Efficient Scheduling for Production Systems with Alternative Machines using Genetic Algorithm Incorporating Effective Knowledge

（効果的な知識を組み込んだ遺伝的アルゴリズムによる
代替機械を有する生産システムのための高性能スケジューリング）

PARINYA　KAWEEGITBUNDIT

March 2019

Contents

Chapter 1

Introduction

1.1 Background and objective

Production scheduling is a decision making for allocating resources to operations of jobs over time with one or more objectives to be optimized. In recent smart manufacturing environments, scheduling plays an important role to achieve efficient production and to survive in competitive global marketplaces. Vast amount of literature about production scheduling has been published so far. They are classified based on the machine configuration and the production flow on shop floor as follows (Pinedo (2008)):

Single machine - The case of a single machine is the simplest of all possible machine environments and is a special case of all other more complicated machine environments.

Parallel machines - There are some machines in parallel. A job requires a single operation and may be processed on any one of the machines or on any one that belongs to a given subset. If all the machines have the same speeds to process an operation, it is referred to as identical parallel machines. If the speeds of the machines are different depending on jobs, then it is referred to as unrelated parallel machines.

Flow shop - There are some machines in series. Each job has to be processed on each machine. All jobs have to follow the same route, i.e., they have to be processed first on machine 1, then on machine 2, and so on. After completion on one machine a job joins the queue at the next machine.

Job shop - In a job shop with some machines, each job has to be processed on some machines but the route of machines to follow depends on the job. Each job has its own predetermined route to follow.

Open shop – In a job shop, each job has a fixed route that is predetermined. When the routes of the jobs are not determined but open, this model is referred to as an open shop.

In addition to the basic model described above, some combinations of these models or more realistic models are as follows:

Flexible flow shop - A flexible flow shop is a generalization of the flow shop and the parallel machine environments. Instead of some machines in series there are some stages in series with at each stage a number of identical machines in parallel. Each job has to be processed first at stage 1, then at stage 2, and so on. A stage functions as a bank of parallel machines; at each stage a job requires processing on only one machine and any machine can do. The queues between the various stages may or may not operate according to the First Come First Served (FCFS) discipline. Flexible flow shops have also been referred to as *hybrid flow shops* or as *multi-processor flow shops* in the literature.

Flexible job shop - A flexible job shop is a generalization of the job shop and the parallel machine environments. Instead of some machines in series there are some work centers with at each work center a number of machines in parallel. Each job has its own route to follow through the shop; job requires processing at each work center on only one machine and any machine can do.

The performance measure for the production scheduling is the key for evaluating the effectiveness of the scheduling system and can take many different forms. One of the most used objective functions in the literature has been *Makespan.* The makespan is equivalent to the completion time of the last job to leave the system. A minimum makespan usually implies a good utilization of the machine(s). Other important objective functions are due date related performance measures such as mean tardiness, number of tardy jobs, weighted tardiness and so on. Among them, the mean tardiness of jobs has been widely adopted as a basic measure for the due date related performance measure. Meeting due dates are one of the most important measures for scheduling as the variety of products increases and meeting customer’s satisfaction is the key to survive in competitive markets.

In this study, we deal with flexible flow shop and flexible job shop scheduling problems with mean tardiness as the objective function. Flexible flow shops and job shops are seen in a number of real-world environments and various approaches have been applied for the problems. The scheduling approaches can be classified into the following three categories:

(1) Exact methods

(2) Meta-heuristics

(3) Heuristics

Most scheduling problems belong to NP-hard. Even a single machine scheduling problem with the tardiness objective has been proved to be NP-hard (Du and Leung, 1990). Therefore, exact methods such as branch and bound or dynamic programming are only suitable for small-scale scheduling problems. When the scheduling problem is getting larger, the exact method will fail because of the large memory requirements and long computational time (Singh and Mahapatra, 2016).

For practical scale of problems, the scheduling that has acceptable performance with a reasonable amount of time is needed. The possible methods that can be applied to the scheduling problems that are NP-Hard in a reasonable time are heuristics and meta-heuristics. Many variations of heuristics and meta-heuristics algorithms have been proposed to solve scheduling problems. Dispatching or priority rules are the most typical heuristics in scheduling. For example, EDD (the earliest due date) rule selects the jobs that has the earliest due dates among the candidate jobs to be processed when a job should be assigned to a machine. Although this approach has strong advantage with regard to calculation time, the performance of scheduling is limited and there is room for improvement.

In these situations, the most prominent approach can be meta-heuristics such as Genetic Algorithm (GA), Simulated Annealing (SA), Taboo Search (TS), and so on. Those methods basically utilize probabilistic search and have no guarantee to obtain optimal solutions in finite calculation time. However, it can obtain better solutions for the practical scale scheduling problems when compared to other approaches. In particular, genetic algorithm is one of the most popular meta-heuristics used for various scheduling problems.

Genetic algorithms are meta-heuristics inspired by Charles Darwin's theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation. GA has first proposed by John Holland (Holland, 1975) as a means to find good solutions for problems that were otherwise computationally intractable. Holland's Schema Theorem and the related building block hypothesis (Goldberg, 1989) provided a theoretical and conceptual basis for the design of efficient GAs. The field grew quickly and the technique was successfully applied to a wide range of practical problems in science, engineering and industry.

Although meta-heuristics like genetic algorithms are one of the most effective methods, there is also room for improvement. First, genetic algorithms have weakness for fine tuning after finding good solutions. Second, because meta-heuristics does not utilize problem-specific knowledge, the quality of the solutions is not high in general. As for the first problem, memetic algorithms that incorporate local search in genetic algorithms have been proposed to compensate the weakness. The aim of this method is to enhance the search by utilizing effective local structure of the problem. The effective definition of local structure depends on the problem. For the second problem, genetic algorithms incorporating problem-specific knowledge have been proposed for searching out high-quality solutions in a reasonable amount of time. Eguchi et al. (2005) proposed a genetic algorithm incorporating priority rules and showed that the method can generate better schedules than using a genetic algorithm alone for larger scale problems. However, this approach considered only job selection and has not been applied to machine selection in flexible job shop scheduling.

The main objective of this thesis is to study effective production scheduling for flexible flow shop and job shop when using genetic algorithm incorporating problem-specific knowledge. The mean job tardiness is the performance measure of scheduling problem in this study.

1.2 Literature Review

Various types of scheduling methods for flexible flow shop scheduling and flexible job shop scheduling are proposed in the literature (Wang, 2005) (Ruben, et al., 2009).

Flexible flow shop is also called hybrid flow shop. The two-stage flexible flow shop scheduling is important to study because the two-stage is basic to consider multistage flexible flow shop scheduling problems. The two-stage flexible flow shop scheduling problem with one stage having at least two machines is proved to be NP hard (Hoogeveen et al., 1996). Gupta and Tunc (1991) and Gupta et al. (1997) studied a two-stage flexible flow shop with single machine and parallel machines on each stage. Chang et al. (2004) presented a two-stage flexible scheduling problem with separated setup and removal times and machine breakdown condition. The two-stage no-wait hybrid flow shop scheduling problem is the process that occurs when the operations of a job have to be processed from start to end without interruptions on or between machines (Liu and Xie, 2003). The no-wait scheduling problems have attracted the attention of many researchers both in practical application area and in theoretical area; see Aldowaisan (2001), Aldowaisan and Allahverdi (1998, 2001, 2002), Allahverdi (1997), Gupta et al. (1997). The no-wait hybrid flow shop that ignores the setup and removal times has been studied by Salvador (1973), Sriskandarajah (1993) and Liu and Xie, (2003). Salvador has developed a branch and bound algorithm to find minimum finish time for a no-wait flow shop with parallel machines model that arise in an actual application in the synthetic fiber industry. The worst case and average case analysis of some heuristic algorithms of this problem has been carried out in Sriskandarajah (1993) and Liu and Xie (2003).

In recent years, a plenty of researchers focused on flexible or hybrid flow shop scheduling problems using meta-heuristics. Engin (2011) and Oguz et al. (2001) proposed an efficient genetic algorithm to solving the problems. Engin and Döyen (2004) proposed a new approach using artificial immune system to solve the problem. Syam and Al-Harkan (2010) presented the comparison of three meta-heuristics; genetic algorithm, simulated annealing and tabu search algorithm to solve problems with regard to minimize makespan. Choong et al. (2011) presented two hybrid heuristic algorithms that combine particle swarm optimization (PSO) with simulated annealing (SA) and tabu search (TS), respectively. The new metaheuristic, called memetic algorithm has presented to solve this problem with regard to makespan by Moghaddam et al. (2009) and Akhshabi (2011). The results show that the memetic algorithm has a competitive performance as compared to genetic algorithm. Akhshabi et al. (2012) showed the flexible flow shop scheduling problem with sequence dependent setup with preventive maintenance. Abyaneh and Zandieh (2012) presented genetic algorithms for a flexible flow shop scheduling problem with sequence dependent setup and limited buffers.

As for flexible job shop scheduling, many optimization approaches have been proposed, such as exact solution algorithms, meta-heuristic algorithms, and heuristic algorithms using priority rules. Exact algorithms such as branch-and-bound methods (Iwata et al, 1978) cannot optimally solve larger scale problems. In general, flexible job shop problems are intractable; even the task of scheduling two identical, parallel machines to minimize the maximum completion time is NP-hard (Lenstra et al., 1977). Flexible or multipurpose-machine job shop scheduling with more than three jobs and two machines is also NP-hard (Brucker et al., 1997). Among the metaheuristic algorithms, many researchers have applied genetic algorithms (GAs) to scheduling with alternative machines (Candido et al., 1998; Jawahar et al., 1998; Norman and Bean, 1999; Morad et al., 1999; Lee and Kim, 2001; Moon et al., 2008; Pezzella et al., 2008; Shao et al., 2009; Li et al., 2010; Phanden et al., 2013). Although such algorithms can solve larger scale problems in reasonable time, the performance of the solutions deteriorates as the scale of problem grows. Among heuristic algorithms, priority rules have been developed and found to be effective in certain environments (Eguchi et al, 2006; Doh et al, 2013).

This study investigates the efficient production scheduling with alternative machines with regard to mean tardiness using genetic algorithms incorporating effective knowledge. There have been a few research studies dealing with flexible flow shop scheduling with regard to mean tardiness. Although lots of research have been carried out for flexible job shop scheduling including mean tardiness objectives, few research papers have examined performance improvement by incorporating heuristic rules.

1.3 Outline

The rest of this thesis is presented as follow:

The chapter 2 demonstrates the efficiency of genetic algorithm incorporating local search using effective local structure. This section deals with flexible flow shop scheduling problem also called hybrid flow shop scheduling problem. Three meta-heuristics: genetic algorithm (GA), memetic algorithm (MA) and random key genetic algorithm (RKGA) are examined.

The chapter 3 focuses on flexible job shop scheduling problem using genetic algorithm incorporating heuristic rules. Incorporating various heuristic rules for job and machine selection are investigated.

Subsequently, chapter 4 investigates the effectiveness of due-date related information for machine selection rules in the flexible job shop problem.

Finally, the chapter 5 mentions the overall conclusions from this study.

Chapter 2

Flexible Flow Shop Scheduling using Genetic Algorithm Incorporating Local Search

2.1 Introduction

This chapter demonstrates the efficient scheduling method using genetic algorithm incorporating local search for flexible flow shop scheduling problem. The genetic algorithm incorporating local search is also called memetic algorithm. For genetic algorithm, two kinds of coding methods, namely job sequencing coding and random key coding, are tested. Therefore, this chapter examines the performances of genetic algorithm (GA), random key genetic algorithm (RKGA), and memetic algorithm (MA).

2.2 Framework and formulation

In the flexible flow shop considered in this study, I jobs J_i ($i = 1, 2, \ldots, I$) must be sequentially processed on a set of N-stage flow shops. Each stage has the same number of machines whose processing rates are the same; each stage has the same number of identical parallel machines. Each job is processed first at stage one, then at stage two, and finally at stage N. The processing time of job J_i on stage j is p_{ij}. All jobs are available at time zero. Each job J_i has its own due-date d_i. The job sequence to release to the first stage is determined by the genetic algorithms used in this chapter. Jobs in each stage can be processed by any machine that idle. After finishing the operation of a job at a stage, the job waits at the intermediate buffer in the first come first served order. Whenever a machine in the next stage became idle, the job is processed on the machine.

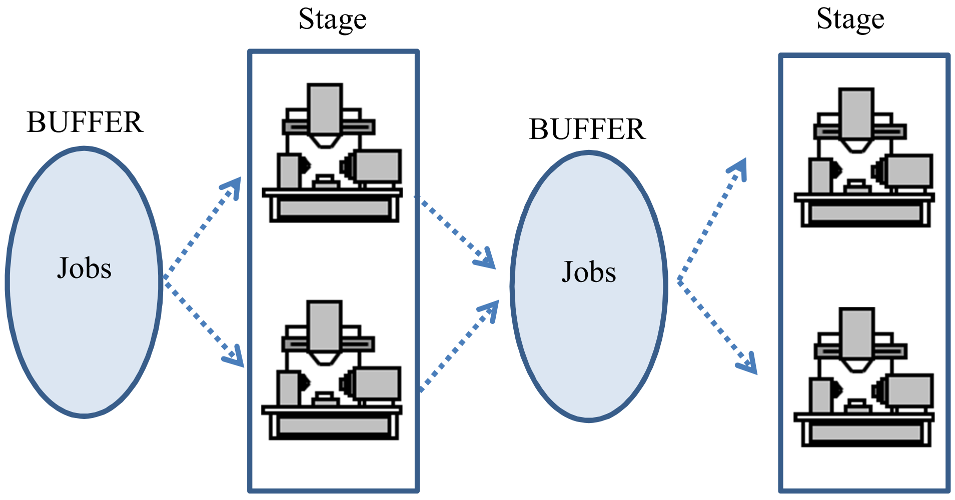


Fig.2.1 The flexible flow shop with identical parallel machine

Each machine cannot process multiple jobs at a time. Preemption is not allowed, namely, processing an operation cannot be interrupted until the processing the operation finishes.

The objective is to minimize the mean job tardiness MT, which is defined as follows:

$$MT = \frac{1}{I}\sum_{i=1}^{I} max(0, C_i - d_i) \tag{2.1}$$

where C_i is the completion time of job J_i at the stage *N*.

2.3 Algorithm procedure

We next describe the GA used in this chapter. GAs search the optimal solutions with holding multiple solutions called a population. A population consists of multiple individuals or chromosomes each of which corresponds to a solution. One of the important design problems in GA is coding of the chromosome. We examine two kinds of coding methods.

2.3.1 Job sequence coding-based GA

Job sequence coding is the most straightforward coding method in GA for scheduling. In this coding, integer numbers which represent job number are used as genes in a chromosome. The sequence of jobs can be readout from the chromosome. For illustration, the list of genes [4 1 3 2 5] represents a chromosome of five jobs being processed in the first stage. The sequence of jobs is $J_4 \rightarrow J_1 \rightarrow J_3 \rightarrow J_2 \rightarrow J_5$. This coding can generate infeasible solutions when crossover operation is applied. To avoid this, partially matched Crossover (PMX) (Goldberg and Lingle, 1989) is used in this study. As for mutation, the swapping is adopted. This mutation randomly selects two genes in a chromosome and exchanges their positions with probability p_m%.

Job sequence coding-based GAs are used for two meta-heuristics in this research: Genetic algorithm (GA) and Memetic algorithm (MA).

2.3.2 Random key based GAs (RKGA)

Bean (1994) proposed the random key coding. In this method, a random value between 0 and 1 is assigned to each locus in a chromosome. The random values are treated as priority values for job sequencing. For illustration, the list of [0.21 0.75 0.37 0.82 0.55] represents a chromosome of five jobs being processed in the first stage. The sequence of jobs is $J_4 \rightarrow J_2 \rightarrow J_5 \rightarrow J_3 \rightarrow J_1$ when the larger value corresponds to the higher priority. This coding method has strong advantage; infeasible solutions cannot be generated by crossover operation. A biased uniform crossover is applied in this study for this coding. First, a random real number is generated using a uniform distribution between [0, 1]. If this value is greater than p_c, the gene for the operation is taken from an individual and copied to the offspring. Otherwise, the gene is taken from the other individual. The value p_c is a constant which determines bias when applying crossover operation. Mutation is applied by regenerating a gene randomly with probability p_m%.

2.3.3 Generational transition of GA

Figure 2 provides the overview of the generational transition. An individual with lower mean tardiness is defined as having higher fitness. The best *r%* individuals in the current population are copied to the next generation, that is, an elitist strategy is adopted. The other (100- *r*) % individuals in the next generation are newly generated by applying a crossover operation in which two individuals are selected randomly from the current population. However, mutation is not applied to the genes of the fittest individual. Therefore, the mean tardiness of the fittest individual either decreases or remains the same with generational transitions.

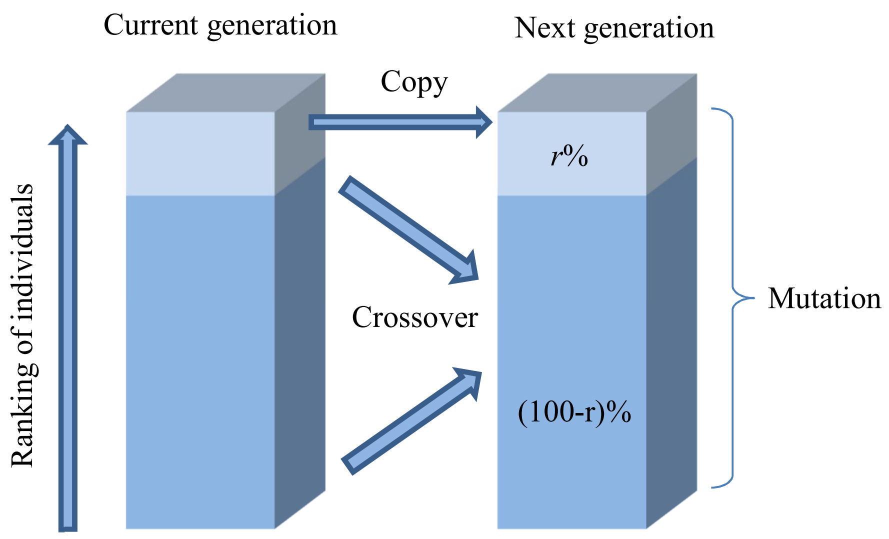


Fig. 2.2 Generational transition in the GA.

