Comparison between SA-based and EA-based Metaheuristics for Solving a Biobjective Unrelated Parallel Machine Scheduling Problem with Sequence Dependent Setup Times

Wei-Shung Chang and Chiuh-Cheng Chyu
Department of Industrial Engineering and Management
Yuan-Ze University, 135 Yuan-Tung Road, Chung-Li, 320, Taiwan, R.O.C
Email: s948905@mail.yzu.edu.tw
iehshu@saturn.yzu.edu.tw

Abstract. The parallel machine scheduling problem with sequence dependent setup times is among the most-studied and hard combinatorial optimization problems, and has been investigated in depth by numerous researchers of both theoretical and practical interests. An extension of this problem to the case of unrelated parallel machines with multi-objective is evidently more complex, and its application can be found in industries such as TFT-LCD and automobile manufactures. The optimization criteria of the problem under study are to minimize total flow time and total tardiness. In this study, several notable multi-objective optimization metaheuristics are employed to solve the proposed scheduling problem: (1) two Pareto converging evolutionary algorithms (PCGAs), one of which uses random key as encoding scheme while the other of which uses job list, and (2) two multi-objective simulated annealing (MOSA) algorithms – SMOSA and UMOSA in the literature. The performances of these algorithms are compared using various convergence and diversity metrics as the evaluation standards, via two test instances generated according to a method introduced in the literature. The experimental results have shown that the PCGA with random key representation outperforms the other three in various convergence measures, and has found most of the reference solutions. Additionally, although the aforementioned PCGA does not provide the best distribution in terms of diversity, it excels in a measure applicable to evaluate the convergence as well as diversity.

Keywords: Unrelated parallel machine scheduling, total tardiness, total flow time, Pareto converging genetic algorithms, simulated annealing, multi-objective optimization.

1. INTRODUCTION

Scheduling jobs on parallel machines is fairly widely encountered in many manufacturing environments, such as during the polarizer sheet cutting process and the glass etch polishing process in TFT-LCD manufacturing, and the drilling operation in a PWB line (Yu et al. 2002). However, in many practical situations, due to different purchase time epochs or manufacturing technology levels, the layout of unrelated parallel machines have more in common than that of identical parallel machines. The unrelated parallel machine scheduling problem (UPMSP) is more difficult than the identical case. Since the latter belongs to NP hard (Karp 1972), the UPMSP is also NP hard. For further knowledge and recent findings regarding the UPMSP, we refer to Logendran et al. (2007).

The goal that the production management attempts to achieve should be clear and measurable, and very often it contains more than one objective. Pfund et al. (2004) pointed out that in the UPMSP, problems with makespan minimization have been fairly widely studied, but problems involving the process characteristics such as release times and sequence dependent setup times, as well as problems with the minimization of the (weighted) number of tardy jobs and total (weighted) tardiness, remains largely unstudied. Pfund et al. (2004) provided a noteworthy survey of algorithms on the UPMSP with single and multiple objectives.

The solution approach to the UPMSP can be categorized into two classes: (1) metaheuristics and (2) exact solution methods. In metaheuristics, Hariri and Potts (1991) proposed a two-phase method to solve the UPMSP
with makespan minimization (Cmax). In the first phase, an integer programming technique is applied, and in the second phase, the earliest completion time rule is used to complete the final schedule of the UPMSP. Weng et al. (2001) proposed seven heuristics for the UPMSP with the objective of minimizing the weighted mean flow time, taking into consideration the job sequence dependent setup times. The experimental results indicated that their algorithms can find high quality solutions in short computational times for problems involving 120 jobs and 20 machines. Bank and Werner (2001) developed a constructive and iterative algorithm to solve the UPMSP with time window constraints on the job release dates and with the objective of minimizing the total weighted lateness.

Glass et al. (1994) developed a genetic algorithm (GA), simulated annealing (SA), and tabu search (TS) to solve the UPMSP without the sequence dependent setup time constraints. Their experiments conclude that GA performs no better than the other two algorithms. Sirvastava (1997) proposed a TS algorithm that could find high quality solutions in a short time for a part of the same instances.

Kim et al. (2002, 2003) proposed an SA to solve the UPMSP with a goal of total tardiness minimization, while taking into consideration job sequence-dependent setup time constraints. Logendran et al. (2004, 2007) developed a TS for the same problem with additional considerations of dynamic release dates and time window machine availability, where the objective was to minimize the sum of weighted tardiness jobs. Chen (2006) presented a record-to-record algorithm with tabu list to solve the UPMSP with the goal of minimizing the maximum tardiness. This paper also presented a threshold accepting algorithm with tabu list to solve the UPMSP to minimize the total tardiness.

