行政院國家科學委員會專題研究計畫 成果報告

求解具有與過去順序相依準備時間以及學習效應之雙目標單機排程問題之研究(I)
研究成果報告(精簡版)

計 畫 類 別 ：個別型
計 畫 編 號 ：NSC 99-2221-E-343-002-
執 行 期 間 ：99 年 08 月 01 日至 100 年 07 月 31 日
執 行 單 位 ：南華大學電子商務管理學系

計 畫 主 持 人：陳世興

計畫參與人員：碩士班研究生-兼任助理人員：白健志
碩士班研究生-兼任助理人員：宋央琳
碩士班研究生-兼任助理人員：Vasar, B
大專生-兼任助理人員：李琳琳
博士班研究生-兼任助理人員：陳萌智
博士班研究生-兼任助理人員：陳木中

報 告 附 件 ：出席國際會議研究心得報告及發表論文

處 理 方 式 ：本計畫涉及專利或其他智慧財產權，2 年後可公開查詢

中 華 民 國 100 年 10 月 14 日

行政院國家科學委員會專題研究計畫年度報告

題目：求解具有與過去順序相依準備時間以及學習效應之雙目標單機排程問題之研究(I)

Bi-criterion Single machine scheduling problem with a past-sequence-dependent setup times and learning effect

計畫編號：99-2221-E-343-002
執行期限：99 年 08 月 01 日至 100 年 07 月 31 日
主 持 人：陳世興
Email：shihhsin@mail.nhu.edu.tw
研究人員：陳萌智、陳木中、白健志、宋央琳

中文摘要

有關在單機排程問題中，考慮過去順序相依準備時間(past-sequence-dependent setup times, PSD)以及學習效應(learning effect, LE)等相關問題已漸漸得到較多研究關注，但從文獻回顧中得知，過去尚未有任何研究有關 PSD 與 LE 之多目標問題，主因多目標問題比單目標問題更加複雜，因此本計畫將會是第一個研究解決此多目標排程問題。採取的方法為先分析 PSD 與 LE 問題的特性，可先計算在第一個目標 (1) 最小化總完工時間(total completion time, TC)與 (2) 各工件完工時間之總絕對時間差值(total absolute differences in completion times, TADC)中，每個位置所帶來的權重之順序，再透過與處理時間的配對，得到針對此雙目標之最佳解。之後再到兩兩解之間，搜尋是否有新的柏拉圖解。如此反覆的運作，能有效的找出部分的柏拉圖解(minimum set of optimal sequences, MSOS)。為了驗證此方法的效率及求解品質，所提出的方法與目前最佳之多目標演算法 MOEA/D 比較，從實驗結果可得知此方法能在極短的時間內，能貼近柏拉圖解，並且所包含的值域比 MOEA/D 更廣，因此求解此問題時，本研究所提出的方法可得到較佳之求解品質。

關鍵詞：單機台排程、雙目標最佳化問題、與過去順序相依準備時間、學習效應、多目標演化式演算法

Abstract

This project solved the bi-criterion single machine scheduling problem of n jobs with a past-sequence-dependent setup times (PSD) and learning effect (LE) together. Although PSD and LE are gradually considered by researchers, there is none who deals with both effects in the bi-criteria problems. The reason is that multi-objective problem is harder than single objective problem because we need to obtain a set of Pareto solutions. As a result, this project is the first one who solves this problem. The two objectives considered are the total completion time (TC) and total absolute differences in completion times (TADC). The objective is to find a sequence that performs well with respect to both the

objectives, the total completion time and the total absolute differences in completion times.

To deal with this new problem, we proposed a method to solve it effectively. We first analyzed the parameters according to the weights on each position of the two objectives. Then, we matched the weights with the processing time of the jobs. So that two optimal sequences were obtained by this matching approach. In addition, we started to search a new Pareto solution located between the two new solutions. The process was repeatedly to search a pair of solutions until no new solutions were found. Our approach was very efficient to find out thte minimum set of optimal sequences (MSOS). In order to evaluate the proposed method, we compared it with the state-of-art multi-objective algorithm, MOEA/D, on numerous instances. The empirical results shown the proposed algorithm was effectively to find out a set of approximate solutions in a very short CPU time when it was compared with MOEA/D. This proposed algorithm was promising to solve the biceriteria scheduling problem in this research.

Keywords: Single machine scheduling, Bi-criterion optimization problem, Past-sequence-dependent setup times, Learning effect, Multi-Objective Evolutionary Algorithms

2 緣由與目的：

In most single machine scheduling problems, the processing time of a job is assumed to be a constant. However, a well-know concept in management science literature is learning effect (LE) first discovered in [18, 6]. Later on, past-sequence-dependent (PSD) effect was studied [9, 8, 11]. No matter the PSD and LE, the processing time of a job is not a constant and depends on its position in the sequence. Both effects, nonetheless, yiled different results on the processing time.

LE is due to an operator who is familiar with an operation so that the processing time is reduced by the number of jobs he/she has processed [3, 18]. On the other hand, PSD increases the processing time because the setup time is depended on the number of jobs has been processed [8]. Since both effects are the trade-off in the processing time if we consider them into together, it shows an interesting point that how do we arrange a sequence that minimizes the considered objective(s).