2.3.4 Incorporating local search

To improve the performance of GA, a search method incorporating local search is examined in this chapter. This method is also well known as memetic algorithm (MA) in which the hill climbing is embedded in GA. In this method, when a new individual is generated, better individuals are searched in the local neighborhood of it. Thus, the local neighborhood of an individual must be defined and the performance of search depends on the definition of this local structure. In this study, the individuals whose positions of two genes are swapped from an individual are defined as ones in the local neighborhood. Swapping is known as an effective local structure for a single machine scheduling (Yagiura and Ibaraki, 2001). In MA, every time a new individual is generated, the performance of swapping randomly selected two genes is evaluated by decoding and they are swapped if it is improved. Swapping is applied continuously while the performance improves.

2.4 Numerical experiments

The computational experiments are used to evaluate the efficiency of three meta-heuristics: GA, MA and RKGA. The details of the experiment are described as follows.

2.4.1 Numerical conditions

For the experiment, the number of operations per job is equal to the number of stages *N*. Each stage consists of two identical machines. The due date d_i of job J_i is set as follows:

$$d_i = k_i \sum_{j=1}^{N} p_{ij} \tag{2.2}$$

This is based on the TWK method (Baker, 1984). The symbol k_i represents the due-date tightness factor. Processing times of jobs are generated using uniform distribution between 5 and 100 ([5,100]). Thirty problems are generated for evaluation. The numerical conditions of the experiment are shown in Table 2.1

Table 2.1 The numerical conditions of the experiment

Number of machines	Number of jobs I	Due date tightness factor k_i
4 machines (2stages, each stage has 2 identical machines)	20	uniform distribution [2.0,4.0]
8 machines (4stages, each stage has2 identical machines)	50	uniform distribution [3.0,5.5]
16 machines (8stages, each stage has 2 identical machines)	100	uniform distribution [3.0,6.5]

For GA procedures, the elite *r%* is 20. Uniform crossover with p_c=0.7 is used for the RKGA. The mutation rate p_m is set as 1%. The number of individuals for GA and RKGA are 400. The population size (the number of individuals) of MA is set to a half (200 individuals) because MA takes twice as much calculation time when using the same number of population size. The number of generations is 1000.

2.4.2 Experimental results

Randomly generated 30 problem instances are used for evaluation. Figures 2.3-2.4 show the experimental results for the two-stage flow shop as a small size problem. Figure 2.3 shows the mean tardiness of jobs for the three methods. The mean tardiness is evaluated based on the mean value $mean_{method}$ and the standard deviation $diff.s.d._{method}$ of the 30 problem instances as follows:

$$mean_{method} = \frac{1}{30}\sum_{l=1}^{30} MT_{method,l} \tag{2.3}$$

$$diff.mean_{method} = \frac{1}{30}\sum_{l=1}^{30}\left(MT_{method,l} - MT_{best\_method,l}\right) \tag{2.4}$$

$$diff.s.d._{method} = \sqrt{\sum_{l=1}^{30}\frac{\left[\left(MT_{method,l}-MT_{best\_method,l}\right)-diff.mean_{method}\right]^2}{30}}. \tag{2.5}$$

where $MT_{method,l}$ represents the mean tardiness of the *l*th problem of method. $MT_{best\_method,l}$ represents the mean tardiness of the *l*th problem with the smallest value of $mean_{method}$ in each figure. Here $diff.s.d._{heuristic}$ denotes the standard deviation of the difference between a particular method and the best method. In the figures, the colored bars show the mean values, and the error bars show the standard deviations. The value on each bar represents $mean_{method}$. The length of the error-bar lines corresponds to twice the value of $diff.s.d_{method}$.

Figure 2.4 shows the generational transition of mean tardiness of jobs of the best individual. Each curve shows the mean value of the 30 problem instances.

From the results of small size problem, memetic algorithm (MA) performs the best. The second is the random key genetic algorithm (RKGA) which has slightly worse performance. Genetic algorithm is the worst for this problem.

Figures 2.5-2.6 show the results for the medium size problem (four stages - eight machines). The experimental results of medium size problem indicate that MA and RKGA are better than GA. The performances of MA and RKGA are similar but RKGA performs slightly better for this problem.

Figures 2.7-2.8 show the experimental results for large size problem (eight stages - 16 machines). RKGA performs best especially at the earlier stage of generational transition. The performances of RKGA and MA are better than that of GA.

These results suggest that MA improves the performance of GA. However, RKGA performs the best especially when the problem size is large.

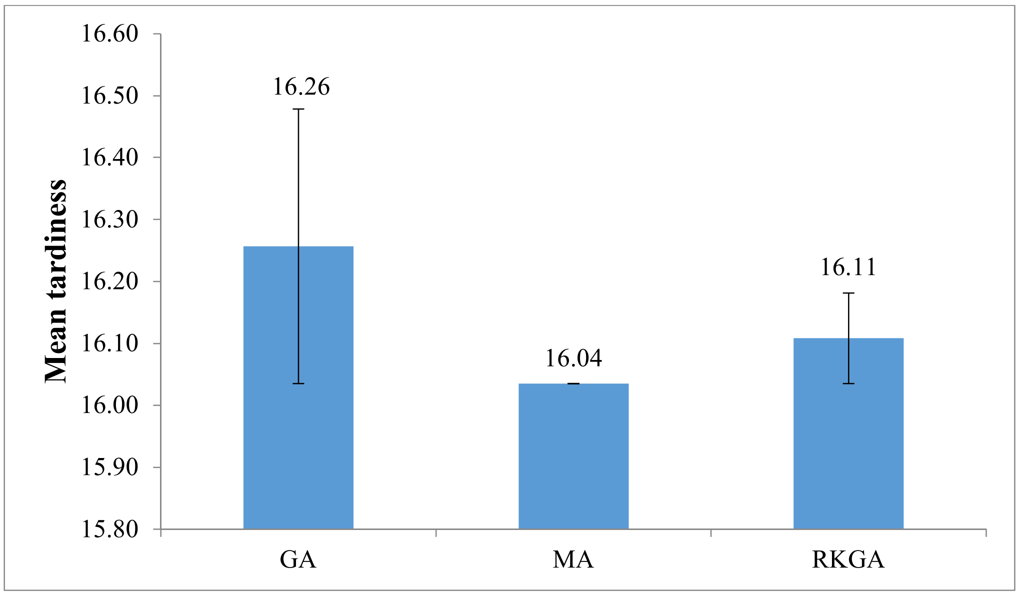


Fig. 2.3 Mean tardiness for small size problem

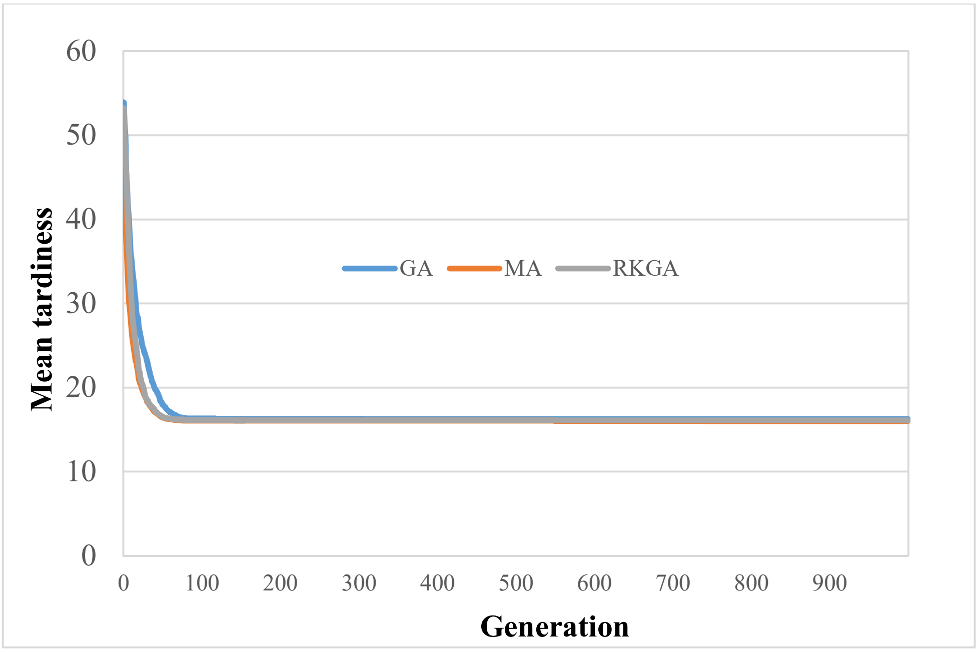


Fig. 2.4 Generational transition for small size problem

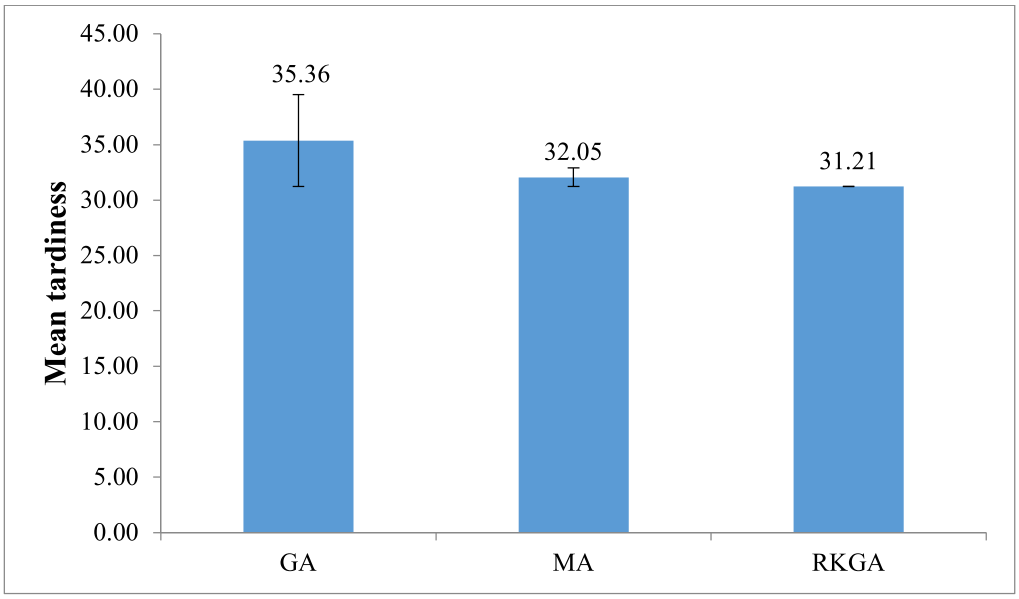


Fig. 2.5 Mean tardiness for medium size problem

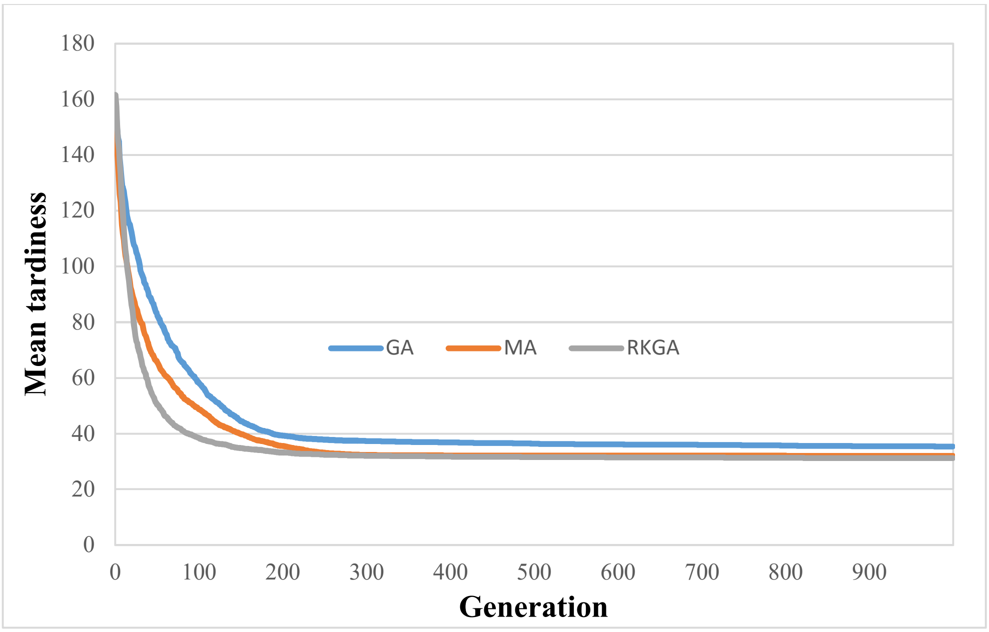


Fig. 2.6 Generational transition for medium size problem

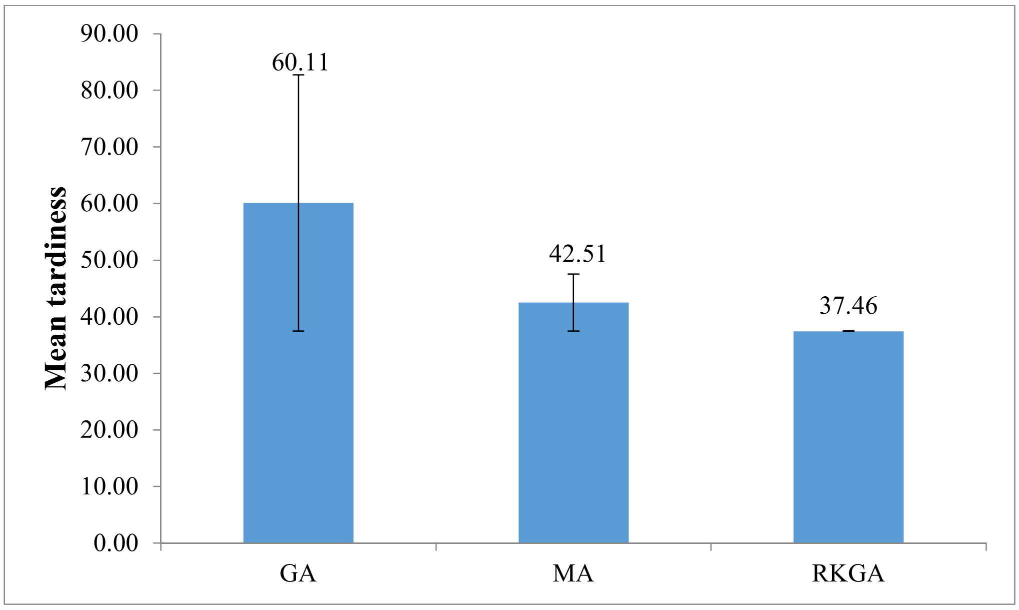


Fig. 2.7 Mean tardiness for large size problem

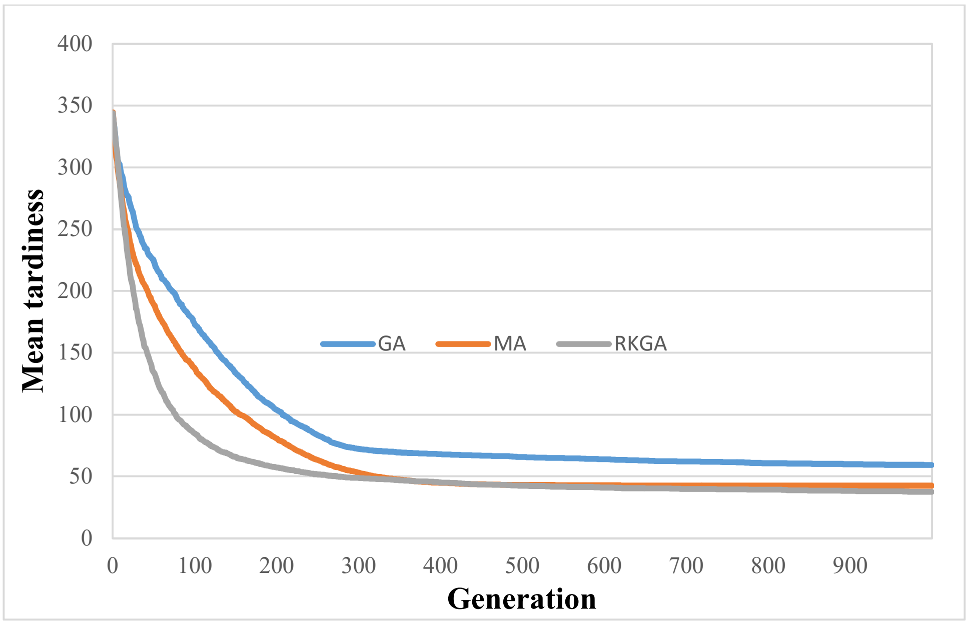


Fig. 2.8 Generational transition for large size problem

2.5 Conclusion

This chapter has considered genetic algorithm for flexible flow shop in which each stage has identical parallel machines. The objective is to minimized the mean tardiness of jobs. Genetic algorhitm has room for improvement because of its weakness for fine tuning after finding good solutions. Incorpoarting hill climbing using effective local strucutre in GA, which is also called memetic algorihtm (MA), is desinged to improve the perforemance. Experimental resutls showed that MA actually improves GA but the random key coding GA (RKGA) performs better expecially for the large size problem. As described in the next chapter, the random key coding is also sutable for the incorporation of heuristic rules for scheduling. Therefore, the RKGA is adopted as the GA used in the following chapters.

