

南 華 大 學

資訊管理學系

碩士論文

具搜尋區域調適性之塔布基因演算法在零工式排  
程問題上的應用

A Tabu Genetic algorithm with Search Area  
Adaptation for the Job-Shop Scheduling Problem

研 究 生：王靜瑜

指 導 教 授：邱宏彬

中 華 民 國 九 十 五 年 六 月 三 日

南 華 大 學  
資訊管理研究所  
碩 士 學 位 論 文

A Tabu Genetic Algorithm with Search Area Adaptation for the  
Job-Shop Scheduling Problem

研究生： 王靜莉

經考試合格特此證明

口試委員： \_\_\_\_\_


李翔北

謝品霖

\_\_\_\_\_

\_\_\_\_\_

指導教授： 邱宏樹

系主任(所長)： 

口試日期：中華民國 95 年 6 月 3 日

## 誌 謝

光陰似箭，兩年的研究生涯即將告一段落。回想起初進研究所時，自己是這樣的懵懂無知，如今我能完成研究所的學業要感謝教導過我的資管所所有老師，謝謝您們的辛勞教誨讓我能順利完成學業。

很幸運能加入 204 研究室這個大家庭，得到老師及眾多同學的幫忙，大家教學相長、相互勉勵，使得我的研究生涯相當充實精采。尤其感謝邱宏彬老師在學術研究及生活思想都給予我很大的啟發與思考，讓我學會站在不同角度去看待事物。還有系助理：伊汝；學長姐：建源、文天、琪雯、美倫；同學：阿聰、琬蓉、施老師、大個、李老師、富一，大家的支持與幫助，我才能有今天的小小成就。最後感謝我的家人，有您們的鼓勵，我才能夠順利求學、畢業。謹以此文表達我誠摯的謝意。

王靜瑜 僅誌於 嘉義  
南華大學資訊管理學系  
中華民國 九十五年 六月

# 具搜尋區域調適性之塔布基因演算法

## 在零工式排程問題上的應用

學生：王靜瑜

指導教授：邱宏彬

南 華 大 學 資 訊 管 理 學 系 碩 士 班

### 摘 要

零工式排程問題(Job shop scheduling problem, JSP)是個不斷被廣泛研究的一項重要議題，並具有 NP-complete 問題的特性。而遺傳演算法(Genetic Algorithm, GA)是種常用於解決最佳化排程問題的演算法，GA 運用多點搜尋的方法使目標解得以多樣化，但須克服在搜尋過程中過早收斂而落入區域最佳解(local minimum)的問題。因此本研究以塔布遺傳演算法(TGA)為基礎搭配區域搜尋的機制，來找到 JSP 的最佳解，稱作改良式塔布遺傳演算法(Modified Tabu genetic algorithm, MTGA)。此方法在產生子代的交配過程中加入禁區(Tabu list)的觀念去避免同系繁殖，更進一步以 Tabu search 的凌駕規則(aspiration criterion)輔助子代的挑選，提升子代母體之強化性，再利用自適性的突變方法(self-adaptive mutation)降低落入 local minimum 的機會。並以區域搜尋的機制加強區域搜尋能力找出最佳排程解。本研究以標竿問題進行實驗，結果顯示 MTGA 和其他方法比起來具有很優異的表現。

**關鍵字：**遺傳演算法、零工式排程問題、塔布搜尋。

# **A Tabu Genetic Algorithm with Search Area Adaptation for the Job-Shop Scheduling Problem**

Student : Ching-Yu Wang

Advisors : Dr. Hung-Pin Chiu

Department of Information Management  
The M.B.A. Program  
Nan-Hua University

## **ABSTRACT**

The job-shop scheduling problem is the important issues to the research of optimal problems. Besides, tabu search is also applied to GA, called TGA for traveling salesman problem (TSP) that has better effectiveness than GA [6]. Thus, this is an interest and important research area for job-shop scheduling problem with TGA. In this paper, we try to discuss this issue.

According to the TGA, it maintains diversity through broad-sense incest prevention. Therefore, the solutions can contain their diversity and prevent premature convergence. But in JSP problem, the crossover and mutation manners of TGA cannot produce the better solutions than GA. So we modified the crossover and mutation phases of TGA called modified TGA (MTGA). First, the modified crossover search phase uses a threshold ( $TH_c$ ) to control the times of crossover for improving the qualities and convergence of solutions. Second, the mutation search phase use two parameters to control the selected points and the times of mutation in order to make the global search wildly and prevent to drop into local minimum more easily. And the experiments results demonstrate the superiority of MTGA in job-shop scheduling problems. Not only balance intensification, but also diversification.

**Keywords: Genetic algorithm, Job-shop scheduling problem, Tabu search.**

# List of Contents

Chapter 1 Introduction .....	1
1.1. Job-shop scheduling problem (JSP).....	2
Chapter 2 Literature review .....	4
2.1 Genetic Algorithm for Job-shop Scheduling Problem.....	4
2.2 Tabu Genetic Algorithm (TGA).....	6
2.2.1 Tabu List.....	11
2.2.2 Aspiration criterion .....	12
2.2.3 Self-adaptive mutation .....	12
Chapter 3 Modified TGA for Job-shop scheduling problem.....	14
3.1 Tabu genetic algorithm for JSP.....	14
3.1.1 Designing chromosomes .....	14
3.1.2 Calculating fitness value .....	15
3.1.3 Tabu list .....	15
3.1.4 Crossover.....	16
3.1.5 Mutation .....	18
3.2 Modified TGA.....	18
3.2.1 Modified crossover search phase .....	21
3.2.2 Modified mutation search phase .....	22
Chapter 4 Performance evaluation .....	23
4.1 Tabu list.....	23
4.2 Performance comparison .....	25
Chapter 5 Conclusion.....	27
References .....	28

## List of Tables

Table 1-1. : Example of $3 \times 3$ JSP problem.....	3
Table 1-2. : Computation-based and Memory-based mating.....	8

## List of Figures

Figure 1-1. : A Chromosome of $3 \times 3$ JSP problem.....	3
Figure 2-1. : Flowchart of simple genetic algorithm .....	5
Figure 2-2. : Flowchart of TGA.....	10
Figure 3-1. : Example of representation for JSP.....	15
Figure 3-2. : Update the tabu list.....	16
Figure 3-3. : Example of crossover in TGA for JSP.....	17
Figure 3-4. : Example of mutation in TGA for JSP.....	18
Figure 3-5. : Flowchart of MTGA.....	19
Figure 4-1. : Comparison of tabu list parameter in TGA ( $10 \times 10$ ).....	24
Figure 4-2. : Comparison of deadlock in TGA ( $10 \times 10$ ).....	24
Figure 4-3. : Comparison GA, TGA, TGA* and MTGA ( $10 \times 10$ ).....	26
Figure 4-4. : Frequency distribution of makespan ( $10 \times 10$ ).....	26



# Chapter 1 Introduction

Current market trends, shorter product life cycles and competitive pressure to reduce costs have resulted in the need for zero inventory systems. In order to maintain market share, the system must be fast responding which implies that more stock has to be maintained. These conflicting requirements demand efficient, effective and accurate scheduling which is complex in all but the simplest production environment [2]. So, scheduling problems need to be solved by good scheduling algorithms and heuristics. Job-shop scheduling problem (abbreviated to JSP) is one of the hardest well-known combinatorial optimization problems. The problem is an allocation of the operations to time intervals on the machines, in order to find a minimum makespan.

