「정보시스템연구」 제25권 제4호
한국정보시스템학회
2016년 12월, pp. 105～119

https://doi.org/10.5859/KAIS.2016.25.4.105

# The Decoding Approaches of Genetic Algorithm for Job Shop Scheduling Problem*

Kim Jun Woo**

〈Table of Contents〉



## Ⅰ. Introduction

Typically, a production procedure consists of several operations, and the goal of scheduling is to determine the optimal start and finish times for each operation. The optimal production schedule is of great importance for efficient production in most manufacturing and service industries, and the scheduling problems have been paid much attention during the last decades (Graves, 1981; Framinan & Ruiz, 2010; Kim & Park, 2010; Hadidi et al., 2012; Lee et al., 2016). Moreover, scheduling is one of the essential parts of modern information systems such as ERP(Enterprise Resource Planning) and MES(Manufacturing Execution System) (Hong et al., 2015).

However, many scheduling problems are modelled as combinatorial optimization problems, and this led the deterministic and numerical methods generally fail to solve such problems in polynomial time (Jeong & Kim, 2016). Instead, the non-deterministic methods such as heuristic and meta heuristic approaches have been practical methods for solving the scheduling problems (Colorni et al., 1996; Mosheiov & Oron, 2004; Gen et al., 2009;

* This paper was supported by Dong-A University URP (University Research Park) in URP projects of Busan metropolitan city.
** Department of Industrial and Management Systems Engineering, Dong-A University, kjunwoo@dau.ac.kr

Jarboui et al., 2013; Kim, 2015).

Almost meta heuristic search methods have common in that they generate some feasible solutions for given problems and evaluate them in order to explore the search space and find the optimal or nearly optimal solutions. Moreover, it is well known that the optimal solution of a production scheduling problem is included within the set of some feasible alternatives called active schedules (Giffler & Thompson, 1960). Hence, many meta heuristic methods for scheduling problems generate and evaluate the active schedules in search procedures (Lee & Salim, 2006). However, the computational cost for generating the active schedules of complex scheduling problems can be high, and this paper aims to compare the performances of the active schedule based method and semi active schedule based method. The set of the semi active schedules is a superset of the set of the active schedules, and the performances of the semi active schedules are generally worse than active ones. However, the semi active schedules are easier to generate, and this is an important advantage over the active schedules.

Especially, this paper considers the genetic algorithm (GA) for solving the job shop scheduling problem (JSP). The GA is one of the well known meta heuristic search method, developed by mimicking the natural process of evolution (Holland, 1975). In order to find the optimal solution of the combinatorial optimization problems such as scheduling problems, a potential solution (individual) must be encoded as a simple string called chromosome, which consists of several characters called genes. Then, the GA maintains a population of individuals during the search procedure, and the population is iteratively evolved by applying the genetic operators. During the last decades, the GA has become one of the popular tools for scheduling tasks (Giovanni & Pezzella, 2010; Zhang et al., 2011; Kim, 2015).

The JSP is a well known scheduling task, characterized by $n$ jobs and $m$ machines. A job consists of $m$ operations to be processed by different machines, and the operations of an individual job must be processed in a specific order. Typically, the goal of the JSP is to determine the start and finish time of each operation that minimize the makespan (Cheng et al., 1996; Gen et al., 2009; Kammer et al., 2011). In this paper, a solution of JSP is encoded as a permutation of all operations, which denotes the ‘assignment order’ of the operations. The permutation encoding is widely used in GAs for sequencing problems, and several genetic operators are available for this encoding scheme (Starkweather et al., 1991; Poon & Carter, 1995). What is important is that the permutation encoding for the JSP can be decoded into both active and semi active schedules easily, and this enables to compare the performances of the active

schedule based GA and the semi active based GA.

The remainder of this paper is organized as follows: Section 2 explains the differences among the schedules for a JSP in more detail. Section 3 introduces the permutation encoding based GA for the JSP, and decoding procedures for generating active and semi active schedules are outlined. The experiment results are provided in Section 4, and discussions and concluding remarks follow in Section 5.

## Ⅱ. Feasible Schedules for Job Shop Scheduling Problem

Table 1 shows an example of 3 by 3 JSP ($n$=3, $m$=3), where each cell denotes the processing time and machine number for the $j$ th operation of the $i$th job. A schedule for JSP is feasible if and only if no machine simultaneously processes two or more operations and the precedence constraints of the operations in all jobs are satisfied. Typically, the feasible schedules of JSP are represented by the Gantt chart as shown in Fig.1~Fig.4.

