Int. J. Production Economics 153 (2014) 139–148

Contents lists available at ScienceDirect

Int. J. Production Economics

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

Coordination of production scheduling and delivery problems with heterogeneous fleet

Chinyao Low [a,*], Chien-Min Chang [b], Rong-Kwei Li [b], Chia-Ling Huang [c]

[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
[c] Department of Logistics and Shipping Management, Kainan University, Taiwan

ARTICLE INFO

Article history:
Received 10 October 2012
Accepted 18 February 2014
Available online 28 February 2014

Keywords:
Production scheduling with delivery
Heterogeneous fleet
Time windows
Adaptive genetic algorithm

ABSTRACT

In a production scheduling with delivery problem, there are different types of products processed by a distribution center and then delivered to retailers. Each retailer order might be consisted of different products. The resolution of this problem is to determine the production sequence, retailers' needs to heterogeneous fleet of vehicles and the visiting sequence of each vehicle for delivery goods within time windows. In this article, a nonlinear mathematical model is proposed with minimizing the total cost which includes transportation cost, vehicle arrangement cost and penalty costs, subjected to satisfy all demands of each retailer. Following, two adaptive genetic algorithms (AGAs) are designed and tested in variety of production and delivery scenarios. The computational experiments show that the total cost gradually decreases as the vehicle type employed in the delivery stage increasing. In addition, more kinds of vehicle types provided in the delivery stage could reduce fixed vehicle cost and variable routing cost.

1. Introduction

Under the current competitive environment, coordination production and delivery problems have been widely discussed in many industries. Consider the distributor stage and the retail stage of two echelon supply chain; it is an important element of reducing the retailer's inventory to achieve the goal of just-in-time. The distribution center has to deliver the products within a limited time to fulfill their order from retailers in order to be more competitive. This paper investigates a practical scheduling and delivery problem of a distribution center (logistics plant). In the distribution center, handling of a retailer's order is looked as an operation such as picking, packing, processing, and should be completed on a single workstation. A variety of products which a retailer required in an order is processed as a batch and put it into a container for delivery convenience. A completed batch might be delivered immediately or group delivered with other batches (retailers' order) to the corresponding retailers depending on the required performance measures such as minimization of total cost, minimization of total delivery distance or minimization of total delivery time deviation. This kind of production scheduling and delivery scenario between distribution center and retailers are popularly seen in the real world such as two echelon supply chain of 7–11 convenience stores, and McDonald's corporation. Traditional research has separately and sequentially investigated scheduling and order delivery without effective coordination. However, making two individual but uncoordinated decisions will not produce a global optimal solution. Furthermore, a good scheduling plan or delivery plan cannot guarantee a good performance of an integrated plan.

Besides, because the retailer's warehouse capacity is quite limited, they prefer to receipt merchandise on time to increase turnover of internal products. Hence, each order requires processing in the distribution center and delivery it to a predetermined location within a time window. The distribution center is required to pay the penalty cost for retailer if the vehicle arrival time later or earlier than retailer's time window. The decision maker has to decide how many vehicles of each type to use given a mix of vehicle types differing in capacity and costs, when the producing orders should start to be picked up and when they can be assigned to a vehicle so that the orders can be optimized into a delivery route. The operational decision-making is to determine the minimum total cost including fixed vehicles cost, variable routing cost and penalty cost of violation time window.

In the recent years, for the two echelon supply chain problems, most of articles focused on production and distribution planning such as Liang (2008) and Safaei et al. (2010), Samaranayake et al. (2011) and Steinrücke (2011). However, this study is related to research concerned with production scheduling and delivery (Chang and Lee, 2004; Zhong et al., 2007; Li and Yuan, 2009; Zdrzalka, 1995; Lu et al., 2008; Wang and Cheng, 2009; Adulyasak et al., 2014a, 2014b). Chang and Lee

* Corresponding author. Tel.: +886 5 5342601x5131; fax: +886 5 5312073.
E-mail address: lowcy@yuntech.edu.tw (C. Low).

http://dx.doi.org/10.1016/j.ijpe.2014.02.014
0925-5273

(2004) studied machine scheduling problems with explicit transportation considerations. Each finished job has a different size during delivery. Three scenarios of the problem are discussed. In addition, the authors provide a proof of NP-hardness and a heuristic with worst-case analysis. Zhong et al. (2007) dealt with two production environments, single machine and parallel machines, and delivery to single customer set. For the first problem, in which jobs are processed on a single machine and a best possible approximation algorithm with a worst-case ratio arbitrarily close to 3/2 is proposed. An improved algorithm with a worst-case ratio 5/3 is proposed where jobs are processed in two parallel machines case. Lu et al. (2008) developed a single machine scheduling problem incorporated the routing decisions of a delivery vehicle which at most c jobs at a shipment. When the preemption is not allowed, they showed that this problem is strongly NP-hard for each fixed $c \geq 1$. 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. In addition, a 5/3-approximation algorithm was proposed for solving this problem. For the related research in production routing problem, Adulyasak et al. (2014a) introduced an optimization-based adaptive large neighborhood search heuristic for this category. In the heuristic, binary variables representing setup and routing decisions are handled by an enumeration scheme and upper-level search operators, respectively, and continuous variables associated with production, inventory, and shipment quantities are set by solving a network flow subproblem. Adulyasak et al. (2014b) extended the topic to inventory routing problem and branch-and-cut algorithms were proposed to solve the different formulations.

Although the production scheduling with delivery problem has received significant attention, few papers applied to practical production environment. The literature on practical issues is discussed as follows: Naso et al. (2007), Chen et al. (2009), Geismar et al. (2008), Day et al. (2009), Bredström and Ronnqvist (2008), and Low et al. (2013). In those papers, Naso et al. (2007) focused their work on ready-mixed concrete delivery. In their work, a genetic algorithm is applied to solve the integrated production planning and distribution routing problem. The strict time-constraints forbid both earliness and lateness of the supply to be taken into account as well. 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. The authors elaborate a solution algorithm composed of the constrained Nelder–Mead method and a heuristic is then proposed for solving the vehicle routing with time windows problem. Low et al. (2013) developed an adaptive genetic algorithm for solving integrated scheduling and delivery problem.

We first present an integer nonlinear programming model for the addressed problem which is the integration of a production scheduling and vehicle routing problem with a time window constraint. Following, two evolutionary algorithms are developed for obtaining better solution for the medium to large scale problems, and numerical results are provided as well.

2. Model formulation

2.1. Description of problem

Considering a two echelon supply chain with one distribution center and N retailers production scheduling – delivery problem. In the distribution center, handling of a retailer's order is looked as an operation such as picking, packing, processing, and should be completed on a single workstation. A variety of products which a retailer required in an order is processed as a batch and put it into a container for delivery convenience. A completed batch might be delivered immediately or group delivered with other batches (retailers' order) to the corresponding retailers depending on the required performance measure, minimization of total cost. The total cost includes transportation cost, vehicle arrangement cost and penalty costs. For a set of retailers, $N = \{1, 2, 3, \ldots, n\}$; each has a geographic location and a demand, d_i, $i \in N$, to be satisfied within a time window $[a_i, b_i]$. The heterogeneous fleet is taken into account in the delivery stage as well.