Genetic algorithms (GAs) are well-known heuristic algorithms and have been applied to solve a variety of complicated problem. In recent years, there is an interest of in using genetic algorithms for solving JSP [16, 18, 13]. GA has shown a good performance regarding its ability to search globally. It searches multiple points in the search space of population. It also uses a crossover operator that enables to search wider region. The diversity in GA is attributed to the form of population, which contains a certain number of encoded individuals for population. Therefore, heuristic algorithms pursue a good balance between exploration and exploitation in consideration of both convergence speed and optimized solution quality. However, in the conventional GA, parents are approved without any further examination after they are chosen at random or just by fitness. So, it's hard to prevent the ancestry mating and control the solution quality.

In this research, we adapt tabu genetic algorithm, (TGA), for solving JSP problem. Furthermore, to modify tabu genetic algorithm, called MTGA that with local search mechanisms in crossover and mutation search phases to contain the diversification and intensification of solutions. Finally, using several experimental results to prove MTGA has better performance than GA and TGA.

In this paper, we try to discuss this issue and its organized as follows. In section2, we review relevant literature related to our study. In section 3, we modified TGA, called MTGA for JSP and explained the process in detail. In section 4, evaluate and discuss the experimental results with our research. Finally, conclusions and future works are summarized in section 5.

### 1.1. Job-shop scheduling problem (JSP)

In general, the classical JSP can be stated as follows [12]: There are  $n$  different jobs and  $m$  different machines to be scheduled. Each job is composed of a set of operations and the operations order on machines is prespecified. The required machine and the fixed processing time characterize each operation. There are numerous constrains on jobs and machines:

- (1) A job does not visit the same machine twice.
- (2) There are no precedence constrains among the operations of different jobs.
- (3) Operations cannot be interrupted.
- (4) Each machine can process only one job at a time.
- (5) Neither release times nor due dates are specified.

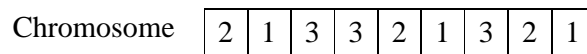
The problem is to determine the operation sequences on the machines for

minimizing the makespan, and the time is necessary to complete all jobs. An example of the three-job three-machine ( $3 \times 3$ ) JSP problem is presented in Table 1-1.

**Table 1-1. Example of  $3 \times 3$  JSP problem**

	Operation (machine number /processing time)		
Job	Op <sub>1</sub>	Op <sub>2</sub>	Op <sub>3</sub>
$J_1$	1 / 3	2 / 3	3 / 2
$J_2$	1 / 1	3 / 5	2 / 3
$J_3$	2 / 3	1 / 2	3 / 3

A chromosome has gene information that shows the order of the job-number for solving the problem in GA. If the number of jobs is  $n$  and the number of machines is  $m$ , the chromosome consists of  $n \times m$  genes. Each job must be depending on an order relation; it will appear  $m$  times exactly. Thus, each chromosome represents a feasible solution. In a  $3 \times 3$  job-shop scheduling problem, 9 genes denote a chromosome, '1' shows job1 ( $J_1$ ), '2' shows job2 ( $J_2$ ), '3' shows job3 ( $J_3$ ), respectively. Each gene, operation ( $Op$ ), is given priority from left to right. Therefore, a left gene has higher priority than right one. An example of a chromosome in a  $3 \times 3$  job-shop scheduling problem is shown in Figure 1-1.



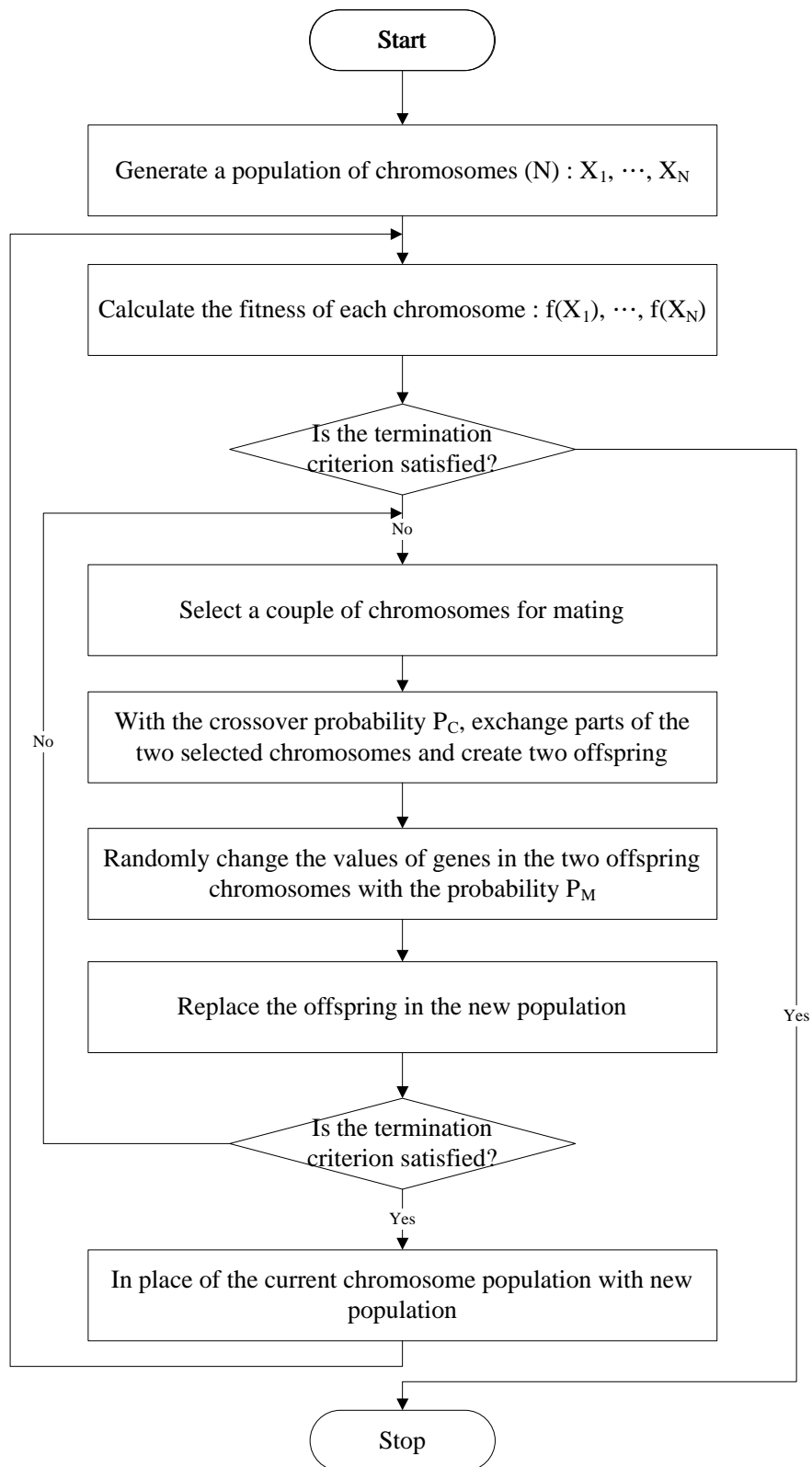
**Figure 1-1. A Chromosome of  $3 \times 3$  JSP problem**

## **Chapter 2 Literature review**

### **2.1 Genetic Algorithm for Job-shop Scheduling Problem**

Genetic algorithm (GA) is one of the stochastic search algorithms based on biological evolution. In order to solve a clearly defined problem and an offspring represented the candidate of solutions. GA is according to crossover and mutation operators with their probabilities to produce a set of offspring chromosomes. A basic GA flowchart can be showed in Figure 2-1 [19]. As we know, GA likes an over and over process, an iteration is called a generation. A run means the whole set of generations. We try to find one or more highly fit chromosomes.