The branch and bound (B&B) methods are commonly used to optimally solve the UPMSP in the literature. Some examples are Martello et al. (1997), Lancia (2000), Yu et al. (2002), Liaw et al. (2003), and Rocha et al. (2008).

Most research on the UPMSP has been focused on a single objective only and there have been comparatively fewer studies on the multi-objective UPMSP. Suresh et al. (1994) developed a TS for UPMSP with two objectives: minimizing the maximum makespan ($C_{\text{max}}$) and the maximum tardiness. The tabu list keeps the record of newly found non-dominated solutions. Jansen et al. (1999) modified the TS by Suresh et al. and solved the UPMSP with the objectives of minimizing $C_{\text{max}}$ and cost of scheduling. In this paper, a simulated annealing that interacts with the commercial software package CPLEX was developed for solving the UPMSP with three objectives—minimizing Cmax, total flow time, and total number of tardy jobs. Since only one schedule will be implemented in a real situation, a decision procedure is suggested to make the most preferable choice of the candidate solutions.

Simulated annealing (SA) has been known as a compact and robust technique to solve many NP-hard problems, including both single objective and multi-objective ones. It can provide excellent solutions to these problems with a substantial reduction in computational time. SA was first introduced by Kirkpatrick et al. (1983). We refer to Collins et al. (1988), Rutenbar (1989) and Eglese (1990) for surveys on single objective SA, and Suman and Kumar (2006) for surveys on multi-objective SA.

In this paper, we will employ the Pareto converging genetic algorithm (PCGA) to solve the UPMSP considering sequence dependent setup times with two objectives (BI-UPMSP-SDST): total flow time and total tardiness. A comparative study will be conducted via three instances of different problem sizes that were generated from a method in the literature. The PCGA was originally introduced by Kumar and Rockett (2002), and has successful applications in solving pair wise nonlinear continuous objective functions, as well as some combinatorial optimization problems such as TSP (Kumar and Singh, 2006).

The remainder of this paper is organized as follows. In Section 2, we describe the problem under study. In Section 3, we present the proposed solution approaches. Section 4 shows the numerical results along with discussions. Finally, Section 5 concludes the paper with implications for future research.

2. PROBLEM DEFINITIONS

If the heading should run into more than one line, the run-over should be flushed left. Suppose that there are $M$ unrelated parallel machines and $N$ jobs, where a job refers to a lot containing a certain number of identical items. Each job may have different processing times depending on the assigned machine. Furthermore, each machine can process one job at a time, the process on a job is non-preemptive, and there is a setup time depending on job sequences as well as the machine assigned to the process. Each job has its own due date.

The aim of this study is to develop an efficient and effective genetic-based algorithm for finding a set of non-dominated schedules to the UPMSP with two objectives: minimizing total flow time and total tardiness, taking into consideration the job sequence dependent setup times. The following notations will be used throughout this paper.

**Symbols:**

$m$: machine index, $m = 1, \ldots, M$

$j$: job index, $j = 1, \ldots, N$

$M$: total number of machines used
3. PROBLEM SOLVING METHOD

In this paper, we propose four meta-heuristics to solve the BIFO-UPMSP-SDST. Two of them use the Pareto converging genetic algorithm (PCGA; Kumar and Rockett 2002) as the algorithm main framework, while the other two take the multi-objective simulated annealing (MOSA) as the main framework. All algorithms are hybrid multi-objective optimization heuristics. An algorithm, named WBM-SMHS (weighted bipartite matching with single machine scheduling heuristic), will be repeatedly employed in the four framework with the aim of finding a near optimal solution to a weighted sum objective problem given a weight vector and a partition of N jobs into M groups, denoted \( G_1, \ldots, G_M \). The two PCGAs differ in encoding scheme and genetic operators. One takes job list (JL) and the other uses random key list (RKL). The two MOSAs are different in determining the acceptance probability of a neighborhood solution.

3.1 Hybrid Pareto Converging Genetic Algorithm

Initially, the problems that Kumar and Rockett (2002) dealt with were continuous. Hence there will be many Pareto optimal solutions even if the two objectives do not conflict significantly with each other. However, for a discrete optimization problem such as the machine scheduling problem, the number of Pareto optimal solutions may be fairly few. Thus, we need to make some adjustments for selecting algorithm parameters such as population size, stopping conditions, number of subpopulations (or called islands), etc.

