An Adaptive Genetic Algorithm with a Fuzzy Logic Controller for Solving Sequencing Problems with Precedence Constraints

YoungSu Yun
Division of Business Administration,
Chosun University
(ysyun@chosun.ac.kr)

In this paper, we propose an adaptive genetic algorithm (aGA) approach for effectively solving the sequencing problem with precedence constraints (SPPC). For effective representation of the SPPC in the aGA approach, a new representation procedure, called the topological sort-based representation procedure, is used. The proposed aGA approach has an adaptive scheme using a fuzzy logic controller and adaptively regulates the rate of the crossover operator during the genetic search process. Experimental results using various types of the SPPC show that the proposed aGA approach outperforms conventional competing approaches. Finally the proposed aGA approach can be a good alternative for locating optimal solutions or sequences for various types of the SPPC.

Received : April 02, 2011 Revision : April 06, 2011 Accepted : April 13, 2011
Type of Submission : English Fast-track Corresponding author : YoungSu Yun

1. Introduction

The sequencing problem with precedence constraints (SPPC) can be applied in a number of optimization problems regarding networks, scheduling, project management, logistics, assembly flow, and routing. The determination of an optimal solution or sequence for the SPPC may be both time- and cost-intensive since infeasible sequences that may be produced in the SPPC greatly affect all downstream stages regarding networks, scheduling, logistics, etc.

The SPPC involves the location of an optimal sequence with the shortest traveling time among all feasible sequences, visiting each node only once on a directed graph, G(A, B), where A = {1, 2, ⋯, I} is the set of nodes, and B = {<i, j>, i, j ∈ A and $i \neq j$} is the set of precedence relations. The precedence relation, <i, j>, determines the sequence of travel between the

* This work was supported by National Research Foundation of Korea-Grant funded by the Korean Government(NRF--2009-327- B00239).

given pair of nodes, *i* and *j*; i.e., *i* must be visited before *j*. This problem can be formulated as a traveling salesman problem (TSP) with precedence constraints.

A number of approaches have been developed for the SPPC. Chen (1990) used an AND/OR SPPC to solve assembly scheduling problems. In this approach, the precedence relations result from geometric and physical constraints regarding the assembled items. He and Kusiak (1992) presented a scheduling model with precedence constraints and the traveling time for each node; they proposed a heuristic approach to solve the scheduling model. Save lsbergh and Sol (1995) presented an SPPC to solve the Dial-A-Ride problem where a vehicle should transport a number of passengers and each passenger should be transported from a given location to a specific destination point. Renaud et al. (2000) suggested a heuristic model to solve a pickup and delivery TSP, which can be formulated as an SPPC. Moon et al. (2002) presented a genetic algorithm (GA) approach with a priority-based representation procedure to solve the SPPC based on a supply chain planning model. Duman and Or (2004) presented an SPPC for a printed circuit board assembly problem and suggested an approach to solve it. This approach initially ignores precedence relations and solves the problem as a pure TSP; it then eliminates the damaged elements in the resulting TSP tour.

More recently, Lambert (2006) formulated a disassembly scheduling problem by modifying the two-commodity network models, which is an SPPC. Su (2007) proposed a case-based reasoning method supported by an evolutionary algorithm to solve an SPPC.

Most of the conventional studies mentioned above show that the SPPC can provide a useful way for representing various industrial optimization problems with precedence constraints. However, there are some drawbacks in modeling the SPPC. First, most of the conventional approaches may not effectively solve various types of the SPPC because of its computational complexity and complicated precedence constraints. Secondly, the computational times required for solving various types of the SPPC greatly increase with the number of nodes considered. Therefore, a new approach is required to improve these drawbacks, that is, it can easily handle complicated precedence constraints and find an optimal solution or sequence quicker than conventional methods as the increase of the number of nodes.

This paper aims at formulating a mathematical model as well as developing an efficient aGA approach for effectively solving various types of the SPPC. The proposed mathematical model is formulated by modifying the standard integer programming for the TSP in order to avoid computational difficulties and ambiguity. In the aGA approach, the topological sort (TS)-based representation procedure is used to generate a set of feasible sequences, which is used for the individuals of the population in the proposed aGA approach. A fuzzy logic controller (FLC) is used for adaptively regulating the rate of the crossover operator in the proposed aGA approach.

2. Mathematical Formulation

The SPPC requires precedence relations with no cycles. Therefore, if a directed graph has a loop, then a linear sequence is impossible, since either *i* is visited before *j* or *j* is visited before *i*. A feasible sequence passes through each node in the given directed graph only once. Therefore, the SPPC is a type of directed acyclic graph. Since most of the directed graphs may have many feasible sequences, it is necessary to find an optimal sequence among all the feasible ones. An optimal sequence, which minimizes the total traveling time, can be decided by comparing all feasible sequences.

In order to avoid computational difficulty and ambiguity, a mathematical model is proposed and formulated by modifying the standard integer programming formulation for the TSP (Baker, 1974; He and Kisiak, 1992; Pinedo and Chao, 1999). The following notation is used to describe the SPPC.

i, j	Node indexes; *i*, *j* = 1, ⋯, *i*, where *i* is the number of nodes.
T_{ij}	Traveling time from node *i* to node *j*.
Ai	Arrival time at node *i*.
P	Set of nodes, (*i*, *j*), where node *i* is visited before node *j*.
R	Set of nodes, (*i*, *j*), where nodes *i* and *j* can be visited in any feasible sequence.
L	An arbitrarily large positive number.