Recently, there have more and more papers used hybrid GA to solve optimum problem. Because of GA provides quite simple structure, process and it has strong abilities of solving and searching. Furthermore, GA searches multiple points in search space of population by evolution of generations and characteristic of search randomly. The abilities can avoid GA dropping in the local optimum and toward the global optimum. Whitley [9] introduced designing GA has two important issues: selection pressure and population diversity. Selection pressure leads GA to exploit information from the fitter individuals and produces more superior offspring iteratively. The diversity in GA is concerned about the population, which contains a certain number of encoded individuals for exploration. Therefore, we must to find a good tradeoff between exploration and exploitation consideration of both convergence speed and optimized solution quality.



**Figure 2-1. Flowchart of simple genetic algorithm**

Masato etc. [18] proposed the modified GA with search area adaptation (mGSA) for solving JSP that does not need such crossover operator in GSA. Goncalves etc. [13] presented a hybrid genetic algorithm for the job-shop scheduling problem. It used the chromosome representation of the problem is based on random keys. The scheduled used a priority rule in which are defined by GA.

## 2.2 Tabu Genetic Algorithm (TGA)

Fred Glover (1989) [11] proposed TS that is a strategy for solving combinatorial optimization problems. In this section, we display TS in a simple form of its conceptions. TS constrains the search by classifying certain of its moves as forbidden and to free the search by a short-term memory function. And utilize aspiration criteria to override the tabu restriction that allow superior solution. In TS, the tabu restrictions and aspiration criteria played a dual role in constraining and guiding the search process. Besides, it uses memory function to record the moving trajectories. According to the used memory structure, we classify these approached into two major categories: computation-based and memory-based mating strategies [6].

Computation-based approach is the most common way to distinguish the degree of conformity among individuals. It is adapting hamming distance be the metric to calculate genotype or phenotype characteristics, and then mate the individuals by different strategies. It has been shown to maintain population diversity and improve the solution. However, memory-based schemes do not require extra computation to measure similarity. And it can

prevent premature convergence and reduce computational overhead in determining bloodline. They are reviewed in Table 2-1.

Ting, Li and Lee proposed tabu genetic algorithm [6], which by incorporates the feature of TS into GA's selection. TGA integrates the tabu list to prevent inbreeding so that population diversity can be maintained, and adapts the aspiration criterion to provide moderate selection pressure. Then it utilizes the self-adaptive mutation to overcome the hard of deciding mutation rate. It uses the classic traveling salesman problem (TSP) as a benchmark to verify the effectiveness of the proposed algorithm.

The main emphases are bellows:

- The memory structure of TS: In order to control the diversity of subpopulation and avoid the condition of inbreeding, it use tabu list to record the search trajectory.
- Under the restrictions of tabu list: TGA can apply the aspiration criterion to support the subpopulation intensively and diversely.
- Utilizing self-adaptive mutation: According to the situation of population change the mutation dynamically to prevent the performance made by fixed mutation rate.

The presented TGA is according to the structure of evolution of GA and the restrictive features of TS. At the first, selecting parents of TGA is the same with normal GA. At the same time the search strategy of TS to expand the obvious characteristics of both algorithms. The flowchart of TGA is showed in Figure 2-2 [6]. If the part of TS, highlighted in gray, is ignored, TGA becomes into a simple GA. It means that the reproductions of GA needed to be supervised by the elements of TS.

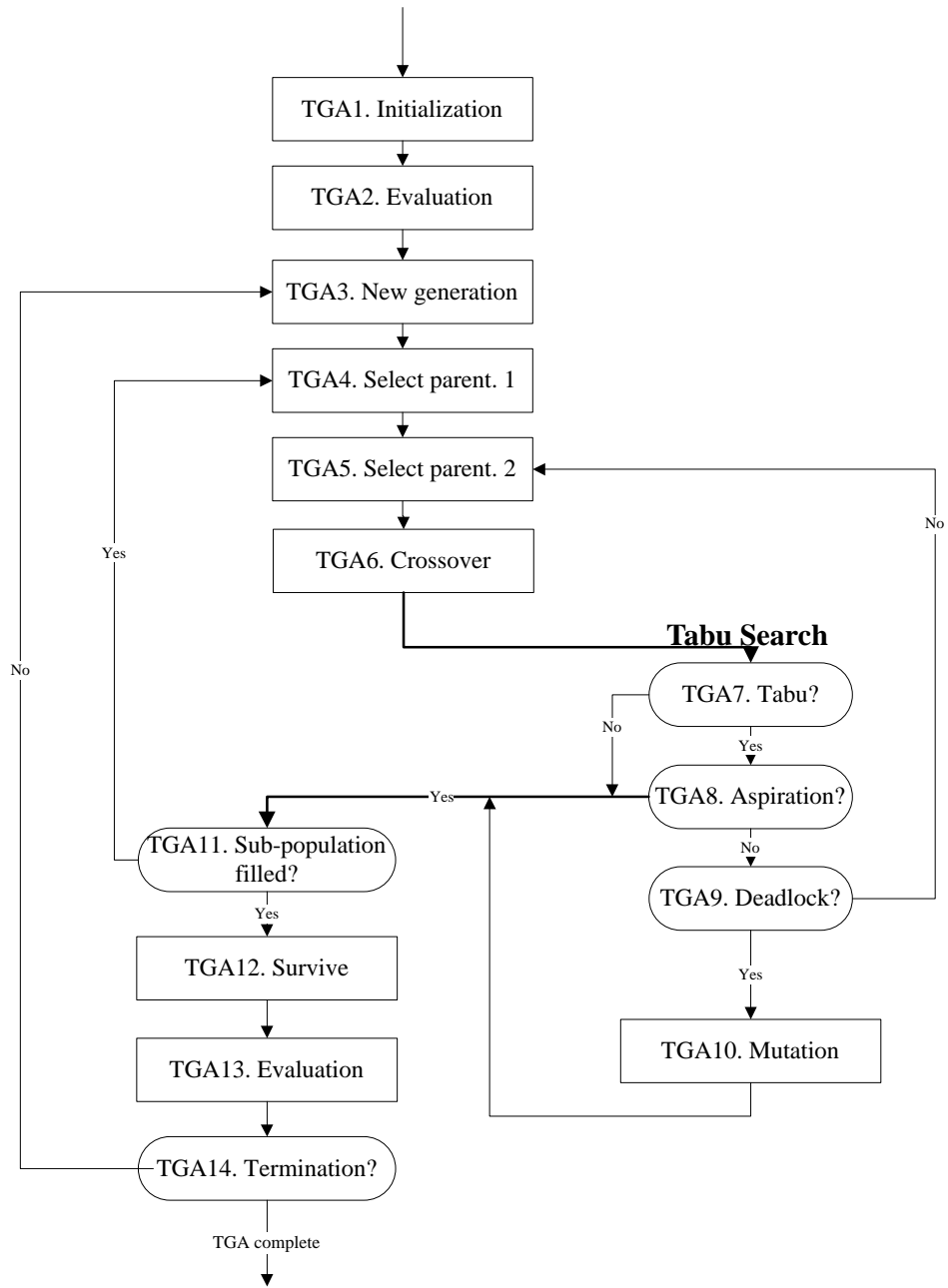
**Table 2-1. Computation-based and Memory-based mating**

	<b>Property</b>	<b>Method</b>
<b>Computation-based mating</b>	Distinguish individuals' similarities (Using hamming distance to measure similarity between to individuals.)	Suggest the clearing policy. [1]  Employ the concept of incest prevention. [14]  Negative assortative mating. [3]  Choosing different crossover method. [20]  The phenotypic assortative mating. [25]  Different assortative mating strategies. [5]  Allowance of mating by a normal function of the normalized fitness. [15]
<b>Memory-based mating</b>	Record the results of mating (Memory structure is appended to individuals to enable their ancestry to be recognized.)	The family tree to disallow incest by adhering to the ancestry-based incest law. [22]  Ancestry table upon GA with variable population size. [3, 4]  Devised three subpopulation schemes to control mating. [26]  The race genetic algorithm. [7]  The genetic algorithm with chromosome differentiation. [23, 24]  On the harmonious mating strategy through tabu search. [6]



After being produced by genetic operators, each pair of offspring went through these restricts of TS, tabu list and aspiration criterion, to confirm that their parents are allowed to mate. When the mating is allowed to mate or is good enough to meet the aspiration criterion, it is grouped into acceptable and the offspring are reserved for the subpopulation. Otherwise, the offspring are rejected and the process returns to select a mate. The process is repeated until the mating is acceptable. If the number of experiments exceeds a predefined threshold, this condition is thought as a Deadlock. Then the selected mate will be mutated. They are regarded as acceptable mate, and delivered to subpopulation.