Chapter 3

Flexible Job Shop Scheduling Using Genetic Algorithm Incorporating Heuristic Rules

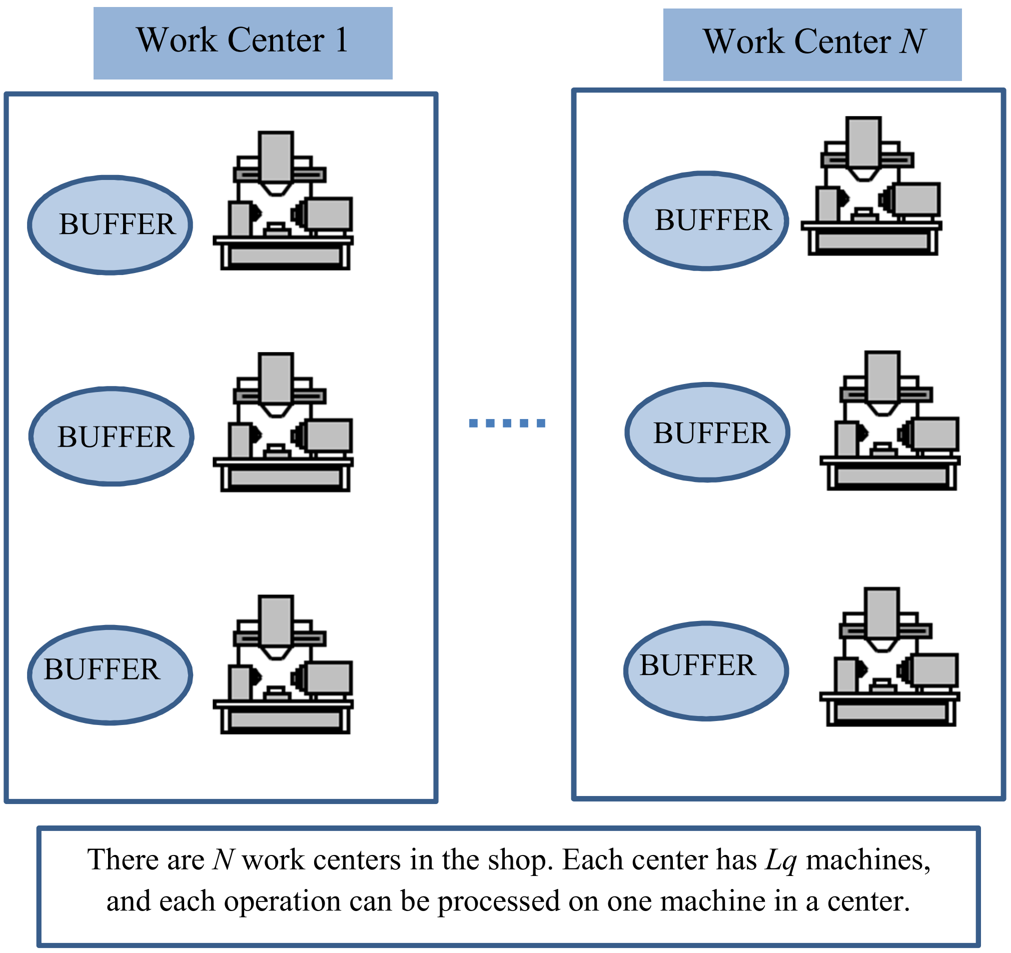
3.1 Introduction

This chapter deals with the flexible job shop scheduling problem. In flexible jobs shops, the sequence of jobs must be determined on each machine. In addition, there are alternative machines with different processing speed in general, a machine must also be selected to process each operation of a job. This problem is more difficult than flexible flow shop in which decision making is need only for the determination of job sequence at the first stage.

In the last chapter, the random key coding GA was found to be effective. This coding method is suitable for the incorporation of heuristic rules that are proposed in the research of job shop scheduling. This chapter proposes an efficient approach to flexible job shop scheduling using GA that incorporates heuristic rules. For the purpose, various combinations of job and machine selection rules are evaluated under different scheduling conditions, and the best rule combination is determined. In flexible job shop scheduling, the relative importance of job and machine selection varies depending on scheduling conditions. In the proposed method, the GA can be applied only to one of the two selections, and the other selection is carried out using a heuristic rule alone. It is further shown that applying genetic algorithm only for either job selection or machine selection can generate good schedules, depending on scheduling conditions.

3.2 Framework and formulation

This chapter considers two types of job shop with alternative machines, as shown in Figure 3.1. The work center base shop consists of N work centers. Each center W_q ($q \in \{1, 2, \dots, N\}$) has L_q machines. Each job J_i requires a set of n_i operations O_{ij} which are processed in order of increasing operation number j. Operation O_{ij} is processed by the center Ws_{ij} ($s_{ij} \in \{1, 2, \dots, N\}$). The routing of the operations through the work centers is flexible, but we assume that $Ws_{ij} \neq Ws_{i,j+1}$ for $j = 1, 2, \dots, n_i-1$, namely, that no two successive operations of a job are processed in the same center. An operation to be processed in center W_q is performed on one of its L_q machines. The machine that processes the operation O_{ij} is denoted M_{ijk} .



There are N work centers in the shop. Each center has Lq machines, and each operation can be processed on one machine in a center.

Fig. 3.1 Work center job shop with alternative machines

This chapter also considers a similar case, a random job shop with alternative machines. In this case, the shop consists of M machines. Each job J_i (i = 1, 2, …, I) requires n_i operations { O_{ij} (j = 1, 2, … , n_i)}, which again are processed in order of increasing j. Each operation can be processed on one of R machines among the M machines in the shop. The specific combination of the R machines able to process an operation varies depending on the nature of the operation. The machine that processes operation O_{ij} is denoted M_{ijk}. ($k \in \{1, 2, \ldots, R\}$), and the processing time is p_{ijk}.

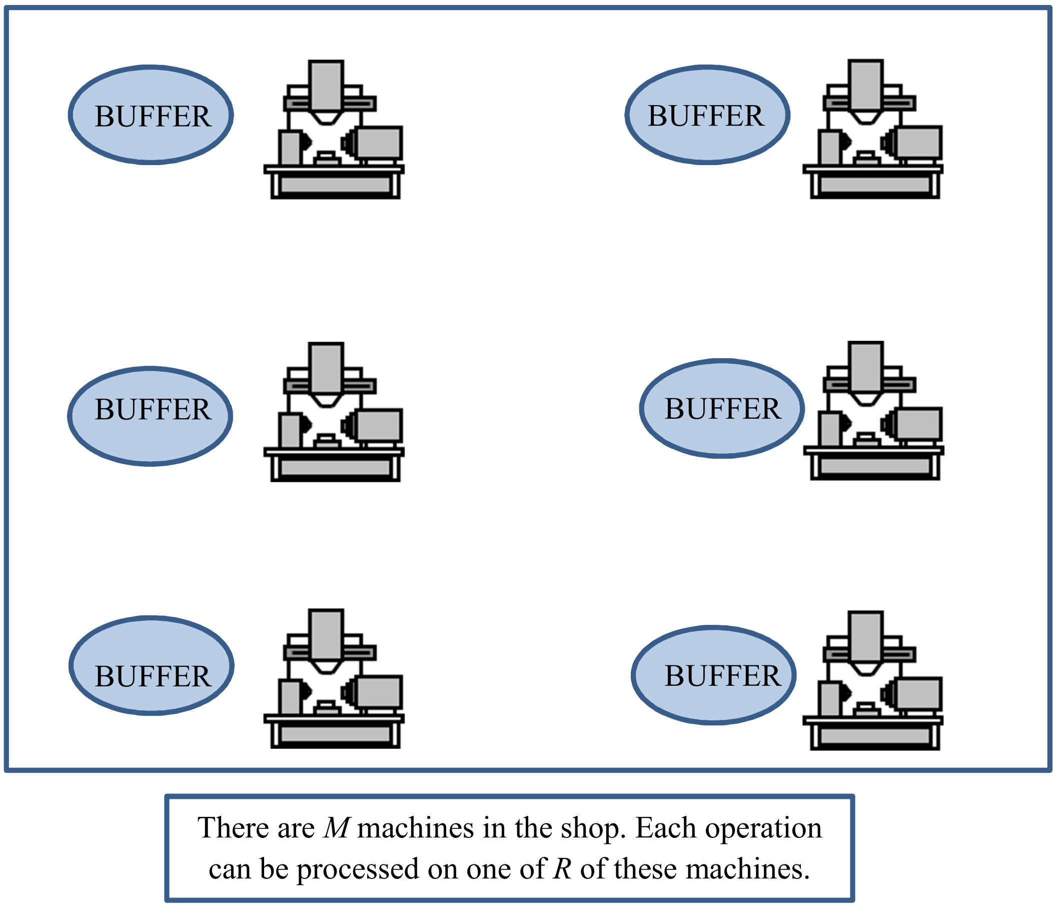

BUFFER
BUFFER
BUFFER
BUFFER
BUFFER
BUFFER
There are *M* machines in the shop. Each operation can be processed on one of *R* of these machines.

Fig. 3.2 Random job shop with alternative machines

In both scenarios, each machine can process only one job at a time, and each job can be processed only on one machine at a time. Operations cannot be interrupted (non-preemption). Transportation times are not considered. The objective of the flexible job shop is to minimize the mean job tardiness *MT*, which is defined as follows:

$$MT = \frac{1}{I}\sum_{i=1}^{I} max(0, C_i - d_i). \tag{3.1}$$

where C_i is the completion time and d_i is the due date of job J_i , respectively.

3.3 Algorithm procedure

3.3.1 Heuristic rules for job and machine selection

This section describes the procedure for generating schedules and the heuristic rules used in the procedure. We assume that each machine is equipped with an input buffer in which jobs wait to be processed. When a new job arrives on the shop floor, a machine is selected to process the first operation, and the job is transferred to its input buffer. When the job operation is finished, the machine to process the next operation is selected and the job is transferred to the input buffer of the selected machine. When a machine has become idle and there are jobs waiting to be processed in its input buffer, one of the jobs is selected for immediate processing. If there is only one waiting job, it is automatically started. This paper considers non-delay scheduling.

There are two types of decision to be made in this approach to scheduling: job selection and machine selection. When a machine has become idle and it has jobs waiting to be processed, one has to be selected for the next round of processing. When a new job arrives on the shop floor or a machine has finished processing a job operation, one of the machines that can process the next operation has to be selected. Heuristic rules are applied to these choices, and the overall scheduling performance depends on their effectiveness.

3.3.1.1 Rules for job selection

Various priority or dispatching rules have been proposed in the literature for the task of job selection. Because the objective here is to minimize mean job tardiness, five rules have been chosen for further examination. In each case, a priority index (*PI*) is calculated, and the job with the highest value is selected to be processed next.

1. **ATC**

The ATC rule (Vepsalainen and Morton, 1987) assigns a priority index PI_{ijk} to each waiting job operation oij as follows:

$$PI_{ijk} = \frac{1}{p_{ijk}} exp\left\{-\frac{max[d_i - t - rpt_i - b(rpt_i - p_{ijk}),0]}{\kappa\bar{p}}\right\} \tag{3.2}$$

where *t* is the current time, $\bar{p}$ is the average processing time of operations of jobs waiting to be processed on a particular machine, and rpt_i denotes the time remaining for the processing of the job. In this paper, when calculating rpt_i, the processing time of the succeeding operation is taken to be the average of the processing times on the candidate machines for the operation. Here κ and *b* are adjustable parameters.

2. **CR+SPT**

The CR+SPT rule (Anderson and Nyirenda, 1990) calculates PI_{ijk} for each waiting operation O_{ij} as

$$PI_{ijk} = \frac{1}{p_{ijk}}\left(max\left[\frac{d_i - t}{rpt_i}, 1\right]\right)^{-1} \tag{3.3}$$

3. **(SL/RPN)+SPT**

The (SL/RPN)+SPT rule (Yoda et al., 2014) assigns a priority value for each waiting operation O_{ij} using

$$PI_{ijk} = \frac{1}{p_{ijk}}\left(max\left[\frac{d_i - t - rpt_i}{rpn_i}, 0\right] + 1\right)^{-1} \tag{3.4}$$

where rpn_i is the number of operations remaining to be completed for job J_i

4. **SLACK**

The SLACK rule assigns a priority value for each waiting job operation O_{ij} as

$$PI_{ijk} = \left[exp\left(\frac{d_i - t - rpt_i}{k_s}\right)\right]^{-1} \tag{3.5}$$

where k_s is a constant. When used alone, this rule simply selects the job with the smallest value of (d_i - t - rpt_i). The exponential function is introduced for incorporation of this rule into the GA dealt with in this paper.

5. **EDD**

The EDD rule's priority index for each waiting operation O_{ij} is

$$PI_{ijk} = \frac{1}{d_i} \tag{3.6}$$

This rule selects the job with the earliest due date.

3.3.1.2 Rules for machine selection

In contrast to the rules for job selection, not many heuristics for machine selection have been developed in the literature. Here we examine five candidate rules for machine selection. Priority indexes are again assigned, and the machine with the highest priority value is selected.

1. **PT**

The PT rule selects the machine with the lowest processing time among the candidates. When a machine to process the next operation O_{ij} of a job has to be selected, the priority index is calculated for each candidate M_{ijk} using

$$PI_{ijk} = \frac{1}{p_{ijk}} \tag{3.7}$$

2. NINQ

The NINQ rule selects the machine with the fewest jobs in its input buffer. When a machine is needed to process the next operation O_{ij} of a job, the priority index for each candidate M_{ijk} is calculated as

$$PI_{ijk} = \frac{1}{n_k + 1} \tag{3.8}$$

where n_k represents the number of jobs waiting in the input buffer of a machine M_{ijk} that can process O_{ij}.

3. WINQ

The WINQ rule selects the machine with the smallest total processing time for all of the jobs waiting in its input buffer. When a machine to process the next operation O_{ij} of a job has to be selected, the index value for each candidate machine M_{ijk} is calculated as

$$PI_{ijk} = \frac{1}{winq_k} \tag{3.9}$$

where $winq_k$ represents the total of the imminent processing times of all jobs waiting in the input buffer of a machine M_{ijk} that can process the next operation.

4. WINQ+RPT+PT

The WINQ+RPT+PT rule selects the machine with the smallest total value of (1) processing times for all waiting jobs in its input buffer, (2) the remaining processing time $rptc_k$ of the operation currently being processed, and (3) the processing time of the next operation if the operation is processed on the machine. The priority index is given by

$$PI_{ijk} = \frac{1}{winq_k + rptc_k + p_{ijk}} \tag{3.10}$$

5. (WINQ+RPT+PT)×PT

The (WINQ+RPT+PT)×PT rule (Eguchi et al., 2006) combines the WINQ+RPT+PT and PT rules. This rule selects the machine with the smallest value of the times {(1) + (2) + (3)} × (3) described in Section 3.2.4. The priority index is calculated as

$$PI_{ijk} = \frac{1}{(winq_k + rptc_k + p_{ijk})p_{ijk}} \tag{3.11}$$

The PT rule is effective at reducing total workload by selecting machines that can process operations with smaller processing times. NINQ and WINQ are the basic rules for workload balancing. The WINQ+RPT+PT rule is effective for more detailed workload balancing. The combined (WINQ+RPT+PT)×PT rule aims to both balance workload and reduce total workload (Eguchi et al., 2006).

3.3.2 Scheduling using a genetic algorithm

Our goal is to find an efficient scheduling method for job shops with alternative machines using a GA that incorporates heuristic rules. The GA used in this chapter is random key based GAs (RKGA) that described in the previous chapter. The random keys can be considered as priority values for job selection.

3.3.2.1 Random key based genetic algorithm

The random keys can be considered as priority values for job selection and machine selection. As for job selection, the gene j_gene_{ij} is assigned to each operation o_{ij}. The value of j_gene_{ij} takes the real number between 0 and 1. When an operation of a job has to be selected as the operation to be processed next, the operation which has the largest value of j_gene_{ij} is selected.

For machine selection, Norman and Bean (1999) proposed a machine number coding approach. In the approach, the genes using the number [1, *k*] were used for machine selection; the gene values directly correspond to the machines to process the next operation. However, this study adopts real numbers for coding also for machine selection because of the convenience for incorporating heuristic rules. The gene m_gene_{ijk} is assigned for each candidate machine of an operation o_{ij}. The value of m_gene_{ijk} takes the real number between 0 and 1. When a machine among candidate machines has to be selected as the machine to process the next operation of a job, the machine which has the largest value of m_gene_{ijk} is selected.

A chromosome consists of the genes, *j_gene*$_{ij}$ and *m_gene*$_{ijk}$, for all the operations of jobs. The generational transition and genetic operators are designed as described in the previous chapter. The genes in the chromosomes or individuals in the first generation are generated randomly using the uniform distribution between 0 and 1. The fitness value of an individual is calculated by carrying out scheduling simulation using the genes as priority values for job selection and machine selection. The individual which has the smaller mean tardiness is defined to have the higher fitness. The parameters r%, p_c , and p_m% are also used in this chapter.

3.3.2.2 Genetic algorithm incorporating with heuristic rules

The performance of the GA can be improved by incorporating problem-specific knowledge. Embedding heuristic rules found to be effective for scheduling into the GA can facilitate the search for optimal schedules, and these can be incorporated when evaluating the fitness of an individual. When selecting an operation to be processed next, the operation with the highest priority value is selected. If the rules are used alone, the priority values are calculated using their heuristic targets. If the GA is used alone as described in Section 4.1, the priority values are given by genes. With a heuristic rule for job selection incorporated into the GA, the priority values are given by the combination of genes and a job selection rule; the priority value *j_priority*$_{ijk}$ for each waiting job operation O_{ij} is calculated following Eguchi et al. (2005):

$$j\_priority_{ijk} = j\_gene_{ij} \times PI_{ijk} \tag{3.12}$$

A heuristic rule for machine selection can be similarly incorporated into the GA; the priority value for each candidate machine M_{ijk} is calculated as follows:

$$m\_priority_{ijk} = m\_gene_{ijk} \times PI_{ijk} \tag{3.13}$$

When incorporating heuristic rules into the GA, an individual in which all the genes have the same value is included in the initial population. This value is set to 0.5 in this paper. The scheduling performance corresponding to this individual is the same as that generated using the heuristic rules alone, because the large/small relation of priority values of Eq. (3.12) and (3.13) are determined only by PI_{ijk}. The best individual is copied to the next generation, and the mutation step is not applied to it. Therefore, the mean tardiness of a schedule obtained using this method is guaranteed to be smaller than or at least equal to that of a schedule generated using the heuristic rules alone.

3.4 Numerical experiments

3.4.1 Experimental conditions

Two types of scheduling problems are generated for performance evaluation: a job shop that consists of work centers and a random job shop. For the work-center shop, performance is examined under three different scheduling conditions: moderate load, a state of low machine utilization, and the imposition of tight job due dates. As a result, we examine scheduling performance using four conditions: the work-center shop with three conditions and the random job shop. Detailed conditions for each problem are described below.

