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

Applied column generation-based approach to solve supply chain scheduling problems

Yung-Chia Chang [a], Kuei-Hu Chang [b] & Teng-Kai Chang [c]

[a] Department of Industrial Engineering and Management, National Chiao Tung University, Hsinchu, 300, Taiwan

[b] Department of Management Sciences, R.O.C. Military Academy, Kaohsiung, 830, Taiwan

[c] Graduate School of Industrial Engineering and Management, National Chiao Tung University, Hsinchu, 300, Taiwan

Published online: 03 Apr 2013.

To cite this article: Yung-Chia Chang, Kuei-Hu Chang & Teng-Kai Chang (2013) Applied column generation-based approach to solve supply chain scheduling problems, International Journal of Production Research, 51:13, 4070-4086, DOI: 10.1080/00207543.2013.774476

To link to this article: http://dx.doi.org/10.1080/00207543.2013.774476

International Journal of Production Research, 2013
Vol. 51, No. 13, 4070–4086, http://dx.doi.org/10.1080/00207543.2013.774476

Taylor & Francis
Taylor & Francis Group

Applied column generation-based approach to solve supply chain scheduling problems

Yung-Chia Chang[a], Kuei-Hu Chang[b]* and Teng-Kai Chang[c]

[a]*Department of Industrial Engineering and Management, National Chiao Tung University, Hsinchu 300, Taiwan;* [b]*Department of Management Sciences, R.O.C. Military Academy, Kaohsiung 830, Taiwan;* [c]*Graduate School of Industrial Engineering and Management, National Chiao Tung University, Hsinchu 300, Taiwan*

(*Received 3 March 2012; final version received 27 January 2013*)

More and more enterprises have chosen to adopt a made-to-order business model in order to satisfy diverse and rapidly changing customer demand. In such a business model, enterprises are devoted to reducing inventory levels in order to upgrade the competitiveness of the products. However, reductions in inventory levels and short lead times force the operation between production and distribution to cooperate closely, thus increasing the practicability of integrating the production and distribution stages. The complexity of supply chain scheduling problems (integrated production and distribution scheduling) is known to be NP-hard. To address the issues above, an efficient algorithm to solve the supply chain scheduling problem is needed. This paper studies a supply chain scheduling problem in which the production stage is modelled by an identical parallel machine scheduling problem and the distribution stage is modelled by a capacitated vehicle routing problem. Given a set of customer orders (jobs), the problem is to find a supply chain schedule such that the weighted summation of total job weighted completion time and total job delivering cost are minimised. The studied problem was first formulated as an integer programme and then solved by using column generation techniques in conjunction with a branch-and-bound approach to optimality. The results of the computational experiments indicate that the proposed approach can solve the test problems to optimality. Moreover, the average gap between the optimal solutions and the lower bounds is no more than 1.32% for these test problems.

Keywords: dynamic programming; column generation approach; identical parallel machine; supply chain scheduling

1. Introduction

Supply chain management has been one of the most widely discussed topics in the modern business world. A supply chain represents all stages at which raw materials and components are transformed into finished products to be delivered to end customers. The main objective of supply chain management is to satisfy customer demands through the most efficient use of resources. Traditional approaches consider production and distribution separately and sequentially, with little or no coordination between these two stages. When there is a sufficient amount of inventory between production and distribution, the operations of these two stages can be decoupled; hence, traditional approaches are able to deliver reasonable and effective solutions. However, a high inventory level leads to increased inventory holding costs and longer material flow times through the supply chain; this would impair a company's ability to respond promptly to the demand changes and eventually worsen the total supply chain profitability.

In the current competitive global market, companies are forced to lower the amount of inventory needed across their supply chain but still have to be more responsive to customers' requirements. Reduced inventory creates a closer interaction between production and distribution activities and thus increases the practical usefulness of integrated models (Sarmiento and Nagi 1999). Nowadays, many enterprises have adopted a make-to-order business model, where products are custom-made and are delivered to customers within a very short lead time directly from the factory without the intermediate step of finished product inventory (Su et al. 2010; Shao and Dong 2012). Co-ordinating the production and distribution activities is complex. The complexity of supply chain scheduling problems (integrated production and distribution scheduling) is known to be NP-hard. A decision that is optimal with respect to both stages (i.e. production and distribution) might not be an optimal decision for each stage, especially when each stage has its own performance measure. The optimal solution to one stage may adversely affect the other. Consider a simple example in which a company produces different products for multiple customers who reside at different locations. Products have to be delivered to customers by some vehicle right after they are produced. The travel time and transportation cost are

*Corresponding author. Email: evenken2002@yahoo.com.tw

non-negligible. To save the distribution cost, orders from closely located customers may be delivered together within one batch. It is then reasonable to schedule these products such that they are produced at similar times. However, producing these products may require very different setups and thus would result in high production cost. On the other hand, scheduling similar products together to save the production cost might result in higher distribution costs if they are ordered by different customers who are located far apart from one another.

Among the existing models that explicitly address the integration of production scheduling and distribution scheduling and routing, the main focus is on clarifying the problem complexity of various special cases for a general model with different objectives, followed by developing dynamic programming algorithms with respect to some special cases, or analysing the worst-case and asymptotic performance of some heuristics developed for these special cases (Lee and Chen 2001; Chang and Lee 2004; Li, Vairaktarakis, and Lee 2005; Pundoor and Chen 2005). They either assume direct shipment (i.e. no routing decisions need to be made) or consolidation of customers into one or two areas to simplify the distribution stage. So far, only some of the problems studied by Chen and Vairaktarakis (2005) do explicitly consider the routing decisions. They develop heuristics and perform computational experiments on the scale of 100 jobs and five customer locations. Another set of studies assumes that job delivery can be made instantaneously without any transportation time and use batch delivery cost to account for distribution activities (Wang and Cheng 2000; Hall and Potts 2003). Routing decisions are not considered in these studies. As pointed out by Chen and Vairaktarakis (2005), more academic research is needed to model direct production-distribution interactions and develop problem-solving techniques that can be used in practice. Most of the special cases of this class of problems have already been classified as NP-hard, especially when there are multiple machines for production and when there are multiple customer locations. Actually, even standalone single-stage problems (i.e. only scheduling or only vehicle routing) are very complicated. For example, the problem that involves one machine, multiple delivery locations, multiple delivery vehicles available, an equal-sized job when considering loading to a vehicle, and with the objective of minimising the time when the last job is delivered to its customer is NP-hard in the strong sense, even if the processing times at the first stage are all equal to zero (Lee and Chen 2001).

The proposed work is to address the supply chain scheduling problems (integrated production and distribution scheduling) from an operational perspective by considering the detail scheduling at an individual job level. The problems can be viewed as two-stage supply chain scheduling problems. At the first stage, jobs are arranged to be processed by some manufacturing facility that can be modelled as a single machine, a set of parallel machines, or a series of flow shop machines. After processing, jobs must be delivered to some customers who may reside at different locations by some transportation means (e.g. delivery vehicles). The problems at the second stage involve detailed vehicle dispatching and routing decisions that are typically referred to as vehicle routing problems, which have also been extensively discussed in the literature (e.g. Bramel and Simchi-Levi 1997).

This research planned to develop exact algorithms to find the optimal solutions of the study problems. To achieve that, problems are first formulated by integer programming (IP) and solved by a branch and bound approach in conjunction with the column generation technique. The rest of this article is organised as follows. We briefly review the literature in Section 2. Section 3 presents the research method, which jointly considers scheduling activities of production and distribution in an integrated manner. Computational experiments were tested in Section 4. The final section makes conclusions.