All the researches have indicated that genetic algorithm and job-shop scheduling problem are the important issues to the research of optimal problems. Besides, tabu search is also applied to GA, called TGA for traveling salesman problem (TSP) that has better effectiveness than GA [6]. Thus, this is an interest and important research area for job-shop scheduling problem with TGA.



**Figure 2-2. Flowchart of TGA**

### 2.2.1 Tabu List

The contribution of tabu list in TGA is to prevent incest in a memory-based manner. Each successful mating in GA is regarded as a move in TS, thus it's the same with tabu list in TS to prevent some moves be trapped in local optimum. TGA employs it to track reproduction and forbid incest. The selection is monitored and restricted by tabu list, just like the move in TS. So the clans of mating parents will record in tabu list of the other side. The following function 1 can defined the two chromosomes  $C_i(G_i, \lambda_i, T_i)$  and  $C_j(G_j, \lambda_j, T_j)$  are tabu [6].

$$\text{Tabu} (C_i, C_j) \begin{cases} \text{true,} & \text{if } \lambda_i \in T_j \text{ or } \lambda_j \in T_i, \\ \text{false,} & \text{otherwise.} \end{cases} \quad (1)$$

In TGA, after a successful mating, the parents update their tabu list with the mate's clan, so that the mating history can be traceable. The offspring inherit the clan and updated tabu list entirely from one of the parents, rather than from a combination of both parents.

The size of tabu list is proportional to the size of the population, and can be defined as function 2 [6]:

$$N_T = \delta \times N \quad (2)$$

$N_T$  is the size of the tabu list;  $\delta$  is the parameter of proportionality ( $0 \leq \delta < 1$ ), and  $N$  is the size of the population. So a large population will accompanies a large tabu list. Thus, a bigger parameter of proportionality

tightens the restriction and the searches focus on exploration. On the contrary, a smaller one loosens the restrictions and supports the search exploitation. However, if the parameter closes to zero, TGA degrades into GA.

### 2.2.2 Aspiration criterion

The aspiration criterion is another way to measure the mating. It is designed to permit the better solution overthrow the tabu restrictions. If the fitness of the offspring ( $C_i'$ ) is  $F_i'$  and the best solution as so far is  $S$ ; then the aspiration criterion is defined as function 3 [6].

$$\text{Aspiration}(C_i') = \begin{cases} \text{true,} & \text{if } F_i' > S, \\ \text{false,} & \text{otherwise.} \end{cases} \quad (3)$$

If the mating parents are trapped in tabu, their fitness of offspring is better than the best solution as so far. The offspring able to regardless of tabu restrictions. Therefore, aspiration criterion supports intensification under the consideration of diversification. So that, it can harmonizes the selection pressure and population diversity.

### 2.2.3 Self-adaptive mutation

In natural environment, the probability of mutation is not fixed. In order to adapt to the changing environment, organisms mutate by themselves. In GA, there is usually given a mutation rate of fixed probability. This manner cannot reflect the truly of nature condition. So TGA use self-adaptive mutation to imitate organism mutation in natural environment.

In the TGA, the population of diversity is too low to achieve an acceptable parents, the selection will be repeated indefinitely, causing a dead lock. Technically, the number of clans or the diversity shrinks with evolution. In order to prevent this condition, we use threshold (TH) to restrict the number of repeated trial mating. Deadlock is defined as function 4 [6].

$$\text{Deadlock} = \begin{cases} \text{true,} & \text{if } r \geq \text{TH,} \\ \text{false,} & \text{otherwise.} \end{cases} \quad (4)$$

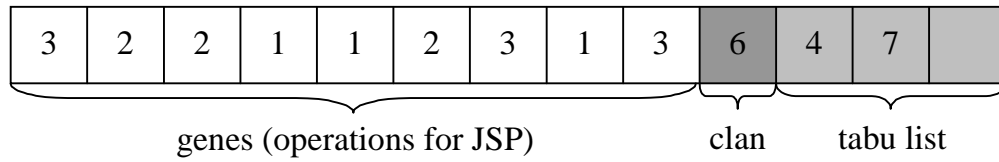
# Chapter 3 Modified TGA for Job-shop scheduling problem

## 3.1 Tabu genetic algorithm for JSP

### 3.1.1 Designing chromosomes

A chromosome has genes information for solving problem in GA. However, in TGA, a clan number represents identification, it appended to each chromosome with tabu list in the course of recording the clans of offspring after evolution. The process will complete the TS strategy. So, a chromosome in TGA is showed in three parts,  $(G, \lambda, T)$ , which  $G$  is a set of normal genes;  $\lambda$  is a clan number of chromosome, and  $T$  is a set of tabu list.

The genes information used the operation-based representation [21]. It encodes a schedule as a sequence of operations and each gene stands for one operation. Gene is given priority from left to right. Therefore, a left gene has higher priority than right one. At the initialization stage, a clan number is assigned uniquely. The offspring will inherit the clan number from one of their parents. If there are the same clans in the population, it means they are from the same ancestry. The tabu list records several numbers of clan to prevent inbreeding. It shows in Figure 3-1 that such a chromosome structure presents an example of  $3 \times 3$  job-shop schedule, an additional clan (6) and tabu list (4, 7).



**Figure 3-1. Example of representation for JSP**

### 3.1.2 Calculating fitness value

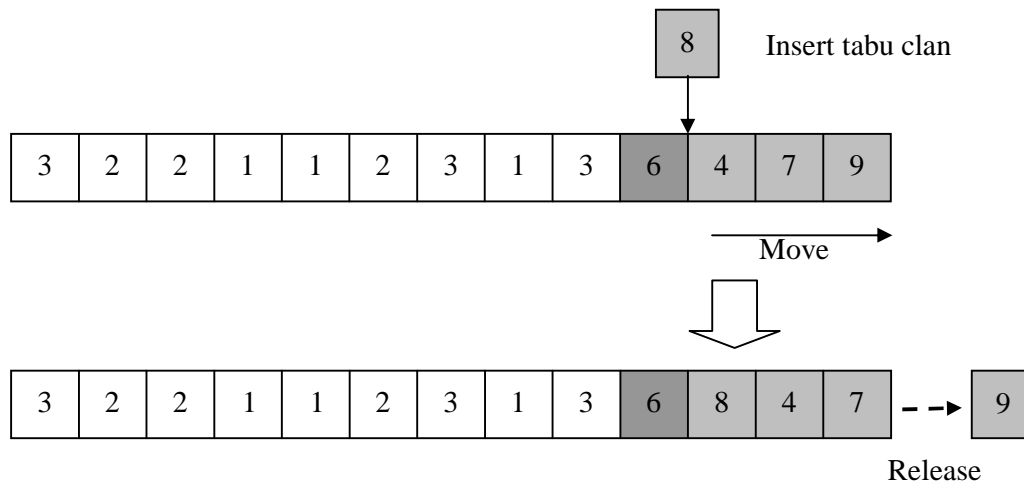
In normal, the chromosomes are selected when the fitness values are bigger. That means the fitness value is bigger, the probability of being chosen is higher. However, the JSP problem is better with smaller makespan. So the makespan is transferred into the fitness value as follow [8]:

$f(x)$  represents fitness value;  $C_{\max}$  represents the max makespan;  $g(x)$  represents makespan.

$$f(x) = \begin{cases} C_{\max} - g(x), & \text{when } g(x) < C_{\max} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

### 3.1.3 Tabu list

The operator of updating the tabu list works like a queue. A new clan forbidden to mate will insert into the tabu list. If the tabu list is full, all the clans in it will shift to the right position. The last clan will be released from the tabu list, and regains the qualification to be mated. Figure 3-2 illustrates the operation of updating the tabu list in TGA.