In the work-center model, the shop floor consists of eight work centers ($N = 8$), and each center has two machines ($L_q = 2$). Any operation assigned to a work center can be processed on either of its machines. The total number of machines is thus 16. There are 100 jobs assigned. The number of operations n_i for each job is set randomly to lie between 4 and 8. The order of work stations to be visited is determined randomly. The processing times for an operation on the two candidate machines are taken to be different. We assume that the processing time on the first machine is shorter than that on the second for all operations. The processing time on the first machine is sampled from a uniform distribution between 5 and 100. The processing time on the second machine is determined by multiplying the processing time on the first machine by a speed factor, which is uniformly distributed between 1.0 and 2.0.

The due date d_i of job J_i is as follows:

$$d_i = r_i + k_i \sum_{j=1}^{n_i} \sum_{k=1}^{2} \frac{p_{ijk}}{2} \tag{3.14}$$

This is based on the TWK method. Namely, the interval between the release time r_i and the due date d_i is set to be proportional to the total processing time of all the operations for the job. Here we estimate the processing time of an operation by averaging the processing times on the two candidate machines. The parameter k_i is the due date tightness factor. For problems with moderate load, the tightness factor for each J_i is taken from a uniform distribution between 1.5 and 3.0. Approximately 10%–15% of the jobs are tardy at this tightness when using the GA with heuristic rules. For problems with tight due dates, k_i is set by using a uniform distribution between 1.0 and 2.0. Approximately 25%–30% of the jobs are tardy at this tightness.

Twice as many jobs as the number of machines in the shop are released to the floor at time zero. Additional jobs arrive on the shop floor at exponentially distributed random intervals. For conditions of moderate load, the arrival rate is set so that the average machine utilization reaches 80%–90%. For low machine utilization, the arrival rate is set so that the average machine utilization becomes about 60%–70%.

For random job shop problems, there are simply 16 machines and 100 jobs. The number of operations n_i for each job is randomly chosen to lie between 4 and 8. Each operation can be processed on the first and the second (alternative) machine. The order of the first machine to process each operation is randomly determined under the condition that no successive operations are processed on the same machine. The processing time of an operation on the first machine is taken from the uniform distribution between 5 and 100. The second machine is selected randomly, and its processing time is chosen randomly between the range of the processing time on the first machine and twice the processing time on the first machine. The job due dates and arrival rates are set in the same way as for the work-center problems with moderate load; there are about 10%–15% tardy jobs, and the machine utilization is about 80%–90% when using the GA with heuristic rules.

Thirty scheduling problems are generated for each scheduling condition.

3.4.2 Results using heuristic rules alone

First, the scheduling performance using heuristic rules alone is examined for the four scheduling conditions. Figures 3.3–3.6 show the results. Twenty-five combinations using five job selection rules and five machine selection rules are tested. In this chapter, the parameters for the heuristic rules are set by using preliminary experiments as follows: $\kappa = 1.5$ and $b = 1$ for the ATC rule, and $k_s = 100$ for SLACK. In each figure, the mean tardiness of jobs when using each combination of job selection and machine selection rule is evaluated based on the mean value $mean_{rules}$ and the standard deviation $diff.s.d._{rules}$ of the 30 problems as show in previous chapter (see Eq.2.3-Eq.2.5).

Figures 3.3–3.5 show that the results using the PT rule for machine selection are very bad. This is natural, because the second machine in each work center is never used when using the PT rule. Figure 3.6, by contrast, shows that using the PT rule in a random job shop is not very bad. This is because the first machine for each job operation is randomly selected from all the machines in the shop.

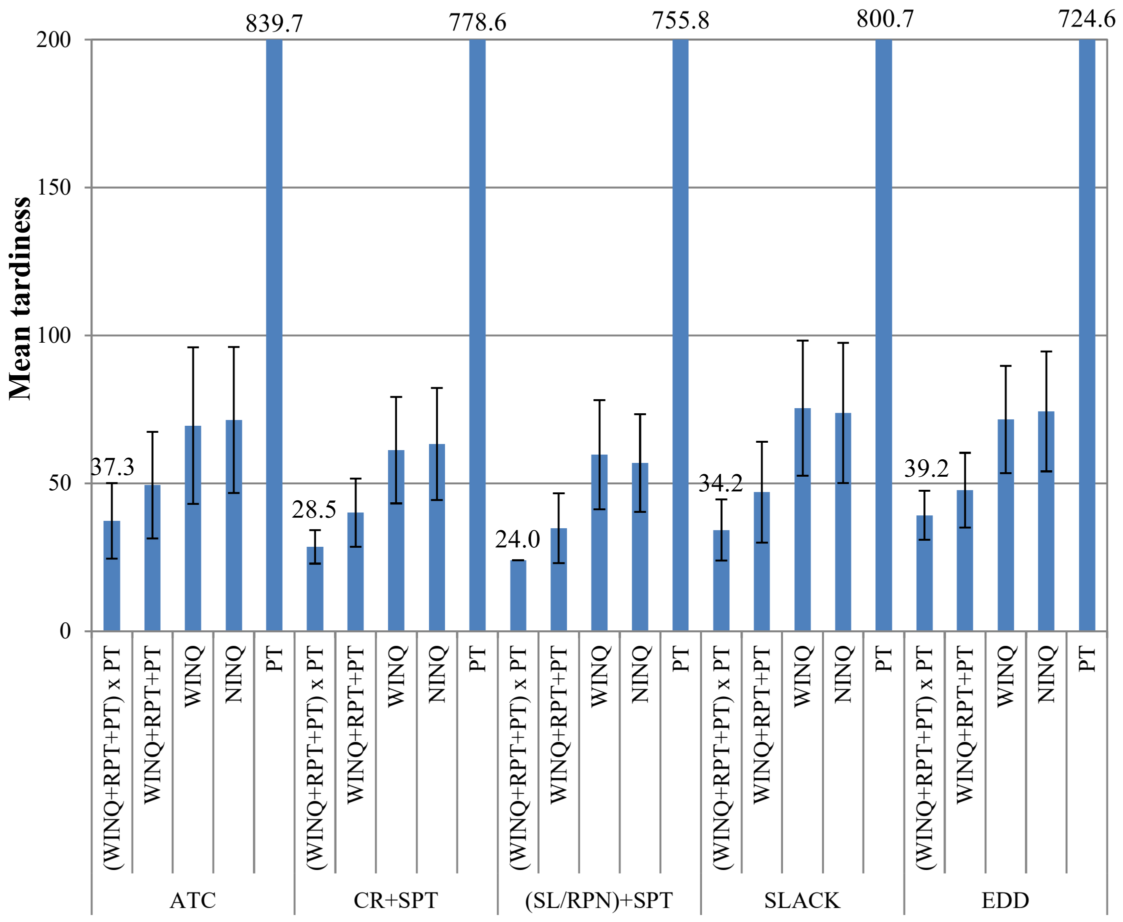


Fig. 3.3 Mean tardiness using heuristic rules alone for work-center problems under moderate load.

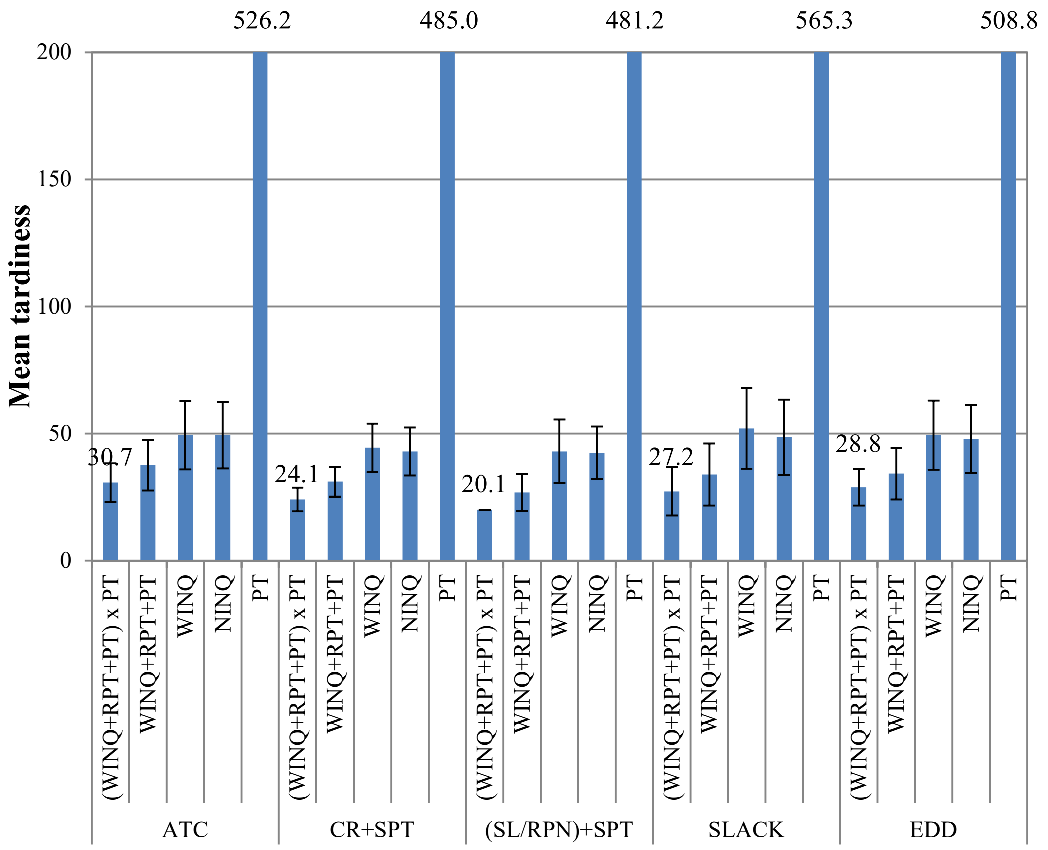


Fig. 3.4 Mean tardiness using heuristic rules alone for work-center problems under low machine utilization.

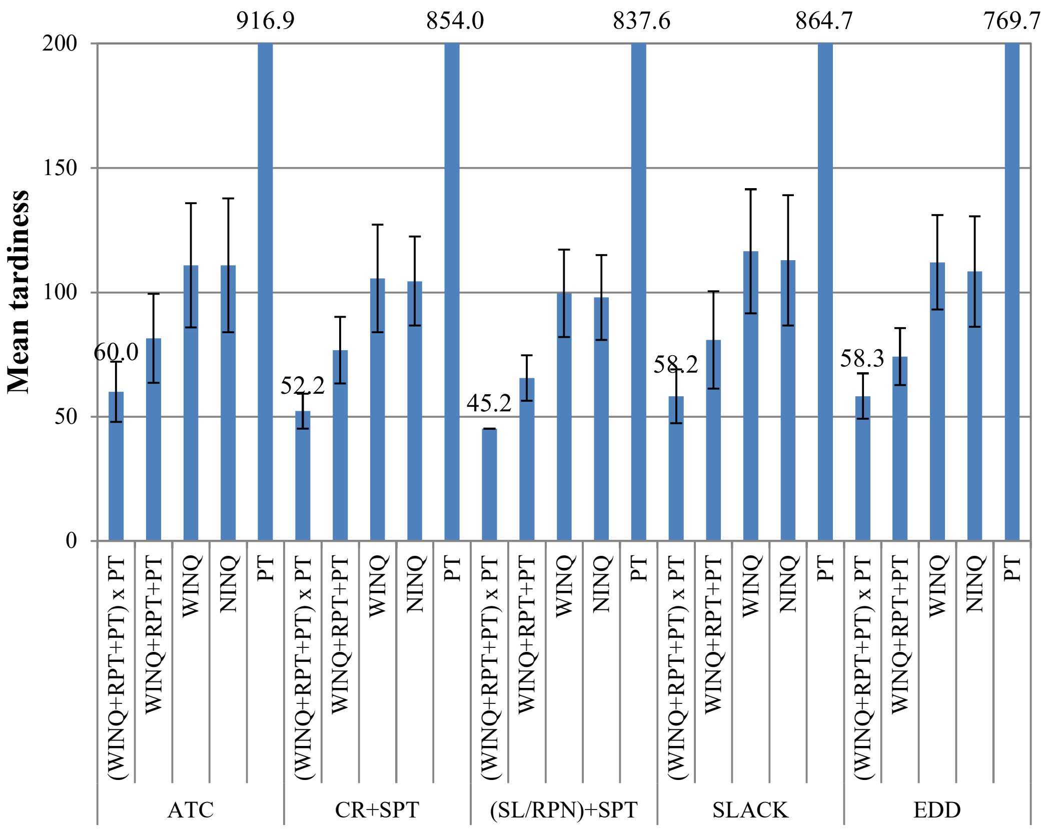


Fig. 3.5 Mean tardiness using heuristic rules alone for work-center problems under tight due dates

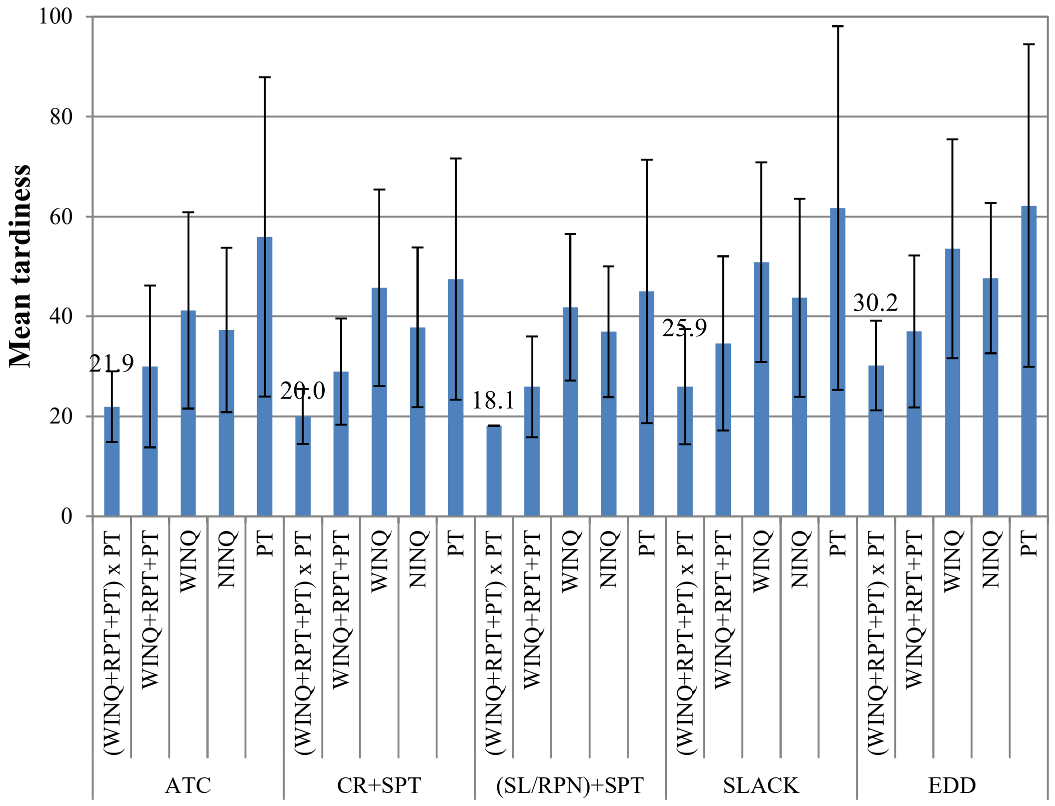


Fig. 3.6 Mean tardiness using heuristic rules alone for a random job shop.

Figures 3.3–3.6 indicate that the machine selection rules have a greater impact on performance than the job selection rules. For all conditions, the (WINQ+RPT+PT)×PT rule performs best. There are relatively small differences due to the job selection rules. Among them, the (SL/RPN)+SPT rule has the smallest mean tardiness for all the conditions. This rule can be considered as one of the best job selection rules if we have to select one rule. As a result, we conclude that the best combination of heuristic rules is (SL/RPN)+SPT plus (WINQ+RPT+PT)×PT rule when using heuristic rules alone.

3.4.3 Effectiveness of random key based coding for machine selection

In this chapter, we adopt a GA using random key based coding not only for job selection but also for machine selection. To examine the effectiveness of this coding method for machine selection, the method is compared with the coding method based on machine numbers proposed by Norman and Bean (1999). In this experiment, the parameters in the GA are set as follows: $r = 20$, $p_c = 0.7$, $p_m = 1$. These values were determined based on preliminary experiments. The number of individuals and generations are 400 and 1000, respectively. The work-center problems with moderate load are solved using the two types of GAs described in Section 3.1. Figure 3.7 shows the generational transition of the GAs using the two coding methods for machine selection. Each curve in the figure is the average generational transition of the best individuals for 30 problems. This illustrates that the proposed random key based coding outperforms machine number based coding.