2. Literature review

2.1 *Production scheduling*

Due to the popularity of the just-in-time philosophy and total quality management concept, schedule plays an important role in achieving the goal of on-time delivery, which in turn is one of the crucial factors for customer satisfaction. Most researchers studied the machine scheduling problems without taking into account the underlying physical material handling systems. The assumption was made that an unlimited capacity is associated with the material handling system and that transportation times between machines are negligible. Only a few researchers have considered the joint optimisation of machine scheduling and job transporting in an underlying material handling system. The first research that explicitly considered the transportation issue is probably by Maggu and Das (1980). They studied a two-machine flowshop makespan problem, in which they assumed that there are unlimited buffers on both machines and that a sufficient number of transporters are available to transport jobs from one machine to the other with job-dependent transportation times. Kise (1991) considered a similar problem but with only one transporter capable of transporting at most one job at a time. He shows that this problem is ordinarily NP-hard, even with job-independent transportation times. Other related studies that considered a similar problem, with limited buffer space and unit transporter capacity, was presented by Stevens and Gemmill (1997) and Ganesharajah, Hall, and Sriskandarajah (1998).

Another line of research has focused on problems in which job completion time is defined as the time when a job is delivered to the destination. Hall and Shmoys (1992) studied a single-machine problem with unequal job arrival times and delivery times. In their model, they implicitly assumed that a sufficient number of vehicles are available in the system in order to deliver a processed job to the customer immediately. They provided a heuristic, justified by a worst-case analysis. Lee and Chen (2001) studied machine scheduling problems with explicit transportation considerations. Two types of transportation situations are considered in their models. The first type, called Type-1 transportation, involves intermediate transportation of jobs from one machine to another for further processing. The second type, called Type-2 transportation, involves the transportation that is provided to deliver finished jobs to their destinations. Jobs are delivered in batches by the transporter(s). They assumed that all jobs require the same physical space on the transporter. Both transportation capacity and transportation times are considered in their models.

Hurink and Knust (2001) independently studied robotic operations in the manufacturing environment, which is an intermediate type of transportation. They assumed that there is only one transporter available to carry each single job and that the return time of the transporter is zero. Chang and Lee (2004) extended Lee and Chen's (2001) work to the situation in which each job occupies a different amount of space in the vehicle. Li, Vairaktarakis, and Lee (2005) extended Chang and Lee's work by considering delivery to multiple customers at different locations.

2.2 *Vehicle routing problem*

If we consider only the vehicle routing problem, a tremendous amount of research has been conducted in the field of vehicle routing problems in the last three decades. Various vehicle routing problem models have appeared. For the results on basic vehicle routing problem models without time windows, many survey papers have appeared, including, among others, Bodin (1990) and Laporte (1992). A more complicated model that involves customer time windows has been studied extensively more recently. Some representative results for this model include, among others, Desrochers, Desrosiers, and Solomon (1992), Bramel and Simchi-Levi (1997), and Fisher, Jornsten, and Madsen (1997). The most general vehicle routing problem model is the so-called pickup-delivery model. For results on this model, see the papers by Savelsbergh and Sol (1998).

Many researchers have considered so-called inventory-routing problems, where customers' demands are replenished over time and where the frequency and size of customer deliveries are decision variables as well. These problems can be viewed as an extension of pure vehicle routing problems, where customer demands are filled only once. There are also results addressing the integration of vehicle routing with other decisions. Many researchers consider problems of integrating location and routing, where routing decisions are incorporated into traditional location models. Some representative papers include, among others, Stowers and Palekar (1993) and Min, Jayaraman, and Srivastava (1998).

2.3 *Column generation approach*

Dantzig and Wolfe (1960) pioneered this fundamental idea, developing a strategy to extend a linear programme column-wise, as needed in the solution process. The column generation that was devised for linear programmes is a success story in large-scale integer programming (Lübbecke and Desrosiers 2005). Among all existing exact methods, branch and bound approaches that are based on the column generation method have recently been proven to be the best for both machine scheduling problems and vehicle routing problems. A number of studies have reported promising results obtained by applying column generation-based algorithms to solve machine scheduling problems (e.g. Van den Akker, Hoogeveen, and Van de Velde 1999; Chen and Powell 1999; Chen 2004; Bard and Rojanasoonthon 2006). Applying this approach, the size of solvable problems is larger than others. For example, for the parallel machine problem with the objective of minimising the total weighted completion times, Chen and Powell (1999) can solve the problem with 100 jobs and 20 machines, which is more than three times larger than the problem size solved by the best existing result using other techniques. The column generation approach has also successfully solved various types of vehicle routing problems (e.g. Desrochers, Desrosiers, and Solomon 1992; Savelsbergh and Sol 1998; Desaulniers 2007). For problems with tight customer time windows, it is reported (Desrochers, Desrosiers, and Solomon 1992) that the column generation approach can solve instances with 100 customers and is faster than any other approach. In addition, column generation-based approaches are also superior for many other difficult combinatorial optimisation problems, including airline crew scheduling problems, cutting stock problems, the graph colouring problem, lot scheduling problems, the generalised assignment problem, and telecommunications network design problems. It has been shown that the column generation-based approaches are computationally more effective than any other approach for the corresponding problems. A general framework of the column generation approach is given by Barnhart et al. (1998).

3. Proposed approach

This paper applied a column generation-based algorithm to solve supply chain scheduling problems (integrated production and distribution scheduling). First, it must establish a mathematical programming model. The original complex master problem will decompose into a restricted master problem and several subproblems by the column generation method's characteristics. Each subproblem is representing the single machine scheduling problem and the single batch delivery problem, using the dynamic programming method to obtain the solution of a subproblem.

3.1 *Problem statement and notation definition*

The problem is described as follows. There are n jobs, k customers, m unrelated parallel machines, and an unlimited number of homogeneous vehicles available in the system. Each vehicle can only carry at most z jobs in one delivery trip. The objective is to minimise $\alpha \sum wC + (1-\alpha)TC$, where α is a given constant ($0 \leq \alpha \leq 1$) representing the decision-maker's relative preference on customer service level and total distribution cost. In this expression, TC is the total distribution cost that includes the fixed and variable parts of the transportation cost, and wC is the order completion time multiplication weight to transform cost.

This research that explored the problem has made the following assumptions:

- In the production part

  (1) All jobs and machines are available at time 0.
  (2) Each machine can process only one job at any given time.
  (3) Does not consider the job rework or the machine breakdown.
  (4) Each job $j \in N$ needs to be processed by any one of the machine without setup time.

- In the distribution part

  (1) Only a logistics centre (manufacturing facility) and each vehicle can only transport once.
  (2) Delivery vehicles without quantity and distance limit.
  (3) The capacity upper bound and speed of each vehicle are the same.
  (4) All vehicles are available and parking at the manufacturing facility at time 0.
  (5) Each vehicle must finally return to logistics centre.

According to assumptions mentioned above, this research constructs the problem into a binary integer programming model. The corresponding parameters are defined as follows:

N set of n jobs, $N = \{1, 2, \ldots, n\}$,
K set of k customers, $K = \{1, 2, \ldots, k\}$,
N_h set of jobs that are ordered by customer $h \in K$,
n_h number of jobs that are ordered by customer $h \in K$, i.e. $n_h = |N_h|$,
V set of homogeneous vehicles, $V = \{1, 2, \ldots, v\} (v \geq n)$,
M set of m identical parallel machines, $M = \{1, 2, \ldots, m\}$,
B_j set of jobs that can be processed before job $j \in N$,
p_j processing time of job $j \in N$,
w_j weight of job $j \in N$,
t_{lh} the transportation time from customer $l \in K$ to customer $h \in K$,
v_{hl} the variable cost travelling from customer $h \in K$ to customer $l \in K$,
F_{fix} the fixed cost by sending out one vehicle to deliver jobs, and
z the capacity of each vehicle.
α preference coefficient, $0 \leq \alpha \leq 1$.

The decision variables are defined as:

c_j the time when job $j \in N$ is finished processing by the manufacturing facility,
C_j the time when job $j \in N$ is delivered to its customer,
d_0^u the time when vehicle $u \in V$ departs from the manufacturing facility,

d_h^u the time when vehicle $u \in V$ arrives at customer $h \in K$,
$x_{ij} = 1$ represents job $j \in N$ is processed immediately before job $j \in Ni$ on this machine; otherwise $x_{ij} = 0$,
$y_{hl}^u = 1$ represents vehicle $u \in V$ travels from customer $h \in K$ to customer $l \in K \ \{h\}$; otherwise $y_{hl}^u = 0$,
$a_j^u = 1$ represents job is delivered by vehicle $u \in V$; otherwise $a_j^u = 0$.