The vehicle capacity Q_k determines the size of different type of vehicles T, $T = \{1, 2, 3, \ldots, t\}$, and corresponds to the fixed acquisition cost of w_k, $k \in T$. We assume that as long as the total physical space of the products loaded into the vehicle does not exceed Q_k, they can be arranged to fit in the physical space provided by the vehicle. Each trip starts at the distribution center (location 0), travels to a sequence of retailer locations, and returns to the distribution center. We assume that each retailer is visited only once and that all demands must be satisfied. Other assumptions in formulation are given as follows:

(1) The setup time for producing different batches is negligible.
(2) The vehicle loading time is negligible.
(3) The travel distances of delivery network are symmetric and satisfy the triangle inequality.
(4) An infinite supply of each vehicle type is available.

2.2. Notations

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 retailer i in the distribution center for vehicle type k,
Z_{ij}	a binary decision variable indicating if retailer i is performed before retailer j in the distribution center,
y_{0i}	a binary decision variable indicating if retailer i is delivered after depot,
x_{ijk}	a binary decision variable indicating if vehicle type k travels the arc (i, j),
t_i	arrival time at retailer i,
u_{ij}	the penalty due to the violation the time windows at the end of each arc (i, j).

Non-decision variables and parameters:

p_i	unit processing time of retailer i,
d_i	demand of retailer i,
τ_{ij}	cost of travel from retailer i to j,
s_i	service time at retailer i,
r_i	flow variables associated with retailer i,
w_k	fixed acquisition cost of vehicle k,
P_e	early delivery penalty per order,
P_l	late delivery penalty per order,
N	retailer set,
N_0	retailer set including depot,
M	A very large positive number.

2.3. Mathematical models

The addressed problem with minimization of the total cost objective can be formulated as follows:

$$\min \sum_{k \in T} w_k \sum_{j \in N} x_{0jk} + \sum_{k \in T} \sum_{i \in N_0} \sum_{j \in N_0} \tau_{ij} x_{ijk} + \sum_{k \in T} \sum_{i \in N_0} \sum_{j \in N_0} u_{ij} x_{ijk} \quad (1)$$

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

Production scheduling constraints:

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

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

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

$$C_{ik} \leq M \sum_{j\in N_0} x_{ijk}, \quad i \in N; \quad k \in T \tag{5}$$

Flow conservation constraints:

$$\sum_{i\in N}\sum_{k\in T} x_{ijk} = 1, \quad j \in N, \tag{6}$$

$$\sum_{j\in N_0} x_{ijk} = \sum_{j\in N_0} x_{jik}, \quad i \in N_0; \quad k \in T, \tag{7}$$

$$r_0 = 0, \tag{8}$$

$$r_j - r_i \geq (d_j + Q_t)\sum_{k\in T} x_{ijk} - Q_t, \quad i \in N_0; \quad j \in N, \tag{9}$$

$$r_j \leq \sum_{k\in T}\sum_{i\in N_0} Q_k x_{ijk}, \quad j \in N. \tag{10}$$

Definitional constraints:

$$t_j \geq t_i + s_i + \tau_{ij} - M(1 - x_{ijk}), \quad i \in N; \quad j \in N; \quad k \in T, \tag{11}$$

$$t_i \geq y_{0i} + \tau_{0i} - M(1 - \sum_{k\in T} x_{0ik}), \quad i \in N, \tag{12}$$

$$y_{0i} \geq \max(\sum_{k\in T} C_{ik}, \quad \sum_{k\in T} C_{jk}) - M(2 - \sum_{k\in T} x_{0ik} - \sum_{k\in T} x_{ijk}), \quad i \in N; \quad j \in N, \tag{13}$$

$$u_{ij} = \frac{1}{2}[(a_i - t_i)^+ P_e + (t_i - b_i)^+ P_l] + \frac{1}{2}[(a_j - t_j)^+ P_e + (t_j - b_j)^+ P_l], \quad i \in N; \quad j \in N,$$
$$(a_i - t_i)^+ = \max\{0, \ a_i - t_i\}$$
$$(t_i - b_i)^+ = \max\{0, \ t_i - b_i\} \tag{14}$$

Nonnegative constraints:

$$C_{ik} \geq 0, \quad i \in N; \quad k \in T. \tag{15}$$

$$t_i \geq 0, \quad i \in N, \tag{16}$$

$$y_{0i} \geq 0, \quad i \in N, \tag{17}$$

Eq. (1) minimizes the total cost, the first term gives the total fixed cost; the next gives the total routing cost; the last gives the penalty cost. Constraints (2)–(5) are production scheduling constraints. Constraint (2) ensures that the batch at the distributor stage can be completed only after it has been processed on the workstation for the necessary processing time. Constraint (3) imposes that the one batch (retailer's order) at the distribution stage cannot start before the previous batch has been picked for different batches. Constraint (4) defines that a batch can only be performed either before or after the other for any pair of batch i and j. Constraint (5) guarantees that each retailer's order has to be processed in distribution center before delivery.

Constraints (6)–(10) are flow conservation constraints of vehicle route problems. Constraints (6) enforce that one vehicle arrives at retailer j exactly once and every route starts and ends at the depot. Constraint (7) allows that for each retailer i, the entering vehicle must eventually leave this retailer. Constraints (8)–(10) guarantee that vehicle capacities are not exceeded. The variable r_i gives the total demand that a vehicle has serviced on its route after it reaches retailer i. Constraints (13)–(15) are definitional constraints. Constraints (11) and (12) define the arrival time at retailer j. Constraints (13) describes the relationship between two variables y_{0i} and x_{ijk}. Constraints (14) measures the penalty due to the violation of the time window for each trip (i, j). Constraints (15)–(17) are nonnegative constraints.

3. Developing evolutionary algorithms

Introduced by Holland (1992), genetic algorithms (GAs) apply some techniques and procedures inspired by evolutionary biology to solve complex optimization problems. In this article, each chromosome in GAs is encoded by integer with order (or permutation) based on the priority assignment. An integer coding could keep the chromosome simple and reduce the overhead of coding/decoding (see e.g. Michalewicz, 1996; Ishibuchi et al., 2003; or Iyer and Saxena, 2004). The retailers are specified by a whole series of genes that have to be presented. The decoding process is based on the characteristics of the production scheduling problem with delivery. This article provides a methodology for route first-cluster second heuristics to ensure the feasibility of chromosomes and the efficiency of decoding. The detailed process of the decoding is described in Section 3.1.

The basic structure of GAs is including selection, crossover operators, mutation operators and its detail process is described in Section 3.2. Crossover occurs only with some probability p_c (the crossover rate). When the solutions are not subjected to crossover, they remain unmodified. Mutation involves the modification of the value of each gene of a solution with some probability p_m (the mutation rate). However, from the experience of using GAs, it is not hard to discover that usually a lot of trial-and-errors need to be done to achieve good results for a particular application. This article provides an efficient scheme to avoid the computationally expensive from typical GAs. The adaptive rates of crossover and mutation are adopted in the GAs to realize the twin goals of maintaining diversity in the population and sustaining the convergence capacity of the GAs. The proposed adaptive genetic algorithm (AGA) starts with initial parameter values and dynamically modifies its parameters set to eliminate time-consuming of the experiment of setting parameters, is described in Section 3.3.