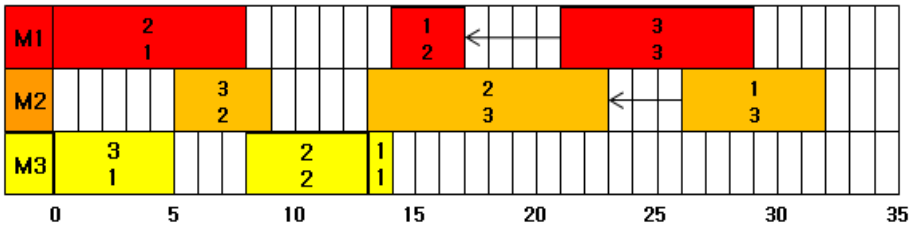
A feasible schedule specifies the processing sequences of the machines, which are the sequences of the jobs that visit the machines. For example, Fig. 1 shows a feasible sequence of the JSP in Table 1, and the processing sequences of the machine 1, 2 and 3 are '2-1-3', '3-2-1' and '3-2-1', respectively. A feasible schedule is non-active if any operation can be started sooner without changing any processing sequence, and the schedule in Fig. 1 is non-active since the two operations, (3, 3) and (1, 3) can move to left. Moreover, it is obvious that the non-active schedules are not recommendable since their performances can be enhanced by simply moving some operations such as (3, 3) and (1, 3). Note that the makespan of the non-active schedule in Fig. 1 is 32.

<Table 1> An Example of 3 by 3 JSP

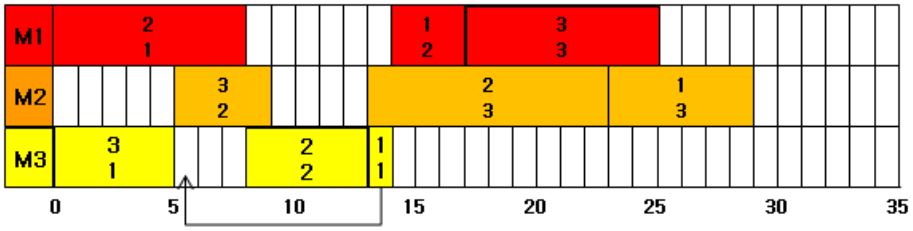
| $i$ \ $j$ | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 1, 3 | 3, 1 | 6, 2 |
| 2 | 8, 1 | 5, 3 | 10, 2 |
| 3 | 5, 3 | 4, 2 | 8, 1 |

Next, a feasible schedule is semi active if no operations can move to left without 'jumping over' other operations. For example, if the operations (3, 3) and (1, 3) in Fig. 1 move to the earliest possible position without changing any processing sequence, we obtain a new feasible schedule as shown in Fig. 2. This is a semi active schedule and the makespan is 29, which is better than the non-active schedule. That is, the semi active schedules are more suitable for the scheduling purpose, however, some operations still can move to left by

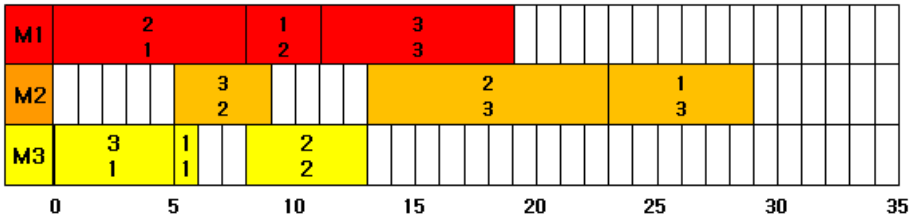
jumping over other ones. In Fig. 2, the operation (1, 1) can move to left without violating the precedence constraints in the given jobs, which results in a new feasible schedule in Fig. 3.