For convenience, this paper introduces two dummy jobs, job 0 and job $n + 1$, with zero processing time and zero sizes; c_0 is defined as 0. Moreover, let customer 0 represent the location of the manufacturing facility where each vehicle is initially located, and customer $k + 1$ represents the location in the manufacturing facility where each vehicle will return to. There is no travel time or cost between customer 0 and customer $k + 1$. Based on the definition above, this research can construct a binary integer programming mathematical model as follows:

$$Min\ \alpha\left(\sum_{j\in N} w_j C_j\right) + (1-\alpha)\left(F_{fix}\sum_{u\in V}\sum_{h\in K} y_{0h}^u + \sum_{u\in V}\sum_{\substack{h,l\in K\cup\{0,k+1\}\\ h\neq l}} v_{hl} y_{hl}^u\right) \tag{1}$$

Subject to

$$\sum_{j\in N} x_{0j} \le m \tag{2}$$

$$\sum_{i\in B_j\cup\{0\}} x_{ij} = 1 \qquad \forall j \in N \cup \{n+1\} \tag{3}$$

$$c_j = \sum_{i\in B_j\cup\{0\}} c_i x_{ij} + p_j \qquad \forall j \in N \cup \{n+1\} \tag{4}$$

$$\sum_{l\in K\setminus\{h\}} y_{hl}^u - \sum_{l\in K\setminus\{h\}} y_{lh}^u = 0 \qquad \forall u \in V\ ,\ \forall h \in K \tag{5}$$

$$\sum_{h\in K} y_{h,k+1}^u - \sum_{h\in K} y_{0,h}^u = 0 \qquad \forall u \in V \tag{6}$$

$$\sum_{h\in K} y_{0h}^u \le 1 \qquad \forall u \in V \tag{7}$$

$$n_h \sum_{l\in K\setminus\{h\}} y_{lh}^u - \sum_{j\in N_h} a_j^u \ge 0 \qquad \forall u \in V\ ,\ \forall h \in K \tag{8}$$

$$\sum_{u\in V} a_j^u = 1 \qquad \forall j \in N \tag{9}$$

$$\sum_{j\in N} a_j^u \le z \qquad \forall u \in V \tag{10}$$

$$d_h^u = \sum_{l\in K\cup\{0\}\setminus\{h\}} (d_l^u + t_{lh}) y_{lh}^u \qquad \forall u \in V\ ,\ \forall h \in K \tag{11}$$

$$c_j - \sum_{u\in V} d_0^u a_j^u \le 0 \qquad \forall j \in N \tag{12}$$

$$C_j - \sum_{u \in V} d_h^u a_j^u = 0 \qquad \forall j \in N_h,\ \forall h \in K \tag{13}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in N \cup \{0\},\ j \in N \cup \{n+1\} \tag{14}$$

$$y_{hl}^u \in \{0,1\} \qquad \forall h \in K \cup \{0\},\ l \in K \cup \{k+1\}, u \in V \tag{15}$$

$$a_j^u \in \{0,1\} \qquad \forall j \in N,\ u \in V \tag{16}$$

The objective function (1) minimises the weighted sum of the total weighted job completion time and total distribution cost. Constraint (2) represents at most m jobs processed in m partial machines at time 0. Constraints (3) ensure that each job either is processed at time 0 or has a job processed immediately before it. Constraints (4) define the time when each job is finished processing by a machine and is ready for delivery. Constraints (5) and (6) represent the conservation-flow requirements. Constraints (7) ensure each vehicle is used at most once. Constraints (8) make sure that if a job is delivered to its customer by a vehicle, then that vehicle has to visit that customer. Constraints (9) guarantee that each job will be delivered to its customer exactly once. Constraints (10) associate with the vehicle capacity. Constraints (11) and (13) define the job completion time; i.e. the time when a job is delivered to its customer. Constraints (12) make sure that a job will not be delivered until a machine completes processing that job. Constraints (14) to (16) give the binary constraints for the decision variables.

Clearly, this formulation is very complicated. Constraints (11) even incorporate nonlinear constraints that make this problem intractable. Therefore, it is not appropriate to solve this formulation directly. This research applied Dantzig-Wolfe decomposition to decompose this mathematical model to become a set-partitioning master problem and a sub-problem. The parameters for the set partition-type formulation are defined as follows.

S set of all possible single-machine partial schedules for processing jobs on a single machine. Each job schedule specifies what jobs are processed in what order and completed at what times.
N set of n jobs, $N = \{1, 2, \ldots, n\}$,
M set of m identical parallel machines, $M = \{1, 2, \ldots, m\}$,
Q set of all possible delivery schedules for a vehicle. Each delivery schedule specifies what jobs are delivered to their corresponding customers at what times and also specifies when a vehicle starts to deliver jobs. Note that a delivery schedule could be empty; i.e. it may not deliver any job.
w_j weight of job $j \in N$,
$a_{js} = 1$ if partial schedule s ($s \in S$) covers job j ($j \in N$); 0 otherwise,
$b_{jq} = 1$ if delivery schedule q ($q \in Q$) covers job j ($j \in N$); 0 otherwise,
d_q the time when delivery schedule q ($q \in Q$) starts to deliver jobs,
c_{js} the completion time of job j ($j \in N$) in partial schedule s ($s \in S$),
C_{jq} the completion time of job j ($j \in N$), defined as the time when it is delivered to its customer under delivery schedule q ($q \in Q$),
F_q total transportation cost for delivery schedule q ($q \in Q$),
F_{fix} the fixed cost by sending out one vehicle to deliver jobs,
α preference coefficient, $0 \leq \alpha \leq 1$.

Decision variables:

$x_s = 1$ if partial schedule s ($s \in S$) is selected in the optimal solution.
$y_q = 1$ if delivery schedule q ($q \in Q$) is selected in the optimal solution.

The set partitioning-type formulation, denoted as [SP], is as follows:

$$[\text{SP}] \quad \textit{Min} \qquad \sum_{q \in Q} \left[\left(\alpha \sum_{j \in N} w_j C_{jq} b_{jq} \right) + (1 - \alpha)(F_{fix} + F_q) \right] y_q \tag{17}$$

Subject to

$$\sum_{s\in S} a_{js}x_s = 1 \qquad \forall j \in N \tag{18}$$

$$\sum_{s\in S} x_s \le m \tag{19}$$

$$\sum_{q\in Q} b_{jq}y_q = 1 \qquad \forall j \in N \tag{20}$$

$$\sum_{q\in Q} y_q \le n \tag{21}$$

$$\sum_{s\in S} c_{js}a_{js}x_s - \sum_{q\in Q} d_q b_{jq}y_q \le 0 \qquad \forall j \in N \tag{22}$$

$$x_s \in \{0,1\}, \forall s \in S \tag{23}$$

$$y_q \in \{0,1\}, \forall q \in Q \tag{24}$$

Equation (17) is the weighted sum of the total weighted job completion time and total distribution cost. Constraints (18) make sure that each job has to be processed once by a certain machine. Constraint (19) guarantees that at most m partial machine schedules will be selected in any feasible solution. Constraints (20) ensure that each job will be delivered to its customer exactly once. Constraint (21) ensures the distribution path of at most n deliveries. Constraints (22) make sure that the departing time of delivery schedule q that contains job j is no earlier than the completion time for job j on a certain machine. Constraints (23) and (24) are the binary requirements for the decision variables.

The column generation approach needs to relax the integer constraint when solving the master problem; this research assume that this linear relaxation of the set partitioning master problem, denoted as [LSP], makes π_j, σ, λ_j, β, and δ_j the dual variables for the constraints (18) to (22). Due to the relationship of the dual property when all reduced costs are greater than or equal to 0, the optimal solution is obtained. The sub-problem is divided into two types in this research. One sub-problem is associated with the set of variables x, which regards the machine scheduling problems; this research is called the single machine scheduling sub-problem. The other one is related to the set of variables y, which deals with the job delivering problems; this research is called the single batch delivery sub-problem. The master problem is solved to connect the single machine scheduling sub-problem and the single batch delivery sub-problem, generating columns to improve the objective value to obtain the smallest total cost.

