Integrated process planning and scheduling in a supply chain

Chiung Moon a, Young Hae Lee b,*, Chan Seok Jeong c, YoungSu Yun d

a Department of Information and Industrial Engineering, Yonsei University, Seoul, Republic of Korea
b Department of Information and Industrial Engineering, Hanyang University, Ansan, Gyeonggi-do, Republic of Korea
c Korea Army, Seoul, Republic of Korea
d Department of Business Administration, Chosun University, Gwangju, Republic of Korea

Available online 16 June 2007

Abstract

This paper deals with the integration of process planning and scheduling, which is one of the most important functions in a supply chain to achieve high quality products at lower cost, lower inventory, and high level of performance. Solving the problem is essential for the generation of flexible process sequences with resource selection and for the decision of the operation schedules that can minimize makespan. We formulate a mixed integer programming model to solve this problem of integration. This model considers alternative resources: sequences and precedence constraints. To solve the model, we develop a new evolutionary search approach based on a topological sort. We use the topological sort to generate a set of feasible sequences in the model within a reasonable computing time. Since precedence constraints between operations are handled by the topological sort, the developed evolutionary search approach produces only feasible solutions. The experimental results using various sizes of problems provide a way to demonstrate the efficiency of the developed evolutionary search approach.

© 2007 Elsevier Ltd. All rights reserved.

Keywords: Process planning; Scheduling; Evolutionary search approach; Supply chain; Topological sort

1. Introduction

Supply chain is an integrated business process where various business entities work together in an effort to (a) acquire raw materials, (b) convert these raw materials into specified final products, and (c) deliver these final products to retailers (Beamon, 1998). A global chain covers multiple manufacturing sites and consists of suppliers, fabrication and assembly shops, as well as outsourcing entities. These supply chains’ planning and scheduling activities are very complex and have to take place within the enterprise and across the entire supply chain in order to achieve high quality products at lower cost, lower inventory, and high levels of performance. As a result, to efficiently provide global optimal solutions, enterprises are migrating from separated planning processes toward more coordinated and integrated planning processes.

This paper was part of The 27th International Conference on Computers & Industrial Engineering, held in Beijing, China in October 2000.

* Corresponding author. Tel.: +82 31 400 5262; fax: +82 31 602 7730.
E-mail address: yhlee@hanyang.ac.kr (Y.H. Lee).

0360-8352/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cie.2007.06.018
Recently, researchers have placed increasing attention on integrated models for supply chain planning. Process planning and scheduling are possibly the most important functions in a supply chain system. This is because these functions are essential for inventory to be available-to-promise for customers. Process planning determines how an item will be manufactured; it acts as a bridge between design, manufacturing, and scheduling. Process planning considers alternative resources.

In practice, any customer order, assigned to a site for processing, can be scheduled on various resources and may have a flexible process sequence. If some orders have certain process sequences, they should be considered for an integrated model using alternative resources. Hankins, Wysk, and Fox (1984) showed that using alternative resources reduces lead-time and improves overall resource utilization. Nasr and Elsayed (1990) presented two heuristics to determine an efficient schedule for the \( n \) orders and the \( m \) resources problem with alternative routings allowed for each operation. Brandimarte and Calderini (1995) developed a two-phase hierarchical Tabu search for planning and scheduling. Palmer (1996) developed a method based on simulated annealing to solve integrated problems.

However, the researchers mentioned above did not consider the precedence rule for an operation sequence; they only considered the time aspect with a non-constraint operational sequence. Tan (2000) presented a review of various research efforts considering the process planning and scheduling area and discussed the extent of applicability of various approaches in solving process planning and scheduling. Guinet (2001) proposed a primal-dual approach to solve a production planning model with a network.


The main focus of our work is not only the formulation of a mathematical model that incorporates the process planning of resource selection and operation sequencing and the determination of their schedules so that the total processing time is optimized, but also the development of an efficient evolutionary search approach with a topological sort (TS) technique (Horowitz & Sahni, 1984). This technique is to determine a set of operation sequences for orders. The evolutionary search approach has been successfully applied to a number of combinatorial optimization problems (Cochran et al., 2003; Gen & Cheng, 2000; Moon et al., 2002).

