International Journal of Industrial Engineering, 17(2), 128-141, 2010.

# A GENETIC ALGORITHM APPROACH FOR OPTIMIZING CHEMICAL TOWERS CONSTRUCTION PROJECT SCHEDULING WITH DYNAMIC RESOURCES CONSTRAINTS

**Chun-Wei R.Lin[1] and Hsian-Jong Hsiau[*2]**

Department of Industrial Management
National Yunlin University of Science and Technology
123 University Road, Section 3
Touliu, Yunlin, Taiwan, 640, ROC
[1]lincwr@yuntech.edu.tw
[*2]Corresponding Author's e-mail: g9421805@yuntech.edu.tw

In this paper we formulate the real-life chemical towers construction project as a dynamic resource constrained project scheduling problem (DRCPSP) with minimum makespan. Unlike the traditional resource constrained project scheduling problem, the DRCPSP model is able to consider both independent and dynamic resources which depend on the welding jobs processing sequence (JPS) of the major cylindrical columns components. A modified genetic algorithm with auto-shift mechanism (GAASM) is proposed to search for the optimal solution. A real-life example is presented to demonstrate the applicability of GAASM as well. Simulation experiments of eight illustrative problems with 30 runs show that GAASM outperforms the conventional GA based method (GABM). Furthermore, from eight groups with a total of 240 problems comparing two common rules, namely Top-down and Bottom-up rules, adopted in the current company plans (CP), GAASM demonstrates 5.09% ~ 14.60% average reduction in makespan successfully.

**Significance:** In the DRCPSP model, the resources characteristics are not only independent but also dynamic, which depend on the processing sequence of jobs. The proposed GAASM solution method of DRCPSP can be applied in practice for optimizing chemical towers construction project scheduling.

**Keywords:** Chemical Tower, Project Scheduling, Resource-Constrained, Genetic Algorithm

*(Received 1 April 2009; Accepted in revised form 17 May 2010)*

## 1. INTRODUCTION

Chemical towers are usually used in petroleum refineries, petrochemical and chemical plants and natural gas processing plants. The chemical towers can be classified by function as distillation column, reaction column, extraction column, absorption column, and washing column, etc. In practice the procedure of a chemical towers construction project includes data sheets provided by the process, detail design, material purchasing, and the tower to be fabricated in shop, to be transported and installed in the site. A typical industrial distillation tower operating system and a flow diagram of a chemical towers project are illustrated as Figure 1. In this paper we focus on the scheduling of tower fabrication and its assembly in the shop.

Normally a tower is composed of two heads (the top head and bottom head), shell and skirt. Industrial chemical towers are typically constructed in large, vertical cylindrical columns with diameters ranging from about 1meter to 5 meters and heights ranging from about 10 meters to 50 meters or more. Therefore the main structure of the shell is assembled by welding with several cylinder layers made by steel plate. The internal of section usually has a tray support ring, tray, distributor or other accessories in each layer. The specification for each layer of the shell is different. Each part of the layer before being assembled has to be fabricated individually. The main jobs of producing the tower are to fabricate and assemble all of the parts.

In a plant construction project the fabricated makespan of chemical towers is very important, as it will impact the total duration and cost. The makespan of scheduling and cost are the key factors of project performance. But the related work has received little attention in the literature. Companies fabricate the towers always following their experiences or company rules. However these experiences or company rules may not be the optimal solution. How to derive an optimal solution of scheduling for minimum makespan is an important goal for a manufacturing company.

The optimal scheduling problem for chemical towers construction project is similar to the resource-constrained project scheduling problem (RCPSP) as found in the literature, but the material resources of fabricating require the parts which are independent resources, the material resources of assembling need the parts for assembling which are dynamic resources based on the assemble sequence. The characteristic of resources is not only independent but also dynamic, and it is unlikely to be only independent in the RCPSP.

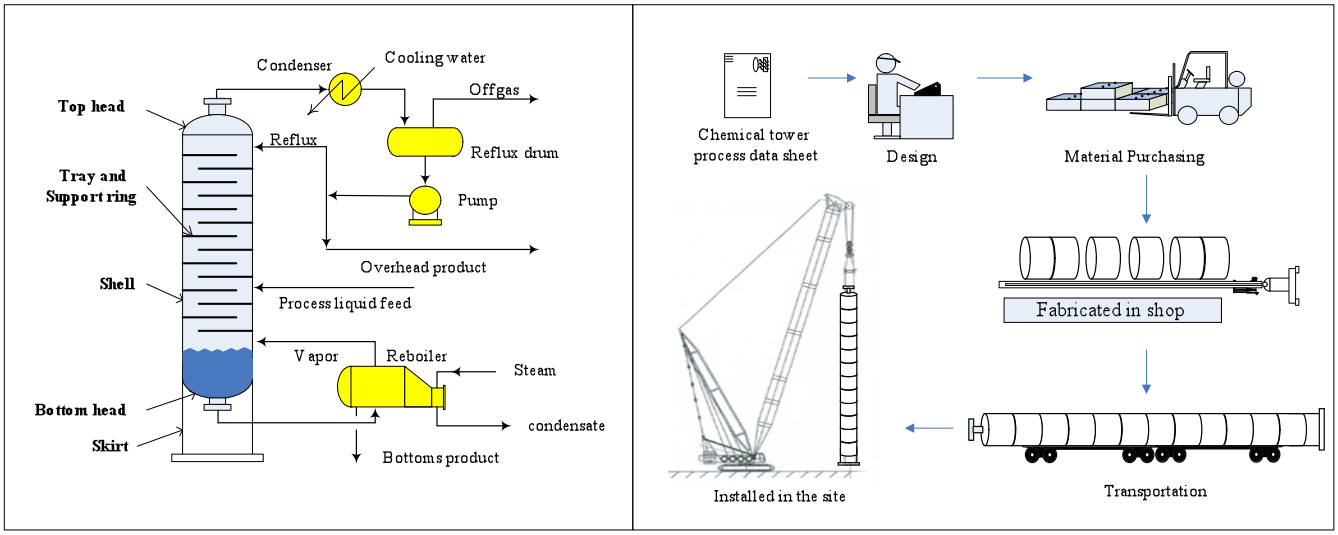


**Figure 1. A typical industrial tower operating system and project flow diagram**

The RCPSP is a classical problem. The related research has expanded greatly over the past few decades, e.g. (Sawski et al., 1997; Nudtasomboon et al., 1997; Brucker et al., 1998), and excellent reviews can be founded in Herroelen et al. (1998) and Brucker et al. (1999). In recent years, several effective algorithms for solving the RCPSP have been investigated continuously. Hartmann and Kolish (2000) evaluated the performance of several state-of-art heuristics for RCPSP from the literature, and analyzed the behavior of the heuristics with respect to their components. Abeyasingehe et al. (2001) presented an efficient method for construction projects scheduling. Zamani MR (2001) described a high-performance exact algorithm based branch and bound method. Heilmann (2003) developed an exact procedure for a general RCPS where multiple modes are available for the performance of the individual activities and minimum as well as maximum time lags between the different activities may be given. Brucker and Knust (2003) presented a destructive lower bound for the multi-mode RCPSP with minimal and maximal time-lags. Bouleimen and Lecocq (2003) used a new simulated annealing for RCPS and its multiple mode versions. Carlier et al. (2003) proposed efficient methods for RCPSP based on makespan lower bounds, which linearly depend on the processing times of the activities. Valls et al. (2003) presented a new meta-heuristic algorithm. Kim et al. (2003) used a hybrid genetic algorithm with fuzzy control for RCPSP. Fleszar et al. (2004) implemented a heuristic based on a variable neighborhood search. Kim et al. (2005) proposed a hybrid genetic algorithm with fuzzy logic controller. AL-Fawzan et al. (2005) introduced the concept of schedule robustness, a bi-objective resource-constrained project scheduling model and developed a tabu search algorithm for this problem. Debels et al. (2006) proposed a new meta-heuristic with hybrid scatter search/electromagnetism. Tseng and Chen (2006) proposed a hybrid metaheuristic method. Valls et al. (2008) proposed a hybrid genetic algorithm.