3.2 *Single machine scheduling subproblem*

Let R_s indicate that the decision variable x corresponds to the reduced cost. Then, the reduced cost R_s is as follows:

$$R_s = -\sum_{j\in N} \pi_j a_{js} - \sigma - \sum_{j\in N} c_{js}\delta_j a_{js} \tag{25}$$

The decision variables are a_{js} and c_{js} in this sub-problem. In Equation (25), π_j $(j \in N)$, σ, and δ_j $(j \in N)$ are the known constants. c_{js} represents the completion time of job $j \in N$ in partial schedule $s \in S$. Therefore, this sub-problem can be seen as the minimum weighted completion time of the single machine scheduling problem; its objective is to find out a set of jobs beginning from time 0 sequence, processing to make the minimum R_s.

In order to fit the branch strategy of the branch and bound algorithm, this research added a job processing precedence constraint in the sub-problem. Let E_j be the set of job processings before job j. Use the branch and bound algorithm to solve the root node. E_j will cause the branching sub-node to divide into two kinds – one kind is jobs that were contained in set E_j, and another kind is jobs that were not contained in set E_j. E_j is updated to follow the branch behaviour in the solution process.

This research follows Chen and Powell (1999), who proposed a dynamic programming method to solve the single machine scheduling sub-problem (denoted as DP1). Let $F(j,t)$ denote job j, finished at time t in the partial schedule, to reduce cost to a minimum, and E_j is the set of job processings before job j; the recursive relation is as follows:
DP1:
Initial state:

$$F(j,t) = \begin{cases} -\sigma & \text{if } j = 0 \text{ and } t = 0 \\ \infty & \text{otherwise} \end{cases} \tag{26}$$

Recursive relation:

$$\begin{gathered} F(j,t) = \min_{i \in E_j}\{F(i,t) - \delta_j t - \pi_j\} \\ t = 0, \cdots, d_{\max}\ ,\ j \in N \end{gathered} \tag{27}$$

Optimal solution:

$$F^* = \min\{F(j,t) | j \in N,\ t = 0, \cdots, d_{\max}\} \tag{28}$$

From Equation (26), we can see that the reduced cost is a known constant that has not arranged any job of a single machine scheduling subproblem in the initial state time 0. In order to avoid arranging a job before time 0, let the reduced cost be infinity to avoid this situation happening. In Equation (27) $F(i,t)$ is the reduced cost of the partial scheduling before job j, and $= -\delta_j t - \pi_j$ represents the reduced cost contribution value after completion of processing job i to arrange job j. Find a processing job with the minimum reduced cost to satisfy Equation (28), representing a single machine scheduling sub-problem that has already obtained the optimal solution.

3.3 *Single batch delivery subproblem*

Let R_q indicate that the decision variable y corresponds to the reduced cost. Then, the reduced cost R_q is as follows:

$$\begin{aligned} R_q &= \alpha \sum_{j \in N} w_j C_{jq} b_{jq} + (1-\alpha)(F_{fix} + F_q) - \sum_{j \in N} \lambda_j b_{jq} - \beta + \sum_{j \in N} \delta_j d_q b_{jq} \\ &= \sum_{j \in N} (d_q \delta_j - \lambda_j) b_{jq} + \alpha \sum_{j \in N} w_j C_{jq} b_{jq} + (1-\alpha)(F_{fix} + F_q) - \beta \end{aligned} \tag{29}$$

In Equation (29), $\delta_j (j \in N)$, $\lambda_j (j \in N)$, F_{fix}, and β are the known constants. The decision variables are b_{jq}, d_q, C_{jq}, and F_q in this sub-problem represents the delivery schedule q that contains job j, the vehicle departure time of delivery schedule q, the time of job j when it is delivered to its customer under delivery schedule q, and the variable cost of delivery schedule q, respectively.

This research lists all the possible combinations of the delivery schedule. Let the quantity of delivery schedule at most be $q_{max} \cdot \Phi_{qj}$ is the set of jobs that are ordered by path q, and j is the customer order of the path q. If we know the vehicle departure time for job j containing the path q, then will know how much the reduced costs contribute value. Let $F_{d,q}(j,\ c)$ be the selection path q, vehicle departure time be d, and job j be the last job included in the batch, and the vehicle capacity of all jobs for reduced costs is c in this batch. Let τ_{qj} be the transportation time from the factory to the customer for job j of path q. F_q is the transportation cost for selected path q and selects the smallest processing time of job on the machine, p_{min}. In order to fit the branch strategy of the branch and bound algorithm, this research added a job delivery processing precedence constraint in the single batch delivery sub-problem. Let D_j be the set of delivery jobs before job j. In the process, use the branch and bound algorithm to solve the node; its sub-node is divided into two types – the jobs included in this set D_j and the jobs not included in this set D_j. The updated D_j follows branch behaviour in the process of the branch and bound algorithm. This research proposes using dynamic programming to solve the single batch delivery sub-problem; the recursive relationship is as follows:

DP2:
Initial state:

$$F_{d,p}(j,c) = \begin{cases} (1-\alpha)(F_{fix} + F_q) - \beta & \text{if } j = 0;\ c = 0 \\ \infty & \text{otherwise} \end{cases} \tag{30}$$

$$For\ q = 1, \cdots, q_{max};\ d = p_{\min}, p_{\min} + 1, \cdots, d_{\max}$$

Recursive relation:

$$F_{d,q}(j,c) = \min_{i \in \{D_j \cup \{0\}\} \cap \Phi_{qj}} \{F_{d,q}(i, c-1) - \lambda_j + d\delta_j + \alpha w_j(\tau_{qj} + d)\}$$
$$for\ j \in \{N \cap \Phi_{qj}\} \cup \{n+1\};\ d = p_{\min}, p_{\min} + 1, \cdots, d_{\max};\ c = 0,\ 1 \cdots,\ z \tag{31}$$

Optimal solution:

$$F^* = \min\{F_{d,\,q}(n+1, c)\}$$
$$for\ \ d = p_{\min}, p_{\min} + 1, \cdots,\ d_{\max}; c = 0, 1, \cdots, z; q = 1, 2, \cdots, q_{\max} \tag{32}$$

From Equation (30), we can see that the reduced cost is a known constant that has not delivered any job in the single batch delivery sub-problem in the initial state. In order to avoid a job delivery before a job process has finished, let the reduced cost be infinity to avoid this situation happening. In Equation (31), $F_{d,q}(i,\ c-1)$ is the reduced cost of the batch before job j, and $-\lambda_j + d\delta_j + \alpha w_j(\tau_{qj} + d)$ represents the reduced cost contribution value after add job j. $\alpha w_j(\tau_{qj} + d)$ is the wait time for it to be delivered to customers. Find a delivery path with the minimum reduced cost to satisfy Equation (32), representing the single batch delivery sub-problem that has already obtained the optimal solution.

3.4 *Procedure of the proposed approach*

This paper applies a column generation-based algorithm combined with a branch and bound algorithm to solve supply chain scheduling problems (integrated production and distribution scheduling). The procedure of the proposed approach is organised into five steps and explained as follows:

Step 1. Generate a restricted master problem.

Use the big M method to add artificial columns to generate a restricted master problem.

Step 2. Calculate the reduced cost and whether conditions of the optimal solution are satisfied.

After obtaining the dual variables from Step 1, calculate the reduced cost of decision variables to satisfy the condition of the optimal solution. If any reduced cost is less than 0, then enter Step 3, and solve the single machine sub-problem and the single batch delivery sub-problem; otherwise, go to Step 4.

Step 3. Solve the single machine sub-problem and the single batch delivery sub-problem.

Using the dual variable of Step 1, generate the single machine scheduling sub-problem and the single batch delivery sub-problem, respectively. Then, use dynamic programming to solve the sub-problems. Use the updated decision variables to obtain results to solve the single machine scheduling and the single batch delivery sub-problem, and add them to the restricted master problem. Return to Step 1 to solve the updated restricted master problem.