PCGA is a steady-state algorithm and can be viewed as an example of \( (\mu + 2) - ES \) in terms of its selection mechanism; that is, at each iteration, we select two individuals from the parent population and perform genetic operations to produce two offspring. These two offspring will join the parent population and the two lowest ranked individuals will be discarded. At each iteration step, the whole population size \( \mu \) is ranked using Fonseca and Fleming’s ranking algorithm (1993, 1998). In this ranking procedure, the rank of an individual is equal to the number of individuals by which it is dominated, and all non-dominated individuals are given the same rank. In our PCGA, the score of rank one is 10, and the rank score decreases geometrically with ratio 0.5.

Roulette wheel method is used for reproduction, where the fitness of each individual is the score corresponding to its rank in the population. In order to have PCGA work efficiently and effectively, we adopt the following two strategies:

1. Multiple islands: An island is a randomly generated population. In order to produce diversified Pareto
solutions, each island is non-migrating in nature; i.e., individuals do not migrate from one land to another as is the case in a conventional island scheme. The proposed PCGA uses five islands which use weight vectors \((1.0, 0), (0.8, 0.2), (0.5, 0.5), (0.2, 0.8),\) and \((0, 1.0),\) respectively. When all five islands attain the termination conditions, they are compared with each other and non-dominated solutions are maintained.

(2) Stopping criteria: Each island terminates when one of the following conditions is met: (a) all individuals in the island have been retained in the first rank for 20 consecutive \((\mu + 2) - \text{ES} \) iterations, and (b) no new individual is added to the first rank during 40 consecutive iterations when the number of individuals in the first rank is less than the island size \(\mu\).

The choice of island size will depend on the problem characteristics, such as the number of jobs, the number of machines, as well as the due dates and processing times of jobs. The island size should be large when these problem characteristics have yielded many Pareto solutions.

In the proposed PCGA for solving the BIO-UMPS-P-SDST, two encoding schemes are used: job list and random key list. We shall refer to the PCGA using job list as PCGA-JL, and the one using random key list as PCGA-RKL.

Figure 1 depicts the encoding scheme and the decoding scheme for PCGA-JL. In the job list, the number in each cell represents the group that the job is assigned to. To obtain a high quality solution, a slightly sophisticated decoding scheme (called WBM-SMSH) is developed and described as follows:

Step 1: Single machine scheduling heuristic (SMSH). For each group and each machine, the early due date (EDD) rule is applied to obtain the initial solution, which will then be improved by a sequence of 2-opt or 3-opt local search based on the weighted sum objective. Notice that each island will be assigned a weight vector. Thus, we have obtained an \(M \times M\) matrix.

Step 2: Weighted bipartite matching (WBM). Apply the Hungarian method to \(M \times M\) matrix and obtain the best group-machine assignment with respect to the weight sum objective.

Figure 2 shows an example of the biased uniform crossover used in PCGA-JL. Firstly, a random number, say \(r\), is generated. Then we randomly select \(r\) positions and exchange of the numbers of the two parents.

The algorithm PCGA-RKL differs from PCGA-JL in crossover operation and initial solution in the decoding scheme. Figure 3 presents an example of the encoding scheme as well as the beginning step of the decoding scheme. For each cell in the RKL, the integral number corresponds to the group that the job belongs to, and the decimal number represents the priority order when these jobs are assigned to a machine for processing. The remaining steps of the decoding scheme are the same as PCGA-JL.

The crossover in PCGA-RKL uses the decimal number. If the sum of two decimal numbers in the same cell position is greater than one, then offspring 1 takes the number from parent 1; otherwise it takes the number from parent 2. Such an operation will be employed one by one to every cell of RKL. On the other hand, offspring 2 is generated using the rule contrary to offspring 1.

The structures of the two PCGAs are basically the same, except for the encoding and decoding schemes. Figure 5 presents the pseudo code of the PCGA.
3.2 Multi-Objective Simulated Annealing Algorithm

We present two multi-objective simulated annealing (MOSA) algorithms to solve the BIO-UPMSP-SDST. These two MOSAs differ mainly in two aspects: (1) the calculation of acceptance probability; one takes the formula by Suppapitnarm et al. (2000), and the other by Ulungu and Teghem (1998); (2) the method to update the external archive. We shall refer to the first one as SMOSA, and the second as UMOSA. The job list is used as solution representation for both SMOSA and UMOSA. Similar to PCGA, both SMOSA and UMOSA apply five search streams with weight vectors, (1.0, 0), (0.8, 0.2), (0.5, 0.5), (0.2, 0.8), and (0, 1.0), to solve the multi-objective UPMSP. These weight vectors will be used for computing the single machine scheduling problem. Like the PCGA, the WBM-SMSSH is used to decode JL as a new schedule. However, unlike the PCGA, SMOSA and UMOSA keep an external archive to update non-dominated solutions during the search process.