An evolutionary search approach based on a TS can generate a good solution to the model within a reasonable amount of computation time. Since precedence relations between operations are handled by the TS, an evolutionary search approach produces only feasible solutions.

2. Problem definition

The manufacturing process for a customer order is made up of various processes used for converting raw materials or semi-finished products, or assembly parts into finished products. The set of processes can be constrained by the precedence relations. These relations are imposed by the technological requirements of the product. Process planning contains the determination of operations and their sequences considering precedence relations and alternative resources for effectively producing a customer order. Scheduling involves allocating resources over time to produce a set of orders. The integrated process planning and scheduling problem is to determine optimal schedules with process plans considering precedence relations between operations, processing time, and the set of alternative resources. The schematic diagram of the problem is shown in Fig. 1.

In Fig. 1, a set of \( n \) orders are to be processed using \( m \) resources with alternative operation sequences. Since the orders that should be manufactured have a typical operational structure constrained by sequentially related operations or machining functions, a process plan representation should have the capability to represent all possible precedence constraints that occur among the planning and processing decisions.
All orders are loaded and processed according to a determined process sequence with resource selection in the process planning. Each order requires a number of operations. Each operation can be performed on a number of alternative, non-identical resources. The processing times with alternative resources for a specific operation are not equal since each resource has finite capacities with different capabilities.

In general, the completion time for all customer orders can be calculated by summing machining times, setup times, transportation times, and waiting times. The total length of time for finishing all customer orders is called makespan which is an important criterion when the number of orders is finite.

3. Integrated process planning and the scheduling model

In order to describe the integrated model, we introduce the following notations:

- $i, j$: the index for the order, $i, j = 1, \ldots, I$, where $I$ is the number of orders,
- $k, l$: the index for the operation, $k, l = 1, \ldots, K$, where $K$ is the number of all of the operations,
- $p, q$: the index for the resource, $p, q = 1, \ldots, M$, where $M$ is the number of resources,
- $K_i$: the number of operations in order $i$,
- $R_i$: the set of precedence relations of two operations, $R_i = \{r_{ij}\forall j = 1, \ldots, J_i\}$, where $r_{ik} = \langle k, l \rangle$, $J_i$ is the number of precedence relations for order $i$, and $k$ and $l$ are indices for operations,
- $B_i$: the set of pairs of operations for order $i$, $B_i = \{r_{ij}\forall j = 1, \ldots, J_i\}$, where $r_{ij} = \langle k, l \rangle$ or $r_{ij} = \langle l, k \rangle$ in any order, $J_i$ is the number of all pairs of two operations without precedence relations for order $i$, and $k$ and $l$ are indices for operations,
- $G_p$: the set of operations to be performed on resource $p$,
- $p_{ikp}$: the processing time for operation $k$ of order $i$ on resource $p$,
- $M$: an arbitrary large positive number

\[
    d_{ikl} = \begin{cases} 
        1, & \text{if operation } k \text{ precedes operation } l \text{ of order } i, \\
        0, & \text{otherwise.} 
    \end{cases}
\]

We introduce two variables to adapt the integrated model:

- $x_{ikp}$: the completion time of operation $k$ of order $i$ on resource $p$
- $y_{ikp}$: the selection of resource $p$ for operation $k$ of order $i$

\[
    y_{ikp} = \begin{cases} 
        1, & \text{if resource } p \text{ is selected for operation } k \text{ of order } i, \\
        0, & \text{otherwise.} 
    \end{cases}
\]

The objective of the model is to minimize the makespan. The mixed integer programming model for solving the integrated model is presented below.
Minimize \( F = \max_{i, k \text{ and } p} \{x_{ikp}\} \)

subject to

\[
x_{ilq} - x_{ikp} \geq p_{lq} \quad \forall (k, l) \in R_{i}, i, p \text{ and } q, \quad (1)
\]

\[
x_{ilq} - x_{ikp} + M(1 - d_{ik}) \geq p_{lq} \quad \forall (k, l) \in B_{i}, i, p \text{ and } q, \quad (2)
\]

\[
x_{ikp} - x_{ilq} + Md_{ikl} \geq p_{ikp} \quad \forall (k, l) \in B_{i}, i, p \text{ and } q, \quad (3)
\]