Variables are introduced to formulate the SPPC as follows :

$$x_{ij} = \begin{cases} 1, & \text{if node } i \text{ is visited before } j, \\ 0, & \text{otherwise.} \end{cases}$$

The objective function for the SPPC is to find a node with the maximum arrival time among all nodes, since this node denotes the completion of all the tasks in the given network. Let *MaxAT* be the procedure for finding a node with the maximum arrival time. Then, *MaxAT* can be represented as follows.

$$MaxAT = \max_{\forall i} A_i \tag{1}$$

Therefore, a mixed-integer programming model for the SPPC can be formulated as follows.

$$\text{minimize } MaxAT \tag{2}$$

subject to

$$A_j - A_i \geq T_{ij} \; \forall (i, j) \in P \text{ and } i \neq j \tag{3}$$

$$A_j - A_i + L_{\xi j} \geq T_{ij} x_{ij} + T_{ji} x_{ji}, \quad \forall (i, j) \in R \text{ and } i \neq j \tag{4}$$

$$x_{ij} + x_{ji} = 1, \; \forall (i, j) \in R \text{ and } i \neq j \tag{5}$$

$$x_{ij} + x_{ji} = 0, 1 \; \forall (i, j) \in R \text{ and } i \neq j \tag{6}$$

Constraints (3) and (4) ensure that two nodes cannot be visited at the same time. The constraints in (5) imply that any two nodes are to be visited in only one sequence. These constraints also mean that there are no cycles in the precedence relations. Constraint (6) restricts the variables to integer.

3. aGA Approach

The detailed procedures of implementing the proposed aGA approach appear in the following subsections. First, a TS-based representation procedure is suggested for effectively treating the precedence constraints that appear in the SPPC. By the use of the TS-based representation procedure, the initialization process for the proposed aGA population is adopted. Secondly, a fitness test and a selection procedure for the individuals resulting from the proposed aGA search process are undertaken. Third, crossover operator is used for genetic operators. Lastly, a FLC procedure to adaptively regulate the rate of the crossover operator is proposed.

3.1 Representation and Initialization

Of the well-known methods for treating precedence constraints in the SPPC, the priority-based representation procedure has been developed by several researchers (Gen and Cheng, 2000; Moon et al., 2002; Moon and Seo, 2005; Gen et al., 2006). The priority-based representation procedure has been widely used in representing feasible sequences. The key issue of the procedure is to assign randomly generated priorities to each node. Therefore, the selection of each node always relies on the priority assigned. Finally, various types of feasible sequences may not be produced because of the priority constraint that is assigned to each node. This case is called idling and may restrict the efficiency of the search process.

For effectively treating precedence constraints in the SPPC, a TS-based representation procedure is developed in this paper. In general, a precedence relation between nodes i and j is represented as predecessor or successor of each node in any sequence. If they visit each other, this situation is called as cycle. To represent precedence relations, we can use the precedence matrix, $D = [d_{ij}]$, where :

$$x_{ij} = \begin{cases} 1, & \text{if node } i \text{ is visited immediately before } j, \\ \text{null}, & \text{otherwise.} \end{cases}$$

where $d_{ij} = 1$ indicates the precedence relation, $<i, j>$.

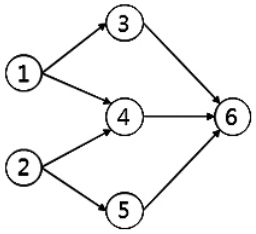
The precedence matrix for an example with six nodes and precedence relations is shown in <Table 1>.

<Table 1> An Example of Precedence Matrix

		Node number					
		1	2	3	4	5	6
Node number	1			1	1		
	2				1	1	
	3						1
	4						1
	5						1
	6						

For example, $d_{24} = 1$ in <Table 1> means that node 2 should be visited immediately before node 4, that is, node 2 precedes node 4 in linear sequencing. We call node 2 an immediate predecessor of node 4, and node 4 an immedi-

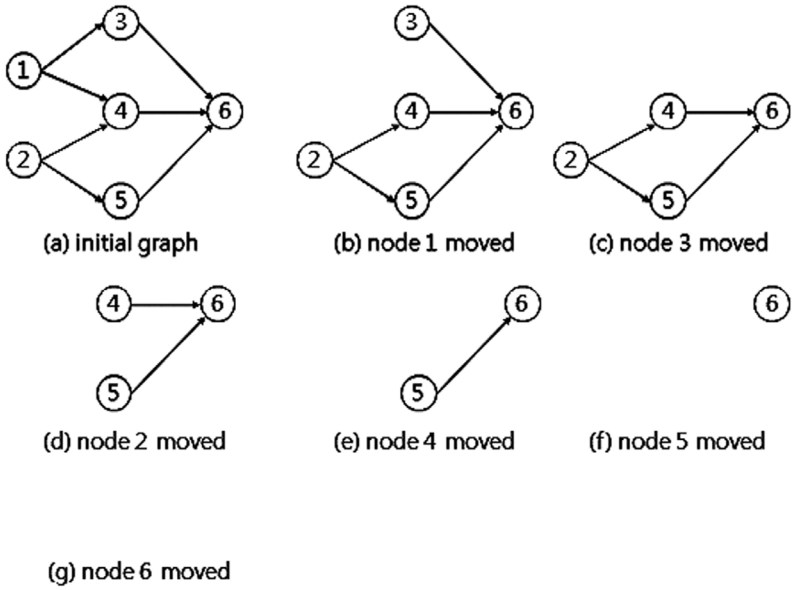
ate successor of node 2. On the other hand, if d_{ij} = null, there is no relationship between the two nodes, i and j. However, if d_{12} = 1, d_{23} = 1, and d_{31} = 1, the graph cannot generate a feasible sequence because it has a cycle. Also, the matrix columns, 1 and 2, in Table 1 have 'null' values. This indicates that nodes 1 and 2 have no predecessors. From the precedence relations in <Table 1>, we can draw the directed graph shown in <Figure 1>.



<Figure 1> A Directed Graph with Precedence Constraints

procedure : TS-based representation procedure
input : a precedence matrix and its directed graph
i : number of nodes to visit
output : feasible sequence in the directed graph
step 0 : **for** all nodes
If all nodes have a predecessor or there is no node *i*;
then the directed graph is infeasible, stop;
else randomly select a node *i* without any predecessors;
return node *i*;
remove node *i* and its arc from the directed graph;
end if
end for

<Figure 2> TS-based Representation Procedure



<Figure 3> Implementation of the TS-based Representation Procedure

The TS-based representation procedure for producing a feasible sequence in a directed graph is described in <Figure 2>.

The TS-based representation procedure in <Figure 2> plays the key role of producing a set of feasible sequences in a directed graph. By the TS-based representation procedure, a feasible sequence with respect to <Figure 1> can be produced. The detailed procedure is shown in <Figure 2>.