**Figure 3-2. Update the tabu list**

### 3.1.4 Crossover

There are a pair of parents be selected randomly from population. When they crossover, the process can be divided into two parts at the same time, the process is showed in Figure 3-3. The one part is genetic operators on the genes as in the normal GA, and the other one is update clan and tabu list by the TS restrictions.

First of all, introducing the genes part, a pair of parents think as  $P_A$  and  $P_B$  that two crossover points are selected randomly and look at the middle parts of genes present  $A_{P1}$  and  $B_{P1}$ . Find the same genes in different locations from  $A_{P2}$  and  $B_{P2}$  and exchange them and their locations. They are became  $A_{C1}$  and  $B_{C1}$ . After that, we assign the different genes from  $A_{P1}$  in  $A_{C1}$  respectively, so do the  $B_{P1}$  in  $B_{C1}$ . The middle parts of offspring showed in  $A_{C2}$  and  $B_{C2}$ . Then, the outside parts of two points are appended to  $A_{C2}$  and  $B_{C2}$  from  $P_A$  and  $P_B$ . The offspring are presented as  $C_A$  and  $C_B$ . This processing can make the offspring to keep the diversity but not go against the restrictions of JSP.



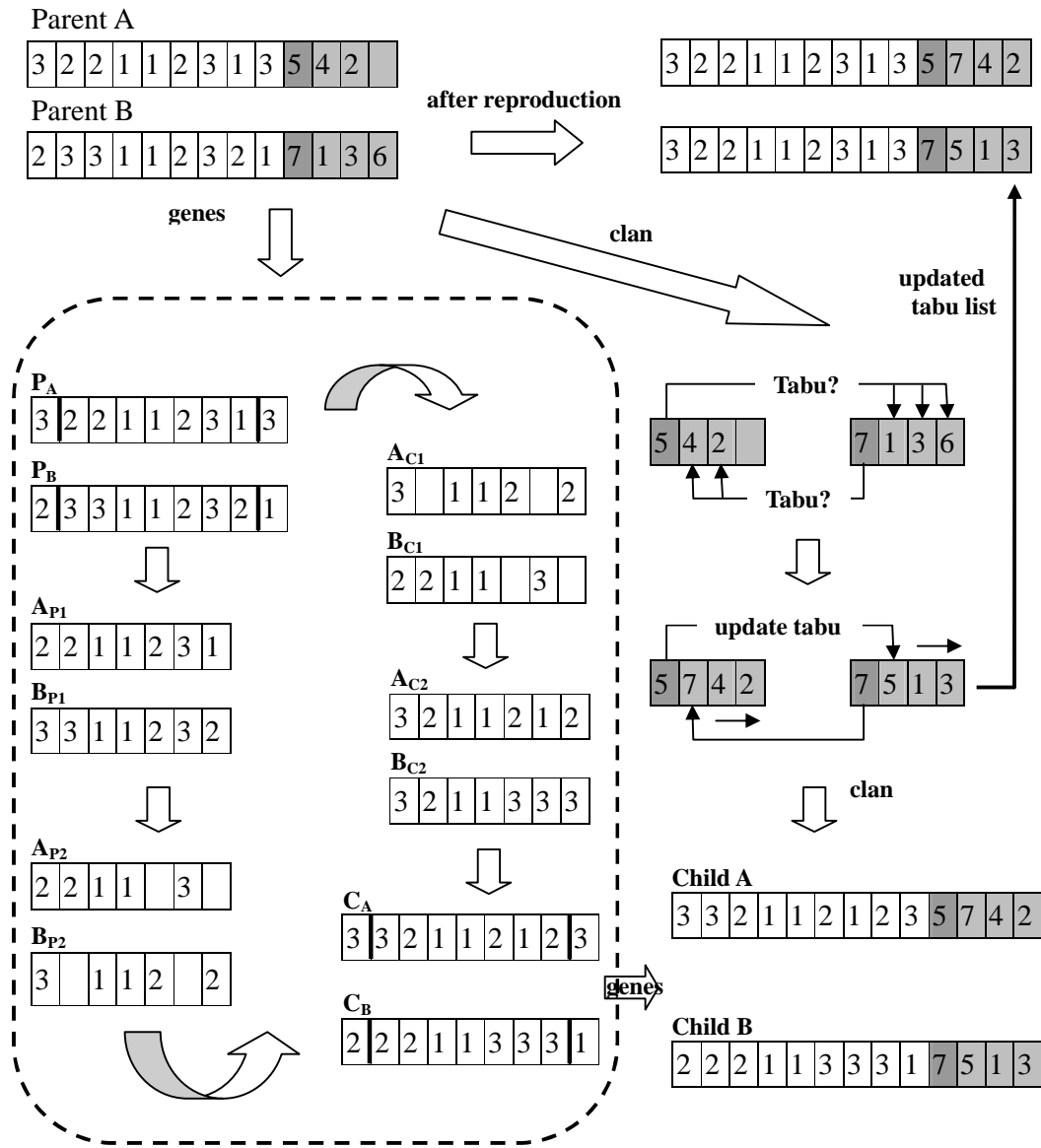
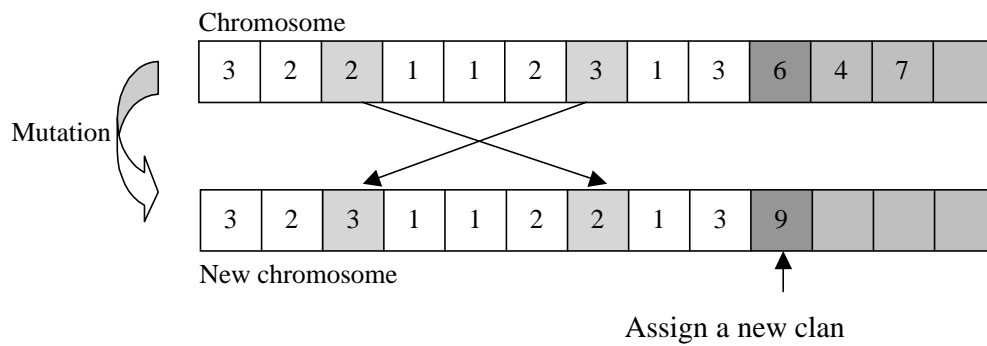