\[
x_{ilq} - x_{ikp} + M(1 - d_{ik}) \geq p_{lq} \quad \forall (k, l) \in G_{p}, i, p, q \text{ and } i \neq j, \quad (4)
\]

\[
x_{ikp} - x_{ilq} + Md_{kl} \geq p_{ikp} \quad \forall (k, l) \in G_{p}, i, p, q \text{ and } i \neq j, \quad (5)
\]

\[
\sum_{p=1}^{M} Y_{ikp} = 1 \quad \forall i \text{ and } k, \quad (6)
\]

\[
x_{ikp} \geq \begin{cases} p_{ikp} & \text{for } i, k \in B_{i}, p \\ 0 & \text{for all other } i, k \end{cases} \quad (7)
\]

\[
y_{ikp} \in \{0, 1\} \quad \forall i, k, \text{ and } p. \quad (8)
\]

Constraint (1) means that the operations of each customer order are processed according to the precedence required. Constraints (2) and (3) ensure that any two operations belonging to the same customer order cannot be processed at the same time. Constraints (4) and (5) ensure that a resource cannot process more than one customer order at the same time. The constraints from (1)–(5) are called disjunctive constraints because one or the other alone must hold. Constraint (6) ensures that only one resource for each operation should be selected. Constraints (7) and (8) imply non-negativity and integrality of the corresponding variables.

4. Evolutionary search approach

Evolutionary search method can provide optimal or near optimal solutions for combinatorial optimization problems. It has been applied to a number of fields including engineering, biology, computer science, and social sciences. One of the most attractive features of the evolutionary search approach is its flexibility in handling various objective functions with fewer requirements for mathematical properties (Gen & Cheng, 1997).

The main issues in developing an evolutionary search approach are chromosome representation, population initialization, evaluation measure, crossover, mutation, and selection strategy. In addition, parameters such as population size, \( pop\_size \); number of generations, \( max\_gen \); probability of crossover, \( p_{c} \); and probability of mutation, \( p_{m} \); should be determined before executing the approach.

In this section, we propose an efficient evolutionary search approach. This approach contains a TS algorithm for solving the integrated process planning and scheduling model with precedence constraints.

4.1. A topological sort for process sequencing

We employ the TS algorithm (Horowitz & Sahni, 1984) to solve the problem of determining the operation sequence for an order. In a directed graph \( G \), the vertices represent operations and the edges represent precedence relationships between operations. It is clear that a topological order (TO) is impossible if the graph contains a cycle; thus, a directed graph with no directed cycles is an acyclic graph. This linear ordering is prioritized such that if a vertex \( v_{1} \) is the predecessor of a vertex \( v_{3} \) in the network, then \( v_{1} \) precedes \( v_{3} \) in the linear ordering. The algorithm that sorts the data into a TO is straightforward and proceeds by eliminating from the network any vertex that has no predecessor. Such vertices, together with all edges leading from them, are deleted from the network. The TS procedure which can be used to produce an operation sequence is described in Fig. 2.
4.2. Representation and initialization

In an ordering problem using an evolutionary search approach, a critical issue is the development of a representation scheme to represent a feasible solution. It is very difficult to represent a path with precedence constraints in a direct graph. In order to generate TOs, the representation scheme has to be capable of generating all possible TOs for a given activity on vertex (AOV) network. In addition, any tour of the solution always corresponds to TOs. Suppose there is one order, which consist of six vertices (operations), $v_1$ through $v_6$, and produces a part. The chromosome structure can be represented as shown in Fig. 3. The main goal of this study is to develop a flexible algorithm capable of solving an integrated model of a real manufacturing environment represented by generalized job-shop scheduling (Papadimitriou & Steiglitz, 1982).

In Fig. 3, the first row indicates the vertex number of operations which can match with randomly selected resource numbers for each operation. Therefore, the second row indicates that each operation randomly selects a resource number from the possible alternative resources. For example, $v_2$ is randomly selected for the operation-sequence because $v_1$ and $v_2$ have no precedent constraints simultaneously. After selecting the $v_2$, we can remove $v_2$ and all edges leading out of $v_2$. Now $v_1$ has no predecessors and $v_1$ is selected for the next operation sequence. By the same manner, the operation sequence, $v_2-v_1-v_4-v_3-v_5-v_6$, is determined for an order and the resources, $p_1-p_2-p_1-p_2-p_3-p_5$, are used for each operation sequentially.