Neighborhood solutions

A biased uniform method is applied to produce neighborhood solutions. Firstly, a random number, say \( r \), is generated. Then we randomly select \( r \) positions and replace each position with a randomly selected machine.

Acceptance probability (AP)

In SMOSA, the acceptance probability is defined as follows:

\[
P_a = \exp \left\{ -\frac{\Delta s_i}{T_n} \right\} \quad (7)
\]

where \( \Delta s_i = f_i(x_{s,i}) - f_i(x_{s,i+1}) \) is the amount of improvement or deterioration of the \( k \)th neighborhood solution in the \( i \)th objective; \( T_n \) is the temperature after the \( n \)th decrease.

On the other hand, a weight vector for objectives, \((w_1, w_2)\), will be used before computing the acceptance probability in the UMOSA. Define \( \Delta s = w_1 \cdot \Delta s_1 + w_2 \cdot \Delta s_2 \) and

\[
P_a = 1 \text{ if } \Delta s \leq 0, \text{ and } P_a = \exp \left\{ \frac{\Delta s}{T_n} \right\} \text{ if } \Delta s > 0.
\]

External archive update

In SMOSA, a neighborhood solution will be put into external archive if this solution is not dominated by the current solution; on the other hand, in UMOSA, a
neighborhood solution is put into external archive for competition only if it is accepted.

Algorithm SMOSA

Input: a set of weight vectors \( \{w_1, \ldots, w_Q\} \) and SA parameters

Begin

\[ \text{For } q = 1 \text{ to } Q \text{ do} \]

\[ \text{Randomly generate a JL,} \]

\[ \text{Current} = \text{JL,} \]

While the stopping criteria not met do

\[ \text{Apply based uniform to generate a Neighborhood,} \]

WBM-SMSH(Current) using \( w_q \);

WBM-SMSH(neighbor) using \( w_q \);

If Neighborhood and Current are incomparable, then

\[ \text{put Current into external archive for competition;} \]

\[ \text{Current} = \text{Neighborhood,} \]

Else if \( \text{Neighbor} \) is accepted, then

\[ \text{Current} = \text{Neighbor,} \]

End (While).

End (For).

End.

4. NUMERICAL RESULTS

An experiment was conducted to evaluate the performance of the four algorithms and make comparison based on three benchmark instances. In the test environment, the four algorithms were coded in Visual Studio C++ NET and executed on a computer with Intel core dual 1.8GHz and 2 GB DDR566. Performance metrics include: (1) Convergence measure - Error ratio (ER), Generational distance (GD), and Overall nondominated vector generation ratio (ONVGR); (2) Diversity measure – Spread; (3) Both convergence and diversity – D1R.

4.1 Date Generation

Two test instances with problem size, 100 (jobs) x 5 (machines) and 100 x 10, were generated referring to the method proposed by Lee and Pinedo (1997). The process time of each job is a random number from an interval \([50, 150]\). Thus, the mean of the process time of a job is \( \mu = 100 \). The sequence dependent setup time of two consecutive jobs on machine \( m \), denoted \( s_{ijm} \), comes from a uniform distribution on an interval \([10, \eta \cdot \mu] \). Here we take the parameter value \( \eta = 0.2 \). The estimated makespan \( \hat{C}_{\text{max}} = (\beta \cdot \bar{s} + \mu) \cdot \mu \), where \( \mu \) is the total number of jobs divided by the total number of machines, \( \bar{s} \) is the mean setup time, and \( \beta = 0.4 + 10/\mu^2 - \eta/7 \). The due date \( \hat{d}_j \) of job \( j \) is generated using the two factors \( \tau \) and \( R \). Here, \( \tau \) is the tardiness factor, and the higher the \( \tau \) value, the tighter the tardiness; \( R \) is the due date range factor. \( \tau \) is defined as \( 1 - \frac{\hat{d}}{C_{\text{max}}} \) and \( R = (d_{\text{max}} - d_{\text{min}}) \cdot \frac{\hat{C}_{\text{max}}}{2} \). For the three test instances, we set \( \tau \) and \( R \) to 0.8 and 0.2, respectively. The due date \( d_j \) is uniformly distributed over the interval \([(1-R) \cdot \hat{d}, \hat{d}] \) with probability \( \tau \) and uniformly distributed over the interval \([\hat{d}, d_j + (R \cdot \hat{d}) \cdot \hat{C}_{\text{max}}] \) with probability \( 1 - \tau \). After some simple algebraic operations, we can see that the average of the due date generated is \( \hat{d} \). Therefore, once \( \hat{d} \) and the related parameters are determined, all due dates can be generated based on the above formula with specified parameter values. Finally, the weights of each job, \( \lambda_j \) and \( \mu_j \), are random numbers generated from the interval \((0, 1)\).