In <Figure 3>, either nodes 1 or 2 without any predecessors is randomly selected. If the selected node is 1, it is removed from the directed graph and the remaining nodes are represented as in situation (b). Secondly, a node between nodes 2 and 3 without any predecessors is randomly selected. If the selected node is 3, it is removed from the directed graph and the remaining nodes are represented as in situation (c). Thirdly, only node 2 should be selected and removed from the directed graph, because there is no node with any predecessors except for node 2. The remaining nodes are represented as in situation (d). Situations (e), (f), and (g) can also be represented in the same way as for situations (b), (c), and (d). Finally, a feasible sequence is produced as follows.

$$< 1 \quad 3 \quad 2 \quad 4 \quad 5 \quad 6 > \tag{7}$$

This feasible sequence can be used as an individual of the population in the proposed aGA approach. The next step is to produce the initial population for implementing the proposed aGA approach. By the TS-based representation procedure, we can easily produce some feasible sequence sets for as many individuals as the population size, pop_size.

If *pop_size* is 5, the initial population can be generated as shown in <Figure 4>.

$V_1 =$	1	3	2	4	5	6
$V_2 =$	2	5	1	3	4	6
$V_3 =$	1	3	2	5	4	6
$V_4 =$	1	2	5	4	3	6
$V_5 =$	2	1	5	4	3	6

<Figure 4> Initial Population

3.2 Fitness Test and Selection

To improve the performance of the individuals in the population in the proposed aGA approach, each individual should be evaluated by measuring its fitness. The fitness value of each individual is computed and the objective is to find the individual with the shortest traveling time. We consider the objective function of the mixed-integer programming model, which is to minimize the total traveling time as per Equation (1). Let *f(k)* be the fitness function of the k^{th} individual; then, *f(k)* can be expressed as follows.

$$f(k) = \min\{\max_{\forall i}\{A_i\}\} \tag{8}$$

The selection strategy is to choose the respective individuals from the current population; the chosen individuals are considered as the population of the next generation. For selection, the elitist selection strategy in an enlarged sampling space (Gen and Cheng, 1997) is used.

3.3 Crossover Operator

For improving the solution quality during the proposed aGA search process, a crossover operator is needed for exchanging genes between individuals. It works by combining subsequences from different parent individuals to produce a new individual. Unfortunately, some general types of crossover operator are not suitable for the SPPC, since most of the individuals resulting from crossover operators may violate the precedence constraints. Therefore, a new crossover operator that considers the TS-based representation procedure is proposed as follows.

Step 1 : Randomly select an individual, I_n, from the current population.

Step 2 : Randomly select two positions, P_1 and P_2, in the selected individual.

Step 3 : Randomly generate an integer, S_g, from (0, 2). If S_g = 0, then select the left subsequence genes of the first cut-off point. If S_g = 1, then select the subsequence genes between the first and second cut-off points. If S_g = 2, then select the right subsequence genes of the second cut-off point.

Step 4 : Use the TS-based representation procedure to produce alternative subsequence genes for the segment selected in Step 3.

Step 5 : Replace the subsequence genes selected in Step 3 with the alternative from Step 4.

Steps 1 to 3 of the above procedure can be represented as the following string.

$$I_n,\ P_1,\ P_2,\ S_g \tag{9}$$

In the above, the range of I_n is (1, *pop_size*); the range of P_1 and P_2 is (1, I); and the range of S_g is (0, 2).

Let us consider the initial population shown in <Figure 4> to implement the crossover operator. If(In, P_1, P_2, S_g) = (5, 1, 5, 1), the selected individual and its segment are shown in <Figure 5>.

$V_5 =$ | 2 | 1 | 5 | 4 | 3 | 6 |

<Figure 5> Before the Crossover Operation

For producing alternative subsequence genes for the segment selected in Step 3, the TS-based representation procedure is used in Step 4. Finally, the feasible subsequence genes, (1, 4, 5), are produced. The new individual with the alternative feasible subsequence genes for Step 5 is shown in <Figure 6>.

$V'_5 =$ | 2 | 1 | 4 | 5 | 3 | 6 |

<Figure 6> After the Crossover Operation

3.4 Adaptive Scheme using FLC

The adaptive scheme in the proposed aGA approach is for adaptively regulating the rate of the crossover operator. Various adaptive schemes for regulating the rate of the crossover operator have been suggested in many conventional studies (Srinivas and Patnaik, 1994; Wang et al., 1997; Song et al., 1997; Wu et al., 1998; Mak et al., 2000; Hong et al., 2002; Yun, 2002; Yun et al., 2003).

Especially, several adaptive schemes using FLCs have been successfully adopted for improving the performance of GAs (Song et al., 1997; Subbu et al., 1998; Gen and Cheng, 2000; Cheong and Lai, 2000). Gen and Cheng (2000) surveyed various adaptive schemes using several FLCs. Subbu et al. (1998) suggested a fuzzy logic-controlled genetic algorithm (FLC-GA) using a fuzzy knowledge base. In the FLC-GA, the rate of the crossover operator is adaptively regulated. Song et al. (1997) used a FLC for regulating the rate of the crossover operator. This FLC is considered as the input variable of the GA. For successfully applying FLCs to GAs, Subbu et al. (1998) and Song et al. (1997) proposed the production of well-formed fuzzy sets and rules. Recently, Cheong and Lai (2000) suggested an optimization scheme for the fuzzy sets and rules. Therefore, the GAs that are controlled by these types of FLC are more efficient in terms of the search speed and quality than the GAs without them.

In this paper, we also use an FLC to adaptively regulate the rate of the crossover operator. We use the basic concept of Song et al. (1997) and improve it in some aspects. The main idea behind this concept is to use a FLC : the crossover FLC, which is implemented to adaptively regulate the rate of the crossover operator during the genetic search process. The heuristic updating strategy for the rate of the crossover operator considers the changes in the average fitness values of the proposed aGA populations in two successive generations. That is, the rate of the crossover operator (P_c) is increased, if it consistently yields better offspring during genetic operations (selection and crossover). However, the P_c is also reduced, if poorer offspring are continuously produced. This scheme encourages well-performing operators to produce more individuals, while also reducing the chance for poorly performing operators to destroy the respective individuals during genetic operations.