For example, let us suppose that the alternative resources and their processing times for each operation are as shown in Table 1. The start-time of $v_2$ on resource number 1 is zero and its end-time is 5. For the next operation, $v_1$ on resource number 2, the start time should be the end-time of $v_2$ and the end time is 10 (end-time of $v_2$ + process time of $v_1$). By the same manner, the result is as shown in Table 2.

Table 2 can be presented as a Gantt chart as shown in Fig. 3. The makespan is 23. When the chromosomes are merged and modified by genetic operators such as crossover or mutation, the makespan may be reduced. When the makespan is minimized enough, the chromosome can show us the best resource selection and the operation sequences. The procedure for generating the feasible solution is shown in Fig. 4.

The first step of the evolutionary search method is to initialize the population composed of chromosomes. The chromosomes contain randomly selected resources from the set of alternative resources for each operation. The initialization process is executed with procedures as shown in Fig. 4. It generates as many chromosomes as population-size.

---

**Procedure: Topological sort**

**Input data**

- Activity on vertex (AOV) network;
- Let $J$ be the number of vertices;

For $j = 1$ to $J$ do

Begin

If every vertex has a predecessor then

the network is infeasible stop;

Else pick a vertex $v_j$ which has no predecessors;

Output $v_j$;

Delete $v_j$ and all edges leading out of $v_j$ from the network;

End

End

Fig. 2. Procedure of topological sort.
Table 1
Alternative resource processing times for operation \( v_j \)

<table>
<thead>
<tr>
<th>Resource no.</th>
<th>Operation</th>
<th>( v_1 )</th>
<th>( v_2 )</th>
<th>( v_3 )</th>
<th>( v_4 )</th>
<th>( v_5 )</th>
<th>( v_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>6</td>
<td>–</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>–</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>3</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>–</td>
<td>–</td>
<td>6</td>
<td>4</td>
<td>–</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2
Start and end-time of each operation (start-time, end-time)

<table>
<thead>
<tr>
<th>Resource no.</th>
<th>Operation</th>
<th>( v_1 )</th>
<th>( v_2 )</th>
<th>( v_3 )</th>
<th>( v_4 )</th>
<th>( v_5 )</th>
<th>( v_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>(0, 5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>(5, 10)</td>
<td>(13, 17)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(17, 20)</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(20, 23)</td>
</tr>
</tbody>
</table>
4.3. Selection and fitness evaluation

The chromosomes have large amounts of information. They are an allocation of operations to each resource which are constrained by determined operation sequences. The procedure for fitness evaluation is shown in Fig. 5.

In our work described in this paper, the formulation considers only one objective function for integrating process planning and scheduling. The objective is to minimize the makespan of fabrication. The makespan is the time required for completing all of the operations for total orders; thus, the completion time of the last operation is the makespan time. After a fitness evaluation for all generated chromosomes, a selection strategy is concerned when choosing chromosomes from the population space. It may create a new population for the next generation based on parents and offspring. We employed a well-known elitist strategy in the enlarged sampling space (Gen & Cheng, 1997) for the selection process.

4.4. Crossover and mutation

To create the new population for the next generation, the execution of genetic operators such as selection, crossover, and mutation forms a new set of offspring chromosomes. In particular, crossover is the main oper-
For crossover, the two point swapping crossover operator (Michalewicz, 1994) is used. This operator simply selects two chromosomes from the parent population, points out two genes in the chromosomes randomly, and then swaps their contents (resource number). For mutation, the random mutation operator (Michalewicz, 1994) is used. This operator simply selects one chromosome and points out one gene in the chromosome. Then, it regenerates randomly a new resource number within the feasible ones for the gene. An example of the two point swapping crossover operator is shown in Fig. 6.

If after crossover and mutation operators, infeasible chromosomes may be generated, they will be rejected by fitness evaluation, and the offspring with feasible chromosomes will have a chance to be selected in the population for the next generation by the selection process.