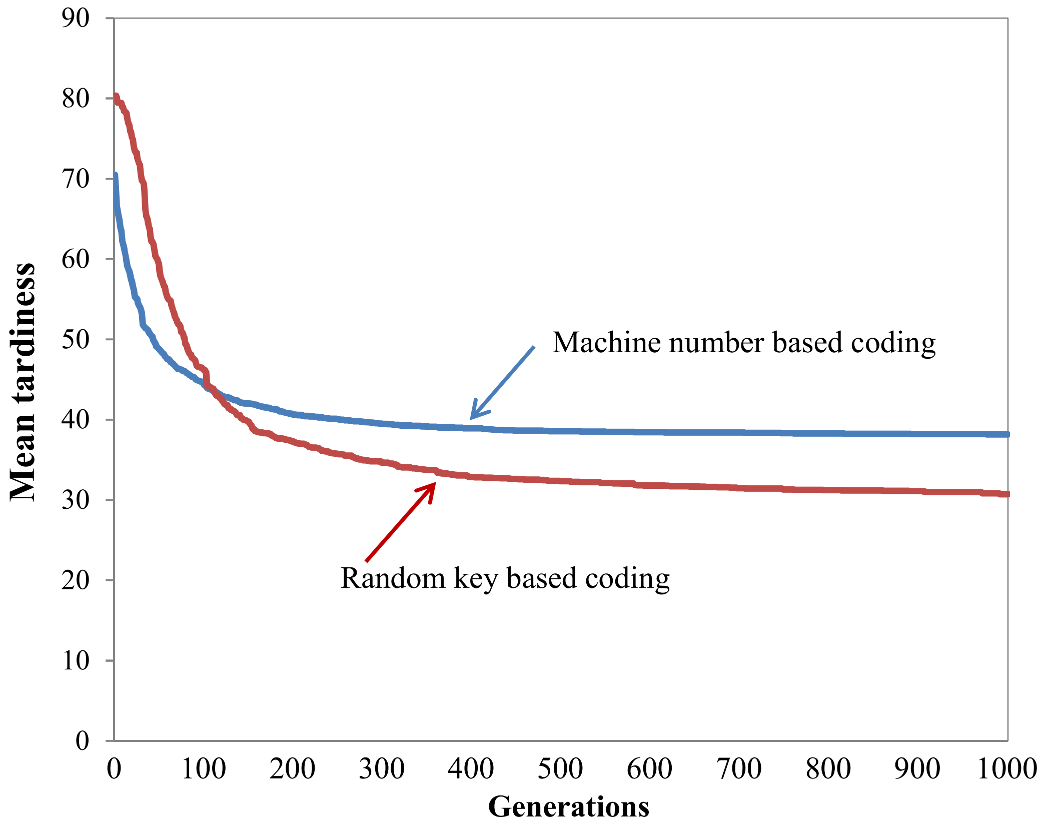


Fig. 3.7 Generational transition of the GAs using random key based coding and machine number based coding for machine selection

3.4.4 GA results incorporating heuristic rules

Next, we examine the performance of the GA incorporating heuristic rules. Three types of GAs incorporating heuristic rules are compared: (1) the method in which the GA is applied both to job and machine selection, (2) the method in which the GA is applied only to job selection and machine section is carried out using an independent rule, and (3) the method in which the GA is applied only to machine selection and job selection is carried out using an independent rule. Each method is evaluated for each of the four problems types. The parameters of the GA are set as in the previous session.

3.4.4.1 Work-center job shops under moderate load

Figures 3.8–3.10 show the mean job tardiness obtained by using the GA incorporating heuristic rules for the work-center type problems with moderate load. For all three GA methods, the best results were obtained when using the (SL/RPN)+SPT rule for job selection and the (WINQ+RPT+PT)×PT rule for machine selection. For most combinations of heuristic rules, good performance was obtained when applying the GA only to machine selection in this condition. The overall best result was 7.7 in Fig. 3.10 when using the (SL/RPN)+SPT rule alone for job selection and applying the GA incorporating the (WINQ+RPT+PT)×PT rule to machine selection.

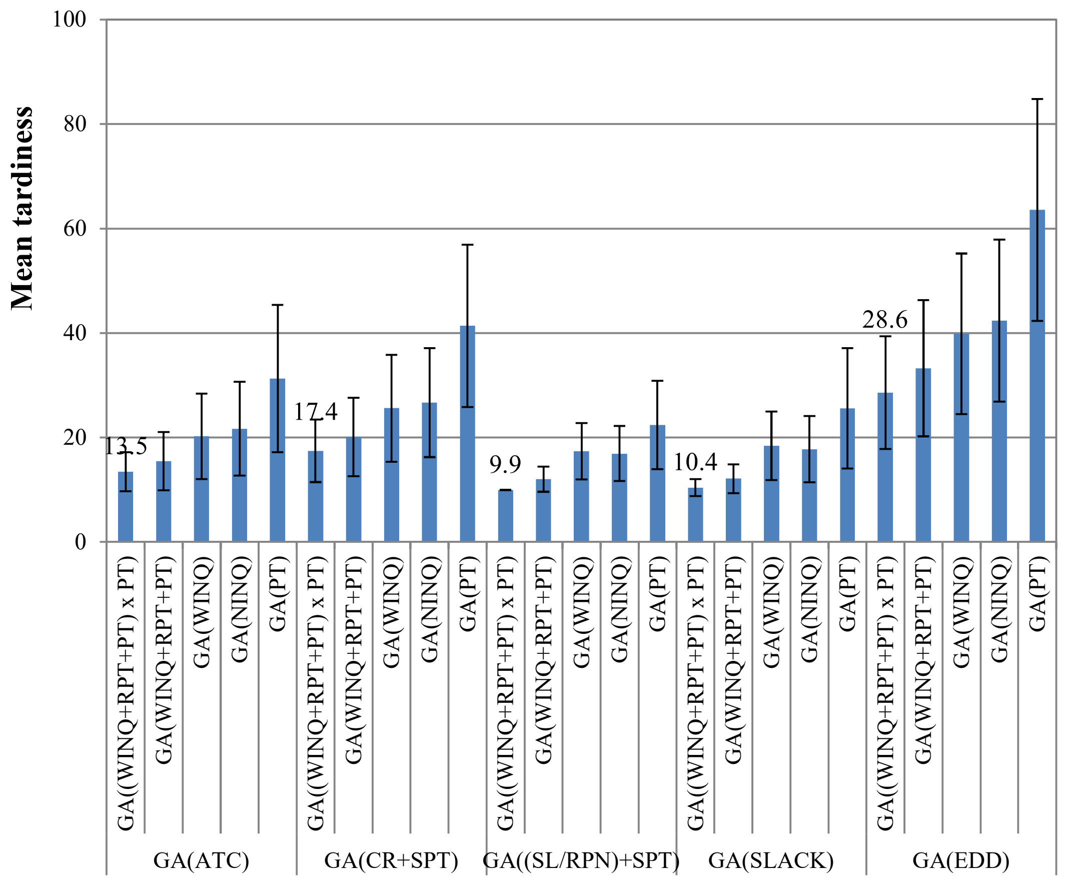


Fig. 3.8 Mean tardiness when the GA is applied to both job and machine selection in work-center job shops under moderate load.

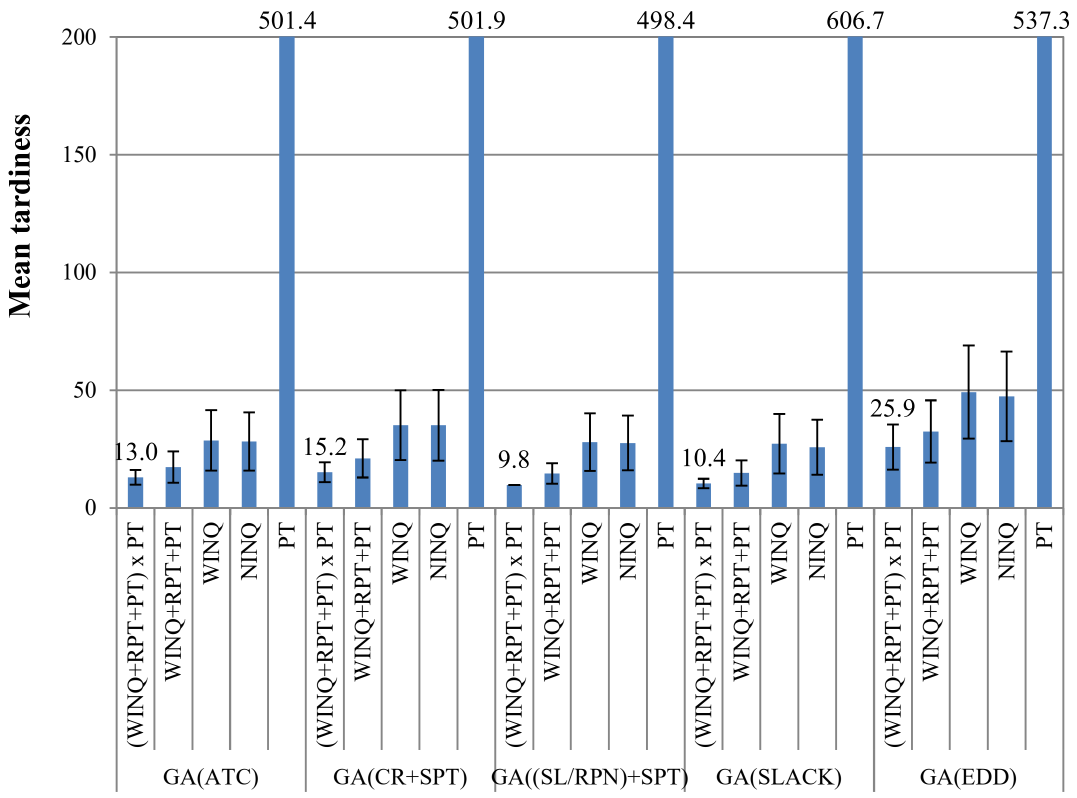


Fig. 3.9 Mean tardiness when the GA is applied only to job selection in work-center job shops under moderate load.

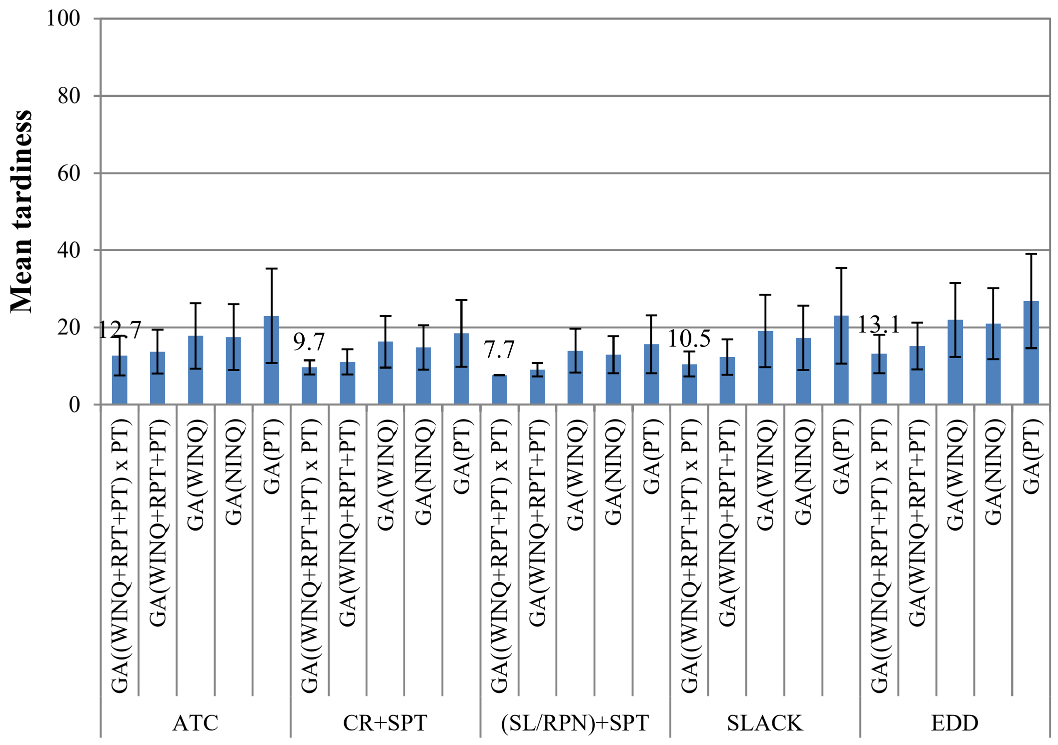


Fig. 3.10 Mean tardiness when the GA is applied only to machine selection in work-center job shops under moderate load.

3.4.4.2 Work-center job shop with low machine utilization

Figures 3.11–3.13 show the results for work-center problems with low machine utilization. Results similar to those under moderate load were obtained, although the mean tardiness values are smaller than those of the moderate conditions. For all three methods of applying the GA, the best results were obtained when using the (SL/RPN)+SPT rule for job selection and (WINQ+RPT+PT)×PT for machine selection. For most combinations of the heuristic rules, good performance was also obtained when applying the GA only to machine selection in this condition. However, applying the GA only to job selection performs best when using the ATC rule or SLACK for job selection and the (WINQ+RPT+PT)×PT rule for machine selection. The overall best result was 6.2 in Fig. 3.13 when using the (SL/RPN)+SPT rule alone for job selection and applying the GA with the (WINQ+RPT+PT)×PT rule for machine selection.

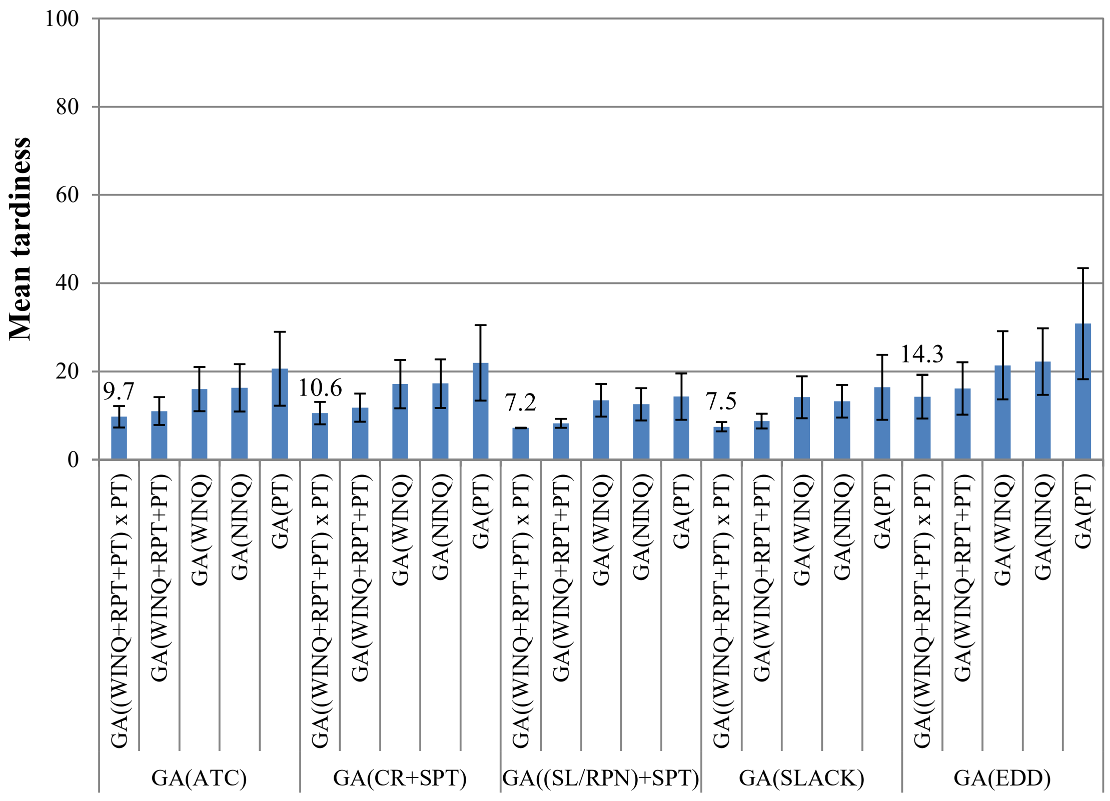


Fig. 3.11 Mean tardiness when the GA is applied to both job and machine selection in work-center job shops with low machine utilization.

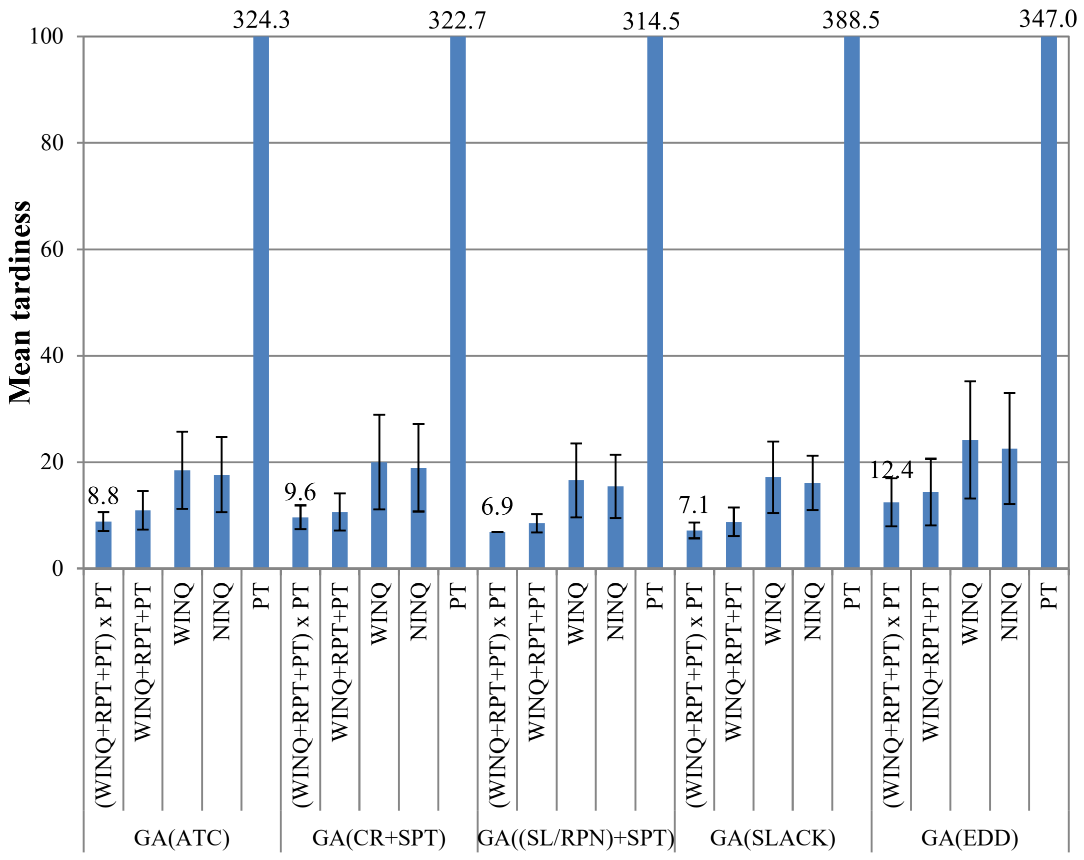


Fig. 3.12 Mean tardiness when the GA is applied only to job selection in work-center job shops with low machine utilization.