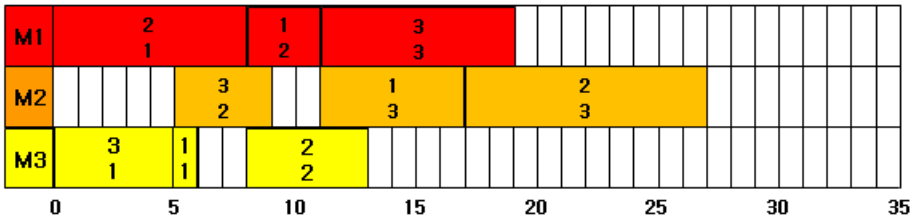

<Fig. 1> A Non-active Schedule for JSP




<Fig. 2> A Semi Active Schedule for JSP




<Fig. 3> An Active Schedule for JSP




<Fig. 4> An Active Schedule with Shorter Makespan

Fig. 3 shows an active schedule, where no operations can move to left without modifying the positions of other operations. Note that the operation (1, 1) has moved to the earliest position, and this enabled the operations (1, 2) and (3, 3) to move left, too. Consequently, the processing sequence of the machine 3 have been modified as '3-1-2'.

The set of all active schedules is a subset of the set of all semi active schedules, and it is well known that the optimal schedules are included in the set of the active schedules. In this context, most meta heuristic methods for solving the scheduling problems focus on the active schedules during the search procedures. However, the makespan of the active schedule in Fig. 3 is the same with the semi active schedule in Fig. 2. That is, the jumping over does not always produce better schedules.

In addition, Fig. 4 shows another active schedule for the JSP in Table 1, where the processing sequence of the machine 2 is '3-1-2'. The makespan of this schedule, 27, is better than the makespan of schedules in Fig. 2 and Fig. 3, and we can see that the active schedules might produce better results than the semi active schedules. However, the active schedule in Fig. 4 can not be generated from the semi active schedule in Fig. 2 by simple jumping over. Indeed, generating the active schedule can be complex and involves high computational cost. On the contrary, the semi active schedules can be obtained more easily and they might produce good results during the search procedure. In this context, this paper aims to compare the performances of the active schedule based method and the semi active schedule based method for solving scheduling problem.

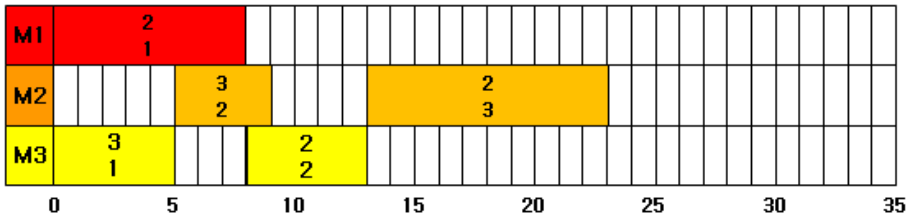
## Ⅲ. Genetic Algorithm for Job Shop Scheduling Problem

Here, the author of this paper explains the elements of the GA for solving JSP investigated in this paper. In order to solve a combinatorial optimization problem by using GA, a potential solution must be represented as a simple string, and this paper uses the permutation encoding scheme for representation. Although the permutation with repetition encoding has been widely used in GAs for JSP (Cheng et al., 1996), the permutation encoding scheme has an important advantage that the traditional genetic operators for sequencing problems can be used. The permutation encoding specifies the order in which the operations are scheduled, and we denote the $j$th operation of the $i$th job as the element $k(=(i-1)\times n+j)$ for efficiency. That is, the JSP in Table 1 contains 9 elements to be sequenced in order to create a solution.

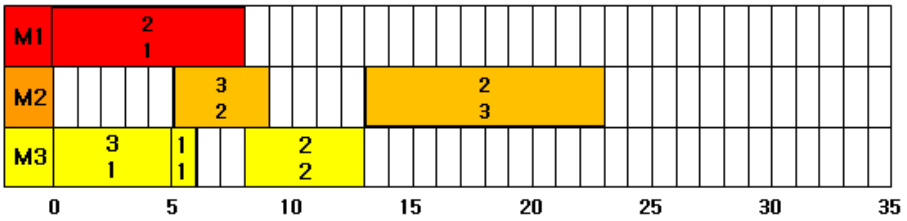
An encoded solution can be decoded into a feasible schedule for JSP by scheduling an element in sequence. For example, let's

consider an encoded solution ‘4-7-8-5-6-1-2-3-9’. It is straightforward that each element should start at earlier time in order to obtain better schedules, hence, the first five elements will be scheduled as shown in Fig. 5.