4.5. Overall procedure for the integrated model

Let $P(t)$ and $C(t)$ be parents and offspring in a generation $t$, respectively. The overall procedure of the proposed evolutionary search approach is described as shown in Fig. 7.
5. Experiments

We describe the design of the first and second examples with various orders and operations in this section. The last example uses a conventional model taken from Sundaram and Fu (1988). We tested all examples to prove the efficiency of the proposed evolutionary search approach.

For experimental comparison under a same condition, the parameters used in the proposed evolutionary search approach were set: \( \text{pop}_{-} \text{size} \) was 20, \( \text{max}_{-} \text{gen} \) was 2000, \( p_c \) was 0.4, and \( p_m \) was 0.1. Altogether 20 iterations were executed to eliminate the randomness of the searches in the proposed evolutionary search approach. The proposed approach was implemented using the Visual Basic language running on an IBM compatible PC with a Pentium 1.2 GHz CPU and 1GB RAM.
5.1. Experiment 1

An example consisting of five orders and five resources were used for experiment 1. Each order involved different operations in a specified order. Let $v_j$ ($j = 1, \ldots, 13$) be the operations for five orders arranged in an orderly manner. This experiment should consider the precedence constraints in each operation. The set of directed graph is shown in Fig. 8. The order related-information and the alternative operation sequences for each order are shown in Table 3.

The best chromosome, the determined resource selection, and best operation sequence for all five orders are shown in Tables 4 and 5, respectively. Using the best chromosome, the best operation sequence was determined: $v_{10} - v_{12} - v_{11} - v_{13} - v_6 - v_9$. Finally, when the operation sequence in each order was considered, the minimized makespan was 16 as shown in Fig. 9. Using the optimal makespan, the start and end times of each operation can be calculated in each machine and their results are shown in Table 6.

The experimental results shown in Tables 4–6 and Fig. 9 demonstrate that each order could select the best operation sequence in order to minimize the makespan. Thus, the proposed evolutionary search approach can greatly reduce the makespan of the total schedule when considering the precedence constraints between orders. Finally, we could prove that the proposed evolutionary search approach was capable of producing the optimal schedule.

### Table 3

<table>
<thead>
<tr>
<th>Order no.</th>
<th>Order type (operation no.)</th>
<th>Operation for alternative resources (operation no., resource no., processing time)</th>
<th>Alternative operation sequences considering precedence constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(v_1, v_2)</td>
<td>(v_1, 1, 5), (v_1, 2, 3), (v_2, 2, 5)</td>
<td>$v_1 - v_2$</td>
</tr>
<tr>
<td>2</td>
<td>(v_3, v_4)</td>
<td>(v_3, 3, 6), (v_4, 4, 5), (v_5, 4, 4)</td>
<td>$v_3 - v_4$</td>
</tr>
<tr>
<td>3</td>
<td>(v_5, v_6, v_7)</td>
<td>(v_5, 2, 4), (v_6, 3, 5), (v_7, 3, 4), (v_8, 2, 3), (v_9, 4, 3), (v_10, 2, 5)</td>
<td>$v_5 - v_6 - v_7$, $v_8 - v_9 - v_{10}$, $v_{11} - v_{12}$</td>
</tr>
<tr>
<td>4</td>
<td>(v_8, v_9)</td>
<td>(v_8, 3, 4), (v_9, 4, 5)</td>
<td>$v_8 - v_9$</td>
</tr>
<tr>
<td>5</td>
<td>(v_{10}, v_{11}, v_{12}, v_{13})</td>
<td>(v_{10}, 1, 4), (v_{10}, 2, 3), (v_{11}, 1, 2), (v_{11}, 2, 4), (v_{12}, 3, 5), (v_{13}, 3, 4), (v_{13}, 5, 3)</td>
<td>$v_{10} - v_{11} - v_{12} - v_{13}$, $v_{12} - v_{10} - v_{11} - v_{13}$</td>
</tr>
</tbody>
</table>