The RCPSP can be classified by resource categories, types, and values. The resource category is described in greater detail see Heilmann (2003). However these kinds of resources requirements for the above related works are independent with jobs scheduling. In this paper on chemical towers construction project scheduling with dynamic resource constraints problem, it is the RCPSP with resource constraints that depends on the jobs processing sequence (JPS). We extend this problem as a dynamic resource-constrained project scheduling problem (DRCPSP) which has not have much research thus far. In DRCPSP, the JPS of project scheduling will affect the resources required. For example, three parts, say parts $p_a$, $p_b$, $p_c$ are assembled to become one product in the construction project. Two assembling jobs need to be done. Job $J_1$ is to assemble parts $p_a$ and $p_b$, $J_2$ is to assemble parts $p_b$ and $p_c$. Suppose the JPS is ($J_1$, $J_2$), then the required resources by $P(J_1)=\{p_a, p_b\}$. After $J_1$ is finished, part $P(J_1)$ is a semi-finished product, and $P(J_2)=\{p_a, p_b, p_c\}$. If the JPS is changed to ($J_2$, $J_1$), then the resource required by $P(J_2)=\{p_b, p_c\}$, and $P(J_1)=\{p_b, p_c, p_a\}$. Obviously, different JPS will have different resource requirements for the same activity. Required resources depend on project scheduling.

The DRCPSP for a chemical towers is stated and formulated in the next section. Section 3 proposes a mathematical model. The modified genetic algorithm with the auto-shift mechanism for solving DRCPSP is introduced. In Section 4, a real-life example demonstrates that the GAASM algorithm is feasible. In Section 5, computational experiments present the results of comparing the GABM and company plans. The conclusion follows in Section 6.

## 2. PROBLEM STATEMENTS AND FORMULATION

### 2.1 Notations

| | |
|---|---|
| $n$ | The number of chemical tower parts |
| $N$ | The set of jobs $N=\{J_0,J_1,J_2,...,J_j,..,J_{2n-1},J_{2n}\}$ $J_0$ and $J_{2n}$ are dummy |
| $j$ | The index of jobs number, $j\in\{1,...,2n-1\}$ |
| $q_j$ | Processing sequence of job $J_j$, (Decision variable) |
| $V(q_j)$ | Processing sequence $q_j$ whose job is $J_{V(q_j)}$,define $V(q_j)=j$ |
| $V$ | An array of jobs, which is ranked by JPS $V=[V(1),V(2),...,V(2n-1)]$. |
| $P(J_j)$ | The set of chemical tower parts required by activity $J_j$ |
| $P_j$ | The set of precedence jobs for job $j$ |
| $R_P$ | The set of parts $R_P=\{p_1,p_2,...,p_l,...,p_n\}$ |
| $m$ | The used number of construction teams |
| $m_a$ | The maximum available number of construction teams |
| $R_T$ | The set of construction teams, $R_T=\{T_1,T_2,...,T_k,...,T_m\}$ |
| $J_{M(T_k)}$ | Processing job $J_j$ done by team $T_k$ |
| $P_c(J_j)$ | The set of conflict parts for job $J_j$ |
| $U_j$ | The construction team $T_k$ assigned to processing job $J_j$ , define $U_j=T_k$, $M(U_j)=j$ |
| $s_j$ | Start time of job $J_j$ |
| $d_j$ | Duration of job $J_j$ |
| $f_j$ | Finish time of job $J_j$ |
| $S$ | The set of jobs schedule $S=\{s_0,s_1,s_2,...,s_{2n}\}$ |
| $N(S,t)$ | The set of jobs work in process at moment $t$, $N(S,t)=\{J_j \mid s_j\le t\le s_j+d_j, J_j\in N\}$ |

### 2.2 Problem Statements

A chemical tower consists of $n$ parts. There are $2n-1$ jobs that need to be done. The jobs can be divided into two portions. The first portion of the job is to fabricate the $n$ parts of the chemical tower. The number of first portion jobs is $n$. The second portion of the job is to assemble the $n$ parts. The number of second portion jobs is $n-1$. The jobs of $J_j$ and $J_{j+1}$ are the precedence of job $J_{n+j}$, therefore $J_j$ and $J_{j+1}$ jobs have to be finished before $J_{n+j}$ job starts to be worked, where $j\in\{1,...,n-1\}$.

To process the project requires renewable resources of construction teams consisting machines and manpower. The maximum available resources of the construction teams are $m_a$ teams. All jobs can be assigned to any construction team $T_k, T_k\in R_T$. Besides construction team resources, the resources of parts are required. All tower parts $p_j, p_j\in R_p$ cannot to be exchanged with each others during their assembly, since the thickness of the steel plate and accessories for each layer part is not the same. The first portion for fabricating jobs needs the resource of material part $p_j$ for the $J_j$ activity. The second portion for assembling jobs basically needs the resource of parts $p_j$ and $p_{j+1}$ for the $J_{n+j}$ activity. It is at this point that the dynamic resources required problem occurs. Suppose the start time of the $J_{n+j}$ activity is earlier than the $J_{n+j+1}$ activity, where $j \le$ n - 2 . While the $J_{n+j}$ activity has been finished by assembling parts $p_j$ and $p_{j+1}$. The $J_{n+j+1}$ activity needs parts $p_{j+1}$ and $p_{j+2}$ But part $p_{j+1}$ has been combined with part $p_j$. To perform the $J_{n+j+1}$ activity, the resources required will be forced to become $p_j$, $p_{j+1}$ and $p_{j+2}$. In other words, the dynamic resources required by the second portion jobs depend on the JPS. The dynamic resources required, base on activity sequence, is the main significance compared to RCPSP**.** The diagram of the precedence and dynamic resource relations for the DRCPSP is illustrated as Figure 2.

The duration of activity $J_j\in N$ is denoted by $d_j$. The start time and finish time of activity $J_j$ is $s_j$ and $f_j$ respectively. Define $s_0=0$ and $f_{2n}=f_{V(2n-1)}$ stands for the start time and finish time of the project. The duration of activities for fabricating and assembling parts are given. The objective is to find a feasible schedule, and the makespan is the minimum.