3.1. Decoding

In this section, we show the decoding process in the genetic algorithm for a chromosome. For any chromosome σ of N retailers, this algorithm starts with a trip that begins from the central depot, visits each retailer exactly once, and then returns to the depot. Assuming a chromosome can be written as $\sigma(1)$–$\sigma(2)$–$\sigma(3)$–…–$\sigma(N)$ where $\sigma(i)$ is i_{th} position in the chromosome sorting. The string 0–$\sigma(1)$–$\sigma(2)$–…–$\sigma(i)$–0–$\sigma(i+1)$–$\sigma(i+2)$–…–$\sigma(j)$–0–$\sigma(j+1)$–…–$\sigma(N)$–0 presents a set of feasible sub-trips. The sum of fixed vehicle costs and variable routing costs between retailer $\sigma(i+1)$ and $\sigma(j)$ is denoted by $\pi(i, \ j)$ for $i < j$. When heterogeneous fleet of vehicles are taken into account, the vehicle type k is chosen as the cheapest vehicle type with a capacity Q_k not smaller than the load between retailer $\sigma(i+1)$ and $\sigma(j)$.

In this article, the addressed production scheduling and delivery problem is looked as a two stage flowshop scheduling problem, and then solve it by the famous Johnson's algorithm. First, without the time window restriction consideration, the vehicle routing plan for the retailers are determined (Section 3.1.1) based on the retailers' and fleet information. Following, sum the required processing time for each retailer on the same route as the dummy processing time at distribution center (stage 1) to form a two stage flowshop scheduling problem, and then solved by the Johnson's algorithm (Section 3.1.2). Finally, adjust the obtained solution with the time window restriction consideration (Section 3.1.3).

3.1.1. Finding a set of feasible trips from a chromosome

To decode a sequence of a chromosome, we need to decide number of retailers visited in a vehicle trip and vehicle type used for the trip. Like in most GAs for vehicle routing problem (VRP), a chromosome simply presents a sequence of N retailers, without trip delimiters. This kind of method was put forward by Prins (2004) and Liu et al. (2009) as the second phase heuristic for solving a VRP in this article. In normative scenario studies, we have to decode a chromosome into solution pool obtained many feasible solution sets δ_j. For each set δ_j, many feasible trips $\delta_j(k)$ are generated according to capacity limit of vehicle k. A procedure for finding a set of feasible trips is described as follows.

A network for the a chromosome σ can be presented by an auxiliary graph $H=(X, A, Z)$. X contains $N+1$ nodes indexed from 0 to N. A contains $\pi(i, j)$ if a trip visiting retailers $\sigma(i+1)$ to $\sigma(j)$ is feasible in terms of the capacity limitation of a vehicle. For each status $\pi(i, j)$, it computes two labels for retailer $j=1, 2, \ldots, N$ of X:V_j, the cost of the feasible path from $\sigma(i+1)$ to $\sigma(j)$ in H excluding penalty cost, and P_j, the predecessor of $\sigma(j)$ on this trip. In other words, each retailer j may be marked multiple group labels, depending on whether a feasible trip contain this node, i.e. if $\pi(1, 3)$ and $\pi(2, 3)$ are feasible route, then $\sigma(3)$ would be marked as ($V'_3=\pi(1, 3)$, $P'_3=1$) and ($V''_3=\pi(2, 3)$, $P''_3=2$). Through the enumeration process, all feasible sub-sequences $\delta_j(k)$ are evaluated and the corresponding cost V_j and path P_j are updated as well. The pseudo-code as this procedure as follows.

procedure: Split algorithm
input: $H=(X, A, Z)$
output: V_N, P_N
begin
  $V_0 \leftarrow 0$
  for $i := 0$ to N **do** $V_{i+1} \leftarrow +\infty$ **endfor**
  for $i := 0$ to N **do**
    $load \leftarrow 0$; $j \leftarrow i+1$; $\pi(i, j) \leftarrow 0$
    while $j \le N$ and $load \le Q_t$ **do**
      $load \leftarrow load + d_{\sigma(j)}$
      if $i = j-1$ **then**
        $\pi(i, j) \leftarrow \tau_{0,\sigma(j)} + \tau_{\sigma(j),0} + w_{0,\sigma(j)}$
      else $\pi(i, j) \leftarrow \pi(i, j) - \tau_{\sigma(j-1),0} + \tau_{\sigma(j-1),\sigma(j)} + \tau_{\sigma(j),0} + w_{0,\sigma(j)}$
      endif
      if $load \le Q_t$ **then**
        $V_j \leftarrow V_i + \pi(i, j)$; $P_j \leftarrow i$
        $j \leftarrow j+1$
      endif
    endwhile
  endfor
  return all feasible set of feasible trips of H
end

Example 1. Considered an integrated scheduling problem with a distribution center and 5 retailers. There is one operation process within distribution center and two types of fleet at depot. The required information for each vehicle is shown in Table 1. The objective is to minimize the sum of fixed vehicle costs and variable routing costs. The required information for each retailer is shown in Table 2.

Table 1
Fleet information for Example 1.

Vehicle	Capacity	Cost
1	120	300
2	180	500

Table 2
Retailer information for Example 1.

i	p_i	d_i	s_i	$[a_i, b_i]$
1	3	50	10	[300, 400]
2	2	60	10	[300, 400]
3	1	40	10	[400, 500]
4	4	80	10	[700, 800]
5	2	70	10	[700, 800]

For a chromosome $\sigma=(1, 2, 3, 4, 5)$, an auxiliary graph is presented as Fig. 1(a). There are 5 retailers and two vehicle types. The corresponding acyclic graph is constructed in Fig. 1(b). $\pi(2, 4)$ represents trip (0, 3, 4, 0) with a total cost, i.e. fixed vehicle costs and variable routing cost, of $80+40+60+300=480$ using vehicle type 1. Fig. 1(c) and (d) gives two sets of feasible trips with its corresponding information and the vehicle used for each trip. The first set has a better performance 1270 to service requirements of each retailer. The mix of vehicles in the fleet of second set use two larger vehicles for its feasible set and the performance is 1330.

3.1.2. Minimizing the total cost for a given set of trips

According to the auxiliary graph $H=(X, A, Z)$ for a chromosome σ, we could get all feasible solution set. Applying splitting procedure (in Section 3.1.1), each trip not only would be calculated its routing costs, but the lead time, operation time, in the distribution center could be backward integrated by its corresponding retailers. Additionally, we use the existing trip as production unit to obtain the optimization processing sequence in integrated scheduling stage. In order to minimize total cost, the proposed integrated production scheduling and delivery problem is transformed as a two-stage flowshop system. Thus, the Johnson's algorithm (Johnson, 1954) is introduced to compress the penalty cost. The processing time of $\delta(k)$ at each stage can be expressed to the operate time $M^{(1)}_{\delta(k)}$ in the distribution center and delivery time $M^{(2)}_{\delta(k)}$ outside the plant. Johnson's algorithm can be implemented as follows.

Johnson's algorithm. Divide K-trips into two disjoint subsets S_1 and S_2, where $S_1=\{K_k | M^{(2)}_{\delta(k)} \ge M^{(1)}_{\delta(k)}\}$ and $S_2=\{K_k | M^{(2)}_{\delta(k)} < M^{(1)}_{\delta(k)}\}$. Order the jobs in S_1 in nondecreasing order of $M^{(1)}_{\delta(k)}$ and those jobs in S_2 in nonincreasing 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 a trip $\delta(k)$ are known at the distribution center stage, the operate time $M^{(1)}_{\delta(k)}$ is the sum of the processing times of the demands of each retailer on trip $\delta(k)$. And $M^{(2)}_{\delta(k)}$ is the sum of the variable routing cost and service time of each retailer on trip $\delta(k)$.