**Fig. 8.** A set of directed graph with precedence constraints in each operation.
Table 4
Best chromosome

<table>
<thead>
<tr>
<th>Vertex no.</th>
<th>$v_1$</th>
<th>$v_2$</th>
<th>$v_3$</th>
<th>$v_4$</th>
<th>$v_5$</th>
<th>$v_6$</th>
<th>$v_7$</th>
<th>$v_8$</th>
<th>$v_9$</th>
<th>$v_{10}$</th>
<th>$v_{11}$</th>
<th>$v_{12}$</th>
<th>$v_{13}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource no.</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 5
Determined resource selection and best operation-sequence

<table>
<thead>
<tr>
<th>Operation sequences and selected resource number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order 1 : Resource no. 2</td>
</tr>
<tr>
<td>Order 2 : Resource no. 4</td>
</tr>
<tr>
<td>Order 3 : Resource no. 1</td>
</tr>
<tr>
<td>Order 4 : Resource no. 3</td>
</tr>
<tr>
<td>Order 5 : Resource no. 1</td>
</tr>
</tbody>
</table>

Fig. 9. Schedule output by Gantt chart for experiment 1.

Table 6
Start and end-times of each operation (start-time, end-time)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Resource no.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>(0,3)</td>
</tr>
<tr>
<td>$v_2$</td>
<td>(3,8)</td>
</tr>
<tr>
<td>$v_3$</td>
<td>(0,5)</td>
</tr>
<tr>
<td>$v_4$</td>
<td>(4,9)</td>
</tr>
<tr>
<td>$v_5$</td>
<td>(9,11)</td>
</tr>
<tr>
<td>$v_6$</td>
<td>(11,16)</td>
</tr>
<tr>
<td>$v_7$</td>
<td>(0,4)</td>
</tr>
<tr>
<td>$v_8$</td>
<td>(5,10)</td>
</tr>
<tr>
<td>$v_9$</td>
<td>(11,14)</td>
</tr>
<tr>
<td>$v_{10}$</td>
<td>(0,4)</td>
</tr>
<tr>
<td>$v_{11}$</td>
<td>(9,11)</td>
</tr>
<tr>
<td>$v_{12}$</td>
<td>(4,9)</td>
</tr>
<tr>
<td>$v_{13}$</td>
<td>(11,14)</td>
</tr>
</tbody>
</table>
5.2. Experiment 2

For experiment 2, the various order and resource sizes were considered, and they were implemented by a varied genetic environment as shown in Table 7. The best makespan obtained after the proposed evolutionary search approach are summarized in Table 7.

In Table 7, the first environment with 16 orders, 40 operations and 5 resources had the best makespan of 63. This result had a fixed value under the varied genetic environment, which means that the proposed evolutionary search approach has a robustness in generating the best makespan though the genetic environment was

<table>
<thead>
<tr>
<th>Order</th>
<th>Operation</th>
<th>Resource</th>
<th>pop_size</th>
<th>max_gen</th>
<th>p_c</th>
<th>p_m</th>
<th>Makespan</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>40</td>
<td>5</td>
<td>50</td>
<td>1000</td>
<td>0.4</td>
<td>0.1</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>1000</td>
<td>0.5</td>
<td>0.5</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>100</td>
<td>1500</td>
<td>0.4</td>
<td>0.1</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>100</td>
<td>1500</td>
<td>0.5</td>
<td>0.5</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>150</td>
<td>2000</td>
<td>0.4</td>
<td>0.1</td>
<td>63</td>
</tr>
<tr>
<td>32</td>
<td>80</td>
<td>5</td>
<td>50</td>
<td>1000</td>
<td>0.4</td>
<td>0.1</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>1500</td>
<td>0.4</td>
<td>0.1</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>100</td>
<td>1500</td>
<td>0.4</td>
<td>0.1</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>150</td>
<td>3000</td>
<td>0.4</td>
<td>0.1</td>
<td>114</td>
</tr>
<tr>
<td>64</td>
<td>160</td>
<td>5</td>
<td>50</td>
<td>1000</td>
<td>0.4</td>
<td>0.1</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>1500</td>
<td>0.4</td>
<td>0.1</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>100</td>
<td>1500</td>
<td>0.4</td>
<td>0.1</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>150</td>
<td>2000</td>
<td>0.4</td>
<td>0.1</td>
<td>227</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>150</td>
<td>3000</td>
<td>0.4</td>
<td>0.1</td>
<td>227</td>
</tr>
</tbody>
</table>