For example, for a minimization problem, we can set the change of the average fitness value at generation *t*, $\triangle AvgFit(t)$, as follows :

$$\triangle AvgFit(t) = \left(\frac{\sum_{k=1}^{par\_size} f_k(t)}{par\_size} - \frac{\sum_{k=par\_size}^{par\_size+off\_size} f_x(t)}{off\_size}\right) \times \lambda \quad (10)$$

where *k* is the generation index and λ is a scaling factor to normalize the average fitness value for applying defuzzification in the FLC; λ is varied according to the problem under consideration.

The parameter, λ, was not used in the original study (Song et al., 1997). However, λ is definitely

required for normalizing the average fitness value because the fitness value is varied according to the problem under consideration. Both $\triangle AvgFit(t-1)$ and $\triangle AvgFit(t)$ are used to regulate p_c, as shown in <Figure 7>.

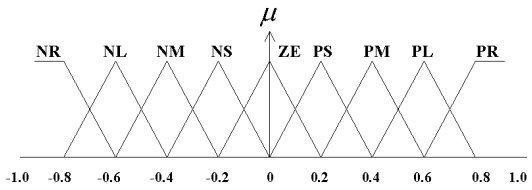
procedure : regulation of p_c using the average fitness value
begin
  if $\varepsilon \le \triangle AvgFit(t-1) \le \gamma$
    and $\varepsilon \le \triangle AvgFit(t) \le \gamma$
    then increase p_c for next generation;
  if $-\gamma \le \triangle AvgFit(t-1) \le -\varepsilon$
    and $-\gamma \le \triangle AvgFit(t) \le -\varepsilon$
    then decrease p_c for next generation;
  if $-\varepsilon < \triangle AvgFit(t-1) < \varepsilon$
    and $-\varepsilon < \triangle AvgFit(t) < \varepsilon$
    then rapidly increase p_c for next generation;
  end
end

<Figure 7> The Regulation of p_c Using the Average Fitness Value.

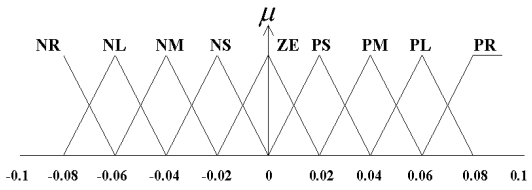
In the above, ε is a given real number in the proximity of zero, and γ and $-\gamma$ are given maximum and minimum values of a fuzzy membership function, respectively. The implementation strategy for the crossover FLC is as follows.

- *Input and output of the crossover FLC.*
  The inputs of the crossover FLC are $\triangle AvgFit(t-1)$ and $\triangle AngFit(t)$; the output is the change in the crossover rate, $\triangle c(t)$.

- *Membership functions of* $\triangle AvgFit(t-1)$, $\triangle Avg$ Fit(t), and $\triangle c(t)$.
  The membership functions of the fuzzy input and output linguistic variables are shown in <Figure 8>~<Figure 9>, respectively. Both $\triangle AvgFit(t-1)$ and $\triangle AvgFit(t)$ are respectively normalized in the range, [-1.0, 1.0]. Also, $\triangle c(t)$ is normalized in the range of [-0.1, 0.1] according to the corresponding maximum values.



<Figure 8> Membership Functions for $\triangle AvgFit(t-1)$ and $\triangle AvgFit(t)$

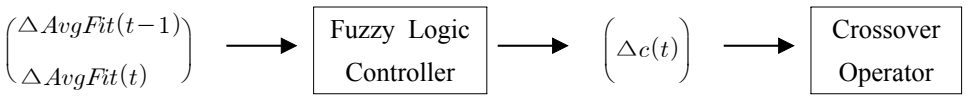


<Figure 9> Membership Function of $\triangle c(t)$

In <Figure 8>~<Figure 9>, NR means Negative larger, NL Negative large, NM Negative medium, NS Negative small, ZE Zero, PS Positive small, PM Positive medium, PL Positive large, and PR Positive larger.

- *Fuzzy decision table*
  For the fuzzy decision table, we use the table from Song et al. (1997).

- *Defuzzification table for control actions*
  The defuzzification table for simply represent-



<Figure 10> Regulation strategy of the FLC used for the aGA loop

ing the control actions of the crossover FLC was required, and the table is taken from Song et al. (1997).

The regulation strategy of the FLC used for the proposed aGA loop is summarized in <Figure 10>

The detailed procedure for its application is as follows.

Step 1 : The input variables of the FLC for regulating the rate of the crossover operator are the changes in the average fitness value in two successive generations, (*t-1* and *t*), as follows :

$$\triangle AvgFit(t-1),\ \triangle AvgFit(t) \qquad (11)$$

Step 2 : After normalizing $\triangle AvgFit(t-1)$ and $\triangle AvgFit(t)$, assign these values to the indexes *i* and *j* corresponding to the control actions in the defuzzification table (Song et al., 1997).

Step 3 : Calculate $\triangle c(t)$ as follows :

$$\triangle c(t) = Z(i,\ j) \times 0.02 \qquad (12)$$

where the contents of $Z(i, j)$ are the corresponding values of $\triangle AvgFit(t-1)$ and $\triangle AvgFit(t)$ for defuzzification (Song et al., 1997). The value of 0.02 is given to regulate the increasing and decreasing ranges of the rate of the crossover operator.

Step 4 : Update the changes in the rate of the crossover operator by using the following equations :

$$p_c(t) = p_c(t-1) + \triangle c(t) \qquad (13)$$

The adjusted rates should lie between 0.5 and 1.0 for $p_c(t)$.

3.5 Overall Procedure of the Proposed aGA Approach

For the proposed aGA approach, we use a real-number representation instead of a bit-string representation. The detailed heuristic procedures are as follows.

Step 1 : Representation

The TS-based representation procedure is used to effectively represent individuals.

Step 2 : Initialization

The initial population is composed of the individuals obtained by the TS-based representation procedure.