Example 2. To facilitate the understanding of our decoding procedure, the same data of Example 1 and two feasible sets by its individual trips, Fig. 1(c) and (d), is considered. The necessary information is collated in Table 3. The notations $M^{(1)}$ within the plant (distribution center) and $M^{(2)}$ for outside the plant are used throughout the example. After implementation of Johnson's algorithm, the optimal production sequence in the distribution center B-C-A(2-3-4-5-1) for Fig. 1(c) and A-B(1-2-3-4-5) for Fig. 1(d) are obtained. The scheduling result (production schedule and delivery schedule) is summarized and demonstrated in Fig. 2. It is noted that the number in the parentheses denotes the arrival time, t_i, at retailer i; e.g. the partial schedule in Fig. 2(a) can be interpreted as the demand orders for retailers 2, 3 and 4 should be processed first, and delivered to retailer 2, 3, 4 sequentially after completed the required processing at 480 in distribution center.

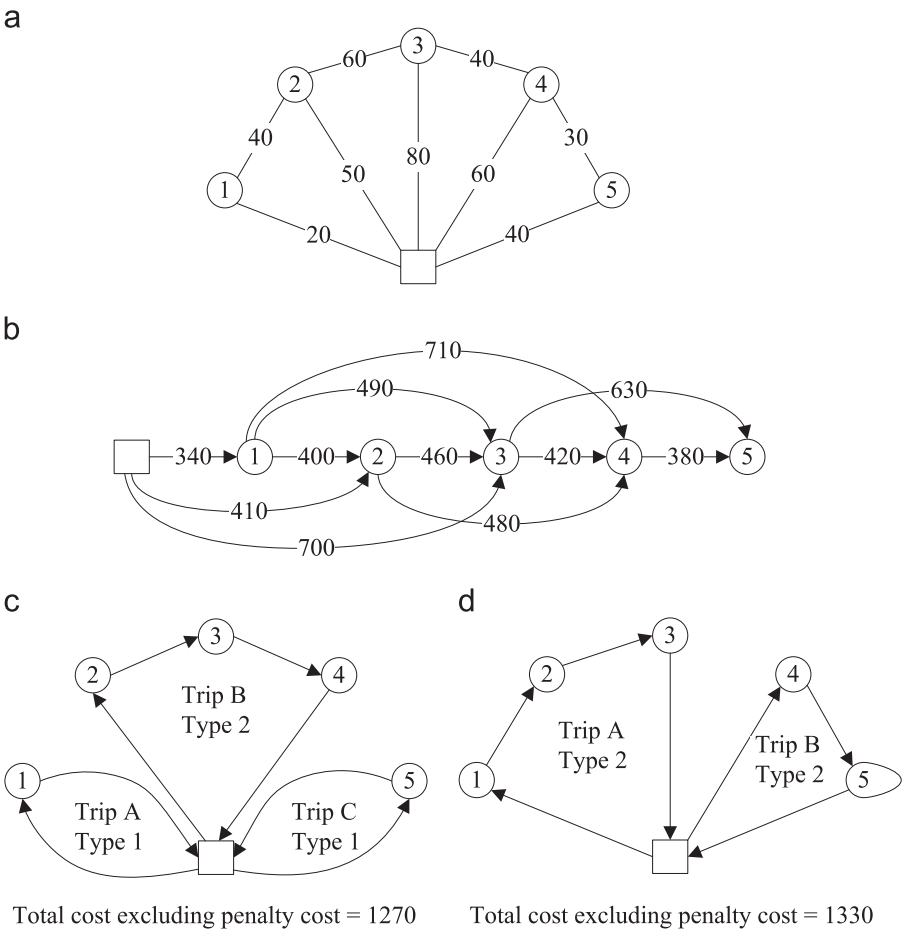

Fig. 1. Decoding of a chromosome. (a) Example data, (b) graph, (c) and (d) set of feasible trips.

Table 3
Trip information for Example 2.

(a) $\delta_1(k)$			(b) $\delta_2(k)$		
Trip($\delta_1(k)$)	$M^{(1)}$	$M^{(2)}$	Trip($\delta_2(k)$)	$M^{(1)}$	$M^{(2)}$
A(0, 1, 0)	150	50	A(0, 1, 2, 3, 0)	310	230
B(0, 2, 3, 4, 0)	480	240	B(0, 4, 5, 0)	460	150
C(0, 5, 0)	140	90			

From the result of Fig. 1, it can be observed that the first feasible set has a better performance regardless of the VRP stage. However, when the operation sequence and the time window restriction is taken into consideration, the penalty cost for the retailers in each trip have to be recalculated based on their corresponding arrival time. For example, when two penalty costs per order p_e and p_l are set as 0.5, the arrival time at retailer 1 is 790 and the penalty cost is equal to 195 in the first feasible set of trip A, $\delta_1(A)$. Compared to the second feasible set of trip B, $\delta_2(B)$, the arrival time at retailer 4 is 830 and the penalty cost is equal to 15. After recalculation process for each individual retailer in the trip, these objective values will be amended to 1605 and 1380, respectively. It can be obviously observed that a decision investigates production scheduling and orders delivery separately and sequentially without coordination under the trade-off of three costs can not obtain the global optimum performance for a production scheduling and delivery problem. Thus, when time window restriction is taken into the addressed problem, a revision process for recalculation of the objective value and a local search procedure for pursuing a better delivery sequence are introduced and described in the Section 3.1.3.

3.1.3. *Enhancing the algorithm*

Although a large number of feasible solutions might be generated and evaluated in the decoding procedure, it is still in a small fraction of all feasible solutions. Thus, some enhancing procedures are proposed to examine the potential improvement for a solution generated from Section 3.1.2. When the penalty cost is not taken into consideration, the addressed problem can be facilitate solved by applying the Johnson's algorithm. However, introducing the penalty cost term in the objective function emphasizes timely delivery and it is consistent with the just-in-time philosophy. In order to have the better performance for the address problem, minimization of total tardiness for the given set of retailers becomes an important issue. When the time windows of most retailers are more loosen compared to their corresponding completion time in the first stage of a specific vehicle. The delay optimization concept is applied to introduce a procedure (called revision process) for resolving the address problem with time windows.

When the feasible trips and the operation sequence would be generated from Sections 3.1.1 and 3.1.2 respectively, the difference between the arrival time t_i for each retailer i and its corresponding time window $[a_i, b_i]$ could be calculated. The variables h_i^- and h_i^+ stand for the corrected value of retailer i after or before its corresponding time window respectively, i.e. it is necessary for only one of the two variables availability. The variable h_i^* determine the flexibility could still be regulated when the adjustment does not violate time window restriction, i.e. $b_i - t_i - \min h_i^-$. In addition, the binary variables d_i^- and d_i^+ determine the arrival time for vehicle of retailer i after or before its corresponding time window. These binary variables indicate whether h_i^- or h_i^+ is non-zero respectively. The pseudo-code of the revision process is described as follows.