Table 8
Order related-information Sundaram and Fu (1988)

<table>
<thead>
<tr>
<th>Order no.</th>
<th>Order type (operation no.)</th>
<th>Operation for alternative resources (operation no., resource no., processing time)</th>
<th>Operation-sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(v1, v2, v3, v4)</td>
<td>(v1, 1, 5), (v1, 2, 3) (v2, 2, 7) (v3, 3, 6) (v4, 4, 3), (v4, 5, 4)</td>
<td>v1–v2–v3–v4</td>
</tr>
<tr>
<td>2</td>
<td>(v5, v6, v7, v8)</td>
<td>(v5, 1, 7) (v6, 2, 4), (v6, 3, 6) (v7, 3, 7), (v7, 4, 7) (v8, 5, 10)</td>
<td>v5–v6–v7–v8</td>
</tr>
<tr>
<td>3</td>
<td>(v9, v10, v11, v12)</td>
<td>(v9, 1, 4), (v9, 2, 5) (v9, 3, 8) (v10, 4, 5) (v11, 4, 6), (v12, 5, 5) (v12, 5, 4)</td>
<td>v9–v10–v11–v12</td>
</tr>
<tr>
<td>5</td>
<td>(v17, v18, v19, v20)</td>
<td>(v17, 1, 3), (v17, 3, 5) (v18, 3, 7) (v19, 4, 9), (v19, 5, 6) (v20, 5, 3)</td>
<td>v17–v18–v19–v20</td>
</tr>
</tbody>
</table>
varied. Similar results are also shown in the second and third environments. Finally, we know that the proposed evolutionary search approach was effective in generating the best makespan under various order environments with precedence constraints.

5.3. Experiment 3

For more various comparisons using conventional methods, the work of Sundaram and Fu (1988) was used to prove the efficiency of the proposed evolutionary search approach. Their work consisted of five orders and five resources. Let \( v_j \) \((j = 1, \ldots, 20)\) be the operations for the five orders arranged in an orderly manner. The order related-information is given in Table 8. However, in their work, the operation sequence was not considered and four different operations were performed in each order in sequential order. Therefore, the operations, \( v_1, v_5, v_9, v_{13}, \) and \( v_{17} \), in Table 8 have no precedence constraints and the operation \( v_i \) is a predecessor of \( v_{i+1} \) in the network.

Using the proposed evolutionary search approach, the best makespan obtained was 33 as shown in Fig. 10. This result was compared with the conventional work performed by Sundaram and Fu (1988) and Palmer (1996). Sundaram and Fu (1988) used a heuristic approach; while Palmer (1996) employed a simulated annealing to solve this problem. Table 9 shows the best makespan obtained in each algorithm. In Table 9, the performance of the proposed evolutionary search approach was superior to that of Sundaram and Fu (1988).

Based on the comparison result using three experiments in Sections 5.1, 5.2 and 5.3, we can conclude that the proposed evolutionary search approach is an efficient algorithm to effectively solve the integrated process planning and scheduling in a supply chain problem.

6. Conclusion

We have proposed an evolutionary search approach based on a topological sort to solve the integrated process planning and scheduling problem in a supply chain. Solving the problem is essential for the generation of flexible process sequences with resource selection and for the decision of operations schedules that can minimize makespan. We have formulated a mixed integer programming model to represent the problem. The problem considers alternative resources and sequences, and precedence constraints. The objective is to find the optimal resource selection for assignments and operation sequences and to decide a schedule in order to minimize the makespan. To solve the problem, a new evolutionary search approach based on topological
sort has been developed. Since precedence relations between operations are handled by a topological sort, the evolutionary search approach produces only feasible solutions.

Experimental results have shown that the proposed evolutionary search approach can be a good alternative for locating an optimal solution in solving complicated process planning and scheduling situations because it produces better solutions than traditional approaches. In conclusion, the proposed evolutionary search approach can be effectively used to solve the complex and sizable problem of integrated process planning and scheduling in supply chains.

References