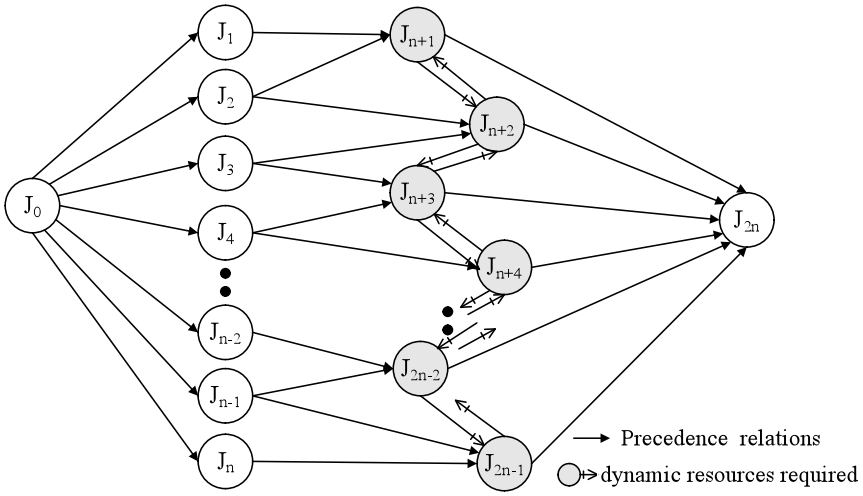


**Figure 2. The precedence and dynamic resource relations for DRCPSP**

### 2.3 Mathematical Model

The model of DRCPSP for chemical towers construction project scheduling is formulated as below.

$$Min.\ f_{2n} = f_{V(2n-1)} \quad \cdots \quad (1)$$

*s.t.*

$$s_j + d_j \le s_{n+j},\ j = 1, 2, ...,\ n \quad \cdots \quad (2)$$

$$s_{j+1} + d_{j+1} \le s_{n+j},\ j = 1, 2, ...,\ n \quad \cdots \quad (3)$$

$$P(J_j) = \{p_j\},\ j = 1, 2, ...,\ n \quad \cdots \quad (4)$$

$$P(J_j) \ne P(J_j),\ \text{where}\ \ i \ne j,\ \ i, j = 1, 2, ...,\ n \quad \cdots \quad (5)$$

$$P(J_j) = \{p_j, p_{j+1}\} \cup \{p_x \mid x \in P_j\} \cup \{p_y \mid y \in P_{j+1}\},\ j = n+1,\ n+2,\ ...,\ 2n-1 \quad \cdots \quad (6)$$

$$P_c(J_j) = P(J_{M(T_1)}) \cap P(J_{M(T_2)}) \cap ... \cap P(J_{M(T_m)}) = \phi, \quad \cdots \quad (7)$$

where $J_{M(T_k)} \in N(S,t),\ k = 1, 2, ...,\ m\ ,\ t = 0, 1, 2, ..., t_f - 1$

$$\{U_{M(T_1)}\} \cap \{U_{M(T_2)}\} \cap ... \cap \{U_{M(T_m)}\} = \phi, \quad \cdots \quad (8)$$

where $J_{M(T_k)} \in N(S,t),\ k = 1, 2, ...,\ m\ ,\ t = 0, 1, 2, ..., t_f - 1$

Objective function (1) is to minimize the maximum of the finish times $f_{2n} = f_{V(2n-1)}$. Inequality (2) (3) are the precedence constraints of the start time for $J_{n+j},\ j = 1, 2, ..., n\text{-}1$. Equation (4) is the parts of independent resource required constraints. Inequality (5) is the parts constraints, none of the parts can be exchanged with each other. Equation (6) is the parts of dynamic resource required constraints. In equation (7), the parts resource required cannot be in conflict during the same period of time. Equation (8) is the resources constraints of the construction team, and it shows one construction team cannot be assigned to do more than one job simultaneously.

### 2.4 Problem analysis

The RCPSP is a NP-hard problem (see Blazewicz et al., 1983). As a RCPSP, the DRCPSP is NP-hard too. The optimal solution cannot be derived directly. In the DRCPSP model, the parameters $n, m, m_a, R_T$ and $d_j$ are given for

the construction project. To search for the optimal solution of minimum makespan for this problem, five steps are considered in the ***Main Algorithm***.

***Main Algorithm***

Step A. To obtain an array $V$ of feasible job sequence $q_j$, $\forall J_j \in N$. The array $V$ of the feasible job sequence has to satisfy constraints (2) (3).

Step B. To compute the dynamic resources of parts $P(J_j)$ required by $J_j$, $\forall J_j \in N$, based on $V$. The results have to satisfy constraints (4) (5) (6).

Step C. Assign each job to construction team based on constraints (7) (8) and compute the makespan $f_{V(2n-1)}$.

Step D. Objective function fitness test. If fitness is satisfied, then stop, else go to Step E.

Step E. To obtain a new array $V$ of feasible job sequence, go to step B.

In the ***Main Algorithm***, suppose $V$ can be obtained from the Step A, then the steps from Step B to Step E can be solved.

**2.5 Company Plan Rules**

In Step A of the ***Main Algorithm***, there are two types of assigning job rules mostly used by the company plan (CP) to derive an array of feasible jobs sequence $V$. The two types of company plan rules are represented by CP1 and CP2.

- *CP1: Top-down.* The JPS is allocated by descending the index of job number $j$. In the first portion with a single part, assigning the jobs from $J_1$ to $J_n$ by descending the index of job number $j$. In the second portion with assembled parts, assigning the jobs for assembling from $J_{n+1}$ to $J_{2n-1}$ by descending order. The computing procedure is shown as ***Algorithm CP1***.

***Algorithm CP1*** : Assign jobs sequence by Top First

Input: $n$. Output: Feasible $V$ of CP1

Step 1. Set $q = j, V(q) = j,$ for $j = 1, 2, ..., n$

Step 2. Set $k = 1, j = n+1$

Step 3. If $q \le 2n-1$ then go to step 4, else output $V$ and STOP.

Step 4. If $j \le 2n-1$ then set $q \leftarrow q+1, V(q) = j, j \leftarrow j + 2^k$, else go to step 5.

Step 5. If $j > 2n-2^{k-1}$ & $q < 2n-1$ then set $q_e = 2n-2^{k-1}$ and go to step 6, else go to step 7.

Step 6. If $q_e > V(q)$ then $q \leftarrow q+1, j = q_e, V(q) = j$, else go to step 5.

Step 7. Set $k \leftarrow k+1, j = n+2^{k-1}$, go to step 3.

- *CP2: Bottom-up.* To allocate the JPS is similar to CP1, but the job sequence is assigned by ascending the index of jobs number $j$. The computing procedure is shown as ***Algorithm CP2***.

***Algorithm CP2*** : Assign jobs sequence by Bottom First

Input: $n$. Output: Feasible $V$ of CP2

Step 1. Set $q = j, V(q) = n-j+1,$ for $j = 1, 2, ..., n$

Step 2. Set $k = 1, j = 2n-1$

Step 3. If $q \le 2n-1$ then go to step 4, else output $V$ and STOP.

Step 4. If $j \ge n+1$ then $q \leftarrow q+1, V(q) = j, j \leftarrow j - 2^k$, else go to Step 5.

Step 5. If $j < n+2^{k-1}$ & $q < 2n-1$, then $q_e = n+2^{k-1}$ and go to step 6, else go to step 7.

Step 6. If $q_e < V(q)$ then $q \leftarrow q+1, j = q_e, V(q) = j$, else go to step 5.

Step 7. $k \leftarrow k+1, j = 2n-2^{k-1}$, go to step 3.

The results of the JPS for makespan of scheduling planned by company rules are feasible, but they may not be optimal. In this paper we propose a modified genetic algorithm with the auto-shift mechanism (GAASM) to solve this problem.

## 3. MODIFIED GENETIC ALGORITHM

### 3.1 Algorithm Overview

The genetic algorithm (GA) is a search technique which is used in computing to find exact or approximate solutions for optimization and search problems. The general procedure for GA was developed by Goldberg which has been widely applied to flow shop scheduling (e.g. Ibrahim et al. 2008) and project scheduling in the literature. But they have to face the problems of infeasible schedules that might be produced by genetic operations. The chromosome may need modification to ensure that project scheduling is feasible. If GA were applied to $V$ jobs in DRCPSP, the same difficulty will occur too. In general genetic operations, chromosomes are randomly selected from the population to perform either crossover or mutation. The offspring produced by a genetic operation may represent an infeasible schedule due to precedence relations. In this problem, the possible solutions space of random chromosomes will have $(2n-1)!$ types. The feasible solutions of chromosomes will have $2^{(n-1)}((n-1)!)^2$ types. For example, for $n=10$ parts, the possible chromosomes will have 19! = $1.21\times10^{17}$ types, including feasible and infeasible types. The infeasible scheduling problem exists, too. The genes of infeasible chromosomes need to be reordered to satisfy the precedence constraints.