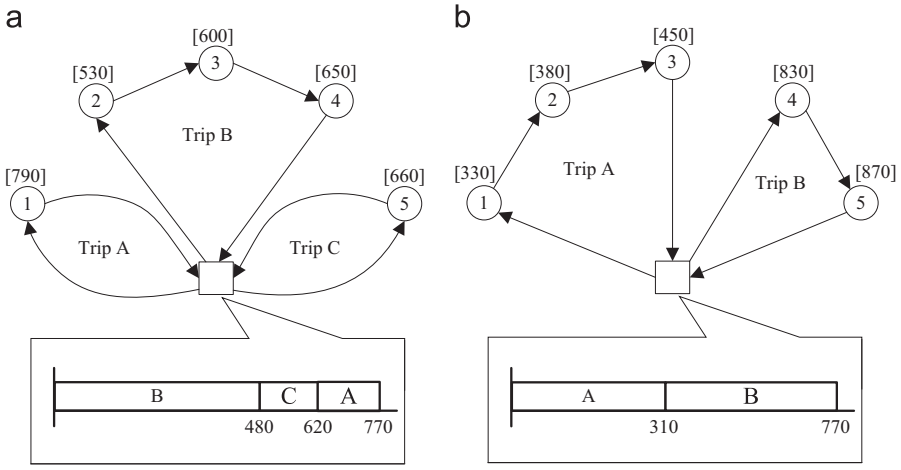

Fig. 2. Production schedule and delivery schedule of Example 2.

procedure: Revision process
input: $\delta(m)$
output: $f(\delta(m))$
begin
  $d_i^+ \leftarrow 0$; $d_i^- \leftarrow 0$; $h_i^+ \leftarrow 0$; $h_i^- \leftarrow 0$; $h_i^* \leftarrow 0$
  for $i := 1$ to $n_{\delta(m)}$ **do**
    if $t_i < a_i$ **then**
      $d_i^- \leftarrow 1$; $h_i^- \leftarrow a_i - t_i$
    else if $t_i > b_i$ **then**
      $d_i^+ \leftarrow 1$; $h_i^+ \leftarrow t_i - b_i$
    end if
  end for
  update $\min h_i^-$
  while $\sum_i d_i^- > \sum_i d_i^+$ **do**
    for $i :=$ 1 to $n_{\delta(m)}$ **do**
      while $a_i \le t_i \le b_i$ and $t_i + \min h_i^- > b_i$ **do**
        $h_i^* \leftarrow b_i - t_i - \min h_i^-$
      end while
    end for
    if $(\sum_i d_i^-)\min h_i^- > (\sum_i d_i^+)\min h_i^- + \sum_i h_i^*$ **then**
      for $i := 1$ to $n_{\delta(m)}$ **do** $t_i \leftarrow t_i + \min h_i$
      end for
    end if
  end while
  return the best scheduling of $f(\delta(m))$
end

The second enhancing procedure is an adaptation of the 2-opt algorithm. In decoding process, a mechanism that converts N retailers into k trips to form a k-job two-stage flowshop scheduling problem is designed. The processing time for each job (trip) within each stage has to be estimated. In the production stage (the first stage), the processing time for a job (trip) can be easily calculated by sum of the corresponding processing time for all members (retailers) in the trip. However, a lack of the optimal service sequences of retailers through a set of feasible trips in the delivery stage. All exchanges in the feasible solutions are examined until there is no feasible exchanged that improves upon the current solution in the second stage.

The pseudo-code of decoding process is formally stated as follows.

procedure: Decoding process
input: $\sigma = \{\sigma(1), \sigma(1), ..., \sigma(n)\}$
output: $f(\sigma)$
begin
  Split algorithm to find all feasible δ_j
  for $k := 1$ to n_{δ_j} **do**
    $M^{(1)}_{\delta_j(k)} \leftarrow \sum_{i \in \delta_j(k)} p_i$
    $M^{(2)}_{\delta_j(k)} \leftarrow \sum_{i \in \delta_j(k)} s_i + \tau_{\delta_j(k)}$
      local search: 2-opt
      update $M^{(2)}_{\delta_j(k)}$
    Johnson's algorithm to schedule the solution set
    Revision process to δ_j
  end for
  return $f(\sigma)$ and it's scheduling
end

3.2. Genetic operators

The solving process of each chromosome can be treated as a two-stage flowshop scheduling after decoding procedure is performed. Plenty of researchers apply GAs for solving the flow shop scheduling problems, such as Reeves (1995), Murata et al. (1996) and Cheng and Chang (2007). The parameters used in the proposed adaptive genetic algorithms (AGAs) are now presented as follows.

(1) Select a new population
  Selection strategy is important in GA search which can guide the solutions into a better performance direction. One effective way of determine the candidates for chromosomes is roulette wheel selection. The purpose of this selection scheme is to propagate high fitness characteristics in the population and to eliminate from the population the weaker chromosomes, thus keeping the search directed toward the optimization of the object function.
(2) Crossover
  The chromosomes in the candidate population go through crossover and mutation processes to produce offspring for the next generation. The longest common subsequence (Cormen et al., 1990) is applied to the mating system. Suppose that A and B are the individuals chosen for crossover $A = (5\ 1\ 6\ 2\ 3\ 9\ 4\ 7\ 8)$

and $B=(1\ 8\ 6\ 3\ 2\ 9\ 4\ 5\ 7)$. First, locate the longest common subsequence, $LCS=(1\ 6\ 2\ 9\ 4\ 7)$, of two candidate individuals. Then, copy common subsequence positions to each offspring and make exchanges for the remaining genes according to their original sequence. The resulting individuals are $A'=(8\ \underline{1}\ \underline{6}\ \underline{2}\ 3\ \underline{9}\ \underline{4}\ \underline{7}\ 5)$ and $B'=(\underline{1}\ 5\ \underline{6}\ 3\ \underline{2}\ \underline{9}\ \underline{4}\ 8\ \underline{7})$. 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.

(3) Mutation
After crossover operation, individuals go through a mutation operation with certain mutation rate. The displacement mutation is implemented in this article. For instance, if the segment (3 9 4) of individual A' were chosen to be mutated, this segment could be randomly selecting an inserting point for its new placement. Hence, the new string becomes $A''=(8\ \underline{3}\ \underline{9}\ \underline{4}\ 1\ 6\ 2\ 7\ 5)$.

3.3. Design of adaptive p_c and p_m

When AGA is implemented, it is essential to be able to identify whether the search is converged prematurely to a local optimum. One possible mechanism is observing the average population fitness value $\bar{f}$ relation to the best value at present f^*. $\bar{f}-f^*$ is likely to be less for a population that has converged to an optimum solution than that for a population scattered in the solution space. In addition, the value of p_c might depend on the fitness of present solutions, i.e., p_c should be directly affected by $f'-f^*$, where f' is the better fitness of candidates to be the crossover. Similarly, the value of p_m depends on the fitness f of offspring. The closer f and f^*, the p_m should be smaller. The expressions for p_c and p_m take the forms

$$p_c=k_1(f'-f^*)/(\bar{f}-f^*),\quad f'\le\bar{f}, \tag{18}$$

$$p_c=k_3,\quad f'>\bar{f} \tag{19}$$

and

$$p_m=k_2(f-f^*)/(\bar{f}-f^*),\quad f\le\bar{f}, \tag{20}$$

$$p_m=k_4,\quad f>\bar{f} \tag{21}$$

where $k_1, k_2, k_3, k_4\le 1.0$.