Figure 3-3. Example of crossover in TGA for JSP

### 3.1.5 Mutation

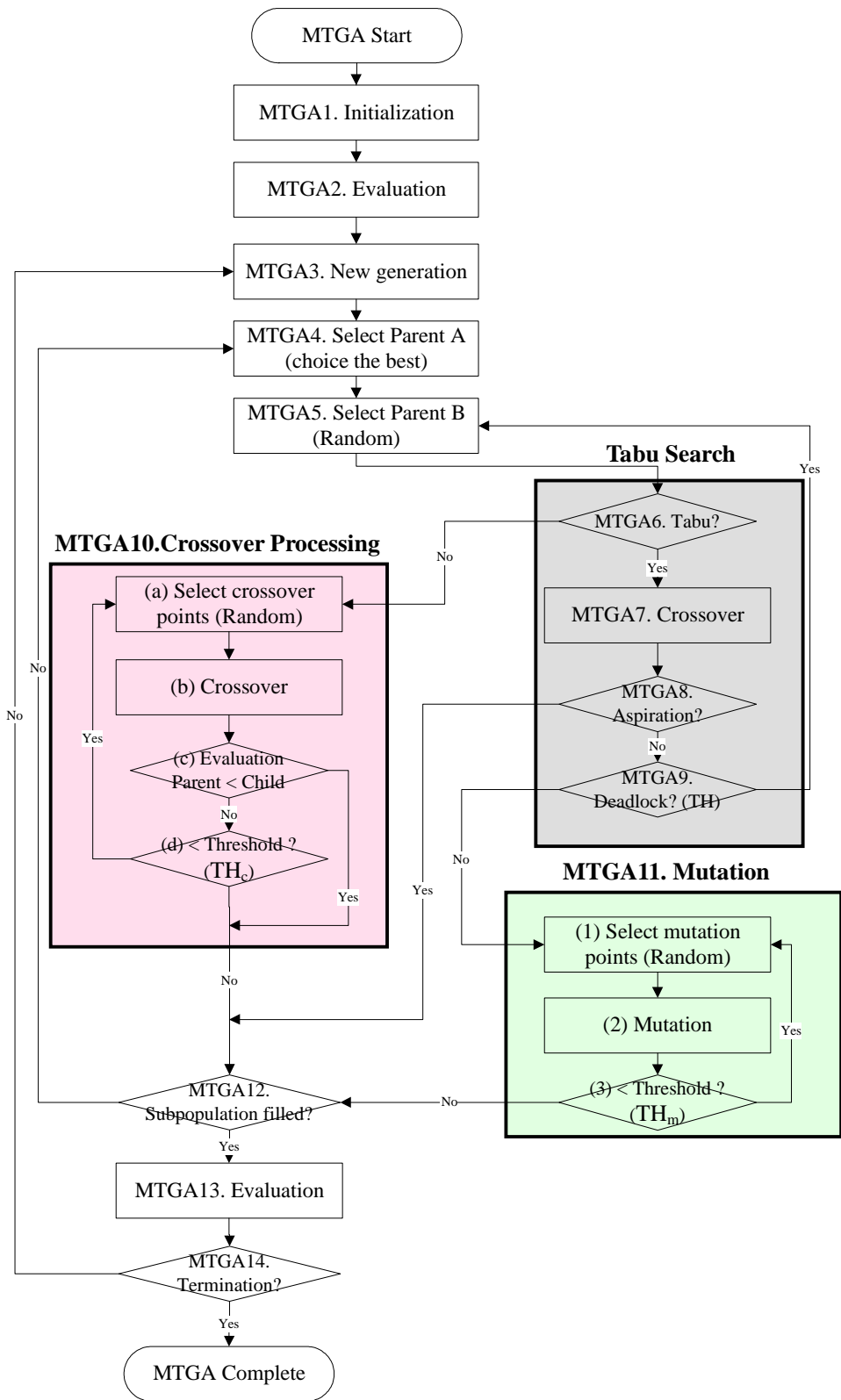
The mutation process is that selection two genes from a chromosome randomly, then exchange the two genes. After mutation, the chromosome is viewed as a newborn and is assigned a new clan number. The example of mutation process in TGA for JSP is presented in Figure 3-4. This mutation operator is given for the population to make up for the lack of diversity. Thus the characteristics of dynamic adaptation in response to population conditions are expected to enhance performance.



**Figure 3-4. Example of mutation in TGA for JSP**

### 3.2 Modified TGA

In TGA for JSP, the results and effectiveness of experiments is not as good as the algorithm for TSP. It inherits the property of diversity, but it can't take the balance between diversification and intensification. Thus, we modified crossover and mutation search phases of TGA, called modified TGA (MTGA) to improve ability of local search and extend search area. And try to convergent solutions under the diversity. The flowchart of MTGA is showed in Figure 3-5.



**Figure 3-5. Flowchart of MTGA**

Based on the above definitions and discussion, the modified algorithm MTGA is formulated as follows.

**Algorithm MTGA.** Assume that the population  $P$  consists of  $N$  chromosomes  $C_1, \dots, C_N$ , with fitness  $F_1, \dots, F_N$ , respectively. The best fitness  $S$  is recoded in each generation  $t$ . The genetic operators are performed to produce offspring  $C'_1, \dots, C'_N$ , and the deadlock criterion TH is defined to prevent infinite loop. The algorithm terminates at  $t_{max}$  generations, at which point the obtained best fitness  $S$  is the optimized result.

**MTGA1.** [Initialization.] Set  $t \mathbf{B} 0$ , and initialize population  $P_t$ .

**MTGA2.** [Evaluation.] Evaluate population  $P_t$  and result in  $F_1, \dots, F_N$ . Set  $S \mathbf{B} \max (F_1, \dots, F_N)$ .

**MTGA3.** [New generation.] Set  $n \mathbf{B} 0$  and  $r \mathbf{B} 0$ . (Where  $n$  is the number of produced offspring, and  $r$  is the repeated times of deadlock.)

**MTGA4.** [Select Parent A.] Set  $i \mathbf{B}$  the best  $(1, \dots, N)$ . (Select parents  $C_i$ .)

**MTGA5.** [Select Parent B.] Set  $j \mathbf{B}$  random  $(1, \dots, N)$ . (Select parents  $C_j$ .)

**MTGA6.** [Tabu?] If Tabu  $(C_i, C_j) = \text{false}$ , go to step MTGA10. (Check parents  $C_i$  and  $C_j$  to see if they are forbidden to mate.)

**MTGA7.** [Crossover.]  $(C'_i, C'_j) \mathbf{B}$  Crossover  $(C_i, C_j)$ .

**MTGA8.** [Aspiration?] If Aspiration  $(C'_i, C'_i) = \text{true}$ , go to step MTGA12.

**MTGA9.** [Deadlock?] Set  $r \mathbf{B} r + 1$ . If  $r < \text{TH}$ , go to step MTGA5, otherwise, go to step MTGA11.

**MTGA10.** [Crossover processing.] Set count\_1  $\mathbf{B} 0$ , and  $\text{TH}_c$  is the repeated times of crossover.

(a) Select crossover points randomly.

- (b)  $(C'_i, C'_j) \mathbf{B}$  Crossover  $(C_i, C_j)$ .
- (c) If  $(F'_i, \dots, F'_j) < (F_i, \dots, F_j)$ , go to step MTGA12.
- (d.) Set count\_1  $\mathbf{B}$  count\_1 + 1. If count\_1  $<$  TH<sub>c</sub>, go to step (a.).  
Otherwise, go to step MTGA12.

**MTGA11.** [Mutation Processing.] Set count\_2  $\mathbf{B}$  0, and TH<sub>m</sub>, is the repeated times of mutation.

- (1) Select mutation points randomly.
- (2)  $(C'_i, C'_j) \mathbf{B}$  Mutation  $(C_i, C_j)$ .
- (3) Set count\_2  $\mathbf{B}$  count\_2 + 1. If count\_2  $<$  TH<sub>m</sub>, go to step (1).  
Otherwise, go to step MTGA12.

**MYGA12.** [Subpopulation filled?] Insert the offspring  $C_i$  and  $C_j$  into  $P_{t+1}$ , and set  $n \mathbf{B} n + 2$ . If  $n < N$ , set  $r \mathbf{B} 0$  and return to step MTGA4.

**MTGA13.** [Evaluation.] Set  $P_{t+1} \mathbf{B}$  Survive  $(P_t, P_{t+1})$  and  $t \mathbf{B} t + 1$ . Evaluation population  $P_t$ , and set  $S \mathbf{B} \max (F_1, \dots, F_N)$ .