Step 3 : Fitness test
Equation (8) is used for the fitness test.

Step 4 : Genetic operators
Selection : The elitist strategy in an enlarged sampling space (Gen and Cheng, 2000).
Crossover : The new crossover operator considering the TS-based representation procedure shown in Section 3.3.

Step 5 : Adaptation
The adaptive scheme using the FLC shown in Section 3.4 is used.

Step 6 : Termination condition
If a predefined maximum number of generations is reached or an optimal solution already known is located during the genetic search process, then all the steps are terminated; otherwise, go to Step 3.

4. Experiments

For proving the efficiency of the proposed aGA approach, two types of the SPPC are considered. The first type uses the directed graph and dataset taken from a conventional approach. In the second type, three cases of the SPPC are considered. The directed graphs in each type are newly made and the datasets are randomly generated in this paper. By using the two types of the SPPC, various comparative analyses are performed between conventional approaches and the proposed aGA approach.

For experimental comparison under the same conditions, all the approaches are run on an IBM-compatible PC with a Pentium 4 CPU 3.2GHz processor, 2GB RAM, and the Windows-XP operating system. Especially, the approaches considered in the second type are programmed in Visual Basic Ver. 6.0. The parameters in the second type are set as follows. The total number of generations is 2,000, the population size is 20, and the crossover rate is 0.5. Altogether 20 iterations are executed to reduce the randomness in running each approach. <Table 2> shows the measures of performance for comparing each approach.

<Table 2> Performance Measures

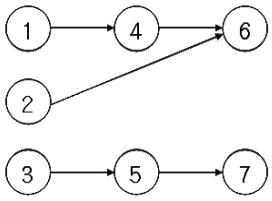
Measure	Description
CPU time	Average CPU time (in sec.)
Best value	Sum of the traveling times in the feasible sequence with minimum traveling time among all feasible sequences
Best sequence	Feasible sequence representing the best value

In <Table 2>, the CPU time is averaged over 20 iterations. The best value and the best sequence mean the best result when each approach has completed a predefined total numbers of iterations (in our case, 20 iterations).

4.1 Test Problem 1 (T-1)

For T-1, the directed graph with precedence

constraints and the corresponding traveling times used in He and Kusiak (1992) are considered, each of which is shown in <Figure 10> and <Table 3>, respectively. Using the dataset from <Figure 10> and <Table 3>, the proposed aGA approach and two conventional approaches (He and Kusiak's approach (1992) and CPLEX approach (2008)) are implemented. The computational results are shown in <Table 4>.



<Figure 10> A Directed Graph with Precedence Constraints

<Table 3> Traveling Time between Each Pair of Nodes

		Node number						
		1	2	3	4	5	6	7
Node number	1		8	4	6	2	5	4
	2	10		9	5	2	6	13
	3	5	13		11	4	9	10
	4	5	2	7		5	8	4
	5	8	5	4	6		3	6
	6	13	5	4	8	4		10
	7	2	11	5	10	8	9	

<Table 4> Computational Results for T-1

Approaches	Best value	Best sequence
He and Kusiak's (1992) approach	31	1-3-5-2-4-7-6
CPLEX approach (2008)	26	3-5-7-1-4-2-6
Proposed aGA approach	26	3-5-7-1-4-2-6

In <Table 4>, He and Kusiak's approach is the worst performer. However, the proposed aGA approach locates the optimal solution with the minimum total traveling time of 26, since the CPLEX approach has been known to be very effective in locating optimal solution and its best value is the same with that obtained by the proposed aGA approach. Therefore, we can confirm that the proposed aGA approach can be used as a useful tool for solving the SPPC.

4.2 Test Problem 2 (T-2)

Four cases of the SPPC are considered in *T-2* and they consist of 7, 20, 40, and 60 nodes, respectively. For the SPPC with 7 nodes, we use the same directed graph and the same corresponding traveling times with <Figure 11> and <Table 3>, respectively. For the SPPCs with 20, 40, and 60 nodes, the directed graphs with precedence constraints and the corresponding traveling times appear in Appendices 1 through 4.

For various comparisons, the GA approach with the priority-based representation procedure (Gen et al., 2006), the GA approach with the TS-based representation procedure (Yun and Moon, 2011), and the proposed aGA approach are compared with each other using the measures of performance defined in <Table 2>. <Table 5> shows the performance results of each approach.

In the SPPCs with 7 and 20 nodes in <Table 5>, all the approaches yield the same solutions in terms of the best value. However, the search speed of the proposed aGA approach is slightly quicker than those of the GA approaches with the priority-based and TS-based representation procedures.