In Step A of the ***Main Algorithm***, a feasible array $V$ ranked by JPS should be satisfied with constraints (2) (3). For example, if $n=5, V=$[2 3 5 7 1 6 4 8 9] is feasible, $V=$[7 2 3 5 1 6 4 8 9] is infeasible because $J_2$ and $J_3$ are the precedence jobs of $J_7$. In this paper the proposed GAASM method can derive the array of feasible job sequences $V$ for finding the optimal solution, and the structure of algorithm is illustrated in Figure 3.

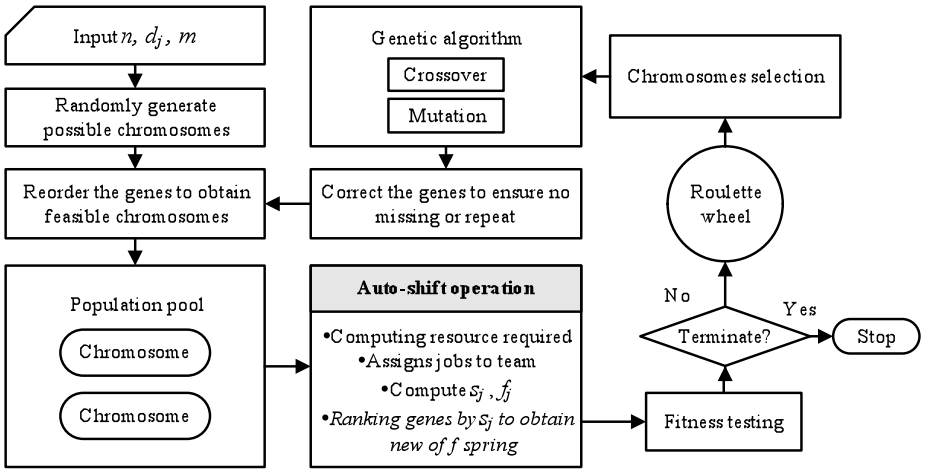


**Figure 3. The structure of GAASM Algorithm**

### 3.2 Solution Encoding and Population Initialization

The most frequently used encoding for the RCPSP is a permutation of the activities. The relative order of the activities in the permutation represents the JPS of the activities by the machines or construction teams. A chromosome represents a solution to the problem and is composed of $2n-1$ genes, where $n$ is the number of activities without dummy jobs.

$Chromosome=[V(1),\ V(2),...,\ V(q_j),...,V(2n-1)]$, where $V(q_j)\in\{1,2,...,2n-1\}$.

The genes of chromosome are used as construction priorities of activities, i.e., the priority of activity $J_j$ is $q_j$, and $V(q_j)=j$. The possible scheduling solution space of chromosome, including feasible and infeasible, will have $(2n-1)!$ combination types of genes.

Traditionally, in a GA, the initial population is generated randomly. In this problem, the initial populations are generated by randomly allocate the integer genes from 1 to $2n-1$. But the generated chromosomes randomly are not guaranteed feasible due to the resource constraints and the precedence of activities. The randomly allocated genes in chromosome have to be checked and reorder by ***Algorithm 1*** in section 3.3 to ensure the solution is feasible.

### 3.3 Genes Reordered

If the job sequence of $J_j, j \geq n+1$, is smaller than the sequence of $J_{j-n}$ or $J_{(j-n+1)}$, then the job $J_j$ has to be moved behind the $J_{j-n}$ and $J_{(j-n+1)}$ for satisfying the precedence relations. To ensure the chromosomes is feasible, the ***Algorithm 1*** is used to check and repair the infeasible chromosome to make it a feasible one.

***Algorithm 1*** Chromosomes check and reorder

Input: Random chromosome $V$. Output: Feasible $V$

Step 1. Set $j = n+1$

Step 2. Set $A = \max\{ q_{(j-n)}, q_{(j-n+1)} \}$

Step 3. If $q_j \leq A$ then move the $V(q_j)$ in a random position between $[V(A), V(2n-1)]$,else go to step 4

Step 4. If $j = 2n-1$ then output $V$, else $j \leftarrow j+1$, go to step 2.

### 3.4 Crossover

The crossover operation produces new sequences of offspring by exchanging two other permutations of the parent. The purpose of crossover is to generate a better offspring for the objective function. In this paper, we apply the method similar to Goldberg's Partially Mapped Crossover Method.

Through exchanging two permutations of the parent, the genes might be repeated or missing. For RCPSP or DRCPSP, the gene of jobs is unique. If crossover is operated, genes need to be checked and repaired for new offspring to avoid being repeated or missing. In order to keep the better chromosomes, the Roulette Wheel Selection method is used. In the general GA crossover operation, parents are selected according to their fitness. The better the chromosomes are, the more chances to be selected they have. The size of the selection section in the roulette wheel is proportional to the value of the fitness function of every chromosome. If the fitness value is better then the selection section is larger.

After crossover operation, new chromosomes have to be reordered to ensure the chromosome is feasible by using ***Algorithm 1***. The modified crossover method is shown in ***Algorithm 2*** and an example of crossover is illustrated in Figure 4.

***Algorithm 2*** Modified Crossover

Input: Chromosomes of parents $V_{P1}, V_{P2}$

Output: Chromosomes of children $V_{CH1}, V_{CH2}$

Step 1. Select the crossover genes length $L$ of chromosome by Roulette Wheel Selection method.
Select start crossover position $C_p$ for parents $V_{P1}, V_{P2}$ between $[1, (2n-1-L)]$ at random.

Step 2. Exchange two crossover genes between parents to produce proto-children $V'_{CH1}$ and $V'_{CH2}$ from start position.

Step 3. Determine the mapping relationship between two crossover genes.

Step 4. Legalize offspring $V'_{CH1}$ and $V'_{CH2}$ with the mapping relationship to make sure the genes is unique in a chromosome, obtain the legal children $V''_{CH1}$ and $V''_{CH2}$.

Step 5. Reorder $V''_{CH1}$ and $V''_{CH2}$ by using ***Algorithm 1***, the results are $V_{CH1}$ and $V_{CH2}$

$V_{P1}$ [ 1 6 9 4 5 8 7 15 2 3 10 17 13 16 19 12 11 18 14 ]
$V_{P2}$ [ 2 7 10 4 3 8 1 13 17 5 12 6 11 9 16 19 14 15 18 ]
$V'_{CH1}$ [ 1 6 9 4 3 8 1 13 17 5 12 6 11 9 19 12 11 18 14 ]
$V'_{CH2}$ [ 2 7 10 4 5 8 7 15 2 3 10 17 13 16 16 19 14 15 18]

$V''_{CH1}$ [ 7 2 16 4 3 8 1 13 17 5 12 6 11 9 19 15 10 18 14 ]
$V''_{CH2}$ [ 6 1 11 4 5 8 7 15 2 3 10 17 13 16 9 19 14 12 18]
$V_{CH1}$ [ 7 2 4 3 8 1 13 17 5 12 6 11 9 15 10 18 19 14 16]
$V_{CH2}$ [ 6 1 4 5 8 7 15 2 3 10 17 13 16 9 19 14 12 11 18]