4.2 Algorithm Parameters

In the PCGA-JL and the PCGA-RKL, the number of islands and the island size are, respectively, set to 5 and 10. Each island starts with a randomly generated population with a specified weight vector for the two objectives. These individual weight vectors are used in WBM-SMSH. For other parameter settings of these two algorithms, see section 3.1 for details. Although the termination conditions of these two algorithms are the same, their encoding schemes will result in different computational times for convergence. For the purpose of fair comparison, in the SMOSA and the UMOSA, the number of cooling steps and the length of Markov chain are set to a certain pair of values so that their computational times are approximately the same as that of the PCGA-RKL. The cooling coefficient is set to \( \alpha = 0.95 \).

4.3 Performance Measures and Comparisons

In the absence of known true Pareto-front, we use the reference set as the non-dominated solutions over all the solutions obtained by the four algorithms. Tables 1-3 display the performance of these algorithms, including the number of reference solutions found, the number of non-dominated solutions found by the algorithm itself, convergence and diversity measures, as well as the computational time of each algorithm on the two benchmark instances. To compare fairly, except for the PCGA-JL, the termination conditions of the two MOSA algorithms are set to approximately the same computational time as that of the PCGA-RKL (see Table 3). The PCGA-JL meets the convergence condition earlier than the PCGA-RKL in both instances due to its less variability encoding scheme.
In the experiment, each instance was solved by each algorithm for three replication runs. The results are presented in Tables 1 and 2. As observed from the tables, a total of 30 and 18 reference solutions have been found for instance 1 (100 x 5) and instance 2 (100 x 10), respectively. For instance 1, the PCGA-RKL found 28 out of 30 and the PCGA-JL found the rest. The distribution becomes slightly smooth for instance 2, where the PCGA-RKL found 13 out of 18, PCGA-JL found 3, and UMOSA obtained 2.

Regarding the local non-dominated solutions found by each algorithm, the two PCGA algorithms produce much more local non-dominated solutions than the MOSA algorithms in instance 1; however, for instance 2, the first three algorithms produced roughly the same number. Figures 7 and 8 present the distributions of the reference solutions and local non-dominated solutions for the two instances. They indicate that the PCGA-RKL significantly outperforms the other three.

Other than the figure presentation, the algorithms are evaluated using the following metrics.
4.3.1 Convergence Measure

Three metrics are proposed to measure the closeness of a solution set to a Pareto set or a reference set.

The first metric is error ratio (ER), proposed by Veldhuizen (1999). It is defined as the ratio of the number of solutions not contained in the reference set over the number of representative solutions found by the algorithm. The drawback of this measure is that it will not accurately reflect true performance when the reference set is not large and dense enough. The second metric is overall non-dominated vector generation ratio (ONVGR; Van Veldhuizen, 1999), which is similar to the first one, except that the denominator is replaced with the number of solutions in the reference set. The third one is generational distance (GD; Deb, 2002), which is defined as follows:

\[
GD(A) = \frac{\sum_{i=1}^{\min} d_i}{|A|}
\]  

(8)

where

\[
d_i = \min_{f_k^{\text{Min}}} \left( f_k^i - f_k^j \right)^2
\]

(9)

In the above mathematical expressions, \( A \) is the representative solution set found by a proposed algorithm, \( P^* \) is the reference set, \( f_k^j \) is the \( j \)th objective value of the \( j \)th reference solution, and \( f_k^i \) is the \( k \)th objective value of the \( i \)th non-dominated solution in \( A \). Since the true Pareto front is unknown, we take Utopian and Nadir objective vectors as the standard coordinates for making normalization transformations. The Utopian objective vector, \(( f_1^{\text{Min}}, f_2^{\text{Min}} )\), is the best known minimum objective value (in \( P^*\)) minus a small number \( \epsilon \), and the Nadir objective vector, \(( f_1^{\text{Max}}, f_2^{\text{Max}} )\), is the best known maximum objective value (in \( P^*\)) plus \( \epsilon \).