<Table 5> Performance of Each Approach

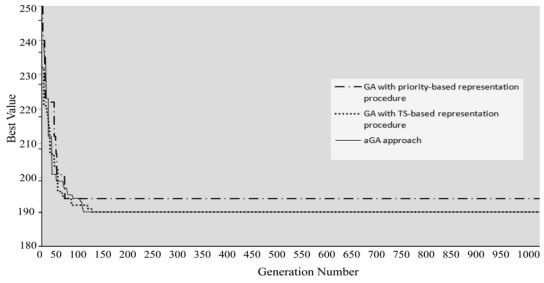
				Number of Nodes	
		7	20	40	60
GA with the priority-based representation procedure (Gen et al., 2006)	CPU time(sec.)	1.02	2.89	5.22	7.45
	Best value	26	98	194	329
	Best sequence	3-5-7-1-4-2-6	1-3-2-6-7-4-5-8-9-11-14-13-10-12-15-17-16-19-18-20	1-3-2-6-4-7-5-10-12-11-9-8-13-15-16-17-14-20-19-18-22-21-24-25-23-26-27-30-31-29-28-32-35-33-36-34-39-37-38-40	1-3-2-4-7-8-6-9-5-11-12-13-10-15-16-17-18-14-22-21-23-20-19-24-25-26-28-27-30-31-32-33-34-29-37-38-36-35-42-41-43-40-39-44-48-46-49-45-47-52-54-51-50-53-58-57-55-59-56-60
GA with the TS-based representation procedure (Yun and Moon, 2011)	CPU time(sec.)	1.01	2.88	4.99	7.23
	Best value	26	98	190	319
	Best sequence	3-5-7-1-4-2-6	1-3-2-6-7-4-5-8-9-11-14-13-10-12-15-17-16-19-18-20	1-3-2-6-5-4-7-10-12-9-11-8-13-15-16-17-14-20-19-18-22-21-24-25-23-26-27-30-31-29-28-32-35-33-36-34-39-37-38-40	1-3-2-4-7-8-6-9-5-11-12-13-10-14-15-16-17-18-22-21-23-20-19-24-25-26-28-27-30-31-32-33-34-29-37-38-36-35-42-41-43-40-39-44-48-46-49-45-47-52-54-51-50-53-58-57-55-59-56-60
Proposed aGA approach	CPU time(sec.)	0.95	2.87	4.91	6.92
	Best value	26	98	190	316
	Best sequence	3-5-7-1-4-2-6	1-3-2-6-7-4-5-8-9-11-14-13-10-12-15-17-16-19-18-20	1-3-2-6-5-4-7-10-12-9-11-8-13-15-16-17-14-20-19-18-22-21-24-25-23-26-27-30-31-29-28-32-35-33-36-34-39-37-38-40	1-3-2-4-7-5-8-6-9-11-12-13-10-14-15-16-17-18-22-21-23-20-19-24-26-28-27-25-31-30-32-33-34-29-37-38-36-35-42-41-43-40-39-44-48-46-49-45-47-52-54-51-50-53-58-57-55-59-56-60

In the SPPC with 40 nodes, the best values of the aGA approach and the GA approach with the TS-based representation procedure have the same value of 190 and their performances are superior to that of the GA approach with the priority-based representation procedure, which means that the search quality using the TS-based representation procedure is better than that using the priority-based representation procedure. In terms of the CPU time, the proposed aGA approach is the quickest, which implies that the adaptive scheme used in the proposed aGA approach well regulates the most desirable degree of exploitation/exploration during its search process.

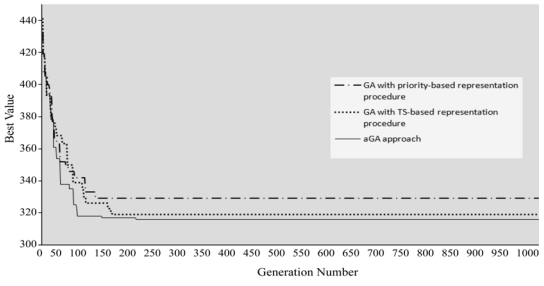
In the SPPC with 60 nodes, the proposed aGA approach is the best performer and the GA approach with the priority-based representation procedure is the worst performer in terms of the best value. A similar result is also shown in terms of the CPU time, that is, the proposed aGA approach is the quickest and the GA approach with the priority-based representation procedure is the slowest. This result implies that the TS-based representation procedure is more efficient in producing feasible sequences for the SPPCs with 60 no-

des than the priority-based representation procedure. Especially, in the comparison between the proposed aGA approach and the GA approach with the TS-based representation procedure, the former outperforms the latter in terms of the best value and the CPU time, though they all use the TS-based representation procedure. This result means that the GA with the adaptive scheme is more efficient than the GA without such a scheme in terms of the search speed and the ability to locate the best solution.

For a more detailed comparison of each approach, <Figures 11>~ <Figure 12> show the convergence process of each approach in the SPPCs with 40 and 60 nodes, until they reach the generation number of 1,000.



<Figure 11> Convergence Process in the SPPC with 40 Nodes



<Figure 12> Convergence Process in the SPPC with 60 Nodes

In <Figure 11>, both the priority-based representation procedure and the TS-based representation procedure show varying and fast convergence processes during the initial generations. However, when the generation number reaches about 40, the TS-based representation procedure has a better fitness value than the priority-based representation procedure. In the comparison between the proposed aGA approach and the GA approach with the TS-based representation procedure, the performance of the former is more efficient than that of the latter. A similar result is also shown in <Figure 12>. The performance of the proposed aGA approach and the GA approach with the TS-based representation procedure is significantly better than that of the GA approach with the priority-based representation procedure.

By the results analyzed using <Table 5>, <Figures 11>~<Figures 12>, we can reach some conclusions as follows;

i) The TS-based representation procedure used in Yun and Moon (2011) and the proposed aGA approach is more efficient in search quality than the priority-based representation procedure used in Gen et al. (2006) as the node sizes of the SPPCs are increased, which means that the former well locates the feasible sequences in the whole search spaces of the SPPCs rather than the latter does.

The major difference between the priority-based representation procedure and

the TS-based representation procedure is that the former uses the priorities randomly generated in each node to produce a feasible sequence, and thus the various types of feasible sequences may not be produced because of the priority constraint assigned to each node, but the latter does not use any priority and it uses the directed graph with precedence constraints to produce a feasible sequence. By these properties of the priority-based representation procedure and the TS- based representation procedure, we can conclude that the latter outperforms the former.

ii) In the comparison between Yun and Moon (2011) and the proposed aGA approach, the former uses the GA approach alone, but the latter adapts both GA approach and the adaptive scheme using FLC, in order to increase search quality within whole search space. Therefore, the rate of crossover operator used in the former remains fixed, but the rate in the latter is adaptively regulated, during genetic search process. In general, most of the adaptive schemes used in GA approach can well regulate the most desirable degree of exploitation/ exploration during its search process. Moreover, many studies have already shown that the GA approaches with adaptive schemes are more efficient in search quality than the GA approach without it (Song et al., 1997; Subbu et al., 1998; Gen and Cheng, 2000; Cheong and Lai, 2000). This mirrors that the proposed aGA approach outperforms the GA approach used in Yun and Moon (2011).

5. Conclusion

In this paper, we have proposed a new aGA approach with the TS-based representation procedure for solving various types of the SPPC. The TS-based representation procedure used in the proposed aGA approach can easily produce a set of feasible sequences in various types of the SPPC. Therefore, the proposed aGA approach locates the optimal sequence with the minimal total traveling time among all feasible sequences.