**Figure 4. An example of modified crossover operation with 19 genes, $L = 10$, $C_p = 4$**

### 3.5 Mutation

A mutation operator in GA is used to avoid the convergence to local optimum and to reintroduce new combination of genes in the population. By mutating a chromosome, the sequence of genes will be slightly changed. In the literature there are basically three different mutation methods: 1) SWAP, 2) POSITION and 3) SHIFT. In this paper, we apply the SHIT mutation operator, since this method outperforms the other two methods (see RUIZ et al. 2006). The SHIFT

mutation is to pick position $X$ randomly and relocated it to another randomly picked position $Y$. The activities between these two positions move along. After the mutation operation, the chromosome $V_M^{'}$ needs to be repaired by using ***Algorithm 1***, and the result is denoted $V_M$. We proposed the Roulette Wheel Selection method to be applied to mutation. The mutation rate in the roulette wheel is proportional to the value of the fitness function of every chromosome. The better the fitness value is, the larger the gene mutation rate is. For example, suppose there are two chromosomes, the fitness value sequence $V_{CH2}$ is better than $V_{CH1}$. The better chromosome $V_{CH2}$ is to be mutated into two genes, the other $V_{CH1}$ is one gene. Two examples of mutation operation results are illustrated in Figure 5.

$V_{CH1}$ [ 7 2 4 3 **8** 1 13 17 5 12 6 11 9 15 10 18 19 14 16 ]

$V_{M1}^{'}$ [ 7 2 4 3 1 13 17 **8** 5 12 6 11 9 15 10 18 19 14 16 ]

$V_{M1}$ [ 7 2 4 3 1 13 8 5 12 6 11 **17** 9 15 10 18 19 14 16 ]

(a) One gene mutation $X = 5, Y = 8$

$V_{CH2}$ [6 1 4 5 8 7 15 **2** 3 10 17 13 16 9 19 **14** 12 11 18 ]

$V_{M2}^{'}$ [6 1 4 **14** 5 8 7 15 3 10 17 13 16 **2** 9 19 12 11 18 ]

$V_{M2}$ [6 1 4 5 **14** 8 7 15 3 10 17 13 16 **2** 9 19 12 11 18 ]

(b) Two genes mutation $X_1 = 8, Y_1 = 13, X_2 = 16, Y_2 = 4$

**Figure 5. An example of mutation with 19 genes**

### 3.6 Dynamic Resources Required

In Step B of the ***Main Algorithm***, suppose $V$ is obtained from Step A, $\forall J_j \in N$, then before computing makespan, dynamic resources have to be computed to ensure the schedule is feasible. The dynamic resource of parts $P(J_j)$ required by $J_j, \forall J_j \in N$, based on $V$ can be computed by ***Algorithm 3***.

***Algorithm 3.*** Computing required sources of parts base on JPS

Input: $n$, $V$. Output: $P(J_j)$, $J_j \in N$

Step 1. Set $i = 1$

Step 2. Find $j = V(i)$

Step 3. If $j \le n$ then $P(J_j) = \{ p_j \}$, $temp\_P(J_j) = P(J_j)$,

else $u = j - n$, $P(J_j) = temp\_P(J_u) \cup temp\_P(J_{u+1})$, $v = \{ \min\ l \mid p_l \in P(J_j) \}$,

$w = \{ \max\ l \mid p_l \in P(J_j) \}$, $temp\_P(J_v) = P(J_j)$, $temp\_P(J_w) = P(J_j)$

Step 4. If $i = 2n-1$ then output $P(J_j)$, $J_j \in N$, else $i \leftarrow i+1$ go to step 2.

Example: Given $n = 5$ and $V = [2\ 3\ 7\ 4\ 5\ 9\ 1\ 8\ 6]$, the resources required for each job can be computed, and the results are as below. $P(J_2) = \{p_2\}$, $P(J_3) = \{p_3\}$, $P(J_7) = \{p_2, p_3\}$, $P(J_4) = \{p_4\}$, $P(J_5) = \{p_5\}$, $P(J_9) = \{p_4, p_5\}$, $P(J_1) = \{p_1\}$, $P(J_8) = \{p_2, p_3, p_4, p_5\}$, $P(J_6) = \{p_1, p_2, p_3, p_4, p_5\}$.

### 3.7 Local Search Procedure

When a chromosome is obtained by GA initialization or evolution, the makespan of objective function can be evaluated. The approach of computing makespan by using the genetic algorithm based method is defined as GABM. The staring time of each job by GABM has to follow the job precedence of genes in the chromosome. The starting time cannot violate the constraints:

$s_j \le s_k$, where $q_j < q_k$, $j, k = 1,2,...,2n-1$

The procedure of computing makespan by GABM is illustrated in ***Algorithm 4.***

***Algorithm 4*** Assigning each job to a construction team and computing project makespan by GABM

Input: $n, m, V, P(J_j), d_j, j = 1,2,...,2n-1$. Output: $f_{V(2n-1)}$

Step 1. Set $i = 1, P(J_{M(T_k)}) = \phi,\ T_k \in R_T$

Step 2. Find $j = V(i)$

Step 3. $P_c(J_j) = P(J_j) \cap (P(J_{M(T_1)}) \cup P(J_{M(T_2)}) \cup ... \cup P(J_{M(T_m)})$

Step 4. If $P_c(J_j) = \phi$ then $s_j = \max\{\min f_{M(T_k)}, \max s_{M(T_k)} \mid T_k \in R_T\}$, $U_j = T_k, \min_{T_k \in R_T} f_{M(T_k)}$,

else $s_j = \max\{f_{M(T_k)} \mid T_k \in R_T\}$, $U_j = T_k, \max_{T_k \in R_T} f_{M(T_k)}$

Step 5. $f_j = s_j + d_j$, $M(U_j) = j$

Step 6. If $i = 2n-1$ then output the makespan $f_{V(2n-1)}$, else $i = i+1$ go to step 2.

In this paper we propose the genetic algorithm with automatic shift method (GAASM) to improve the scheduling and makespan from the GA based solution. In Step C of the ***Main Algorithm***, if the value of $n, m, V, P(J_j), d_j$, where $j = 1,2,...,2n-1$ are given, then assign each job to a construction team base on constraints (4) (5) and then compute the makespan $f_{V(2n-1)}$ by using ***Algorithm 5.*** The main improvement of this algorithm is the shifting of the succeeding job sequence to become the precedence job and change the jobs sequence when $T_k$ and $P(J_j)$ are available. After computing, the makespan and a new offspring can be obtained.

***Algorithm 5*** Assigning each job to a construction team and computing project makespan by GAASM.
Input: $n, m, V, P(J_j), d_j, j = 1,2,...,2n-1$. Output: $f_{V(2n-1)}$, New $V$

Step 1. Set $i = 1, P(J_{M(T_k)}) = \phi, T_k \in R_T$

Step 2. Find $j = V(i)$

Step 3. $P_c(J_j) = P(J_j) \cap (P(J_{M(T_1)}) \cup P(J_{M(T_2)}) \cup ... \cup P(J_{M(T_m)})$

Step 4. If $P_c(J_j) = \phi$ then $s_j = \min_{T_k \in R_T} f_{M(T_k)}$, $U_j = T_k, \min_{T_k \in R_T} f_{M(T_k)}$,
else $s_j = \max\{f_{M(T_k)} \mid T_k \in R_T\}$, $U_j = T_k, \max_{T_k \in R_T} f_{M(T_k)}$

Step 5. $f_j = s_j + d_j$, $M(U_j) = j$

Step 6. If $i = 2n-1$ then go to step 7, else $i \leftarrow i+1$ go to step 2.