**MTGA14.** [Termination?] If  $t < t_{max}$ , return to step MTGA3. Otherwise, the algorithm MTGA is complete.

### 3.2.1 Modified crossover search phase

In TGA, the crossover phase has to obey the tabu restrictions, so it maintained the diversity of population. However, it will due to the insufficient convergence, for this reason, the modified crossover search phase uses a threshold (TH<sub>c</sub>) to control the times of crossover. If the children are better than the parents, they survived in subpopulation. Otherwise, the parents repeat to crossover until the times of crossover equal to the threshold, and keep the last offspring to the subpopulation. It's shown in Figure 3-1 of crossover

processing.

About diversity issue, TGA had good performance in it. But in TGA crossover phase, it's crossover one time, then delivery the offspring in subpopulation. It's a consequence on the lack of convergence. Therefore, we utilize the times of crossover to do some local search for improving the solutions qualities and converging faster.

### 3.2.2 Modified mutation search phase

Mutation operator is in order to adapt evolution environment and adjust it dynamically. But the chromosomes mutate by two selected points that are still too hard to outleap the local minimum. Hence, we added a mechanism in mutation processing. For example, assume a parameter set as 10, a pair from 1 to 10 be selected the mutation points randomly. This manner is in the course of making the global search wildly. It's also preventing to drop into local minimum more easily. After mutation, next step is another mechanism that controls the times of mutation by a threshold ( $TH_m$ ). Finally, the best chromosomes are sent to the subpopulation. It's shown in Figure 3-1 of mutation processing.

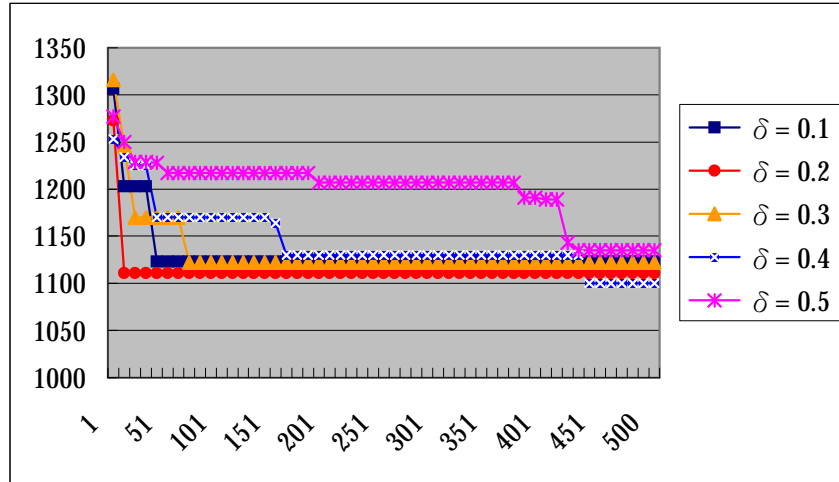
## Chapter 4 Performance evaluation

In our research, several comprehensive experiments are conducted to evaluate the performance of TGA and MTGA in JSP. We apply ft 10, 10×10, benchmark problem which has received the greatest analysis is the instance generated by Fisher and Thompson [10]. Lawler et al [17]. report that within 6000 s when applying a deterministic local search to ft 10 more than 9000 local optima have been generated with a best makespan value of 1006, furthermore emphasizing the difficulty and hardness of this problem. Besides, several researches proved that the optimal makespan is 930 of ft 10.

### 4.1 Tabu list

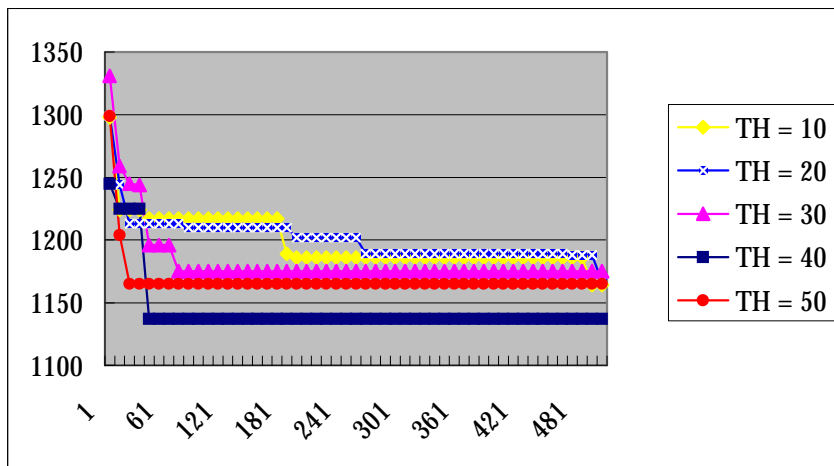
The impact of the size of tabu list on the performance of TGA is considered. The parameter of proportionality,  $\delta$ , determining the size of tabu list, is experimentally changed from 0.1 to 0.5. A population size ( $N = 50$ ), (generation= 500) and deadlock (TH= 10) are simulated. Each parameter runs 20 times and then takes the best solutions for experiment result. The convergence of each parameter can be showed in Figure 4-1.

By the number of parameter increased, the convergence is decreasing. When the parameters are 0.1 to 0.3, the convergences are still having the suspicion of premature convergence. But when the parameters are 0.4 and 0.5, the convergences are slower and slower. In these conditions can observe that the utility of tabu list can prevent premature convergence.



**Figure 4-1. Comparison of tabu list parameter in TGA (10×10)**

Figure 4-2 plots the function of deadlock with the tabu list parameter = 0.4. The mutation probability is controlled by deadlock. When the deadlock is lower, the mutation probability is higher. In Figure 4-2, the deadlock higher than 20, TGA is trapped into premature convergence. However, the deadlock is slower and then mutation rate become higher. So the speed of convergence will slow down.



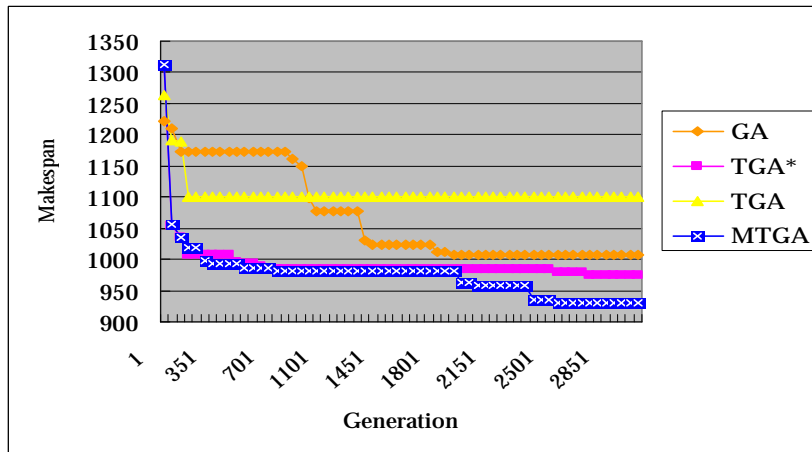
**Figure 4-2. Comparison of deadlock in TGA (10×10)**



## 4.2 Performance comparison

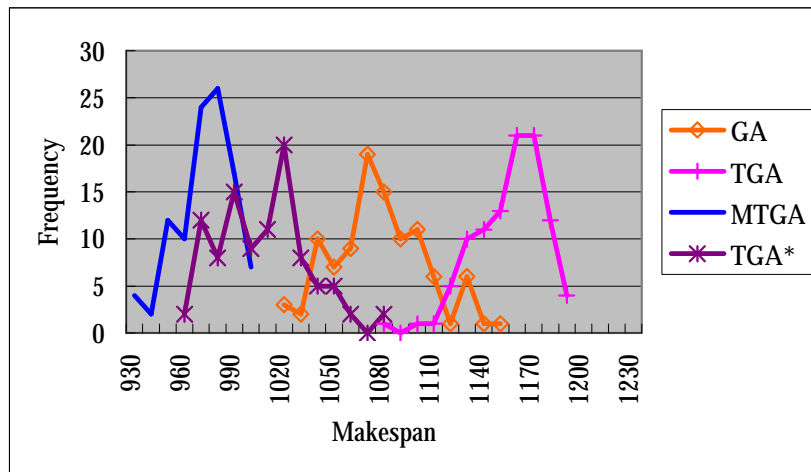
In this experiment, we apply GA, TGA, TGA\* (Selected the best chromosome as one of the parents, the other one selected randomly.) and MTGA to fit 10 JSP problem as a case. In TGA, TGA\* and MTGA, the population size ( $N$ ) is set to 100, the tabu list parameter is set to 0.4 and the deadlock is set to 20. In GA, the population size is set to 100, crossover rate = 0.5, and mutation rate = 0.15.