The significance of p_c and p_m in controlling GA performance has long been acknowledged in GA search. From the past studies such as Grefenstette (1986) and Srinivas and Patnaik (1994) have been devoted to identify optimal parameter setting for GAs. Thus, in this article, we assign k_1 and k_3 a value of 1.0 for ensuring implementation of crossover when both parents' fitness values are inferior compared to the average fitness value. For further control on the crossover rate, p_c decreases as the fitness of the adapter of parents tends to f^*. Moreover, as p_m increases, the next candidate have more opportunity to escape the local optimum. When a parent with inferior average fitness is obtained, it might 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 of parent's fitness value of $\bar{f}$ should also escape the local region to another highly diversified manner. The proposed framework of the AGA is shown as follows.

procedure: Adaptive Genetic algorithm
input: Integrated scheduling problem data set, AGA parameters
output: a near-optimal schedule
begin
  $t\leftarrow 0$
  initialize population P_t with integer coding
  fitness eval(P_t) by decoding procedure
  while (¬ termination condition) **do**
    use genetic operators to create new chromosomes N_c randomly
    fitness eval(N_c) by decoding procedure
    create a set $S=N_c\cup P_t$
    $t\leftarrow t+1$
    adopt a strategy to select chromosomes out of S to form P_t
    update p_c, p_m
  end
  output a near-optimal schedule;
end

4. Computational results

In this section, three computational experiments were conducted to test the performance of the AGA for solving the proposed integrated production scheduling and delivery problem. The first experiment was performed to compare the result of the proposed methods with mathematical programming and AGA heuristics. In this experiment, these methods were be evaluated in the sixteen problems ($N\le 5$). The second experiment was conducted in order to evaluate performance of these heuristics for the medium and large size problem ($N\ge 20$). The last experiment was illustrated the detail search process and logic of AGA heuristics.

In addition, the initial setup of the algorithms after preliminary tests is introduced as well. The initial populations were randomly chosen for each heuristic, each generation had 30 members and the process was continued through 300 generations. The performance of each test problem was presented by averaging the experimental results of 10 replicates. The AGA heuristic with a local search, as the mentioned before, is called AGA-2opt. The alternative without local search is named simple adaptive genetic algorithm, SAGA. For each AGA heuristic, 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.

4.1. Testing data sets generation

In this article, some computational experiments were carried out to observe the searching capabilities of proposed adaptive genetic algorithms. The algorithms were implemented in Visual C++ and run on a personal computer with an Intel Pentium D CPU at 3.40 GHz with 1.00GB of memory. The retailers' information including locations, demands and service times were directly adopted from Solomon's problem set (Safaei et al. (2010), data set contained C, R and RC. The other data were randomly generated, e.g. the unit processing time of each retailer followed $U(1, 10)$ and the total pick-up time was 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 $\times 50\%\ \sum_{i\in N} p_i d_i$

Where all retailers in each test problem set were divided into four groups, and each group had a group number should be 1, 2, 3, or 4. Each retailer in the test problem was randomly assigned into a group, thus, the time window for each retailer can be defined. For example, for a retailer i in the first group, the time window limit b_i of the retailer i was defined as the original b_i plus 50% of the total pick-up time and was plus 100% of the total pick-up time second group when the retailer was assigned into the second group, and so on.

4.2. Performance of the proposed AGA heuristics

To perform the feasibility and the adaptability of the proposed AGA heuristics, an experiment with 16 small size test problems are evaluated, and the results are summarized in Table 4. It was noted that the optimal solutions were obtained by running the commercial mathematical programming software Lingo (the mathematical model is provided in Section 2.3). The results shown in Table 4 revealed that in each case, SAGA and AGA-2opt had the same searching ability and the efficiency to find the solutions was not discernible.

Although the mathematical programming model could provide an accurate optimal solution when the solution space was small, e.g. the case of four retailers with two vehicles, the influence of the trace expansion of solution space made the searching efficiency into unstable. When solving the cases of five retailers with two vehicles, the optimum solutions were searched over 24 running hours and could not been solved. Moreover, the accurate solution was not been found under the commercial software when the demand point growth over five retailers. From the above experimental results, the proposed heuristics could efficiently solve the proposed integrated production scheduling and delivery problem and return a reliable solution. The efficiency of the algorithm makes it suitable for solving real larger cases.

4.3. Performance comparisons of proposed adaptive genetic algorithms

In this section, the performance of two proposed AGA heuristics in a variety of production and delivery environments are evaluated and compared. The experimental environment is defined as a retailer/geographical characteristic flexibility scenario (n/g) in which retailer n can be (20, 50, 80), geographical characteristic g can be (C, R, RC). Five different test problems for each scenario were evaluated for each of AGA heuristic and ten replicates runs for each of test problem. The computational results are summarized in Table 5. The results shown in Table 5 indicated that AGA-2opt performs better when number of retailer equals to 50 or 80. SAGA performs a little better than AGA-2opt with n equals to 20 but insignificance.

Table 4
Comparison of three search heuristics.

N	K	Lingo		SAGA		AGA-2opt	
		Object function	CPU time (s)	Object function	CPU time (s)	Object function	CPU time(s)
4	2	115.5	93	111.5	0.031	111.5	0.015
	2	122.5	68	122.5	0.031	122.5	0.015
	2	198.5	40	198.5	0.015	198.5	0.025
	2	327.5	52	327.5	0.015	327.5	0.016
	2	558.0	34	558.0	0.015	558.0	0.020
4	3	103.0	88	103.0	0.015	103.0	0.023
	3	97.0	235	97.0	0.015	97.0	0.015
	3	123.5	440	123.5	0.015	123.5	0.015
	3	118.5	793	118.5	0.015	118.5	0.015
	3	159.5	27,406	159.5	0.015	159.5	0.016
5	2	172.0	13,647	172.0	0.031	172.0	0.031
	2	312.5	1407	312.5	0.031	312.5	0.031
	2	–	–	119.0	0.031	122.0	0.031
	2	–	–	462.0	0.046	463.0	0.029
	2	–	–	147.0	0.031	147.0	0.034
	2	–	–	300.0	0.026	300.0	0.031

Table 5
Performance comparisons of two search methods.