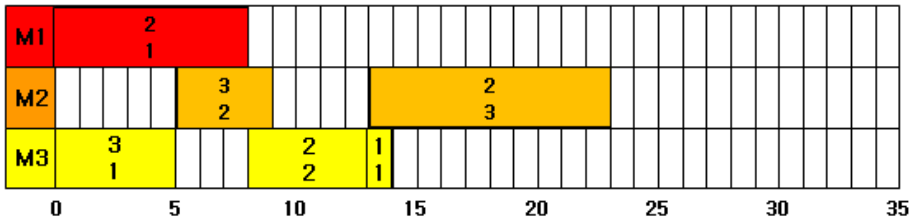
Now, the element 1, the sixth one in the illustrative solution, should be scheduled, and there are two different approaches for scheduling this element. First, the elements can be scheduled in greedy manner by allowing them to jump over other elements previously scheduled. This greedy decoding can be used to produce the active schedules associated with the encoded solutions, and the element 1 is scheduled as shown in Fig. 6 by this approach. However, the schedules of the previous elements must be investigated to schedule an individual element, and this can cause significant computational cost for the large scheduling problems. On the contrary, the elements can be scheduled in FIFO(first-in, first-out) manner where an element can not




<Fig. 5> Gantt Chart for the First Five Elements




<Fig. 6> Element 1 Scheduled in Greedy Manner




<Fig. 7> Element 1 Scheduled in FIFO Manner

start earlier than the other ones previously scheduled, as shown in Fig. 7. Although this FIFO decoding produces the semi active schedules, which might be worse than the active ones, the computational cost of this approach is quite lower than the greedy decoding.

<Fig. 8> Greedy Decoding Procedure

```
For k=1 To m×n
  Set T_j=max_{e∈pj_p(k)}(ft_p(e))
  Set T_m=max_{e∈pm_p(k)}(ft_p(e))

  If T_j ≥ T_m Then
    Set st_p(ES_p(k))=T_j
    Set ft_p(ES_p(k))=st_p(ES_p(k))+t(ES_p(k))
  Else
    For l=1 To n(pm_p(k))
      Set Current_Predecessor=lth element of pm_p(k)
      If T_j ≤ ft_p(Current_Predecessor) Then
        If ES_p(k) can start at ft_p(Current_Predecessor) Then
          Set st_p(ES_p(k))=ft_p(Current_Predecessor)
          Set ft_p(ES_p(k))=st_p(ES_p(k))+t(ES_p(k))
        End If
      End If
    Next l
  End If
Next k
```

<Fig. 9> FIFO Decoding Procedure

```
For k=1 To m×n
  Set T_j=max_{e∈pj_p(k)}(ft_p(e))
  Set T_m=max_{e∈pm_p(k)}(ft_p(e))
  Set st_p(ES_p(k))=max(T_j, T_m)
  Set ft_p(ES_p(k))=st_p(ES_p(k))+t(ES_p(k))
Next k
```

The procedures of the two decoding approaches of GA for JSP are specified in Fig. 8 and Fig. 9, where following notations are used for simplicity. Moreover, we can see that the greedy decoding procedure in Fig. 8 contains a double loop and more complex steps than the FIFO decoding procedure in Fig. 9.

$ES_p$ : the $p$th encoded solution in the population
$ES_p(k)$ : the $k$th elements of the $ES_p$
$m(e)$ : the designated machine of the element $e$
$j(e)$ : the job index of the element $e$
$t(e)$ : the processing time of the element $e$
$st_p(e)$ : the start time of the element $e$ in the $ES_p$
$ft_p(e)$ : the finish time of the element $e$ in the $ES_p$
$pj_p(k)$ : the set of $ES_p(k)$'s predecessors included in the job $j(ES_p(k))$
$pm_p(k)$ : the set of $ES_p(k)$'s predecessors processed by the machine $m(ES_p(k))$

In order to use decoding schemes explained above, the encoded solution must satisfy the precedence constraints between the operation elements. For example, the elements 1, 2 and 3 are contained in the job 1, hence, the element 1 must precede the element 2 and 3 in the encoded solution. Indeed, the JSPs can be viewed as the precedence-constrained sequencing problems. Hence, the initial solutions can be easily obtained by applying the topological sort (Kozen, 1992; Kim, 2016), and PPX (precedence preserving crossover) operator (Bierwirth et al., 1996) and random swapping mutation operator are adopted in this paper.