The performance experimental results are decomposed into two parts: MTGA is compared with GA, TGA and TGA\*. And compared the frequency distribution of GA, TGA, TGA\* and MTGA. In Figure 4-3, we can see that the performance of GA outperforms TGA. Although, TGA prevents premature convergence, but the qualities of solutions are not good enough. It's because that parents are selected by random. It leads the solution easier drop in local minimum. Thus, in TGA\*, one of the parents choose the best chromosome in the population, the other one is selected by random. The qualities of solutions can be proved a lot. However, the performance of MTGA outperforms TGA\*, the reason is that MTGA adapted location search in crossover and mutation search phases. It contains the good solutions in TGA\* and also do local search in the good area that made the performance better. So the efficiency is obvious to the modified operator of crossover and mutation of MTGA and also found the optimal solution 930.



**Figure 4-3. Comparison GA, TGA, TGA\*and MTGA (10×10)**

Figure 4-4 shows the frequency distribution of results, respectively. It is shown here that the distribution of MTGA has width narrower and the solutions are better than GA, TGA and TGA\*. Therefore, it is confirmed that MTGA shows better performance than GA and TGA.



**Figure 4-4. Frequency distribution of makespan (10×10)**

## Chapter 5 Conclusion

This research apply TGA to  $10 \times 10$  JSP problem, it maintains diversity through broad-sense incest prevention by tabu list. Therefore, the solutions can contain theirs diversity. It also can prevent premature convergence by deadlock. But in JSP problem, the crossover and mutation manners of TGA cannot produce the better solutions than GA. So we modified the crossover and mutation phases of TGA called modified TGA (MTGA). First, the modified crossover search phase uses a threshold ( $TH_c$ ) to control the times of crossover for improving the qualities and convergence of solutions. Second, the mutation search phase use two parameters to control the selected points and the times of mutation in order to make the global search wildly and prevent to drop into local minimum more easily. And the experiments results demonstrate the superiority of MTGA in job-shop scheduling problems. Not only balance intensification, but also diversification. Furthermore, it finds out the optimal makespan 930.

## References

- [1] A. Petrowski, A new selection operator dedicated to speciation, in: Proceedings of the Seventh International Conference on Genetic Algorithms, 1997, pp. 144–151.
- [2] A. S. Jain and S. Meeran. Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research* 113, 1999, 390-434.
- [3] C. Fernandes, A. Rosa, A study on non-random mating and varying population size in genetic algorithm using a royal road function, in: Proceedings of IEEE Congress on Evolutionary Computation, Seoul, South Korea, 2001.
- [4] C. Fernandes, R. Tavares, A. Rosa, niGAVaPS—Out breeding in genetic algorithms, in: Proceedings of the 2000 ACM Symposium on Applied Computing, 2000, pp. 477–482.
- [5] C. Fernandes, R. Tavares, C. Munteanu, A. Rosa, Assortative mating in genetic algorithms for vector quantization problems, *Proc. ACM SAC*, 2001, 361–365.
- [6] Chuan-Kang Ting, Sheng-Tun Li, Chungnan Lee. On the harmonious mating strategy through tabu search, *Information Sciences* 156, 2003, 189-214, .
- [7] C. Ryan, Racial harmony and function optimization in genetic algorithms—The races genetic algorithm, in: Proceedings of EP’95. The MIT Press.
- [8] D.E. Goldberg, Genetic algorithm: in search, Optimization and Machine Learning, Addison-Wesley Publishing Co, 1989.

- [9] D. Whitley, The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best, in: Proceedings of 3rd International Conference on Genetic Algorithms, San Mateo, CA, 1989, pp. 116–121.
- [10] Fisher, H., Thompson, G.L., Probabilistic learning combinations of local job-shop scheduling rules. In: Muth, J.F., Thompson, G.L. (Eds.), Industrial Scheduling. Prentice-Hall, Englewood Cliffs, NJ, 1963, pp. 225-251.
- [11] Fred Glover. Tabu Search — Part I. Operation Research Society of America, 1989.
- [12] Gen, M., & Cheng, R.. Genetic algorithms and engineering design. New York: Wiley, 1997.
- [13] Goncalves, Mendes and Resende. A hybrid genetic algorithm for the job shop scheduling problem. European Journal of Operational Research 167, 2005, 77-95.
- [14] H. Shimodaira, DCGA: a diversity control oriented genetic algorithm, IEEE International Conference on Tools with Artificial Intelligence, 1997, pp. 367–374.
- [15] I. Chakraborty, B. Chakraborty, Ideal marriage for .ne tuning in GA, in: Proceedings of IEEE International Conference Systems, Man, and Cybernetics, vol. 1, 1999, pp. 631–636.
- [16] L. Davis. Job shop scheduling with genetic algorithm. In Proc. of the First Int. Conf. on Genetic Algorithms (Edited by J. Grefenstette), pp. 136-140. Lawrence Erlbaum Associates, Hillsdale, NJ, 1985.

- [17] Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., Shmoys, D.B., Sequencing and scheduling: Algorithms and complexity. In: Handbook in operations Research and Management Science 4: Logistics of Production and Inventory, 1993.
- [18] M. Watanabe, K. Ida and M. Gen. A genetic algorithm with modified crossover operator search area adaptation for the job-shop scheduling problem. *Computers & Industrial Engineering* 48, 2005, 743-752.
- [19] Michael Negnevitsky, *Artificial Intelligence A Guide to Intelligent Systems (Second Edition)*, 2005, 219-228.
- [20] R. Bian, Z. Chen, Z. Yuan, Improved crossover strategy of genetic algorithms and analysis of its performance, in: *Proceedings of the Third World Congress on Intelligent Control and Automation*, 2000, pp. 516–520.
- [21] Runwei Cheng, Mitsuo Gen and Yasuhiro Tsujimura. A tutorial survey of job-shop scheduling problems using genetic algorithms — I. Representation. *Computers ind. Engng* Vol. 30, No. 4, pp. 983-997, 1996.
- [22] R. Craighurst, W. Martin, Enhancing GA performance through crossover prohibitions based on ancestry, in: *Proceedings of the Sixth International Conference on Genetic Algorithms*, 1996, pp. 130–135.
- [23] S. Bandyopadhyay, S.K. Pal, Incorporating chromosome differentiation in genetic algorithms, *Informat. Sci.* 104, 1998, 293–319.
- [24] S. Bandyopadhyay, S.K. Pal, Pattern classification with genetic algorithms: Incorporation of chromosome differentiation, *Pattern Recogn. Lett.* 18, 1997, 119–131.

- [25]S. De, S.K. Pal, A. Ghosh, Genotypic and phenotypic assortative mating in genetic algorithm, Inform. Sci. 105, 1998, 209–226.
- [26]W.M. Spears, Simple subpopulation schemes, in: Evolutionary Programming Conference, 1994, pp. 296–307.