n	g	Ex	SAGA			AGA-2opt		
			Object function		CPU time(s)	Object function		CPU time(s)
			Average	Standard deviation		Average	Standard deviation	
	C	1	1163.1	35.40	1.3	**1160.5**	46.24	2.9
		2	**1148.8**	30.90	1.2	1155.5	39.45	2.6
		3	**1152.6**	58.37	1.2	1168.1	40.98	3.1
		4	**1159.7**	44.28	1.3	1161.3	34.09	2.7
		5	**1140.6**	43.02	1.2	1150.4	36.49	2.9
20	R	1	1272.6	44.24	2.3	**1261.9**	43.94	5.8
		2	1213.1	49.32	2.2	**1203.7**	40.22	4.7
		3	1182.3	47.92	2.6	**1174.4**	64.48	6.7
		4	1226.2	48.71	2.6	**1211.1**	55.22	5.8
		5	1455.5	51.31	2.5	**1438.5**	61.37	6.3
	RC	1	**2163.9**	90.01	1.3	2201.8	67.28	2.5
		2	**2285.8**	103.13	1.3	2341.9	112.85	2.1
		3	**2267.2**	81.51	1.3	2324.2	95.60	2.8
		4	**2175.6**	94.63	1.2	2179.5	75.23	3.1
		5	**2236.3**	107.79	1.3	2264.9	83.20	3.0
		1	4211.9	227.90	12.2	**4183.3**	79.92	29.4
		2	4117.5	167.96	9.2	**4091.9**	160.72	36.9
	C	3	3851.6	52.41	9.9	**3813.6**	54.09	36.7
		4	**3785.8**	44.47	12.7	3786.1	51.94	43.2
		5	4230.8	149.03	11.8	**4226.7**	161.10	37.6
50	R	1	5350.8	322.01	18.3	**4956.4**	360.29	50.9
		2	**4136.7**	176.30	15.2	4169.7	170.19	41.0
		3	4783.8	306.02	15.6	**4745.6**	425.48	82.2
		4	**4769.4**	281.76	18.6	4790.8	178.56	68.5
		5	4672.9	510.95	15.6	**4538.7**	425.45	75.2
	RC	1	**6551.8**	149.47	10.4	6580.3	140.31	35.3
		2	6416.7	76.58	10.4	**6379.2**	98.08	47.9
		3	**7005.6**	323.87	9.2	7037.9	444.18	21.7
		4	6672.7	246.69	10.3	**6628.8**	188.31	30.3
		5	**7740.3**	292.38	9.8	7812.5	320.83	28.4
	C	1	**7801.5**	161.39	36.6	7877.9	153.43	119.5
		2	8304.8	178.05	37.7	**8275.8**	183.60	121.1
		3	8763.2	244.38	40.6	**8682.3**	249.64	169.0
		4	**8859.7**	214.10	49.7	8863.1	225.83	163.0
		5	9923.8	101.10	50.7	**9802.1**	279.94	213.4
80	R	1	9738.6	484.78	54.0	**9583.1**	447.24	238.1
		2	10,104.4	513.88	46.2	**10,062.8**	524.65	285.4
		3	9859.2	568.29	58.5	**9684.9**	599.95	270.9
		4	**9682.1**	689.52	59.5	9781.1	704.49	199.1
		5	9768.7	614.98	63.4	**9693.5**	410.77	267.5
	RC	1	12,974.3	720.26	18.7	**12,930.0**	873.92	108.4
		2	15,256.0	587.26	31.2	**15,027.6**	684.62	110.8
		3	**14,837.0**	588.22	33.9	14,880.4	705.99	89.6
		4	15,042.0	869.90	19.8	**14,635.4**	764.80	96.8
		5	15,859.2	911.78	30.2	**15,407.2**	893.34	132.8

For the computational efficiency, the CPU time is usually taken as an efficiency index. With increasing problem size, the experimental results from Table 5 indicated that SAGA has lower CPU time than AGA-2opt performed. It is obvious that the local search took a lot of extra computation time, as a result of experiencing a deterioration of efficiency that both algorithms were efficiency deterioration in the group R especially in AGA-2opt. It is conjectured that variability of

the retailers scattered was an important factor in the expansion of the solution space for our proposed searching mechanisms.

4.4. 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 the testing problems is set same environment as in Chang and Lee (2004) which is N orders are processed on a single machine and then delivered to two areas. A group of customers are located in close proximity to each other as one customer area. The coordinates of each retailer are randomly generated in the interval [0, 30].

When we relax the time window restriction and taken homogeneous fleet into consideration, three scenarios with the number of the retailer equals to 30, 50, and 80 are evaluated, respectively. Six testing problems are run in each scenario. The computational results are summarized and shown in Table 6. The results in Table 6 reveal that both GA and AGA perform better than H3 in each scenario. On average, GA and AGA are more efficient than H3 by 11.38% and 13.19% respectively. Paired t-tests reveal that the performances of the proposed GA based heuristics are statistically better than H3 at the 95% significance level. It is noted that the performance of both GA and AGA is not discernible.

4.5. Analysis of multi vehicle types

To investigate the impact of heterogeneous fleet provided in the delivery stage, an experiment is conducted. The starting point for the construction of the instances used in our computational experiment is extracted from the Solomon benchmark data set (Solomon, 1987) for the vehicle routing problem with time windows. The data set includes geographic information, demand and service time, vehicle capacity and fixed cost. As the magnitude of the potential savings from combination of multiple vehicle types, we have taken an experiment with 20, 50, 80 retailers. Where the vehicle fleet in test problems is divided into five groups, and each group has its corresponding number of vehicle types. Vehicle capacity increasing in the order $Q_1 < Q_2 < \ldots < Q_5$; e.g. vehicle group equal to 3 means that the addressed problem is solved by a combination of the first three types of vehicles. The computational results are summarized and shown in Fig. 3.

The results from Fig. 3 reveal that the total cost gradually decreases as the vehicle type employed in the delivery stage increasing. For example, in the 80 retailers case, the total cost with five kinds of vehicle type is lower than four types 13% and lower than homogeneous fleet 32%. In addition, more kinds of vehicle types provided in the delivery stage is not only reduced the total cost but fixed vehicle cost and variable routing cost. However, the penalty cost would not validity of any inference in favor of a trend of heterogeneous fleet strategy.

5. Conclusions

This article proposes an integration concept for a production scheduling and delivery problem with the minimization of total cost which includes transportation cost, vehicle arrangement cost and penalty costs, subjected to satisfy all demands of each retailer. In the delivery stage, heterogeneous fleet of vehicles is assumed.

For the addressed integrated production scheduling and delivery problem, an integer nonlinear programming model has been presented. However, solving the mathematical model can provide the benchmark solution for the small scale problem ($N \leq 5$), but the influence of the trace expansion of solution space made the searching efficiency into unstable. Therefore, two adaptive genetic algorithms have been proposed for solving medium/large size problems. The solution quality and efficiency of these heuristics were evaluated through randomly generated test problems in various environments. The results indicated that these two heuristics could obtain the benchmark solution for small sized test problems.

For the medium/large size problems, the results from the comparisons of the proposed algorithms revealed that simple adaptive genetic algorithm is more efficient for solving the addressed problems. However, adaptive genetic algorithm with 2-opt has better search ability when the number of retailer increasing. Although the search scheme between the two heuristics was not identical but achieved similar search quality when the number of retailer is less than fifty.

Table 6
Comparison with other heuristics.

N	SAGA/H3	AGA-2opt/H3	SAGA < H3	AGA-2opt < H3
20	0.89	0.86	< 0.001	< 0.001
50	0.89	0.87	< 0.001	< 0.001
80	0.89	0.88	< 0.001	< 0.001

Note: Columns SAGA < H3 and AGA-2opt < H3 lists p-values for the comparison.