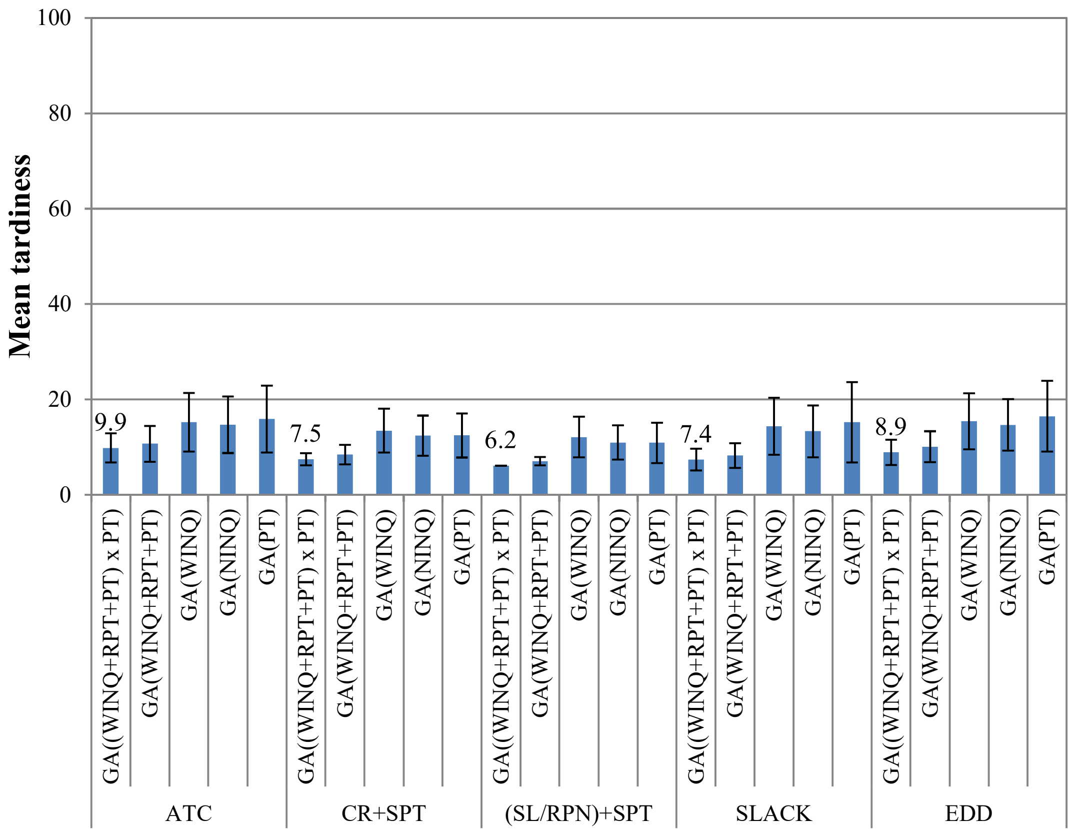


Fig. 3.13 Mean tardiness when the GA is applied only to machine selection in work-center job shops with low machine utilization.

3.4.4.3 Work-center job shop with tight due dates

Figures 3.14–3.16 show the results for work-center problems with tight due dates. Results similar to those under moderate load were obtained, although the mean tardiness values are larger than those of the moderate conditions. For all three methods of applying the GA, the best results were obtained when using the (SL/RPN)+SPT rule for job selection and (WINQ+RPT+PT)×PT for machine selection. For most combinations of the heuristic rules aside from using the (WINQ+RPT+PT)×PT rule for machine selection, good performance was also obtained by applying the GA only to machine selection in this condition. However, the overall best result was 22.7 in Fig. 3.15 when using the GA incorporating the (SL/RPN)+SPT rule for job selection and applying the (WINQ+RPT+PT)×PT rule alone to machine selection.

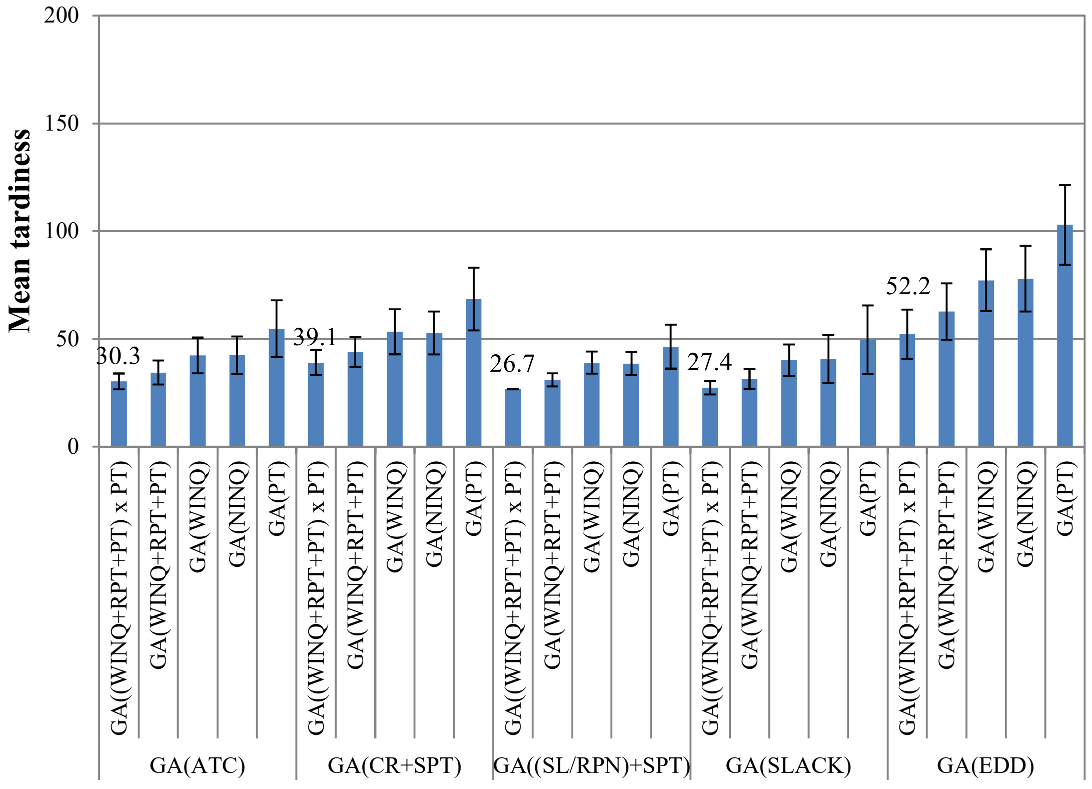


Fig. 3.14 Mean tardiness when the GA is applied to both job and machine selection in work-center job shops with tight due dates.

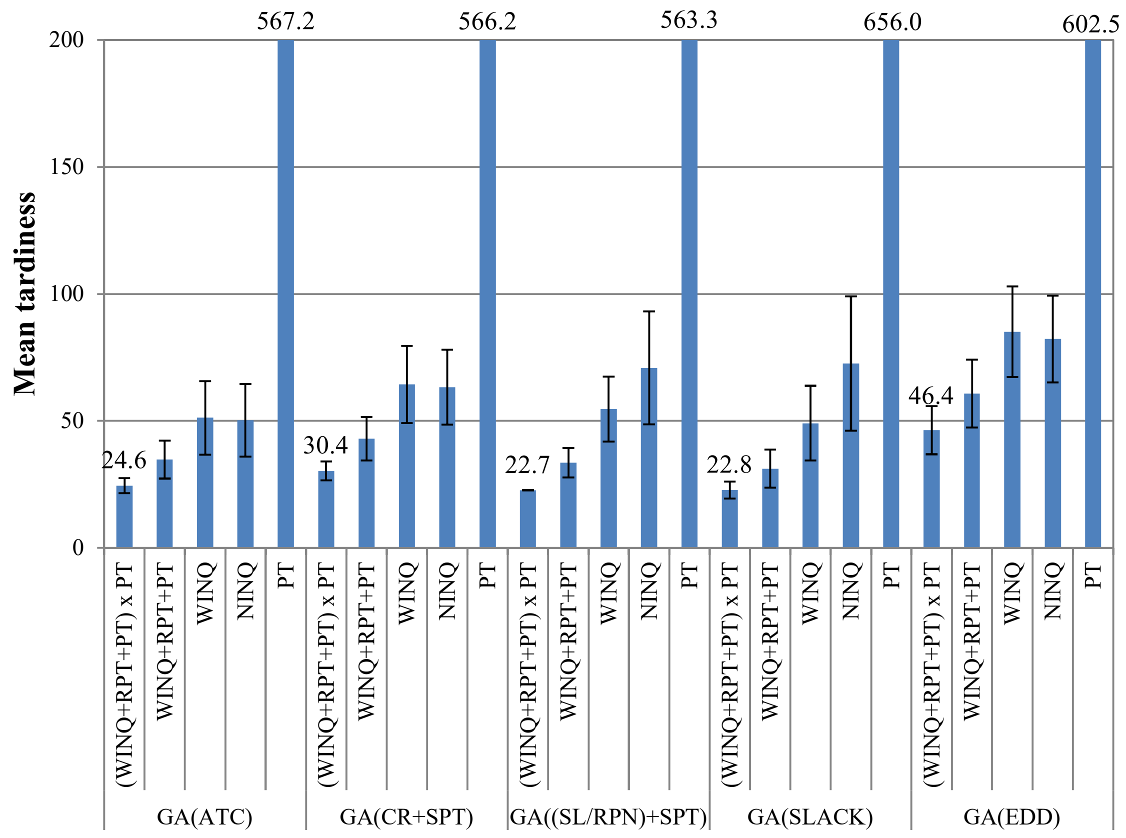


Fig. 3.15 Mean tardiness when the GA is applied only to job selection in work-center job shops with tight due dates.

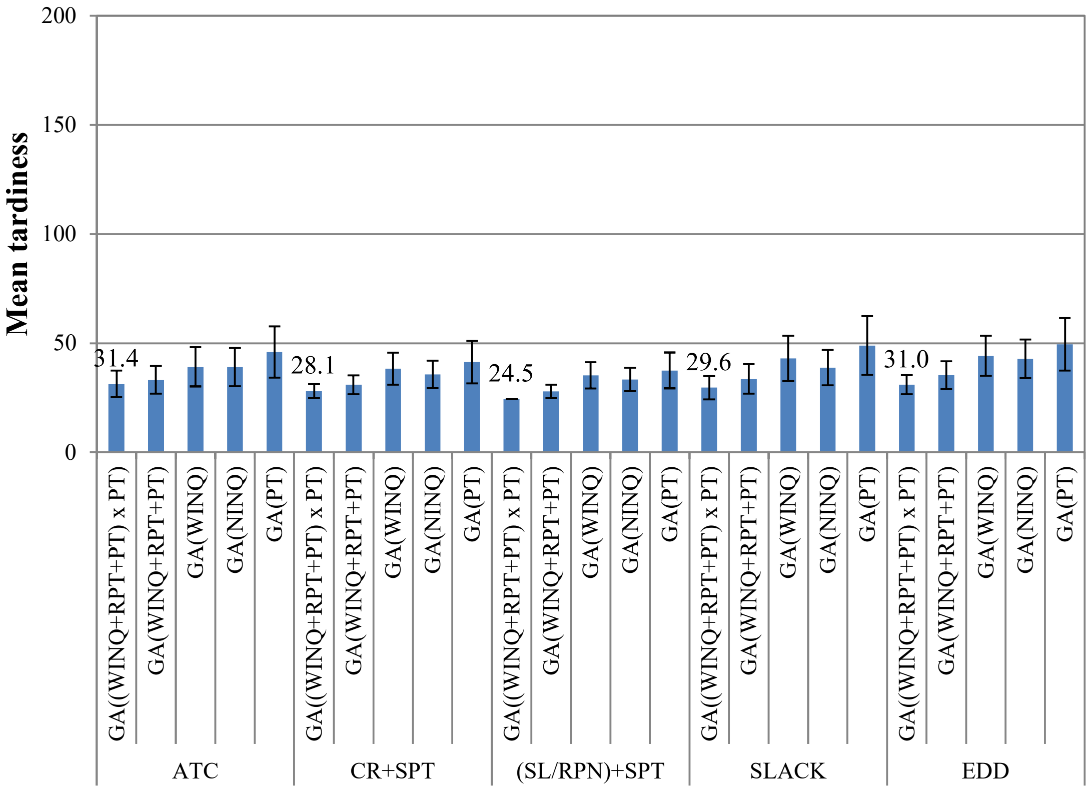


Fig. 3.16 Mean tardiness when the GA is applied only to machine selection in work-center job shops with tight due dates.

3.4.4.4 Random job shop

Figures 3.17–3.19 show the results for the random job shop. For all the three methods of applying the GA, the best results were obtained when using the (SL/RPN)+SPT rule for job selection and (WINQ+RPT+PT)×PT for machine selection. For most combinations of the heuristic rules, good performance was obtained when applying the GA only to machine selection in this condition. The overall best result was 3.4 in Fig. 3.19 when using the (SL/RPN)+SPT rule alone for job selection and applying the GA incorporating the (WINQ+RPT+PT)×PT rule for machine selection.

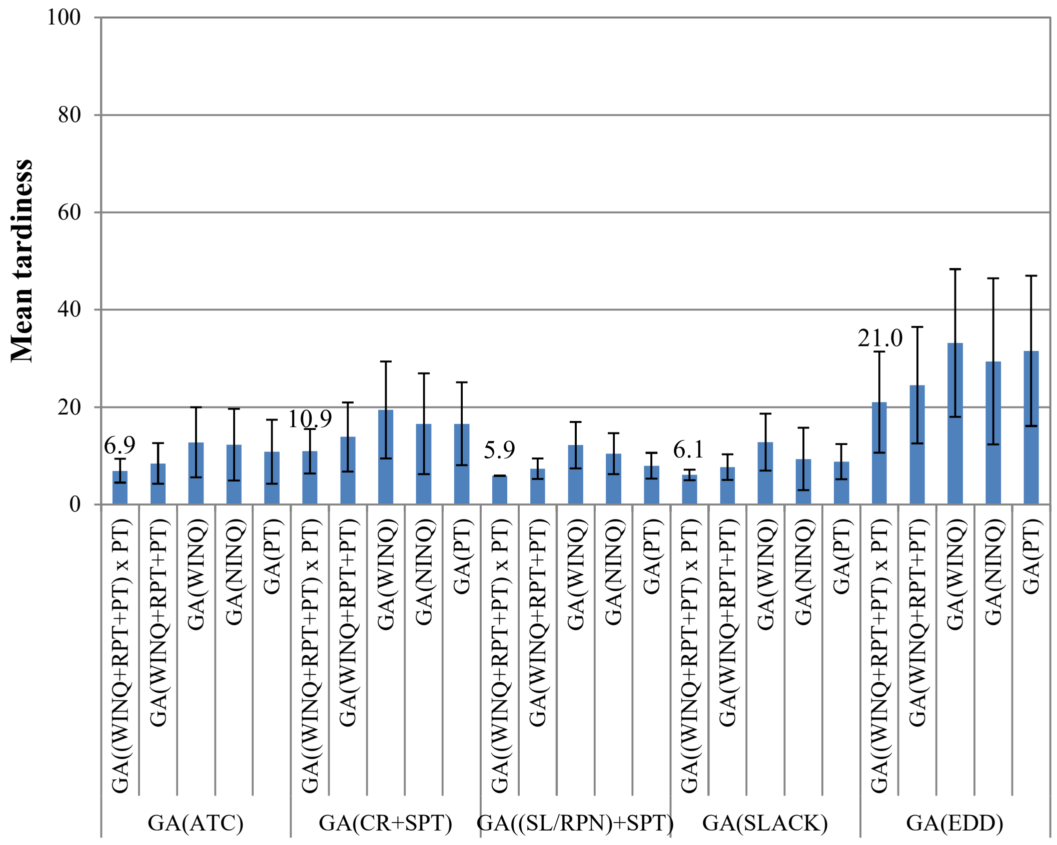


Fig. 3.17 Mean tardiness when the GA is applied to both job and machine selection in the random job shop problem.

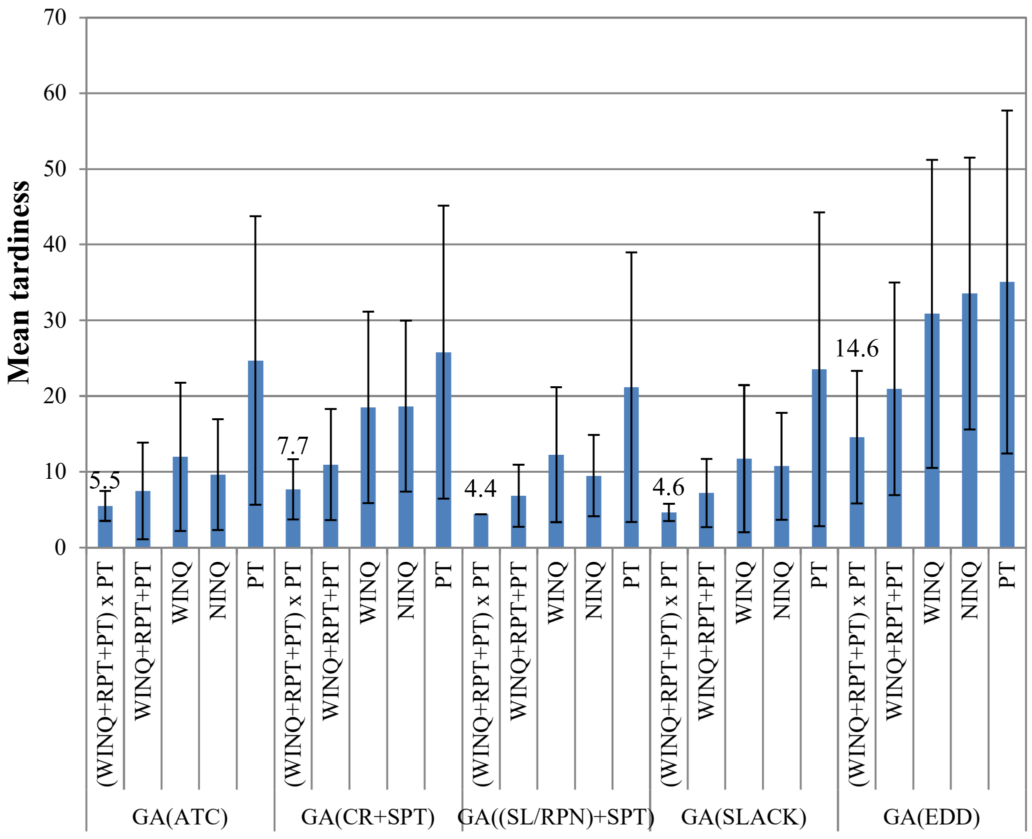


Fig. 3.18 Mean tardiness when the GA is applied only to job selection in the random job shop problem.