## Ⅵ. Experiment Results

Table 2 shows a 5 by 5 JSP, the benchmark problem of this paper. For comparison, both active schedule based and semi active schedule based GAs have been applied to the benchmark problem under crossover rate=0.8, mutation rate=0.01 and maximum iteration number=150 with varying the size of the population and the number of elites to be preserved in each generation. The experiment results are represented in Fig. 10~Fig. 12.

<Table 2> A 5 by 5 JSP

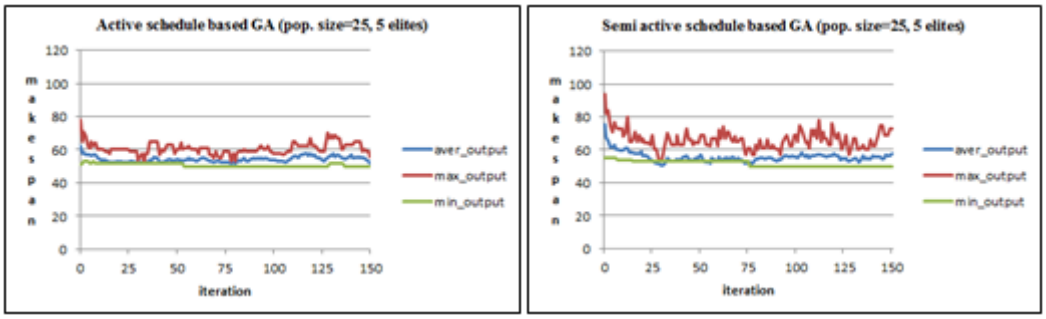
| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 1, 3 | 3, 1 | 6, 2 | 7, 4 | 3, 5 |
| 2 | 8, 2 | 5, 3 | 10, 5 | 10, 4 | 10, 1 |
| 3 | 5, 3 | 4, 4 | 8, 5 | 9, 1 | 1, 2 |
| 4 | 5, 2 | 5, 1 | 5, 3 | 3, 4 | 8, 5 |
| 5 | 9, 3 | 3, 2 | 5, 5 | 4, 4 | 3, 1 |

First, we can see that both GAs successfully discovered the optimal schedules with minimized makespan 50. Although the active schedule based GA tend to discover the optimal schedule in fewer iterations than the semi active schedule based GA, the difference is not significant. Furthermore, the two GAs

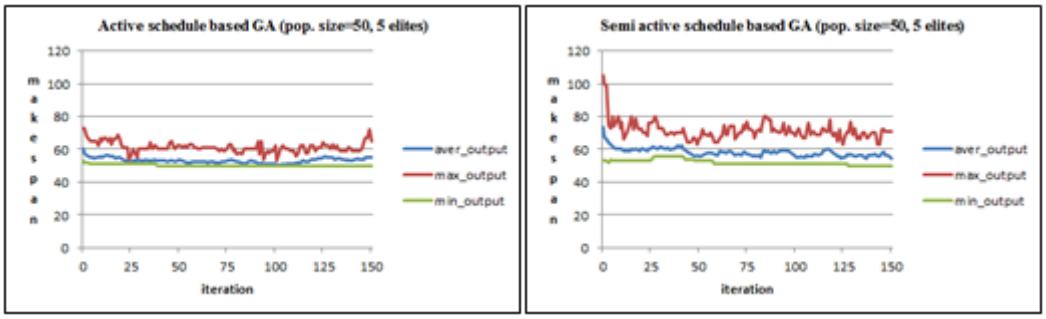
require similar numbers of iterations to find the optimal solution if the size of the population is large as shown in Fig. 12.

On the contrary, the values of the average and maximum makespans of the semi active schedule based GA are generally worse. In other words, if we aim to obtain a good schedule from a single decoded solution, the decoding approach for generating active schedules is more suitable. However, the semi active schedule based GA also succeeded in finding the optimal solutions, which indicates that the alternative decoding approach for generating semi active schedules can be useful in exploring the search space if the iterative search strategy such as GA is adopted.