For various experimental comparisons using the SPPCs with 7, 20, 40, and 60 nodes, four types of conventional approaches (the CPLEX approach, He and Kusiak's approach, the GA with a priority-based representation procedure, and the GA with TS-based representation procedure) have also been presented; their performances have been compared with that of the proposed aGA approach. As a result, we can reach the following conclusions.

i) As the number of nodes in the SPPC increases, the production of feasible sequences by the TS-based representation procedure is more efficient than that by the priority-based representation procedure.

ii) The GA approach with adaptive scheme

can reinforce search speed and quality rather than the GA approach without it.

For our future study, larger-sized SPPCs will be used for comparing the performances in the proposed aGA approach and conventional GA-related approaches.

References

Baker, K., Introduction to sequencing and scheduling, John Wiley and Sons, (1974).

Chen, C., "AND/OR precedence constraint traveling salesman problem and its application to assembly schedule generation", *Proceedings of 1990 IEEE International Conference on Systems*, Man and Cybernetics, (1990), 560~562.

Cheong, F. and R. Lai, "Constraining the optimization of a fuzzy logic controller using an enhanced genetic algorithm", *IEEE Transactions on Systems, Man, and Cybernetics-Part B : Cybernetics*, Vol.30, No.1(2000), 31~46.

Duman, E. and I. Or, "Precedence constrained TSP arising printed circuit board assembly", *International Journal of Production Research*, Vol. 42, No.1(2004), 67~78.

Gen, M. and R. Cheng, Genetic algorithms and engineering design, John Wiley and Son, (1997).

Gen, M. and R. Cheng, Genetic algorithms and engineering optimization, John Wiley and Sons, (2000).

Gen, M., F. Altiparmak and L. Lin, "A genetic algorithm for two-stage transportation problem using priority-based encoding", *OR Spectrum*, Vol.28(2006), 337~354.

He, W. and A. Kusiak, "Scheduling manufacturing systems", *Computers in Industry*, Vol.20(1992), 163~175.

Hong, T. P., H. S. Wang, W. Y. Lin and W. Y. Lee, "Evolution of appropriate crossover and mutation operators in a genetic process", *Applied Intelligence*, Vol.16(2002), 7~17.

Lambert, A. J. D., "Exact methods in optimum disassembly sequence search for problems subject to sequence dependent costs", *Omega*, Vol.34(2006), 538~549.

Mak, K. L., Y. S. Wong and X. X. Wang, "An adaptive genetic algorithm for manufacturing cell formation", *International Journal of Manufacturing Technology*, Vol.16(2000), 491~497.

Moon, C., J. Kim, G. Choi And T. Seo, An efficient genetic algorithm for the traveling salesman problem with precedence constraints", *European Journal of Operational Research*, Vol. 140, No.3(2002), 606~617.

Moon, C. and Y. Seo, "Advanced planning for minimizing makespan with load balancing in multi-plants chain", *International Journal of Production Research*, Vol.43, No.20(2005), 4381~4396.

Pinedo, M. and X. Chao, Operations scheduling, McGraw-Hill, (1999).

Renaud, J., F. F. Boctor and J. Ouenniche, "A heuristic for the pickup and delivery traveling salesman problem", *Computers and Operations Research*, Vol.27(2000), 905~916.

Savelsbergh, M. and M. Sol, "The general pickup and delivery problem", *Transportation Science*, Vol.29(1995), 17~29.

Srinivas, M. and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic

algorithms", *IEEE Transaction on Systems*, Man and Cybernetics, Vol.24, No.4(1994), 656~667.

Song, Y. H., G. S. Wang, P. T. Wang and A. T. Johns, "Environmental/economic dispatch using fuzzy logic controlled genetic algorithms", *IEEE Proceedings on Generation*, Transmission and Distribution, Vol.144, No.4(1997), 377~382.

Su, Q., "Applying case-based reasoning in assembly sequence planning", *International Journal of Production Research*, Vol.45, No.1(2007), 29~47.

Subbu, R., A. C. Sanderson and P. P. Bonissone, "Fuzzy logic controlled genetic algorithms versus tuned genetic algorithms : an agile manufacturing application", *Proceedings of the 1999 IEEE International Symposium on Intelligent Control(ISIC)*, (1988), 434~440.

Wang, P. T., G. S. Wang and Z. G. Hu, "Speeding up the search process of genetic algorithm by fuzzy logic", *Proceedings of the 5th Euro pean Congress on Intelligent Techniques and Soft Computing*, (1997), 665~671.

Wu, Q. H., Y. J. Cao and J. Y. Wen, "Optimal reactive power dispatch using an adaptive genetic algorithm", *Electrical Power and Energy Systems*, Vol.20, No.8(1998), 563~569.

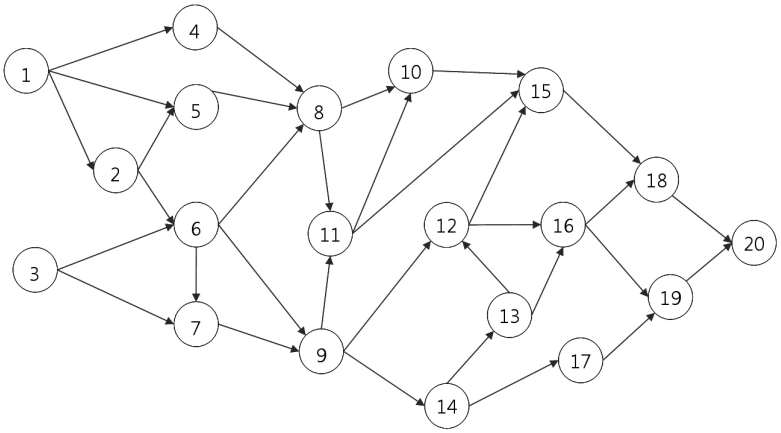
Yun, Y. S., "Genetic algorithm with fuzzy logic controller for preemptive and non-preemptive job shop scheduling problems", *Computers and Industrial Engineering*, Vol.43, No.3 (2002), 623~644.