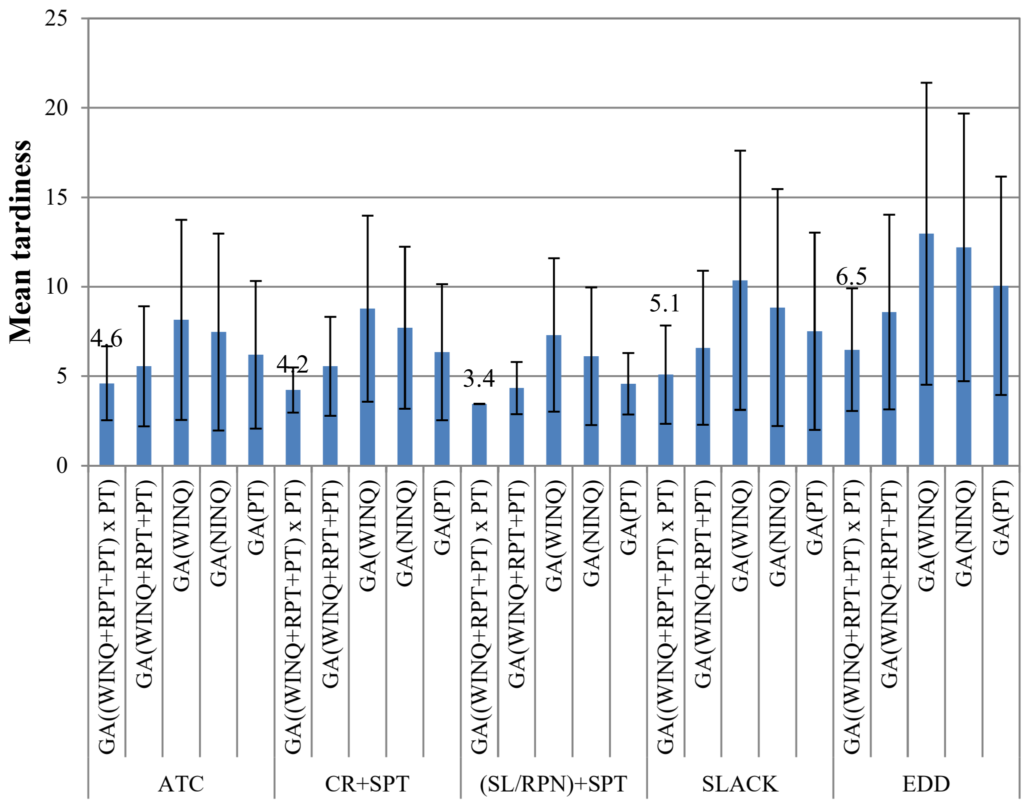


Fig. 3.19 Mean tardiness when the GA is applied only to machine selection in the random job shop problem.

3.4.4.5 Results for smaller scale problems and unbalanced workloads

The results in Sections 3.4.4.1–3.4.4.4 show that effective application of the GA to the two selections depends on the details of the scheduling problem and the heuristic rules used. In most cases, applying the GA only to job or machine selection outperforms applying it to both. To examine the reason for this, two types of new problems are generated: small-scale problems and imbalanced workload problems.

We generated the small-scale problems by decreasing the size of the work-center problem with moderate load by setting the number of work centers to N = 2 and the randomly distributing the number of operations n_i for each job between 2 and 4. The other parameters are for the problems described in Section 3.4.4.1. Thirty problems were randomly generated.

Figure 3.20 shows the generational transitions of the best individuals using the three GA types. The results show the mean values of the 30 problems. In this experiment, the (SL/RPN)+SPT rule is used for job selection and the (WINQ+RPT+PT)×PT rule is used for machine selection. Figure 3.20 indicates that applying the GA to machine selection is more effective than applying it to job selection in this condition. Applying the GA both to job selection and machine selection performed best in this case. This result suggests the reason why applying the GA both to job selection and machine selection did not perform best for the large-scale problems. When the scale of problem increases, the number of decisions for job selection and machine selection also increases. Therefore, the solution space becomes large. It is difficult to search for an optimal schedule in the large solution space within a limited amount of time. Fixing one of the selections by using an effective heuristic rule alone can restrict the search space and can lead to better schedules as a result.

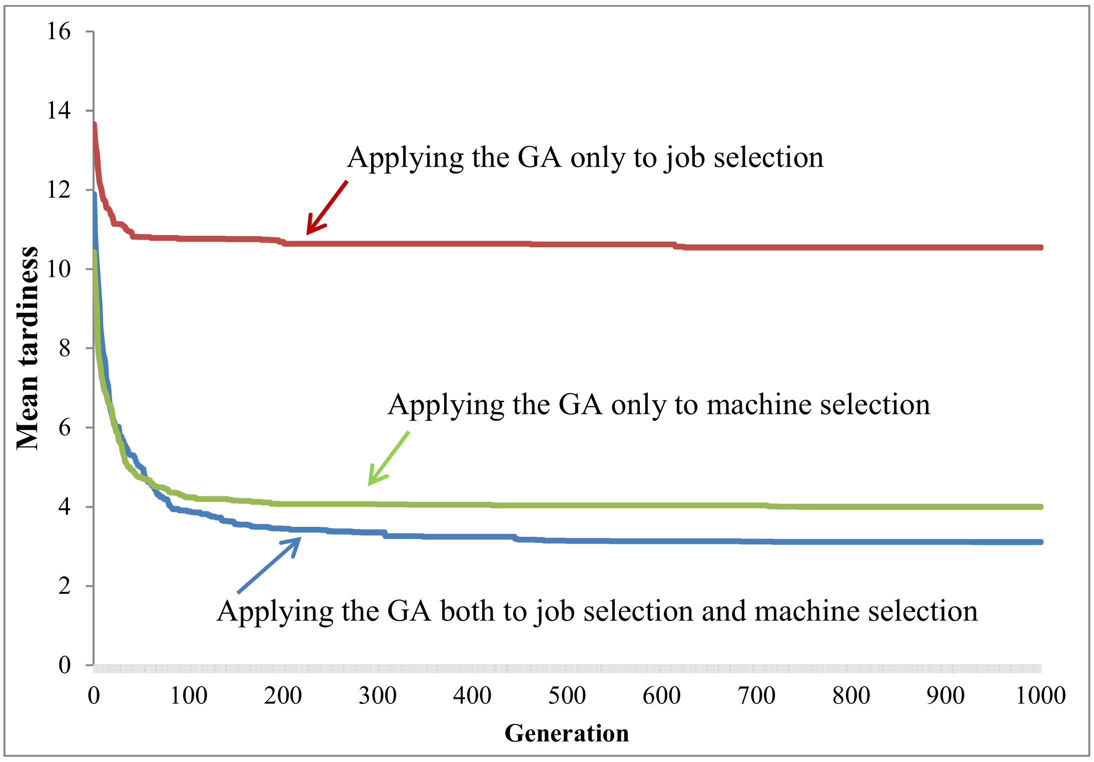


Fig. 3.20 Generational transition of the GAs incorporating heuristic rules for small-scale problems

As a case in which the impact of searching job selections becomes relatively large, we generated problems with imbalanced workloads by modifying the work-center problem with the moderate load. The processing times in two work centers among eight are determined using a uniform distribution between 5 and 200. Those two work centers become bottlenecks in the shop. Other conditions such as the number of machines are set as the same as those for the work center type jobs shop with moderate condition. Thirty problems are generated randomly.

Figure 3.21 shows the generational transition of the best individuals using the three GA types. The results are shown by the mean values of the 30 problems. The (SL/RPN)+SPT rule is used for job selection and the (WINQ+RPT+PT)×PT rule is used for machine selection. Figure 3.21 shows that applying the GA only to job selection outperforms applying the GA only to machine selection in contrast to the results in Figs. 3.8–3.10. In this condition, the lengths of waiting queues in the buffers of certain machines become long and the job selection on the machines becomes more dominant on scheduling performance. In other words, the importance of job selection is relatively low when workload is balanced at the shop floor (Shimoyashiro et al., 1984). In this case, applying the GA both to job selection and machine selection performed best. However, the best method can vary depending on the balance of the impact of job selection and machine selection in the problem to be solved.

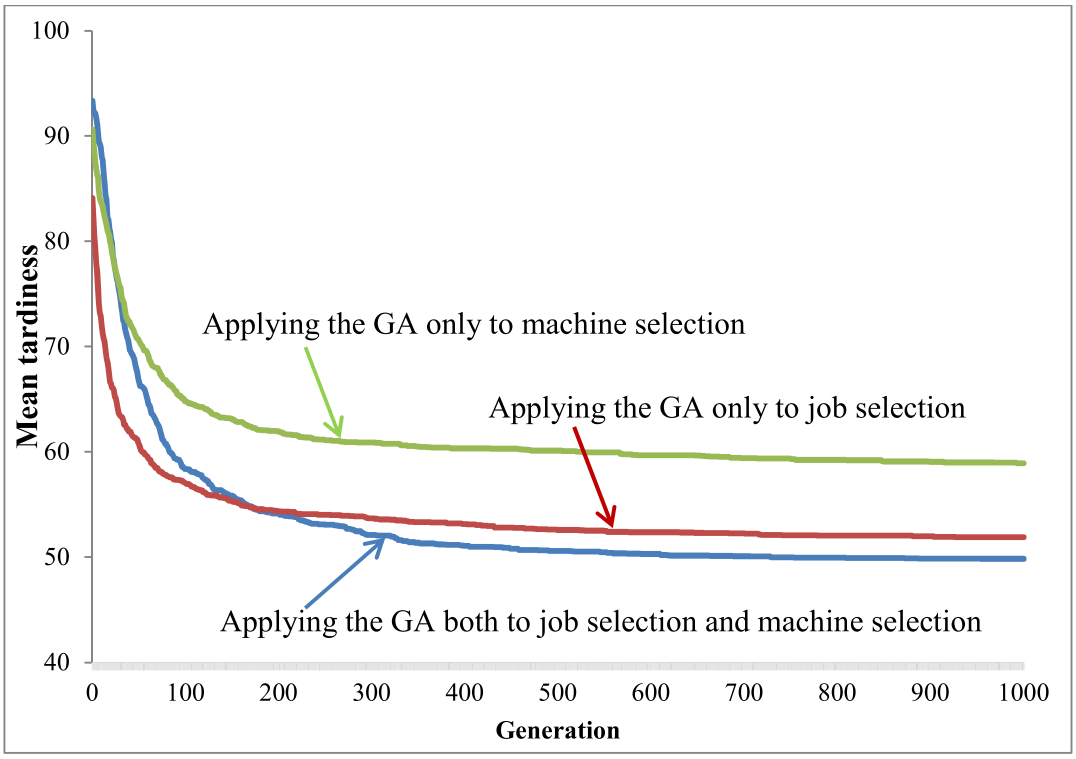


Fig. 3.21 Generational transition of the GAs incorporating heuristic rules for imbalanced workloads.

3.5 Conclusion

This chapter has considered job shop scheduling with the availability of multiple machines to complete an operation. An efficient scheduling method using a genetic algorithm that incorporate heuristic rules has been proposed. The GA uses a random key coding approach not only for job selection but also for machine selection, whose effectiveness has been confirmed through numerical experiment. With this coding approach, heuristic rules can be easily incorporated into the GA. Five job selection rules and five machine selection rules were examined as heuristics for incorporation into the GA. Numerical experiments show that the combination of the (SL/RPN)+SPT rule for job selection and the (WINQ+RPT+PT)×PT rule for machine selection performs best for minimizing the mean tardiness of jobs in various conditions. Numerical experiments also show that applying the GA only to job selection or machine selection performs better than applying the GA both to job and machine selection. These experiments revealed that one of the reasons for this phenomenon is the complexity of searching job and machine selections simultaneously when the scale of the problem is large. The other reason is that the importance of job selection and machine selection varies depending on conditions. In general, if there are sufficient alternative machines available, the effect of job selection diminishes. Therefore, better schedules can be obtained by not applying the GA to job selection in most conditions. However, applying the GA to job selection is also effective under conditions in which job selection is important, such as when bottleneck machines exist.

Chapter 4

Effectiveness of Due-date Related Information for Machine Selection

4.1 Introduction

In the previous chapter, an efficient scheduling method using genetic algorithm incorporating heuristic rules has been proposed. In the method, heuristic rules for job sequencing and machine selection are embedded in the search of genetic algorithm. We have examined various heuristic rules for incorporation when the objective of scheduling is to minimize mean tardiness. As a result, (SL/RPN)+SPT rule performed best for job sequencing. (WINQ+RPT+PT)×PT rule performed best for machine selection. Although the objective of scheduling is to minimize mean tardiness, the machine selection rule does not include due-date related information. Because the objective of scheduling is related to due dates, machine selection using due-date related information seems to be effective. In this chapter, we examine the effectiveness of considering due-date related information for machine selection rules. Numerical experiments are carried out to show the effectiveness of including the information.

4.2 Framework and formulation

The flexible job shop scheduling problem described in the previous chapter is considered in this chapter. The objective is to minimize the mean job tardiness as same as the previous chapter. In this chapter, we focus on the effectiveness of due date related information that included into machine selection rules.

4.3 Algorithm Procedure

4.3.1. Job selection rules

From our previous chapter, the best priority rule for job selection is (SL/RPN)+SPT rule. Therefore, this chapter used this rule for job selection.

4.3.2 Machine selection rules

Also from the previous chapter, five machine selection rules: (WINQ+RPT+PT)×PT, WINQ＋RPT＋PT, WINQ, RPT and PT are used in this chapter. The rules as above are not using due-date related information. In this chapter we also examine new machine selection rules using due-date related information as follows.

- MAX MIN SLACK: The rule calculates slack value (= due-date – current time – sum of remaining processing time) for each waiting job in the input buffer of machine. Then the minimum slack value is selected for each machine. Finally, the machine which has the largest value of the minimum slack value is selected as the machine to process the next operation. The objective of this rule is to select the machine which has larger slack in terms of minimum value.

- MAX TOTAL SLACK: The rule calculates the sum of slack values for all the waiting jobs in the input buffer of machine. Then the machine which has the largest value of the sum of slack values is selected as the machine to process the next operation. The objective of this rule is to select the machine which has larger slack in terms of total value.

- MIN MAX (SL/RPN)+SPT: The rule calculates the value of equation (3.4) for each waiting job in the input buffer of machine. Then the maximum value of it is selected for each machine. Finally, the machine which has the smallest value of the maximum value is selected as the machine to process the next operation. The smaller value of equation (3.4) corresponds to larger slack value. Therefore, the machine with smaller value of the maximum value of equation (3.4) is selected.

- MIN TOTAL (SL/RPN)+SPT: This rule calculates the sum of the values of equation (3.4) for all the waiting job in the input buffer of machine. Then the machine which has the smallest value of the sum of the values is selected as the machine to process the next operation.

- MIN MAX CR+SPT: This rule uses CR+SPT rule (Eq.3.3) instead of (SL/RPN)+SPT in MIN MAX (SL/RPN)+SPT.
- MIN TOTAL CR+SPT: This rule uses CR+SPT rule (Eq.3.3) instead of (SL/RPN)+SPT in MIN TOTAL (SL/RPN)+SPT.
- MIN MAX ATC: This rule uses ATC rule (Eq.3.2) instead of (SL/RPN)+SPT in MIN MAX (SL/RPN)+SPT.
- MIN TOTAL ATC: This rule uses ATC rule (Eq.3.2) instead of (SL/RPN)+SPT in MIN TOTAL (SL/RPN)+SPT.
- MAX MIN EDD: This rule selects the earliest due-date in the input buffer of machine. Then the machine with the maximum value of it is selected as the next machine to process the next operation.
- MAX TOTAL EDD: This rule calculates the sum of the due-date for all waiting job in the input buffer of machine. Then the machine which has the largest value of it is selected as the machine to process the next operation.

4.4 Numerical experiments

4.4.1 Numerical conditions

Numerical experiments are carried out to examine the effectiveness of the method in this chapter. The number of jobs is 100. There are eight work centers (N=8) in the shop floor. Each work center has two machines (L_q=2). Any operation assigned to work center can be processed on both machines in the work center. The number of operations n_i for each job is randomly determined between 4 and 8. The order of work centers to process each operation of a job is determined randomly. It is assumed that one of the machines in a work center can process the operations faster than the other machine. The processing time of an operation on the most efficient machine is determined by the uniform distribution between 5 and 100. The processing time on the other machine in the same work center is determined by multiplying a speed factor by the processing time on the most efficient machine. The speed factor is randomly determined by the uniform distribution between 1 and 2. The due dates of jobs are determined based on TWK method. The problems with two different levels of due-date tightness are generated. When the due-dates of jobs are loose, the number of tardy jobs is set to about 10%-15% by tuning the due-date factor in TWK method. When the due-dates of jobs are tight, the number of tardy jobs is set to about 25%-30%. The thirty problems are randomly generated. The scheduling performance is evaluated by the mean value of equation (3.1) for the thirty problems and the standard deviation from the best rules. For all the conditions, (SL/RPN)+SPT rule is used for the job selection rule.

4.4.2 Experimental results

Table 4.1 and Table 4.2 show the experimental results when using heuristic rules alone. These are the results obtained not using the genetic algorithm. For both due-date tightness levels, (WINQ+RPT+PT)×PT rule for machine selection performed best. The best rule using due-date related information was MAX MIN SLACK.

Table 4.3 and Table 4.4 show the experimental results when using the genetic algorithm incorporating heuristic rules. MAX MIN SLACK rule is used as a machine selection rule which includes due-date related information. When incorporating MAX MIN SLACK rule, the minimum slack value can be negative. Therefore, when the value of minimum slack is s, the value $s^{+}=\exp(s)$ is calculated and the maximum value of the multiplication of s^{+} and the value of gene is used for machine selection in the genetic algorithm. The results in Table 4.3 and Table 4.4 indicate that (WINQ+RPT+PT)×PT rule performed best for incorporation for both due-date tightness levels.