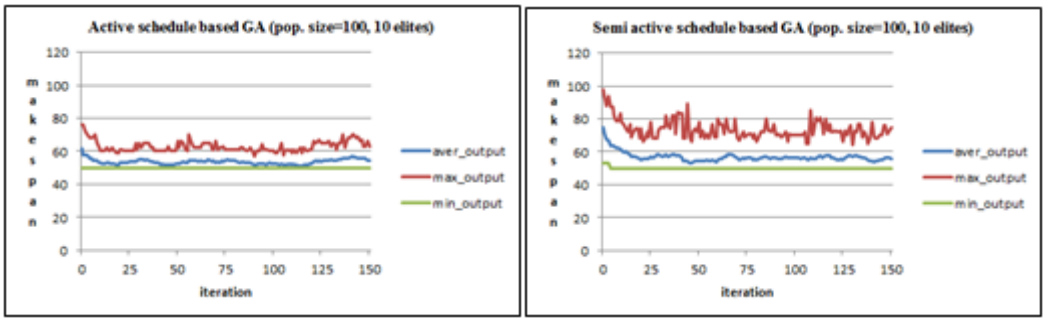
Next, the execution times of the two decoding methods are summarized in Table 3. Note that the experimental GAs were programmed in MicroSoft Visual Basic 6.0 and tested on a 3.40GHz Core i7 PC. In addition, the running times of the other genetic operators including selection, crossover and mutation are not considered in Table 3. We can see that the execution times of the FIFO decoding are significantly shorter than the greedy encoding. In more detial, the execution times of the greedy encoding are about 2.5 times as long as the execution times of the FIFO encoding. As shown in Fig. 8 and Fig. 9, the semi active schedules can be constructed more easily than the active ones, and this enables the semi active schedule based GA to explore the search space more rapidly.




<Fig. 10> Experiment Results for Population Size=25 and 5 Elites




<Fig. 11> Experiment Results for Population Size=50 and 5 Elites




<Fig. 12> Experiment Results for Population Size=100 and 5 Elites

As a result, we can conclude that if an effective search strategy and an efficient decoding method exists, the semi active schedules also can be useful to search the good solutions of JSP, although most conventional search methods for solving scheduling problems have focused on the active schedules during their search procedures.

<Table 3> Comparison of Execution Time (sec.)

| Population size | Greedy decoding | FIFO decoding |
|---|---|---|
| 25 | 3 | 1 |
| 50 | 5 | 2 |
| 75 | 8 | 3 |
| 100 | 11 | 4 |

## Ⅴ. Discussions and Concluding Remarks

In order to solve a hard combinatorial optimization problem by using the meta heuristic search methods such as GA, the potential solutions generally need to be encoded as simple representations. During the search procedure, these encoded solutions are decoded in order to create the candidate schedules to be investigated.

Since the optimal solutions of scheduling problems are included in the set of the active schedules, the encoded solutions are typically decoded into the active schedules in many algorithms for such problems. However, the computational cost for generating the active schedules of complex scheduling problems can be high. On the contrary, the semi active schedules can be generated more efficiently, though their performances are worse than or equal to those of the active schedules.

In this context, this paper aims to compare the performances of active schedule based and semi active schedule based GAs for traditional JSP. To this end, this paper introduces two decoding approaches for permutation encoding scheme, the greedy decoding and the FIFO decoding, and applies them to a 5 by 5

benchmark problem.

The important findings of this paper are as follows: (i) Compared to active schedule based GA, the semi active schedule based GA requires more iterations in order to find the optimal schedule of JSP. However, the difference is not significant. (ii) The execution time of semi active schedule decoding is quite shorter than active schedule decoding. (iii) The operations of semi active schedule decoding is easy to understand and implement.

Consequently, this paper concludes that semi active schedule based search methods also can be useful in exploring the search space of production scheduling problems if effective search strategies are given.

However, this paper has following limitations: (i) Only classical JSP and permutation encoding schemes are considered. (ii) The semi active schedule decoding is applied to a single meta heuristic search method, GA.

The author plans to develop various semi active schedule based search methods by incorporating the semi active schedule decoding into the meta heuristic search methods such as simulated annealing and tabu search, and apply them to a wide range of production scheduling problems in order to evaluate the usefulness of semi active schedules.

## References

Bierwirth, C., Mattfeld, D. C., and Kopfer, H., "On Permutation Representations for Sequencing Problem," In Parallel Problem Solving from Nature-PPSN IV, Springer Berlin Heidelberg, 1996, pp.310-318.