Step 4. Determine whether to apply the branch and bound algorithm.

If the decision variables are not integers, then use the branch and bound algorithm to approach the integer solution; go to Step 5. If the results are not needed to branch and represent the current solution already obtained in the best integer solution, end the algorithm.

Step 5. Selected branch node and start branch.

Firstly, make the variable transformation of decision variables x_s ($s \in S$). Let $\rho_{hj} = \sum_{s \in S} e^s_{hj} x_s$, and $e^s_{hj} = 1$ represents the situation where job h is processed immediately before job j in partial schedule s; otherwise $e^s_{hj} = 0$. Decision variables y_q ($q \in Q$) can also be transformed in a similar method. Let $\theta_{hi} = \sum_{q \in Q} o^q_{hj} y_q$, and $o^q_{hj} = 1$ represents the situation where job j is delivered immediately after job h in delivery schedule q; otherwise $o^q_{hj} = 0$. After this transformation, instead of directly branch on variables x_s and y_q, the new branching variables are ρ_{hj} and θ_{hj}. If some of the ρ_{hj} or θ_{hj} are not integers, two branching rules, called 0.99 and 0.5 rules, can be applied. The 0.99 rule calls to select a branching

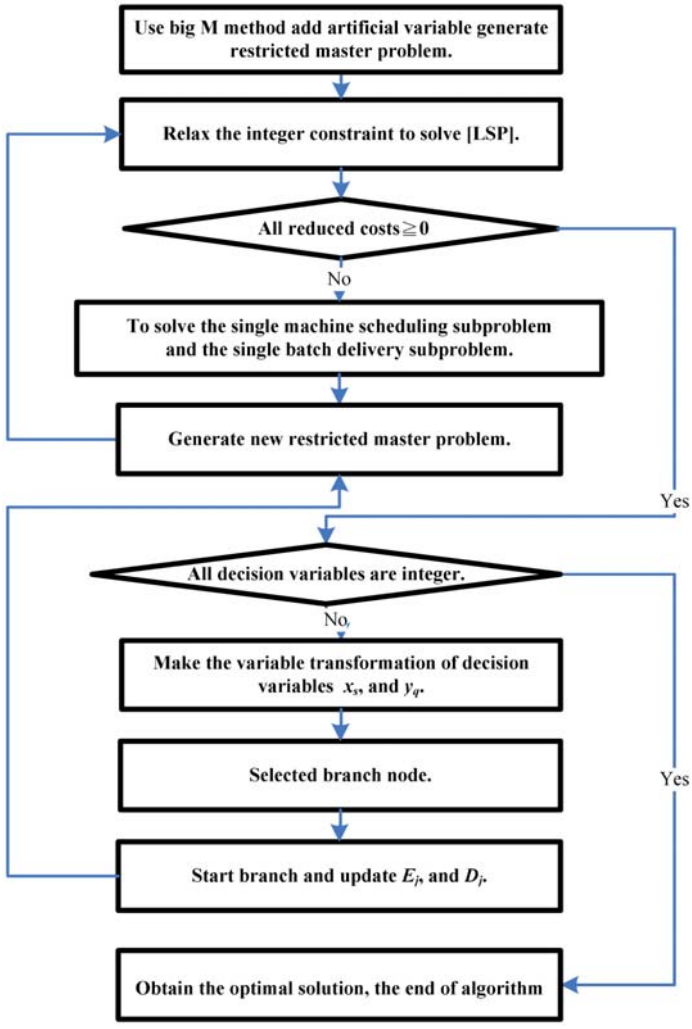

Figure 1. The flowchart of the proposed algorithm.

variable that is the most close to 0.99; the 0.5 rule calls to select a branching variable that is the most close to 0.5. Once a branching variable is selected, two sub-nodes are created to make the branching variable either 1 or 0. If $\rho_{hj} = 1$ is set to be the sub-node in one branch, then job h has to be processed immediately before job j in all partial schedules generated in this sub-node. Likewise, if $\theta_{hj} = 0$ is set to be the sub-node of one branch, then job h cannot be delivered immediately before job j in all delivery schedules generated in this sub-node. Each sub-node is actually one [LSP] problem. If the lower bound value is less than the current upper bound, then the lower bound value is set to the new upper bound and go to Step 1; otherwise, prune this sub-node.

Figure 1 shows the flowchart of the proposed algorithm.

4. Computational experiments

This research used C language and ILOG CPLEX 10.2 to compose the computer program, based on a design algorithm of Section 3. The test hardware environment is an Intel Pentium D 3.00Ghz CPU, 3.25G RAM.

4.1 *Add a column test to the restricted master problem*

In the early period, use of the column generation to solve scheduling problems usually selects the least-reduced cost column to add to the restricted master problem. Instead of adding one column at a time, some scholars propose to add several columns to update the restricted master problem in the recent literature. For example, Chen and Powell (1999) indicate that adding 5 to 10 columns is more efficient than adding 20 columns. This is because by adding 20 columns,

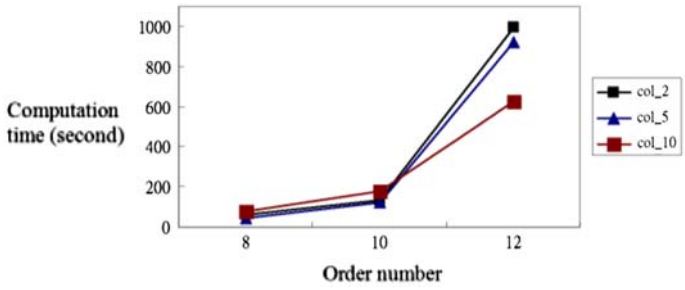

Figure 2. Add different numbers of columns of the computation time difference chart.

the scale of the linear programming problem increases too quickly, causing the solution time to increase. In addition, these 20 columns also possibly contain many insignificant columns for improving the objective value.

The number of machines (m) was 4; the number of jobs (n) was 8, 10, and 12; and preference coefficient $\alpha = 0.5$ for the test criteria. In the machine scheduling stage, all job processing times were drawn from the uniform distribution $U[1,5]$, and job weights were drawn from the uniform distribution $U[1,10]$. In the distribution stage, the fixed cost is 10 by sending out one vehicle and the upper limit of the vehicle capacity is 3 – customer distance and the delivery times corresponding to the uniform distribution $U[1,10]$. It has three kinds of setting values – 2, 5, and 10 – in the part of the add column quantity. Every kind of combination has 10 random test problems by computer in the three kinds of combinations: $n = 8, 10$, and 12. This research used col_2, col_5, and col_10 to add at most 2, 5, and 10 columns to the restricted master problem. The result of adding different numbers of columns is shown in Figure 2.

Because the add-column quantity will affect the algorithm efficiency, this research used the computation time to become the evaluating objective. Observe Figure 2; the col_5 computation time is the shortest when the order numbers are 8 and 10, with the average difference of col_5 and col_10 within 60 s. However, col_10 had significant a difference with two other kinds of settings when the order numbers increased to 12. In $n = 12$, the col_5 average computation time was more than about 300 s than col_10. Moreover, the average difference of col_2 and col_10 was 400 s of computation time. Therefore, this research suggests using column generation techniques to solve supply chain scheduling (integrated production and distribution scheduling) problems, and it can add 2 to 5 columns to the restricted master problem when the order number is less than 10; it can attempt to add at most 10 columns to increase the algorithm efficiency when the order number is greater than 10.

4.2 *Rule test to selected branch node*

Let the computer following the uniform distribution randomly generate 10 test problems, respectively, under the different numbers of machines and orders. The result of the average computation time is shown in Figure 3.

From Figure 3, we can see that the computation time of the 0.99 branching rule is less than 0.5 in three situations: $n = 8$, 10, and 12. Moreover, the computation time difference increases along with order number. Two kinds of rules of the average computation time difference are less than 20 s when the order numbers are 8. But, the average computation time difference increases to close to 200 s when the order numbers are 12.