Step 7. Output makespan $f_{V(2n-1)}$ and compute a new $V$ as offspring by ranking the start time $s_j$ of $V$.

Step D of the ***Main Algorithm*** is used to create an objective function and do the fitness testing with makespan which is obtained by Step C. If fitness is satisfied, then stop. Otherwise, go to Step E to get a new feasible array of $V$, and then repeat Step B.

## 4. A REAL-LIFE EXAMPLE

A petrochemical splitter tower was designed to serve the recovery propylene fluid of refinery plant. Its design pressure is 16.7 kg/cm$^2$G with full vacuum, and design temperature is 65 $^0$C. The total height is 47600 mm and internal diameter is 4300 mm. It is made by one top head, one bottom head, fourteen cylinder layers of shell with individual steel plate thickness (22mm~40mm) and internal components, and four layers of skirts with different size and sharp. Therefore, the number of unit parts $n$ is twenty. Each unit part ( $p_1, p_2, ..., p_{20}$ ) has its specific components including inlet/outlet nozzles, tray supports and accessories, and can be manufactured individually. The manufacture jobs of unit parts are represented by the set of $\{J_1, J_2, ..., J_{20}\}$. The manufacture job of each unit part includes steel plate cutting, forming, grinding, welding, accessory parts installation and non-destructive (NDE) testing. Beside the manufacture jobs, all unit parts need to be assembled together. The set of assemble jobs are $\{J_{21}, J_{22}, ..., J_{39}\}$. The assembly job of each unit part includes unit parts assembling, welding and NDE testing. The set of total project jobs can be represented by $N = \{J_1, J_2, ..., J_j, .., J_{39}\}$. Each job is processed by a construction team which needs three workers, a set of automatic welding machine and other accessory equipments. The number of available construction teams $m_a$ is five teams. The set of construction teams is $R_T = \{T_1, T_2, ..., T_5\}$. The set of job duration which is considered by project manager base on work volume for each job $J_j, j = 1, 2, ..., 39$, is $D_J = \{d_1, d_2, ..., d_{39}\}$={ 40, 33, 17, 16, 14, 20, 25, 18, 30, 27, 40, 31, 23, 37, 33, 37, 17, 33, 31, 15, 13, 31, 35, 11, 37, 20, 40, 29, 20, 33, 37, 14, 35, 16, 25, 20, 13, 29, 11} days. The goal is to find the optimal scheduling and minimum makespan.

The project manager used the existing methods of the company plan rules CP1 and CP2 to plan the job process, the makespan were obtained as $f_{V(39)}(\text{CP1}) = 257$ days and $f_{V(39)}(\text{CP2}) = 253$ days. If proposed GAASM method is employed, the minimum makespan is $f_{V(39)}(\text{GAASM}) = 229$ days, and the improvement rate compared with CP1 and CP2 is 10.89% and 9.49% individually. The optimal JPS can be derived, and the result is $V$ = [7 9 10 11 13 16 4 15 14 30 17 19 6 8 34 5 28 2 35 25 18 3 1 12 37 23 26 38 27 31 21 33 22 20 39 29 36 24 32]. The dynamic resources required for activities $J_j$, where $j = 21, 22, ...., 39$, are shown in Table 1. The optimal solution of job starting time $s_j$, finished time $f_j$ and assigned construction team $T_k$ for each job are summarized in Table 2. The optimal scheduling of the network for project and Gantt-Chart are illustrated in Figure 6 and Figure 7.

**Table 1. Computing results of the dynamic resources**

| $J_j$ | $P(J_j)$ | $J_j$ | $P(J_j)$ |
|---|---|---|---|
| $J_{21}$ | $\{p_1, p_2\}$ | $J_{31}$ | $\{p_{10}, p_{11}, p_{12}\}$ |
| $J_{22}$ | $\{p_1, p_2, p_3, p_4\}$ | $J_{32}$ | $\{p_1, p_2, ..., p_{11}, p_{12}, p_{13}, p_{14}, ..., p_{19}, p_{20}\}$ |
| $J_{23}$ | $\{p_3, p_4\}$ | $J_{33}$ | $\{p_{13}, p_{14}, p_{15}, p_{16}\}$ |
| $J_{24}$ | $\{p_1, p_2, ..., p_{11}, p_{12}\}$ | $J_{34}$ | $\{p_{14}, p_{15}\}$ |
| $J_{25}$ | $\{p_5, p_6\}$ | $J_{35}$ | $\{p_{14}, p_{15}, p_{16}\}$ |
| $J_{26}$ | $\{p_5, p_6, p_7\}$ | $J_{36}$ | $\{p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{19}, p_{20}\}$ |
| $J_{27}$ | $\{p_5, p_6, p_7, p_8, p_9\}$ | $J_{37}$ | $\{p_{17}, p_{18}\}$ |
| $J_{28}$ | $\{p_8, p_9\}$ | $J_{38}$ | $\{p_{17}, p_{18}, p_{19}\}$ |
| $J_{29}$ | $\{p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}\}$ | $J_{39}$ | $\{p_{17}, p_{18}, p_{19}, p_{20}\}$ |
| $J_{30}$ | $\{p_{11}, p_{12}\}$ | | |

**Table 2. The results of the optimal solution**

| $q_j$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $j$ | 7 | 9 | 10 | 11 | 13 | 16 | 4 | 15 | 14 | 30 | 17 | 19 | 6 | 8 | 34 | 5 | 28 | 2 | 35 | 25 |
| $T_k$ | 1 | 2 | 3 | 4 | 5 | 5 | 1 | 3 | 2 | 4 | 1 | 1 | 3 | 5 | 2 | 4 | 5 | 3 | 2 | 4 |
| $s_j$ | 0 | 0 | 0 | 0 | 0 | 23 | 25 | 27 | 30 | 40 | 41 | 58 | 60 | 60 | 67 | 73 | 78 | 80 | 83 | 87 |
| $d_j$ | 25 | 30 | 27 | 40 | 23 | 37 | 16 | 33 | 37 | 33 | 17 | 31 | 20 | 18 | 16 | 14 | 29 | 33 | 25 | 37 |
| $f_j$ | 25 | 30 | 27 | 40 | 23 | 60 | 41 | 60 | 67 | 73 | 58 | 89 | 80 | 78 | 83 | 87 | 107 | 113 | 108 | 124 |

| $q_j$ | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $j$ | 18 | 3 | 1 | 12 | 37 | 23 | 26 | 38 | 27 | 31 | 21 | 33 | 22 | 20 | 39 | 29 | 36 | 24 | 32 | 0 |
| $T_k$ | 1 | 5 | 2 | 3 | 1 | 5 | 4 | 1 | 4 | 3 | 2 | 5 | 2 | 1 | 1 | 4 | 5 | 4 | 4 | 0 |
| $s_j$ | 89 | 107 | 108 | 113 | 122 | 124 | 124 | 135 | 144 | 144 | 148 | 159 | 161 | 164 | 179 | 184 | 194 | 204 | 215 | 229 |
| $d_j$ | 33 | 17 | 40 | 31 | 13 | 35 | 20 | 29 | 40 | 37 | 13 | 35 | 31 | 15 | 11 | 20 | 20 | 11 | 14 | 0 |
| $f_j$ | 122 | 124 | 148 | 144 | 135 | 159 | 144 | 164 | 184 | 181 | 161 | 194 | 192 | 179 | 190 | 204 | 214 | 215 | 229 | **229** |