Cheng, R., Gen, M., and Tsujima, Y., "A Tutorial Survey of Job-shop Scheduling Problem using Genetic Algorithms – I. Representation," *Computers and Operations Research*, Vol.38, No.11, 2011, pp.1556-1561.

Colorni, A., Dorigo, M., Maffioli, F., Maniezzo, V., Righini, G., and Trubian, M., "Heuristics from Nature for Hard Combinatorial Optimization Problems," *International Transactions in Operational Research*, Vol.3, No.1, 1996, pp.1-21.

De Giovanni, L., and Pezzella, F., "An Improved Genetic Algorithm for the Distributed and Flexible Job-shop Scheduling Problem," *European Journal of Operational Research*, Vol.200, No.2, 2010, pp.395-408.

Framinan, J. M., and Ruiz, R., "Architecture of Manufacturing Scheduling Systems: Literature Review and an Integrated Proposal," *European Journal of Operational Research*, Vol.205, No.2, 2010, pp.237-246.

Gen, M., Lin, L., and Zhang, H., "Evolutionary Techniques for Optimization Problems

in Integrated Manufacturing System: State-of-the-art Survey," *Computers and Industrial Engineering*, Vol.56, No.3, 2009, pp.779-808.

Giffler, B., and Thompson, G. L., "Algorithms for Solving Production-Scheduling Problems," *Operations Research*, Vol.8, No.4, 1960, pp.487-503.

Graves, S. C., "A Review of Production Scheduling," *Operations Research*, Vol.29, No.4, 1981, pp.646-675.

Hadidi, L. A., Tl-Turki, U. M., and Rahim, A., "Integrated Models in Production Planning and Scheduling, Maintenance and Quality: A Review," *International Journal of Industrial and Systems Engineering*, Vol.10, No.1, 2012, pp.21-50.

Holland, J. H., Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.

Hong, S., Kim, N., and Lim, S., "A Longitudinal Case Study on Evaluating Operational Performance of Production Information Systems in Korean Small and Medium-Sized Manufacturers," *The Journal of Internet Electronic Commerce Research*, Vol.15, No.5, 2015, pp.59-74.

Jarboui, B., Siarry, P., and Teghem, J., Metaheuristics for Production Scheduling, John Wiley & Sons, 2013.

Jeong, S. W., and Kim, J. W., "Solving the Constrained Job Sequencing Problem using Candidate Order based Tabu Search," *The Journal of Information Systems*, Vol.25, No.1, 2016, pp.159-182.

Kammer, M., Akker, M. V. D., and Hoogeveen, H., "Identifying and Exploiting Commonalities for the Job-shop Scheduling Problem," *Computers and Operations Research*, Vol.38, No.11, 2011, pp.1556-1561.

Kim, J. W., "Developing a Job Shop Scheduling System through Integration of Graphic User Interface and Genetic Algorithm," *Multimedia Tools and Applications*, Vol.74, No.10, 2015, pp.3329-3343.

Kim, J. W., "Candidate Order based Genetic Algorithm (COGA) for Constrained Sequencing Problems," *International Journal of Industrial Engineering: Theory, Applications and Practice*, Vol.23, No.1, 2016, pp.1-12.

Kim, W.-D., and Park, Y.-T., "A Case Study on the Efficiency Improvement on Auto-Parts Painting Process Management by An RFID System," *The Journal of Information Systems*, Vol.19, No.4, 2010, pp.233-252.

Kozen, D., The Design and Anlysis of Algorithms, Springer, 1992.

Lee, H. P., and Salim, S., "Algorithms for Solving Production-Scheduling Problems," *Operations Research*, Vol.8, No.4, 1960, pp.487-503.

Lee, J., Kim, I., and Jeong, T., "Novel Optimal Controlling Algorithm for Real-time Integrated-control Smart Manufacturing

system," *Journal of the Korea Industrial Information Systems Research*, Vol.21, No.2, 2016, pp.1-10.

Mosheiov, G., and Oron, D., "A Note on the SPT Heuristic for Solving Scheduling Problems with Generalized Due Dates," *Computers and Operations Research*, Vol.31, No.5, 2004, pp.645-655.

Poon, P. W., and Carter, J. N., "Genetic Algorithm Crossover Operators for Ordering Applications," *Computers and Operations Research*, Vol.22, No.1, 1995, pp.135-147.