Yun, Y. S., M. Gen and S. L. Seo, "Various hybrid methods based on genetic algorithm with fuzzy logic controller", *Journal of Intelligent Manufacturing*, Vol.14(3-4)(2003), 401~ 419.

Yun, Y. S. and C. Moon, "Genetic algorithm ap proach for precedence-constrained sequencing problems", appeared in *Journal of Intelligent Manufacturing*, (2011).

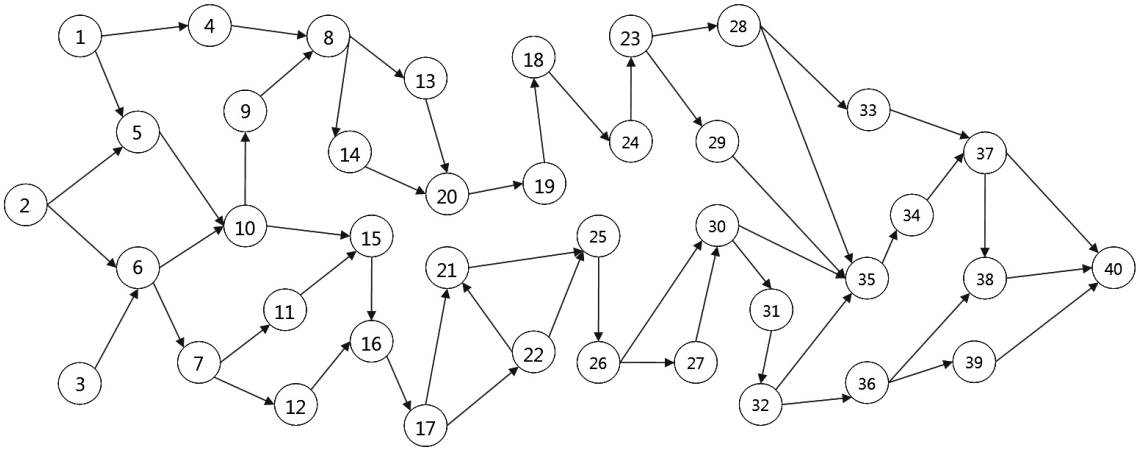
CPLEX optimization method, (2008). http://www.ilog.com/products/cplex/product/algorithms.cfm.

<Appendix 1>



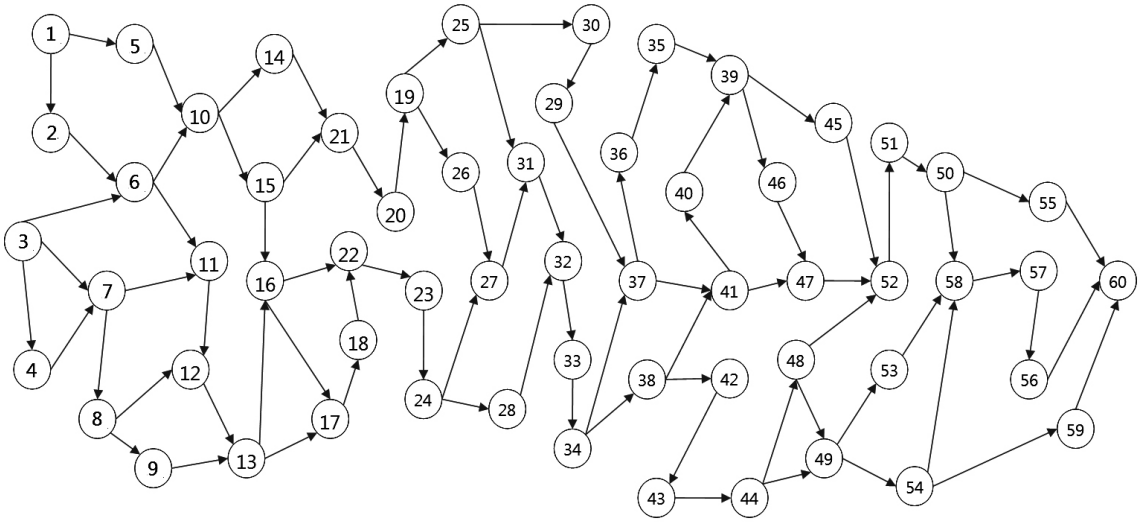
<Figure 1> SPPC with 20 Nodes

<Appendix 2>



<Figure 2> SPPC with 40 Nodes

<Appendix 3>



<Figure 3> SPPC with 60 Nodes

<Appendix 4>

<Figure 4> Traveling Time between Each Pair of Nodes

Abstract

선행제약순서결정문제 해결을 위한 퍼지로직제어를 가진 적응형 유전알고리즘

윤영수

본 논문에서는 선행제약순서결정문제(Sequencing problem with precedence constraints, SPPC)를 효과적으로 해결하기 위한 적응형 유전알고리즘(Adaptive genetic algorithm, aGA)을 제안한다. aGA에서는 SPPC를 효과적으로 표현하기 위해 위상정렬에 기초한 표현절차(topological sort-based representation procedure) 를 사용한다. 제안된 aGA는 퍼지로직제어를 이용한 적응형구조를 가지고 있으며, 유전탐색과정을 통해 교차변이 연산자(Crossover operator)의 비율을 적응적으로 조절한다. 수치예제에서는 다양한 형태의 SPPC를 제시하였으며, 그 실험결과는 제안된 aGA가 기존의 알고리즘보다 우수함을 보여주었다. 결론적으로 말하자면 본 논문에서는 제안된 aGA가 다양한 형태의 SPPC에서 최적해 혹은 최적순서를 발견하는데 아주 효과적이라는 것을 밝혔다.

Keywords : 적응형 유전알고리즘, 선행제약순서결정문제, 위상정렬에 기초한 표현절차, 퍼지로직제어

* 조선대학교 경상대학 경영학부

저 자 소 개

윤영수

대구대학교 산업공학과 학사, 건국대학교 산업공학과 석사・박사, 일본 와세다대학교 지능정보시스템대학원 박사, 현재 조선대학교 경영학부 조교수로 재직중이다. 주요 연구관심분야는 유전알고리즘을 이용한 생산시스템 최적화, 공급망관리(Supply Chain Management)의 최적화 등이다.