In order to examine whether the computation time of using the 0.99 branching rule is significantly less than the one using 0.5 branching rule, we performed an independent sample t test in these 120 test problems; the result is shown in Table 1.

From Table 1, we can see that the p value is 0.027358 which is less than 0.05, representing that the population mean has a significant difference of these two rules. Therefore, the computation time of the 0.99 branching rule will be

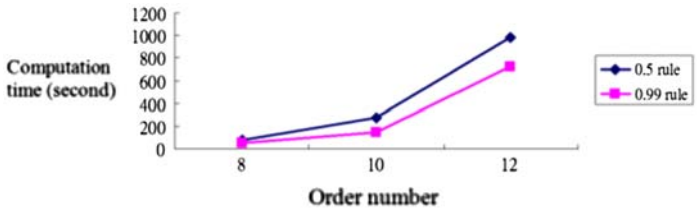

Figure 3. Different branching rules of the computation time difference chart.

Table 1. *T* test for the mean population difference of two branching rules.

		T-tests; Grouping: rule (Spreadsheet1) Group 1: 099 rule Group 2: 0.5 rule			
Variable	Mean 1	Mean 2	*t*-value	df	*p*
tTime	306.2333	448.9400	−2.23413	118	0.027358

obviously less than the 0.5 branching rule when selecting the branch node. Based on this test result, this research used the closest ρ_{hj} and θ_{hj} to 0.99 to become the branching node in the test sample problem.

4.3 *Design of test problems*

This subsection will use a computer simulation to generate test problems and apply column generation to solve supply chain scheduling problems (integrated production and distribution scheduling). Due to this, the research used the exact solution method to solve the NP-Hard problem. It must consider the computation time and the computer's memory limit. Therefore, we will not design more complex test problems.

This research used test numbers of machines (m) of 2, 4, and 6. When $m = 2$, it must consider three situations of numbers of jobs (n): 8, 10, and 12; when $m = 4$, it must consider three situations of numbers of jobs (n): 8, 10, and 12; when $m = 6$, it must consider three situations of numbers of jobs (n): 10, 12, and 14. The setting values of the numbers of customers (k) are 3, 4, and 5. In addition, it also tests different preference coefficients (α), 0.2, 0.5, and 0.8, in the objective function. The number of machines, number of jobs, number of customers, and the preference coefficients mentioned above have 81 kinds of combinations; the total has 405 test problems.

Table 2. Computational results for three customers.

$k = 3$			LP_IP gap		Columns		Nodes		CPU time	
α	m	n	Average	Maximum	Average	Maximum	Average	Maximum	Average	Maximum
0.2	2	6	1.30%	2.50%	754	863	48	60	4.908	14.14
		8	4.80%	8.50%	30,147	33,941	2547	3156	55.938	209.97
		10	0.90%	1.90%	59,321	69,421	7768	9768	993.627	1732.91
	4	8	0.70%	1.10%	1708	2178	178	232	9.18	16.03
		10	0.70%	1.40%	22,959	39,734	1963	3685	153.53	462.03
		12	4.00%	7.60%	72,837	79,446	4705	5857	601.08	1289.30
	6	10	0.90%	1.70%	4598	7826	612	1829	15.65	32.97
		12	0.50%	1.10%	19,598	29,826	1392	2329	168.81	476.38
		14	3.40%	6.90%	54,470	99,624	2041	3732	998.77	1547.11
0.5	2	6	3.80%	4.20%	1144	1768	87	156	3.51	12.38
		8	1.60%	3.00%	10,798	14,404	977	1293	92.4	160.02
		10	0.30%	0.50%	73,514	89,294	2236	3442	1273.75	1333.52
	4	8	1.70%	3.30%	3642	5622	372	633	13.14	49.38
		10	0.10%	0.20%	26,981	30,876	1750	1933	133.40	476.89
		12	1.90%	3.50%	36,642	50,415	1800	3456	407.09	1211.98
	6	10	0.50%	1.20%	7031	8024	1866	3420	13.54	61.59
		12	0.30%	0.50%	23,531	39,024	1866	3420	50.37	113.94
		14	2.00%	2.10%	64,730	108,188	2460	4099	710.95	1708.81
0.8	2	6	0.40%	0.70%	851	1249	64	115	1.04	4.08
		8	0.30%	0.50%	32,443	43,193	5398	7765	16.03	33.56
		10	0.80%	0.90%	69,401	126,841	3197	6004	496.05	632.74
	4	8	1.70%	2.70%	3631	5277	353	427	4.07	7.38
		10	0.10%	0.30%	12,095	20,065	652	1133	33.98	178.80
		12	4.80%	9.40%	75,224	144,905	4683	7102	987.93	1514.84
	6	10	0.20%	0.50%	3962	5968	1011	2245	13.54	18.45
		12	0.40%	0.70%	19,662	24,968	1400	1945	47.72	84.58
		14	0.30%	0.60%	185,560	262,545	29,345	35,358	1480.69	2550.47

Note: The total average of the LP_IP gap is 1.42%.

4.4 *Test results and analysis*

The test problems use computer simulation to test different numbers of customers; the results are listed in Tables 2 to 4. The column 'LP_IP gap' represents the average gap of the optimal solution between [LSP] and [SP], the [LSP] optimal solution is Z_{LP}, and the [SP] optimal solution is Z_{IP}. This formula is shown as below:

$$\left|\frac{Z_{LP} - Z_{IP}}{Z_{LP}} \times 100\%\right|$$

In these tables, 'Columns' represents the average number of columns generated for solving the test problem; the column 'Nodes' represents the total number of branch and bound nodes explored for solving the problem; the column 'CPU' time represents the spent computation time (in seconds) to obtain the optimal solution. 'Average' represents the mean value of the five test problems; 'Maximum' represents the maximum output value of the five test problems. In Table 4, the ' – ' symbol' represents the test problems already achieving the upper limit of computer memory resources before obtaining the optimal solution; then, stop the programme.

From Tables 2 to 4, we can see that the average of the LP_IP gap is 1.42%, 1.13%, and 1.53% in the numbers of customers 3, 4, and 5. The average of the LP_IP gap is 1.32% of all test problems. This result shows that the proposed method can provide a lower bound that is very close to the optimal solution. Therefore, we must consider the computation time and the limit of computer memory resources in real-world problems, even if we have not obtained the optimal solution but still can provide the lower bound of the solution.

This research confers the influence of machine number, order number, customer number, and preference coefficient to LP_IP gap and the CPU time. Moreover, it performs multiple regression analysis in the test result. In $k = 5$, the limit on computer memory resources does not have the best integer solution data when the order number of test problems is greater than 10. Therefore, delete these 70 problems to perform multiple regression analysis in the 335 test problems. The results of the multiple regression analysis are shown in Table 5 and Table 6.

Table 3. Computational results for four customers.