Starkweather, T., MacDaniel, S., Mathias, K. E., Whitley, L. D., and Whitley, C., "A Comparison of Genetic Sequencing Operations," Proceedings of the 4th International Conference on Genetic Algorithms, 1991, pp.69-76.

Zhang, G., Gao, L., and Shi, Y., "An Effective Genetic Algorithm for the Flexible Job-shop Scheduling Problem," *Expert Systems with Applications*, Vol.38, No.4, 2011, pp.3563-3573.

**Kim Jun Woo**

He is an assistant professor of the Department of the Industrial and Management Systems Engineering, Dong-A University. He received his Ph.D. degree in industrial and systems engineering from KAIST in 2009. His current research interests include data mining, artificial intelligence, combinatorial optimization and meta heuristic algorithms, etc.

<Abstract>

# The Decoding Approaches of Genetic Algorithm for Job Shop Scheduling Problem

Kim Jun Woo

**Purpose**

The conventional solution methods for production scheduling problems typically focus on the active schedules, which result in short makespans. However, the active schedules are more difficult to generate than the semi active schedules. In other words, semi active schedule based search strategy may help to reduce the computational costs associated with production scheduling. In this context, this paper aims to compare the performances of active schedule based and semi active schedule based search methods for production scheduling problems.

**Design/methodology/approach**

Two decoding approaches, active schedule decoding and semi active schedule decoding, are introduced in this paper, and they are used to implement genetic algorithms for classical job shop scheduling problem. The permutation representation is adopted by the genetic algorithms, and the decoding approaches are used to obtain a feasible schedule from a sequence of given operations.

**Findings**

The semi active schedule based genetic algorithm requires slightly more iterations in order to find the optimal schedule, while its execution time is quite shorter than active schedule based genetic algorithm. Moreover, the operations of semi active schedule decoding is easy to understand and implement. Consequently, this paper concludes that semi active schedule based search methods also can be useful if effective search strategies are given.

***Keywords:*** job shop scheduling problem, genetic algorithm, decoding, active schedule, semi active schedule

<국문초록>

# Job Shop 일정계획 문제 풀이를 위한 유전 알고리즘의 복호화 방법

김 준 우

**Purpose**

생산 일정계획 문제의 해법들은 일반적으로 총처리시간이 짧은 active 스케줄에 초점을 맞추어 해를 탐색하는 경우가 많다. 그러나 active 스케줄은 semi active 스케줄에 비해 생성하는 것이 까다롭기 때문에, 일정계획을 생성하는데 소요되는 계산 비용을 감안하면 semi active 스케줄을 적절히 활용하는 것이 도움이 될 수 있다. 이에, 본 논문에서는 동일한 생산 일정계획 문제에 active 스케줄 기반 탐색 방법과 semi active 스케줄 기반 탐색 방법을 적용함으로써 이들의 성능을 비교해보고자 하였다.

**Design/methodology/approach**

각 공정들의 작업장 할당 순서를 의미하는 permutation encoding 기반 유전 알고리즘을 고전적인 job shop 일정계획 문제에 적용하기 위해 본 논문에서는 active 스케줄 복호화 및 semi active 스케줄 복호화의 두 가지 복호화 방법을 소개하였으며, 이들은 공정들의 순열로부터 실행가능한 스케줄을 얻는데 사용되었다.

**Findings**

semi active 스케줄 기반 유전 알고리즘은 active 스케줄 기반 유전 알고리즘에 비해 최적해를 탐색하는데 소요되는 반복 횟수가 좀 더 많은 경향이 있었으나, 알고리즘 실행 시간을 훨씬 짧았다. 나아가, semi active 스케줄 복호화는 그 절차가 단순하여 이해하고 구현하기 용이하다는 장점이 있었다. 따라서, 효과적인 해 탐색 전략이 주어지는 경우에는 semi active 스케줄에 기반한 해법이 일정계획 문제 풀이에 도움이 될 수도 있을 것으로 보여진다.

***Keywords:*** job shop 일정계획 문제, 유전 알고리즘, 복호화, active 스케줄, semi active 스케줄

* 이 논문은 2016년 10월 12일 접수, 2016년 12월 12일 1차 심사, 2016년 12월 27일 게재 확정되었습니다.