulation. *International Journal of Production Research*, 49 (11), 3127–3149.

Solomon, M.M., 1987. Algorithms for the vehicle routing and scheduling problems with time windows constraints. *Operations Research*, 35 (2), 254–265.

Steinrücke, M., 2011. An approach to integrate production–transportation planning and scheduling in an aluminum supply chain network. *International Journal of Production Research*, 49 (21), 6559–6583.

Taillard, E., 1993. Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64 (2), 278–285.

Wang, X. and Cheng, T.C.E., 2009. Production scheduling with supply and delivery considerations to minimize the makespan. *European Journal of Operational Research*, 194 (3), 743–752.

Zdrzalka, S., 1995. Analysis of approximation algorithms for single-machine scheduling with delivery times and sequence independent batch setup times. *European Journal of Operational Research*, 80 (2), 371–380.

Zhong, W., Dosa Gyorgy, and Tan, Z., 2007. On the machine scheduling problem with job delivery coordination. *European Journal of Operational Research*, 182 (3), 1057–1072.

Downloaded by [National Chiao Tung University] at 00:27 26 April 2014