Table 4.1 Mean tardiness using heuristic rules alone in the loose due-date condition

Machine selection rules	Mean tardiness	S.D. from the best
PT	755.8	102.5
NINQ	56.9	16.5
WINQ	59.7	18.4
WINQ+RPT+PT	34.9	11.8
(WINQ+RPT+PT)×PT	24.0	0.0
MAX MIN SLACK	60.3	19.9
MAX TOTAL SLACK	70.6	23.2
MIN MAX (SL/RPN)+SPT	68.1	21.9
MIN TOTAL (SL/RPN)+SPT	66.0	20.2
MIN MAX CR+SPT	85.8	27.8
MIN TOTAL CR+SPT	72.7	21.9
MIN MAX ATC	73.2	24.2
MIN TOTAL ATC	66.1	19.8
MAX MIN EDD	61.7	18.6
MAX TOTAL EDD	82.0	29.6

Table 4.2 Mean tardiness using heuristic rules alone in the tight due-date condition

Machine selection rules	Mean tardiness	S.D. from the best
PT	837.6	92.2
NINQ	97.9	17.1
WINQ	99.6	17.6
WINQ+RPT+PT	65.5	9.1
(WINQ+RPT+PT)×PT	45.2	0.0
MAX MIN SLACK	96.4	19.4
MAX TOTAL SLACK	103.2	18.6
MIN MAX (SL/RPN)+SPT	107.8	25.1
MIN TOTAL (SL/RPN)+SPT	101.2	21.5
MIN MAX CR+SPT	126.1	23.4
MIN TOTAL CR+SPT	110.2	22.1
MIN MAX ATC	114.3	25.3
MIN TOTAL ATC	108.5	22.5
MAX MIN EDD	99.4	19.4
MAX TOTAL EDD	123.2	28.5

Table 4.3 Mean tardiness using the genetic algorithm incorporating heuristic rules in the loose due-date condition

Machine selection rules	Mean tardiness	S.D. from the best
GA+PT	22.4	8.5
GA+NINQ	16.9	5.3
GA+WINQ	17.4	5.4
GA+WINQ+RPT+PT	12.0	2.4
GA+ (WINQ+RPT+PT)×PT	9.9	0.0
GA+MAX MIN SLACK	20.4	8.0

Table 4.4 Mean tardiness using the genetic algorithm incorporating heuristic rules in the tight due-date condition

Machine selection rules	Mean tardiness	S.D. from the best
GA+PT	46.4	10.2
GA+NINQ	38.6	5.4
GA+WINQ	39.0	5.2
GA+WINQ+RPT+PT	31.1	3.0
GA+ (WINQ+RPT+PT)×PT	26.7	0.0
GA+MAX MIN SLACK	45.5	9.8

4.5 Conclusion

In this chapter, various machine selection rules were examined in the genetic algorithm incorporating heuristic rules for flexible job shop scheduling. Because the objective function for scheduling is mean tardiness, due-date related information seems to be important not only for job selection also for machine selection. However, the experimental results show that the best machine selection rule is (WINQ+RPT+PT)×PT rule both when using heuristic rules alone and when applying the genetic algorithm incorporating the heuristic rules. (WINQ+RPT+PT)×PT rule works for load balancing. Numerical results suggest that it is not necessary to include due-date related information in machine selection if the due-date related information is considered in job selection and load balancing is considered in machine selection appropriately.

Chapter 5

Conclusions

5.1 Conclusions

This study aims to develop the efficient scheduling method for production systems with alternative machines using genetic algorithm. Genetic algorithm is one of the most popular meta-heuristics applied to production scheduling. Although genetic algorithm is effective for scheduling, it has also weakness for fine tuning after finding good solutions. In addition, genetic algorithm itself does not utilize problem-specific knowledge, there is room for performance improvement. This study has investigated incorporating effective knowledge to improve the efficiency of scheduling using genetic algorithm.

This research consists of two approaches for production systems with alternative machines. First, the multi-stage flexible flow shop scheduling problem with regard to minimize the mean tardiness was discussed. The genetic algorithm incorporating hill climbing using effective local structure, also called memetic algorithm, was examined. In the experiment, three meta-heuristics: genetic algorithm

(GA), memetic algorithm (MA) and random key based genetic algorithm (RKGA) were compared. The numerical experiments showed that MA was better than GA. GA can be improved by incorporating the local search using effective local structure. However, the performances of MA and RKGA are similar, and the RKGA performs better when problem size is getting larger.

Second, flexible job shop scheduling problem with regard to minimize the mean tardiness was discussed. Genetic algorithm incorporating heuristic rules was investigated. For this problem, RKGA was applied with incorporating various heuristic rules for job selection and machine selection. Numerical experiments showed that the combination of the (SL/RPN)+SPT rule for job selection and the (WINQ+RPT+PT)×PT rule for machine selection performed best for minimizing the mean tardiness of jobs in various conditions. Numerical experiments also showed that applying the GA only to job selection or machine selection performed better than applying the GA both to job and machine selection. This suggest that it is very difficult to search optimal schedules for large scale problems with alternative machines; restricting search space by effective knowledge can improve the performance of scheduling.

Furthermore, the effective of due-date related information of machine selection rules was examined. Numerical results suggest that it is not necessary to include due-date related information in machine selection if the due-date related information is considered in job selection and load balancing is considered in machine selection appropriately.

Bibliography

[1] Abyaneh S.H. and Zandieh M., Bi-objective hybrid flow shop scheduling with sequence-dependent setup times and limited buffers, International Journal of Advanced Manufacturing Technology, Vol.58, pp.309–325, (2012).

[2] Akhshabi M., Akhshabi M. and Khalatbari J., A Memetic Algorithm for Hybrid Flow Shop Scheduling with Multiprocessor Task Problems, Journal of Basic and Applied Scientific Research, Vol.1, No.12, pp.3053–3057, (2011).

[3] Akhshabi M., Taghavifard S.M. and Akhshabi M., Hybrid flow shop scheduling problem with set up depend sequence with respect to PM, Indian Journal of Fundamental and Applied Life Sciences, Vol. 2, No.2, pp.244–250, (2012).

[4] Aldowaisan T. A., A new heuristic and dominance relations for no-wait flow shops with setup times, Computers & Operations Research, Vol. 28, No. 6, pp. 563–584, (2001).

[5] Aldowaisan T. A. and Allahverdi A., Total flowtime in no-wait flowshops with separated setup times. Computers & Operations Research, Vol. 25, No. 12, pp.1025–1031, (1998).

[6] Allahverdi A., Scheduling in stochastic flow shops with independent setup processing and removal times, Computers & Operations Research, Vol. 24, No. 11, pp. 995–960, (1997).

[7] Allahverdi A. and Aldowaisan T., Minimizing total completion time in a no-wait flow shop with sequence-dependent additive changeover times, Journal of the Operational Research Society, Vol. 52, pp. 449–462, (2001).

[8] Allahverdi A. and Aldowaisan T., No-wait flow shops with bi-criteria of makespan and total completion time, Journal of the Operational Research Society, Vol. 53, No. 9, pp. 1004–1015, (2002).

[9] Allahverdi, A., Gupta, J., and Aldowaisan, T., A review of scheduling research involving setup considerations, Omega, Vol.27, pp.219–239, (1999).

[10] Anderson, E. J. and Nyirenda, J. C., Two new rules to minimize tardiness in a job shop, International Journal of Production Research, Vol.28, No.12, pp.2277–2292, (1990).

[11] Baker, K.R., Sequencing rules and due-date assignments in a job shop, Management Science, Vol.30, No.9, pp.1093–1104, (1984).

[12] Bean, J. C., Genetic algorithms and random keys for sequencing and optimization, ORSA Journal on Computing, Vol.6, No.2, pp.154–160, (1994).

[13] Brucker, P., Jurisch, B. and Krämer, A., Complexity of scheduling problems with multi-purpose machines, Annals of Operations Research, Vol.70, pp.57–73, (1997).

[14] Candido, M. A. B., Khator, S. K. and Bracia, R. M., A genetic algorithm based procedure for more realistic job shop scheduling problems, International Journal of Production Research, Vol.36, No.12, pp.3437–3457, (1998).

[15] Chan, F.T.S., Wong, T.C. and Chan, L.Y., Flexible job-shop scheduling problem under resource constraints, International Journal of Production Research, Vol.44, No.11, pp.2071–2089, (2006).

[16] Chang J., Yan W., Shao H.., Scheduling a two-stage no-wait hybrid flow shop with separated setup and removal times, Proc. of the 2004 American Control Conference. Boston, pp. 1412–1416, (2004).

[17] Choong F., Phon-Amnuaisuk S. and Alias M.Y., Metaheuristic methods in hybrid flow shop scheduling problem, Expert Systems with Applications, Vol.38, pp.10787–10793, (2011).

[18] Conway, R.W., Priority dispatching and job lateness in a job shop, The Journal of Industrial Engineering, Vol.16, No.4, pp.228–237, (1965).

[19] Doh, H.H, Yu, J.M, Kim, J.S., Lee, D.H. and Nam S.H., A priority scheduling approach for flexible job shops with multiple process plans, International Journal of Production Research, Vol.51, No.12, pp.3748–3764, (2013).

[20] Du, J. and Leung, J. Y.-T., Minimizing Total Tardiness on One Machine is NP-hard, Mathematics of Operations Research, Vol. 15, No. 3, pp. 483-495, (1990).

[21] Eguchi, T., Oba, F. and Kozaki.S., Dynamic Scheduling Using the Mixture of a Genetic Algorithm and a Priority Rule, Transactions of the Japan Society of Mechanical Engineers Series C, Vol.71, No.703, pp.1047–1053, (2005). (in Japanese)

[22] Eguchi, T., Oba, F.,Toyooka, S. and Sato,Y., Machine selection rule for dynamic job shop having alternative machines with different processing times, Journal of the Japan Society for Precision Engineering, Vol.72, No.4, pp.459–464, (2006). (in Japanese)

[23] Eguchi, T., Kaweegitbundit, P., Hoshino, N., Daido, T., Murayama T., Job shop scheduling with alternative machines - Experimental evaluation of the scheduling method using a genetic algorithm incorporating heuristic rules -. In: Manufacturing Systems Division Conference 2015, No.15-8, pp.95–96, The Japan Society of Mechanical Engineers, (2015)

[24] Engin O. and Döyen A., A new approach to solve hybrid flow shop scheduling problems by artificial immune system, Future Generation Computer Systems. Vol.20, p.1083–1095, (2004).

[25] Engin O., Ceran G. and Yilmaz M.K., An efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems, Applied Soft Computing, Vol.11, p.3056–3065, (2011).

[26] Garey M.R., Tarjan R.E. and Wilfong G.T., One-Processor Scheduling with Symmetric Earliness and Tardiness Penalties, Mathematics of Operations Research, Vol.13, p. 330–348, (1988).

[27] Goldberg D.E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, (1989).

[28] Goldberg D.E. and Lingle, R., Alleles, Loci and the Traveling Salesman Problem, Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications, pp. 154–159, Los Angeles, USA. (1985).

[29] Gupta J.N.D., Hariri A.M.A. and Potts. C.N., Scheduling a two-stage hybrid flow shop with parallel machines at the first stage, Annuals of Operations Research, Vol.69, pp.171–191, (1997).

[30] Gupta J.N.D. and Tunc E.A., Schedules for a two-stage hybrid flowshop with parallel machines at the second stage, International Journal of Production Research, Vol.29 pp. 1489–1502, (1991).

[31] Gupta J. N. D., Strusevich V. A. and Zwaneveld C., Two-stage no wait scheduling models with setup and removal times, Computers & Operations Research, Vol. 24, No. 11, pp.1025–1031, (1997).

[32] Holland J.H., Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, MI, (1975).

[33] Hoogeveen, J.A., J.K. Lenstra and B. Veltman, Preemptive scheduling in a two-stage multiprocessor flow shop is NP-hard, European Journal of Operational Research, Vol.89, No.1–2, pp.172–175, (1996).

[34] Iwata, L, Murotsu, Y., Oba, F., Uemura, T. and Okushima, K., Optimization of selection of machine-tools, loading sequence of parts and machining conditions in job-shop type machine systems, Annals of the CIPR, Vol.27, No.1, pp.447–451, (1978).

[35] Jawahar, N., Aravindan, P. and Ponnambalam, A genetic algorithm for scheduling flexible manufacturing systems, International Journal of Advanced Manufacturing Technology, Vol.14, pp.588–607, (1998).

[36] Kaweegitbundit, P. and Eguchi, T., Job shop scheduling with alternative machines using genetic algorithm incorporating heuristic rules, In: Mechanical Engineering Congress, 2014 Japan [MECJ-14], No.14-1, S1420140, The Japan Society of Mechanical Engineers, Tokyo, (2014).

[37] Lee, H. and Kim, S.S., Integration of process planning and scheduling using simulation based genetic algorithm, International Journal of Advanced Manufacturing Technology, Vol.18, pp.586–590, (2001).

[38] Lenstra, J.K., Rinnooy Kan, A.H.G and Brucker, P., Complexity of machine scheduling problems, Annals of Discrete Mathematics, Vol.1, pp.343–362, (1977).

[39] Li, X., Shao, X., Gao, L. and Qian, W., An effective hybrid algorithm for integrated process planning and scheduling, International Journal of production economics, Vol.126, pp.289–298, (2010).

[40] Liu Z. and Xie J. A heuristic for two-stage no-wait hybrid flow shop scheduling with a single machine in either stage. Tsinghua Science and Technology, Vol. 8, No. 1, pp. 43–48, (2003).

[41] Moon, I., Lee, S. and Bae, H., Genetic algorithm for job shop scheduling problems with alternative routing, International Journal of Production Research, Vol.46, No.10, pp.2695–2705, (2008).

[42] Morad, M. and Zalzala, A., Genetic algorithm in integrated process planning and scheduling, Journal of Intelligent Manufacturing, Vol.10, pp.169–179, (1999).

[43] Nasr, N., Elsayed, E., A., Job shop scheduling with alternative machines, International Journal of Production Research, Vol.28, No.9, pp.1595–1609, (1990).

[44] Norman, B. A. and Bean, J. C., A Genetic algorithms methodology for complex scheduling problems, Naval Research Logistics, Vol.46, pp.199–211, (1999).

[45] Oguz, C., Janiak, A. and Lichtenstein, M., Metaheuristic Algorithms for Hybrid Flow-Shop Scheduling Problem with Multiprocessor Tasks. Proceeding of the 4th Metaheuristics International Conference (MIC 2001), Porto, Portugal, July 16-20, (2001).

[46] Pezzella, F., Morganti, G. and Ciaschetti, G., A genetic algorithm for the flexible job-shop scheduling problem, Computers & Operations Research, Vol.35, pp.3202–3212, (2008).

[47] Pinedo M. L., Scheduling Theory, Algorithm, and System, Fifth Edition, New York: Springer Science and Business Media, (2016).

[48] Phanden, R.K., Jain, A. and Verma R., An approach for integration of process planning and scheduling, International Journal of Computer Integrated Manufacturing, Vol.26, No.4, pp.284–302, (2013).

[49] Ruiz, R., Antonio, J. and Rodriguez, V., The hybrid flow shop scheduling problome, European Journal of Operations Research, Vol.205, Issue 1, pp.1–18, (2010).

[50] Salvador M. S., A solution of a special class of flow shop scheduling problems, in Proc. of the symposium on the theory of scheduling and its applications, Springer-Verlag, Berlin, pp. 83–91,(1973).

[51] Shao, X., Li, X., Gao, L. and Zhang, C., Integration of process planning and scheduling–A modified genetic algorithm-based approach, Computers & Operations Research, Vol.36, pp.2082–2096, (2009).

[52] Shimoyasiro, S., Isoda, K. and Awane, H., Input scheduling and load balance control for a job shop, International Journal of Production Research, Vol.22, No.4, pp.597–605, (1984).

[53] Sriskandarajah C., Performance of scheduling algorithms for no-wait flow shops with parallel machines. European Journal of Operational Research, Vol. 70, No. 3, pp. 365–378, (1993).

[54] Syam W. P. and Al-Harkan I. M., Comparison of Three Meta Heuristic to Optimize Hybrid Flow Shop Scheduling Problem with Parallel Machines, World Academy of Science, Engineering and Technology. Vol.62, pp. 271–278, (2010).

[55] Tavakkoli-Moghaddama, R., Safaeic, N. and Sassanib, F., A memetic algorithm for the flexible flow line scheduling problem with processor blocking, Computers & operaitons research, Vol.36, pp.402–414, (2009).

[56] Vepsalainen, A. P. J. and Morton, T. E., Priority rules for job shops with weighted tardiness costs, Management science, Vol.33, No.8, pp.1035–1047, (1987).

[57] Wang, H., Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions, Expert Systems, Vol.22, No.2m pp.78–85, (2005).

[58] Yagiura M. and Ibaraki T., On Metaheuristic Algorithms for Combinatorial Optimization Problems, Systems and Computers in Japan, Vol. 32, Issue 3, pp. 33-55, (2001)

[59] Yoda, M., Eguchi, T. and Murayama T., Job shop scheduling for meeting due dates and minimizing overtime using genetic algorithm incorporating new priority rules, Journal of Advanced Mechanical Design, Systems, and Manufacturing, Vol.8, No.5, Paper No.14–00076, (2014).

PUBLICATIONS

International Journal

[1] Memetic algorithm approach to two-stage hybrid flow shop scheduling problem with identical parallel machines, Advanced Materials Research, 651, pp. 548-552, (2013).

[2] Flexible job shop scheduling using genetic algorithm and heuristic rules, Journal of Advanced Mechanical Design, Systems, and Manufacturing, Vol.10, No.1, Paper No.15–00434, (2016).

International Conference

[1] Job Shop Scheduling with Alternative Machines Using a Genetic Algorithm Incorporating Heuristic Rules -effectiveness of Due-date Related Information-. APMS2015 International Conference, Advance in Production Management Systems, Part I, pp. 439–446, (2015).

Acknowledgements

I would like to express my sincere thankfulness to Assoc. Prof. Toru Eguchi, my great advisor for the continuous supports from the beginning to the completion of this thesis. Without this support, I would not be able to accomplish my study.

I also acknowledge the members of my dissertation committee, Prof. Keiji Yamada, Prof. Nobutaka Wada, and Assoc. Prof. Yasuhisa Sekiguchi for giving their generously time and expertise to improve my work. I sincerely thank them for their helpful suggestions, insightful discussions, and their good-natured support.

I also thank all laboratory members for friendship, many suggestions and all helping that made me very happiness during the time in Japan.

Finally, my special thanks to all of my family and my friend for all everything supporting in my life.