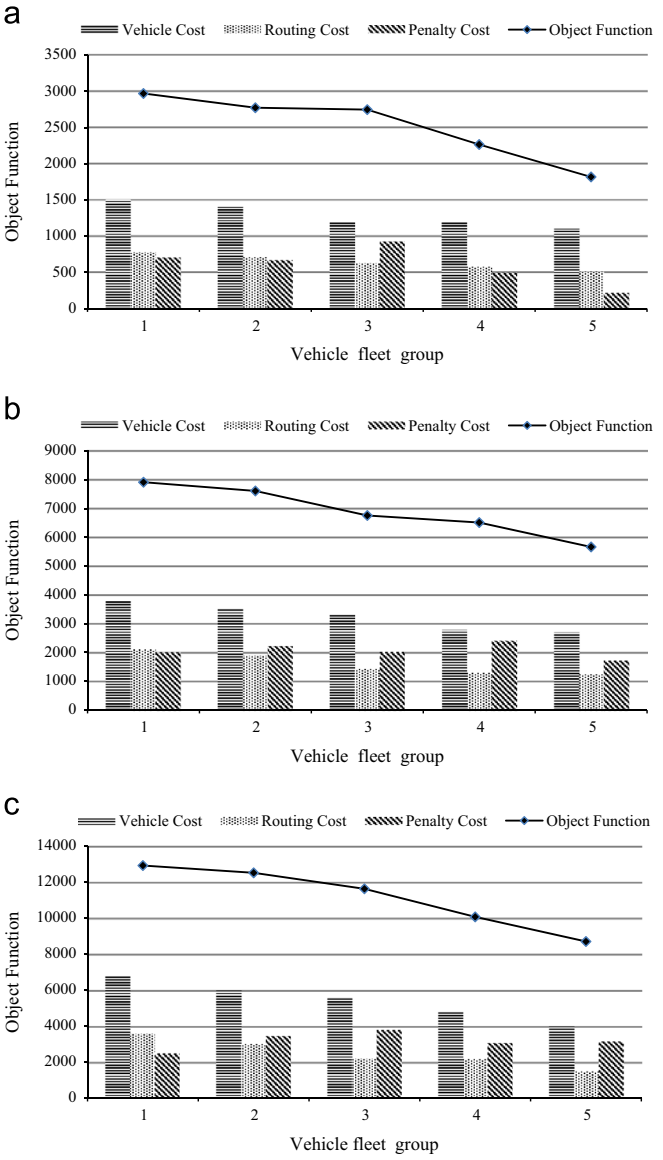

Fig. 3. Object function under different vehicle type. (a) Cost structure analysis (N=20), (b) Cost structure analysis (N=50), and (c) Cost structure analysis (N=80).

Hence, when the number of retailer is less than 50, the standard adaptive genetic algorithm is recommended for solving the problem owing to the searching efficiency consideration. Otherwise ($N > 50$), adaptive genetic algorithm with 2-opt is recommended to ensure the solution quality.

To ensure the proposed AGA heuristics have sufficient solving ability, a performance comparison with existing heuristic, H3, is investigated. The computational result indicates that the SAGA and AGA-2opt are both better than the existing H3 without taking the time window restriction into consideration.

Finally, an experiment is conducted for analyzing the impact of heterogeneous fleet provided in the delivery stage as well. The experimental results indicate that more kinds of vehicle types provided in the delivery stage is not only reduced the total cost but also fixed vehicles cost and variable routing cost. However, the penalty cost would not validity of any inference in favor of a trend of heterogeneous fleet strategy.

In the future study, the proposed coordination of production scheduling and delivery model can be further improved by taken setup cost and inventory holding cost into the model. Furthermore, more kinds of picking procedure can be expanded in the coordination model.

References

Adulyasak, Y., Cordeau, J.F., Jans, R., 2014a. Formulations and branch-and-cut algorithms for multi-vehicle production and inventory routing problem. INFORMS J. Comput. 26 (1), 103–120.

Adulyasak, Y., Cordeau, J.F., Jans, R., 2014b. Optimization-based adaptive large neighborhood search for the production routing problem. Transp. Sci. 48 (1), 20–45.

Bredström, D., Ronnqvist, M., 2008. Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. Eur. J. Oper. Res. 191 (1), 19–31.

Chang, Y.C., Lee, C.Y., 2004. Machine scheduling with job delivery coordination. Eur. J. Oper. Res. 158 (2), 470–487.

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

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

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

Day, J.M., Wright, P.D., 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 J. Comput. 20 (1), 21–33.

Grefenstette, J.J., 1986. Optimization of control parameters for genetic algorithms. IEEE Trans. Syst. Man Cybern. 16 (1), 122–128.

Holland, J.H., 1992. Adaptation in Natural and Artificial Systems. University of Michigan press, Ann Arbor, MI.

Ishibuchi, H., Yoshida, T., Murata, T., 2003. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans. Evolut. Comput. 7 (2), 204–223.

Iyer, S.K., Saxena, B., 2004. Improved genetic algorithm for the permutation flowshop scheduling problem. Comput. Oper. Res. 31 (2), 593–607.

Johnson, S.M., 1954. Optimal two- and three-stage production schedules with setup times included. Naval Res. Logist. Q. 1 (1), 61–68.

Li, S., Yuan, J., 2009. Scheduling with families of jobs and delivery coordination under job availability. Theor. Comput. Sci. 410 (47–49), 4856–4863.

Liang, T.F., 2008. Integrating production–transportation planning decision with fuzzy multiple goals in supply chains. Int. J. Prod. Res. 46 (6), 1477–1494.

Liu, S., Huang, W., Ma, H., 2009. An effective genetic algorithm for the fleet size and mix vehicle routing problems. Transp. Res. Part E: Logist. Transp. Rev. 45 (3), 434–445.

Low, C., Li, R.K., Chang, C.M., 2013. Integrated Scheduling of Production and Delivery with Time Windows. Int. J. Prod. Res. 51 (3), 897–909.

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

Michalewicz, Z., 1996. Genetic algorithms + Data structures = Evolution programs. Springer-Verlag, New York.

Murata, T., Ishibuchi, H., Tanaka, H., 1996. Genetic algorithms for flowshop scheduling problems. Comput. Ind. Eng. 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. Eur. J. Oper. Res. 177 (3), 2069–2099.

Prins, C., 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. Comput. Oper. Res. 31 (12), 1985–2002.

Reeves, C.R., 1995. A genetic algorithm for flowshop sequencing. Comput. Oper. Res. 22 (1), 5–13.

Safaei, A.S., et al., 2010. Integrated multi-site production–distribution planning in supply chain by hybrid modeling. Int. J. Prod. Res. 48 (14), 4043–4069.

Samaranayake, P., Laosirihongthong, T., Chan, T.S., 2011. Integration of manufacturing and distribution networks in a global car company—network models and numerical simulation. Int. J. Prod. Res. 49 (11), 465 3127–3149.

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

Srinivas, M., Patnaik, L.M., 1994. Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans. Syst. Man Cybern. 24 (4), 656–667.

Steinrücke, M., 2011. An approach to integrate production—transportation planning and scheduling in an aluminum supply chain network. Int. J. Prod. Res. 49 (21), 6559–6583.

Wang, X., Cheng, T.C.E., 2009. Production scheduling with supply and delivery considerations to minimize the makespan. Eur. J. Oper. Res. 194 (3), 743–752.

Zdrzalka, S., 1995. Analysis of approximation algorithms for single-machine scheduling with delivery times and sequence independent batch setup times. Eur. J. Oper. Res. 80 (2), 371–380.

Zhong, W., Gyorgy, Dosa, Tan, Z., 2007. On the machine scheduling problem with job delivery coordination. Eur. J. Oper. Res. 182 (3), 1057–1072.