The numerical result indicates that the PCGA-RKL achieves the best values in all three convergence measures, and the PCGA-JL is the second best.

4.3.2 Diversity Measure

Diversity is another important characteristic for measuring the quality of a non-dominated solution set. One popular measure for diversity is Spread (Deb, 2001), which calculates a relative distance between consecutive representative (non-dominated) solutions. It also takes care of the extent of the spread and requires a reference set \( P^* \) to compute the measure. Mathematically, it may be expressed as:

\[
\text{Spread}(A) = \frac{\sum_{m=1}^{M} \sum_{k=1}^{K} |d_i - \bar{d}|}{\sum_{i=1}^{n} d_i + |A|/\bar{d}}
\]

(10)

where \( d_i \) is the distance between two consecutive solutions, \( \bar{d} \) is the distance between the extreme solutions of \( P^* \) and \( A \) corresponding to the \( m \)th objective function. The value of this measure is zero for an ideal distribution of solution set but it can be more than 1 as well for a bad distribution of solution set.

Although the PCGA performs exceptionally better than the MOSA in convergence aspect, there is no significant difference in the Spread measure between all four algorithms.

4.3.3 Convergence and Diversity

\( D_1_r \) (Czyzak and Jaszkiewics, 1996) can measure both characteristics. The mathematical expression is shown...
below. It is unlike GD metric, whose comparison and measurement is based on each best (incomparable) solution that the algorithm has found, and the denominator is the number of best solutions. In contrast, the measurement of D1_R is based on each solution in the reference set. Given a set of representative solutions found by an algorithm, each reference solution aims at finding the nearest representative solution and computes the distance. Thus, D1_R has the function of measuring the convergence as well as the diversity to the solutions produced by an algorithm. Notice that the denominator of the D1_R is the number of solutions in the reference set.

\[
D1_R(A) = \frac{1}{|P^*|} \sum_{q \in P} \min\{d_{fq} | f \in A\} \tag{10}
\]

where

\[
d_{fq} = \sqrt{(f_1 - q_1)^2 + \cdots + (f_L - q_L)^2} \tag{11}
\]

Although its Spread measure is worse than some of the others, the PCGA-RKL performs the best in D1_R, which combines the measures of convergence and diversity.

5. CONCLUSIONS AND FUTURE RESEARCH

Over the years, researchers have focused their attention on the machine scheduling problem with a single objective. However, in practice, the goal that the management hopes to achieve is frequently multi-fold, along with a set of good alternatives available for making a decision. This paper studies the unrelated parallel machine scheduling problem in considering job sequence dependent setup times with two objectives: minimizing total weighted flow time and total weighted tardiness. Four methods based on two approaches are proposed to solve this NP-hard problem: (1) Pareto converging genetic algorithm (PCGA) and (2) multi-objective simulated annealing (MOSA) algorithms. These algorithms have been introduced in the literature, but they have never applied to the problem studies in this paper. Our experimental results, with two test instances generated using a method in the literature, indicate that the PCGA-RKL achieves the best performance in the number of reference solutions found, various convergence measures, as well as a composite measure applicable to both convergence and diversity.

Besides the two aforementioned approaches, there are a variety of techniques, such as NSGA-II (Deb et al. 2002), PAES (Knowles and Corne, 2000), AMOSA (Bandyopadhyay et al. 2008), can be employed to solve the multi-objective unrelated parallel machine scheduling problem with sequence dependent setup times. A good direction of the future research may be to develop algorithms based on these ideas to solve this multi-objective optimization problem and compare their performances with the PCGA-RKL.

ACKNOWLEDGMENT

This research was partially supported by the National Science Council in Taiwan under grant NSC 95-2221-E-155-045.

REFERENCES


AUTHOR BIOGRAPHIES

Wei-Shung Chang is a Ph.D. student in the Department of Industrial Engineering and Management, Yuan-Ze University, Taiwan. His research interests include Supply Chain Management, PCB Assembly, and Production Planning and Scheduling. His email address is <s948905@mail.yzu.edu.tw>

Chiuh-Cheng Chyu is an associate professor in the Department of Industrial Engineering and Management at Yuan-Ze University in Taiwan. His research interests include Applied Operations Research and Bayesian Decision Analysis. His email address is <iehshsu@saturn.yzu.edu.tw>