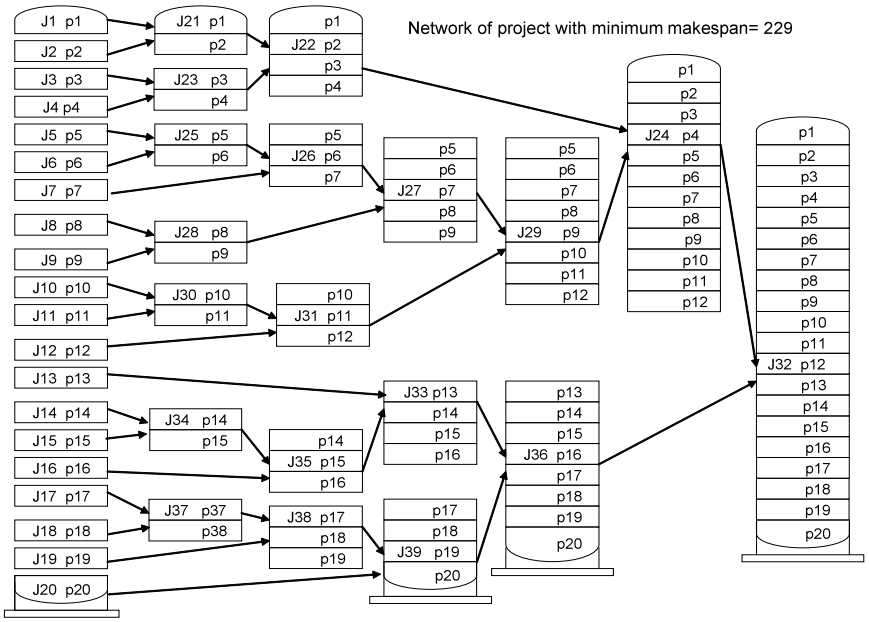


**Figure 6. Network of project with minimum makespan**

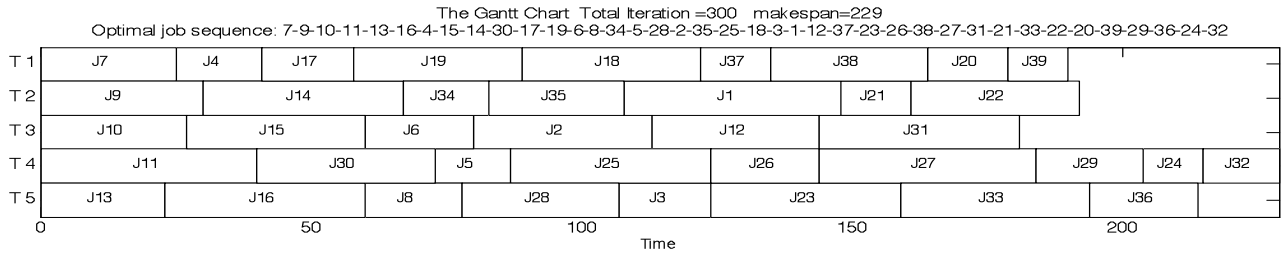


**Figure 7. The Gantt-Chart of optimal scheduling**

## 5. EXPERIMENT AND RESULTS

Two types of experiments are designed to test the performance of proposed GAASM. One is to compare the performance of GAASM with the genetic algorithm based method (GABM). Another is to compare the performance of GAASM with the company plan (CP) rules. In the simulation environment of these experiments, three factors of $n$, $m$ and $d_j$ are considered. Each factor is clarified into two levels. The number of jobs $n$ has the lower level 19 and the higher 39. In other words, the parts of $n$ for the lower level are 10 and 20 for the higher level. In practice the size of a shell plate for tower designed to be used mostly is 8'x 30' (height 2420 mm x width 9100 mm), the applied heights of parts $n = 20$ can reach to 48.4 meters which covers most of the towers height in the petrochemical industry. The maximum available number of construction teams is set so that the low level is 3 and the high is 5. The durations of jobs are randomly generated in [10, 20] for the low level and in [10, 40] for the high level.

### 5.1 The Comparison of GAASM and GABM

Eight problems are designed to test the performance of GAASM and GABM. The experimental factors and levels of comparison for the problem are set as Table 3. The GA operation parameters for GAASM and GABM are set as: population size 30, probability of crossover 0.6, probability of mutation 0.1 and maximum generation is 300. Each problem runs 30 times by GAASM and GABM. The results of the makespan obtained by GAASM and GABM are shown in Table 4, and the makespan obtained by GAASM is much better than by GABM obviously. Define the GAASM better (%) = (makespan of GABM－makespan of GAASM)/ makespan of GABM×100%. The GAASM better rate is between 13.67%-41.51%. When any level of three factors is high, the better rate of the GAASM is increased.

**Table 3. Factors and levels of experimental problems**

| Problems | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 |
|---|---|---|---|---|---|---|---|---|
| Number of jobs | 19 | 19 | 19 | 19 | 39 | 39 | 39 | 39 |
| Number of teams | 3 | 3 | 5 | 5 | 3 | 3 | 5 | 5 |
| Range of jobs duration | [10,20] | [10,40] | [10,20] | [10,40] | [10,20] | [10,40] | [10,20] | [10,40] |

**Table 4. Comparison of the makespan by using GAASM and GABM**

| Problems | | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 |
|---|---|---|---|---|---|---|---|---|---|
| GAASM | Mean | 106.1 | 79.63 | 174.6 | 130.2 | 206.4 | 140.2 | 350.1 | 236.0 |
| | Std dev | 0.305 | 1.159 | 0.928 | 1.305 | 0.504 | 1.157 | 1.147 | 2.767 |
| | Max. | 107 | 81 | 176 | 133 | 207 | 142 | 352 | 243 |
| | Min. | 106 | 78 | 173 | 127 | 206 | 137 | 348 | 229 |
| GABM | Mean | 122.9 | 115.6 | 207.9 | 195.5 | 254.0 | 227.7 | 446.7 | 403.5 |
| | Std dev | 3.166 | 3.654 | 4.097 | 6.892 | 4.556 | 4.591 | 8.582 | 11.89 |
| | Max. | 129 | 120 | 216 | 206 | 261 | 235 | 458 | 422 |
| | Min. | 117 | 108 | 199 | 183 | 242 | 217 | 428 | 378 |
| GAASM better (%) | Mean | 13.67 | 31.11 | 16.01 | 33.40 | 18.74 | 38.42 | 21.62 | 41.51 |

### 5.2 The Comparison of GAASM and CP

Eight groups with 240 problems are designed to test the performance of GAASM and CP rules. Each group has 30 problems. The experimental factors and levels for problems GP1-GP8 are set same as P1-P8 as shown in Table 3. Each problem generates the jobs duration information randomly from Table 5 with uniform distribution. The jobs sequence of company plan rules by section 2.5 are as follows.

- Jobs sequence by company plan rule for 19 jobs
  CP1: $V$ = [1 2 3 4 5 6 7 8 9 10 11 13 15 17 19 12 16 18 14]

CP2: $V$ = [10 9 8 7 6 5 4 3 2 1 19 17 15 13 11 18 14 12 16]

- Jobs sequence by company plan rule for 39 jobs

CP1: $V$ = [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 23 25 27 29 31 33 35 37 39 22 26 30 34 38 24 32 36 28]

CP2: $V$ = [20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 39 37 35 33 31 29 27 25 23 21 38 34 30 26 22 36 28 24 32]

Define the GAASM improvement rate (%)=(makespan of CP－makespan of GAASM)/ makespan of CP×100%. The results of makespan improved by GAASM are shown in Table 5. The GAASM average improvement rate is between 5.09%－14.60% in 8 groups of problems. The GAASM outperforms CP1 and CP2. The improved rate of makespan is increased where the available construction team and the range of job duration are in high level.