$k = 4$			LP_IP gap		Columns		Nodes		CPU time	
α	m	n	Average	Maximum	Average	Maximum	Average	Maximum	Average	Maximum
0.2	2	6	2.30%	4.80%	1635	5331	167	575	1.895	2.31
		8	1.50%	4.80%	19,843	66,387	1428	4881	382.315	421.08
		10	1.40%	3.90%	131,031	218,267	6288	18,753	859.165	1363.39
	4	8	2.10%	6.30%	2747	5456	282	621	7.26	9.73
		10	1.40%	5.00%	42,298	112,834	3911	10,260	324.00	623.44
		12	0.60%	1.90%	135,658	288,354	6533	17,392	1968.20	2468.45
	6	10	0.50%	2.80%	6955	14,550	460	1027	14.44	29.09
		12	0.60%	2.70%	45,354	100,711	3875	10,056	198.33	357.01
		14	1.60%	4.20%	179,420	345,121	11,285	23,390	1561.38	2575.72
0.5	2	6	1.60%	5.20%	1550	2922	173	475	3.51	6.00
		8	0.90%	2.80%	17,502	54,562	1144	3897	65.40	295.00
		10	1.70%	4.50%	97,619	131,654	3771	12,190	1273.75	1536.33
	4	8	1.00%	2.90%	4038	16,447	444	2061	18.71	32.16
		10	1.20%	3.30%	68,685	121,025	3478	11,461	302.19	327.63
		12	0.70%	1.90%	107,544	209,296	6189	16,837	1090.63	1868.92
	6	10	0.80%	4.20%	6038	22,313	443	1989	49.07	64.67
		12	0.90%	3.00%	20,968	41,888	1502	3431	345.82	658.19
		14	0.10%	0.30%	156,087	319,532	8487	18,661	1335.43	2515.58
0.8	2	6	1.50%	3.80%	756	1679	54	129	3.17	5.51
		8	1.40%	2.80%	5237	13,701	318	993	101.43	159.55
		10	1.30%	4.90%	89,855	174,929	4379	23,876	1657.35	2943.22
	4	8	0.90%	2.60%	1365	2216	103	203	23.65	35.77
		10	0.70%	2.10%	13,582	57,485	1115	5502	113.69	211.72
		12	1.10%	3.70%	127,229	252,932	4990	15,899	652.61	1279.55
	6	10	0.80%	2.10%	6038	7454	443	439	11.34	40.39
		12	1.50%	6.90%	15,293	24,758	1132	1917	210.24	298.59
		14	0.50%	1.30%	115,947	208,078	16,140	32,175	1332.35	2507.86

Note: The total average of the LP_IP gap is 1.13%.

Table 4. Computational results for five customers.

$k = 4$			LP_IP gap		Columns		Nodes		CPU time	
α	m	n	Average	Maximum	Average	Maximum	Average	Maximum	Average	Maximum
0.2	2	6	2.50%	3.20%	1897	2519	133	199	19.79	32.08
		8	1.40%	1.90%	38,685	51,025	6478	9461	233.40	376.89
		10	—	—	—	—	—	—	—	—
	4	8	5.50%	6.20%	3787	4650	223	338	48.88	79.25
		10	—	—	—	—	—	—	—	—
	6	10	2.90%	5.80%	12,601	20,459	658	1061	392.36	704.91
		12	—	—	—	—	—	—	—	—
0.5	2	6	0.60%	0.60%	2073	2903	155	266	24.88	43.59
		8	1.90%	3.60%	10,052	12,682	602	758	394.01	563.78
		10	—	—	—	—	—	—	—	—
	4	8	0.10%	0.20%	3390	3399	360	390	82.70	97.55
		10	—	—	—	—	—	—	—	—
	6	10	1.00%	2.00%	15,922	17,717	687	967	491.51	626.09
		12	—	—	—	—	—	—	—	—
0.8	2	6	1.40%	1.90%	1232	1233	90	114	15.54	22.69
		8	0.50%	1.00%	3716	4918	187	273	89.63	126.05
		10	—	—	—	—	—	—	—	—
	4	8	0.20%	0.40%	3647	4760	249	386	58.88	69.25
		10	—	—	—	—	—	—	—	—
	6	10	0.30%	0.60%	9460	11,713	602	737	404.28	574.94
		12	—	—	—	—	—	—	—	—

Note: The total average of the LP_IP gap is 1.53%.

Table 5. Multiple regression analysis for different factor influence LP_IP gap.

	Regression summary for dependent variable: LP_IP gap $R = 0.36,361,715$; $R_2 = 0.13,221,743$; adjusted $R_2 = 0.076,623,146$ $F(4330) = 12.3616$; $p < 0.06,283$; Standard error of estimate: 01,129					
$N = 335$	Beta	Set. error of beta	B	Standard Error of B	$t(330)$	p level
Intercept			0.017722	0.011245	1.57600	0.120115
Coefficient	−0.300832	0.118436	−0.014265	0.005616	−2.54003	0.013604
Machines	−0.189279	0.166034	−0.001335	0.005616	−1.14000	0.258670
Orders	−0.041721	0.170470	−0.000212	0.000868	0.24474	0.807465
Customers	0.094243	0.122862	0.001477	0.001477	0.76707	0.445953

From Table 5, we can see that the machine number, order number, and customer number do not have significant influence; only the preference coefficient to LP_IP gap has the significant influence (p-value = 0.013 < 0.05). Based on the results of Tables 2 to 4, this research combines data at the same preference coefficient to obtain the mean value; the LP_IP gap is 1.89%, 1.12%, and 0.96% in the preference coefficients 0.2, 0.5, and 0.8, respectively. It can discover the LP_IP gap's tendency of gradually decreasing following the increase in preference coefficient. That is because the smaller preference coefficient represents more emphasis, considering the influence of total distribution cost. This research used the time relationship to construct the mathematical model, linking two stages of production and distribution. When the objective function emphatically considers the total distribution cost, it will cause the [LSP] solution and the best integer solution of [SP] to have a bigger error. Otherwise, the bigger preference coefficient represents more emphasis on the job completion times. This used the time relationship to link two stages of production and distribution to gain a bigger benefit, causing the LP_IP gap to be small.

From the analysis results in Table 6, we can see that the p-value is less than 0.05 for the machine number, order number, and customer number. It can explain that these three factors all have a significant influence regarding the CPU

Table 6. Multiple regression analysis for different factor influence CPU time.

	Regression summary for dependent variable: CPU time R = 0.7,719,465; R_2 = 0.5,958,699; adjusted R_2 = 0.56,982,583 F(4330) = 121.66; p < 0.00,000; Standard error of estimate: 303.54					
N = 335	Beta	Set. error of beta	B	Standard error of B	t(330)	p level
Intercept			−1687.30	302.3902	−5.57989	0.000001
Coefficient	0.027976	0.080821	52.28	151.0207	0.34615	0.730402
Machines	−0.589247	0.113302	−163.75	31.4865	−5.20068	0.000002
Orders	1.091019	0.116329	218.82	23.3320	9.37874	0.000000
Customers	0.220079	0.083841	135.90	51.7707	2.62495	0.010899

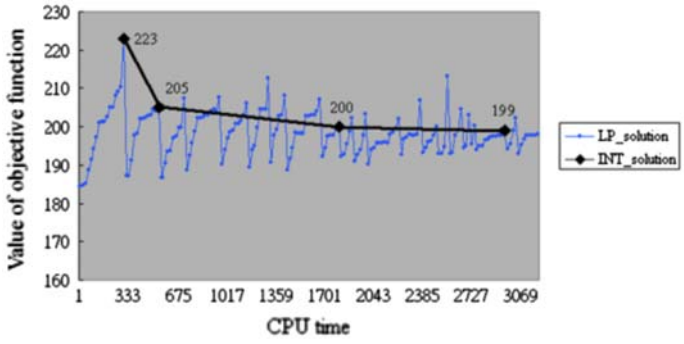

Figure 4. The relationship graph of the objective function value's improvement range and the CPU time.

time. The reason is that when the factor's quantity increases the order number or customer number, the column combination will also be increasing in the [LSP]. It is easy to generate the phenomenon of degeneracy at this time and must spend a long time to solve the linear programming problem. This research used the example of a test problem ($\alpha = 0.5, m = 4, n = 10,$ and $k = 5$) that did not obtain the best integer solution, explaining the phenomenon of degeneracy to influence the value of the objective function in the solution process; the result is shown in Figure 4.

In Figure 4, the LP_solution is the optimal solution of the [LSP] after the restricted master problem is updated every time, and the INT_solution is the integer solution that satisfies the integer constraints. From Figure 4, we can see that the objective function value improvement range is more obvious in the initial period. From the start of CPU time of about 500 s, the LP_solution often appears as the phenomenon of degeneracy to cause the objective function value to circulate between 190 and 210. Observe the INT_solution; we can see that the second-time integer solution improved the first-time by about 8.7%, and obtaining the improved integer solution only needed about 240 s of CPU time. Then, the improved objective function value range gradually flattens. The improved ranged remained 0.5% of the last two integer solutions before the program terminated and spent about 1300 s of CPU time. Because the solution generated too much degeneracy, this kind of improved phenomenon of the range of the objective function value is very small, yet the computation time increased.