**Table 5. The improvement rate of the makespan by comparing GAASM and CP with 240 problems**

| Problems | | GP1 | GP2 | GP3 | GP4 | GP5 | GP6 | GP7 | GP8 |
|---|---|---|---|---|---|---|---|---|---|
| Better than CP1 (%) | Mean | 10.2754 | 11.2801 | 12.7687 | 14.4675 | 6.0922 | 8.7233 | 8.214 | 11.6854 |
| | Std dev | 2.7116 | 2.7188 | 3.6208 | 4.9641 | 1.7918 | 2.1036 | 3.2255 | 4.8626 |
| | Max. | 14.9606 | 18.4466 | 19.8347 | 24.0838 | 9.5238 | 13.0435 | 13.8889 | 21.1268 |
| | Min. | 5.0000 | 5.5556 | 5.7971 | 5.0955 | 2.7778 | 4.7297 | 1.4749 | 1.7857 |
| Better than CP2 (%) | Mean | 9.8249 | 10.6163 | 11.7546 | 14.7465 | 4.1062 | 7.0938 | 7.0609 | 10.4753 |
| | Std dev | 2.4466 | 3.6086 | 4.2711 | 6.0628 | 1.5279 | 2.8007 | 2.9306 | 4.3950 |
| | Max. | 14.2857 | 17.8571 | 19.4805 | 24.000 | 9.1304 | 12.8049 | 13.889 | 22.4913 |
| | Min. | 5.2174 | 4.3478 | 0.5714 | 0.6623 | 1.8957 | 2.1429 | 1.8373 | 2.7132 |
| Average (%) | Mean | 10.0502 | 10.9482 | 12.2616 | 14.6070 | 5.0992 | 7.9086 | 7.6374 | 11.0803 |

## 6. CONCLUSIONS

The chemical towers construction is the bottle neck activity in the entire construction project of a chemical plant. For some critical giant tower cases, the chemical towers have to be built before other facilities owing to the huge construction land space required. The shortening of the makespan of the chemical towers construction project is analogy to the vital cost savings of the entire chemical plant construction. In this paper we have developed a model of dynamic resource constrained project scheduling problem (DRCPSP) for the real-life chemical towers construction project. A genetic algorithm with auto-shift mechanism method (GAASM) is also proposed to generate optimal schedule with minimum makespan. The characteristics of resources in a DRCPSP model can be both independent and dynamic depending on the jobs processing sequences (JPS). This paper adopted both real-life and simulation problems to demonstrate the performance and applicability of DRCPSP with GAASM. The results from the real-life example show that the DRCPSP model with the GAASM solution method can demonstrate 10.89% and 9.49% improvement comparing with two existing company rules, Top-down and Bottom-up, respectively. On the other hand, the simulation experiments containing eight illustrative problems with 30 runs show that GAASM outperforms the conventional GA based method (GABM). Furthermore, from eight groups with a total of 240 problems comparing with two existing company rules show that GAASM can successfully improve the makespan with average 5.09%－14.60%. The simulation results show that GAASM is even more suitable while the available number of construction teams and the range of job durations are in the high level. For the project management administrators, this fruitful outcomes of this paper implies that the DRCPSP model with the GAASM method is both applicable and time-saving for the construction projects with independent and dynamic resource constraints.

## 7. REFERENCES


1. Salewski F., Schirmer A. and Drexl A. (1997). Project scheduling under resource and mode identity constraints: Model, complexity, methods, and application. European Journal of Operational Research, 102: 88-110.
2. Nudtasomboon, N. and Randhawa, S.U. (1997). Resource-constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs. Computers & Industrial Engineering, 32: 227-242.
3. Brucker, P., Knust, S., Schoo, A. and Thiele, O. (1998). A branch and bound algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research 107: 272-288.
4. Herroelen, W., Reyck, B.D. and Demeulemeester, E. (1998). Resource-constrained project scheduling: a survey of recent developments. Computers & Operations Research 25:279-302.
5. Brucker, P., Drexl, A., Möhring, R., Neumann, K., and Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. European Journal of Operational Research, 112:3-41.
6. Hartmann, S. and Kolish, R. (2000). Experimental evaluation of state-of-art heuristics for the resource-constrained project scheduling problem, European Journal of Operational Research, 127:394-407.

7. Abeyasinghe, M., Chelaka, L., Greenwood, David, J. and Johansen D.E. (2001). An efficient method for scheduling construction projects with resource constraints, International Journal of Project Management, 19:29-45.
8. Zamani, M.R. (2001). A high-performance exact method for the resource-constrained project scheduling problem. Computers and Operations Research, 28, 1387-1401.
9. Heilmann R. (2003). A branch-and–bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags. European Journal of Operational Research, 14:348-365.
10. Brucker P, Knust S. (2003). Lower bounds for resource-constrained project scheduling problems. European Journal of Operational Research, 149:302-313.
11. Bouleimen K, Lecocq H. (2003). A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research, 149:268-281.
12. Carlier J, Néron E. (2003). On linear lower bounds for the resource constrained project scheduling problem. European Journal of Operational Research, 149:314-324.
13. Valls V, Quintanilla S, Ballestín F (2003) Resource-constrained project scheduling: A critical activity reordering heuristic. European Journal of Operational Research, 149:282-301.
14. Kim, Kwan Woo, Gen, Mitsuo, Yamazaki, Genji. (2003). Hybrid genetic algorithm with fuzzy logic for resource-constrained project scheduling. Applied Soft Computing Journal, 2, 174-188.
15. Fleszar K, Hindi K.S. (2004). Solving the resource-constrained project scheduling problem by a variable neighbourhood search. European Journal of Operational Research, 155:402-413.
16. Kim KW, Yun YS, Yoon JM, Gen M, Yamazaki G. (2005). Hybrid genetic algorithm with adaptive abilities for resource-constrained multiple project scheduling. Computers in Industry, 56:143-160.
17. Al-Fawzan M.A., Haouari M. (2005). A bi-objective model for robust resource-constrained project scheduling. International Journal of Production Economics, 96:175-187.
18. Debels D, Reyck BD, Leus R, Vanhoucke M. (2006). A hybrid scatter search electromagnetism meta-heuristic for project scheduling. European Journal of Operational Research, 169:638-653.
19. Tseng LY, Chen S.C. (2006). A hybrid metaheuristic for the resource-constrained project scheduling problem. European Journal of Operational Research, 175:707-721
20. Valls V, Ballestín F, Quintanilla, S. (2008). A hybrid genetic algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research, 185:495-508.
21. Blazewicz, J., Lenstra, J.K., Rinooy Kan, A.H.G. (1983). Scheduling subject to resource constraints: Classification and complexity. Discrete Applied Mathematics, 5, 11-14.
22. Ibrahim A. K., Rajashekar M. (2008). Performance analysis of flowshop scheduling using genetic algorithm enhanced with simulation. International Journal of Industrial Engineering- Theory, Applications and Practice, 15, 62-72.

## BIOGRAPHICAL SKETCH

**Chun-Wei R. Lin** is a professor in the Department of Industrial Management at Yunlin University of Science & Technology, Taiwan. He received his MS and PhD degrees from the Department of Industrial Engineering and Management, The Pennsylvania State University, USA. His research activities include global logistics and supply chain management, manufacturing and logistics systems, green production management and health service industry management.

**Hsian-Jong Hsiau** is currently a PhD candidate in the Department of Industrial Engineering and Management at Yunlin University of Science & Technology, Taiwan. He received his MS degree in the same department, in 2004. His present research interests include construction project management and supply chain management.