5. Conclusions

Due to the diversity and rapidly changing customer demand as well as the competitive market environment's change, the product life cycle will shorten. On the other hand, to pursue customised products and simultaneously high lead time, enterprises are faced with such consumer markets that gradually change to 'make-to-order' and 'direct sales' business model. In such a business model, the enterprise pursues high customer service levels simultaneously but cannot take into account the objective of cost minimisation. In practice, it has the trade-off relationship between cost and the customer service level. The purpose of this research is to find out the optimal balance point between these two aspects. This class of problems can be viewed as two-stage supply chain scheduling problems (integrated production and distribution scheduling problem). In the objective function, in addition to considering the time when the products are delivered to

customer, to measure the performance of customer service level, the cost of product delivery is added to customers. Obtain the optimal solution for the objective function definition – namely, find out the result of system optimisation.

The proposed algorithm is an applied column generation-based approach, combining dynamic programming and the branch and bound technique to solve two-stage supply chain scheduling problems. The advantage of the proposed approach can provide a lower bound that is very close to the optimal solution. The results of the computational experiments indicate that the proposed approach can solve the test problems to optimality. Moreover, the average gap between the optimal solutions and the lower bounds arise no more than 1.32% for these test problems. It showed that this algorithm can provide a good-quality lower bound. In addition, this research also added the column quantity of the restricted master problem to find a suitable setting value. The test results showed that when the order number is less than 10, it suggests adding at most five columns each time; when the order number is bigger than 10, add at most 10 columns to the restricted master problem.

There are still some directions remaining to be studied. Further research can be conducted to explore more practical approaches in solving this class of problem. For example, metaheuristics, such as tabu search, genetic algorithms, particle swarm optimisation, and ant colony optimisation, can be applied to find near optimal solutions to this class of problem. Moreover, it is also important to discuss such problems under different cost structures such as including the due-date related performance measures.

Acknowledgements

This research was supported in part by the National Science Council of Taiwan under Grants NSC 96-2221-E009-086, NSC 98-2410-H-009-005, NSC 101-2410-H-145-001, and NSC 101-2410-H-009-005-MY2.

References

Bard, J. F., and S. Rojanasoonthon. 2006. "A Branch-and-Price Algorithm for Parallel Machine Scheduling With Time Windows and Job Priorities." *Naval Research Logistics* 53 (1): 24–44.

Barnhart, C., E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. 1998. "Branch-and-Price: Column Generation for Solving Huge Integer Programs." *Operations Research* 46 (3): 316–329.

Bodin, L. D. 1990. "Twenty Years of Routing and Scheduling." *Operations Research* 38 (4): 571–579.

Bramel, J., and D. Simchi-Levi. 1997. "On the Effectiveness of Set Covering Formulations For the Vehicle Routing Problem with Time Windows." *Operations Research* 45 (2): 295–301.

Chang, Y. C., and C. Y. Lee. 2004. "Machine Scheduling with Job Delivery Coordination." *European Journal of Operational Research* 158 (2): 470–487.

Chen, Z. L. 2004. "Simultaneous Job Scheduling and Resource Allocation on Parallel Machines." *Annals of Operations Research* 129 (1–4): 135–153.

Chen, Z. L., and W. B. Powell. 1999. "Solving Parallel Machine Scheduling Problems by Column Generation." *INFORMS Journal on Computing* 11 (1): 78–94.

Chen, Z. L., and G. L. Vairaktarakis. 2005. "Integrated Scheduling of Production and Distribution Operations." *Management Science* 51 (4): 614–628.

Dantzig, G. B., and P. Wolfe. 1960. "Decomposition Principle for Linear Programming." *Operations Research* 8 (1): 101–111.

Desaulniers, G. 2007. "Managing Large Fixed Costs in Vehicle Routing and Crew Scheduling Problems Solved by Column Generation." *Computers & Operations Research* 34 (4): 1221–1239.

Desrochers, M., J. Desrosiers, and M. Solomon. 1992. "A New Optimization Algorithm For the Vehicle Routing Problem with Time Windows." *Operations Research* 40 (2): 342–354.

Fisher, M. L., K. O. Jornsten, and O. B. G. Madsen. 1997. "Vehicle Routing with Time Windows: Two Optimization Algorithms." *Operations Research* 45 (3): 488–492.

Ganesharajah, T., N. G. Hall, and C. Sriskandarajah. 1998. "Design and Operational Issues in AGV-Served Manufacturing Systems." *Annals of Operations Research* 76: 109–154.

Hall, N. G., and C. N. Potts. 2003. "Supply Chain Scheduling: Batching and Delivery." *Operations Research* 51 (4): 566–584.

Hall, L. A., and D. B. Shmoys. 1992. "Jackson's Rule for Single-Machine Scheduling: Making a Good Heuristic Better." *Mathematics of Operations Research* 17 (1): 22–35.

Hurink, J., and S. Knust. 2001. "Makespan Minimization for Flow-Shop Problems with Transportation Times and a Single Robot." *Discrete Applied Mathematics* 112 (1–3): 199–216.

Kise, H. 1991. "On an Automated Two-Machine Flowshop Scheduling Problem with Infinite Buffer." *Journal of the Operations Research Society of Japan* 34 (3): 354–361.

Laporte, G. 1992. "The Vehicle Routing Problem: An Overview of Exact and Approximate Algorithms." *European Journal of Operational Research* 59 (3): 345–358.

Lee, C. Y., and Z. L. Chen. 2001. "Machine Scheduling with Transportation Considerations." *Journal of Scheduling* 4 (1): 3–24.

Li, C. L., G. Vairaktarakis, and C. Y. Lee. 2005. "Machine Scheduling with Deliveries to Multiple Customer Locations." *European Journal of Operational Research* 164 (1): 39–51.

Lübbecke, M. E., and J. Desrosiers. 2005. "Selected Topics in Column Generation." *Operations Research* 53 (6): 1007–1023.

Maggu, P. L., and G. Das. 1980. "On 2×n Sequencing Problem with Transportation Times of Jobs." *Pure and Applied Mathematika Science* 12: 1–6.

Min, H., V. Jayaraman, and R. Srivastava. 1998. "Combined Location-Routing Problems: A Synthesis and Future Research Directions." *European Journal of Operational Research* 108 (1): 1–15.

Pundoor, G., and Z. L. Chen. 2005. "Scheduling a Production-Distribution System to Optimize the Tradeoff Between Delivery Tardiness and Distribution Cost." *Naval Research Logistics* 52 (6): 571–589.

Sarmiento, A. M., and R. Nagi. 1999. "A Review of Integrated Analysis of Production-Distribution Systems." *IIE Transactions* 31 (11): 1061–1074.

Savelsbergh, M., and M. Sol. 1998. "DRIVE: Dynamic Routing of Independent Vehicles." *Operations Research* 46 (4): 474–490.

Shao, X. F., and M. Dong. 2012. "Comparison of Order-Fulfilment Performance in MTO And MTS Systems with an Inventory Cost Budget Constraint." *International Journal of Production Research* 50 (7): 1917–1931.

Stevens, J. W., and D. D. Gemmill. 1997. "Scheduling a Two-Machine Flowshop with Travel Times to Minimize Maximum Lateness." *International Journal of Production Research* 35 (1): 1–15.

Stowers, C. L., and U. S. Palekar. 1993. "Location Models with Routing Considerations for a Single Obnoxious Facility." *Transportation Science* 27 (4): 350–362.

Su, J. C. P., Y. L. Chang, M. Ferguson, and J. C. Ho. 2010. "The Impact of Delayed Differentiation in Make-to-Order Environments." *International Journal of Production Research* 48 (19): 5809–5829.

Van den Akker, J. M., J. A. Hoogeveen, and S. L. Van de Velde. 1999. "Parallel Machine Scheduling by Column Generation." *Operations Research* 47 (6): 862–872.

Wang, G., and T. C. E. Cheng. 2000. "Parallel Machine Scheduling with Batch Delivery Costs." *International Journal of Production Economics* 68 (2): 177–183.