In single machine scheduling, various objectives are considered, such as mean flow time, mean tardiness, maximum flow time, maximum tardiness, number of tardy jobs, weighted mean of earliness, and tardiness. Moreover, these objectives may present trade-off between or among the objectives to each other. In other words, no single solution can optimize all the objectives at the same time. Pareto optimal solutions, which characterizes the best trade-offs among the objectives, are of practical interest to a decision maker.

To the best of our knowledge, this research is the first research which consider the bi-objective single machine scheduling problem with the consideration of LE and PSD effect. We consider the objective of minimizing the total completion time (TC) and total absolute difference in completion times ($TADC$). In addition, a good approach was used in this research.

When we solve the bi-criteria scheduling problem, the set of all the optimal Pareto solutions in the objective space is called the Pareto front (PF). Because bi-criteria problems and multi-objective problem are hard to solve, many researchers like to employ heuristics or Multi-objective evolutionary algorithms (MOEAs) to find a set of approximation solutions to the PF. In this research, we employed the Aneja-Nair [1] which demonstrated an efficient method to acquire a set of Pareto solutions. This method is able to find minimum set of optimal sequences ($MSOS$) efficiently. When we want to have complete set of optimal sequences ($CSOS$), we should utilize some MOEAs to do the global search. Moreover, we also evaluated the performance of the Aneja-Nair method in the extensive experiments. Finally, even though PSD and LE have received many attentions in recent years [6, 2, 14, 10, 17, 9, 5, 11], there was none who solve the bi-criteria problems. It is of importance of this research in the academic study.

3 研究報告應含的內容

3.1 Methods

This research employs the Aneja-Nair's algorithm and MOEAs to solve the bi-criterion Single machine scheduling problem in the first year. Aneja-Nair's method is able to obtain the minimum set of optimal sequences ($MSOS$) effectively due to this method search a good solution between a pair of Pareto solutions. When we want to have complete set of optimal sequences ($CSOS$), we utilize the MOEAs because it has a better capability of doing the global search. In the second year, we study the Aneja-Nair method works with MOEAs. After Aneja-Nair method generates $MSOS$ efficiently, these Pareto solutions can be used as the initial solutions for MOEAs. The solution quality could be improved.

A past-sequence-dependent setup times (PSD) and learning effect (LE) are considered in this research. The property of PSD is to increase the processing time while the number of jobs are processed. On the other hand, due to the comprehensive of the jobs, the learning effect decreases the processing time. Because both problems are the fundamental of this research before we consider the PSD and LE into together, the problem definition of PSD and LE are shown in the Section 3.2.1 and Section 3.2.2.

After that, we introduce the PSD integrates with LE which hasn't been solved no matter in bi-criteria single objective problem. This research is thus the first research which solves this problem. The combination of the PSD and LE is shown in Section 3.3. As soon as we introduce the problem, the methodology of Aneja-Nair method and MOEAs are discussed. In Section 3.5, we show the approach of the Aneja-Nair's algorithm. Later on, a MOEAs, specifically the MOEA/D, which will be used. Finally, the de-

tail of MOEA/D and the combination with Aneja-Nair method are explained in Section 3.7.

3.2 Problem Definitions of PSD and LE

3.2.1 Problem Definition of Past-Sequence-Dependent Setup Times

A batch of n independent jobs to be processed on a continuously available single machine. The machine can process only one job at a time and job splitting and inserting idle times are not permitted. All the jobs are available at time zero. Each job has a processing time p_j, $(j = 1, 2, ..., n)$. Let $s_{[j]}$ and $p_{[j]}$ be the setup time and processing time of a job occupying position j in the sequence respectively, and $s_{[j]}$ is defined as

$$s_{[j]} = \gamma \sum_{i=1}^{j-1} p_{[i]} \quad j = 2, 3, ..., n \;\; s_{[1]} = 0, \tag{1}$$

where $\gamma \geq 0$ is a normalizing constant. In the above Eq. (1), the actual length of the setup time depends on the value of γ. Koulamas and Kyparisis [8] considered the following scheduling problems with past-sequence-dependent setup times given by equation (1). Problem. (i): $1/s_{psd}/C_{max}$; Problem. (ii):$1/s_{psd}/TC$; Problem. (iii):$1/s_{psd}/TADC$; Problem.(iv):$1/s_{psd}/BC$. It is shown in [8] that the well-known shortest processing time (SPT) sequence is optimal for both the problems Problem. (i) $(1/s_{psd}/C_{max})$ and Problem. (ii) $(1/s_{psd}/TC)$.

3.2.2 Problem Definition of Learning Effect

When we include the learning effect as given in [4], the processing time of a job depends on its position in the sequence and is given as

$$p_{jl} \;=\; p_j l^{\alpha} \tag{2}$$

In the above equation, p_j is the normal processing time of job j, and p_{jl} is the processing time of job j if it is in position l of the sequence, and α is the learning index and $\alpha < 0$. From the above equation (2), we see that $p_{j1} > p_{j2} > p_{j3}... > p_{jn}$. For example, if $p_j = 3$ and $\alpha = -0.515$, then $p_{j1} = 3$, $p_{j2} = 2.0994$, $p_{j3} = 1.7037$, $p_{j4} = 1.4691$, $p_{j5} = 1.3095$, and so on.

3.3 Past-Sequence-Dependent Setup Times Together with Learning Effect

After the basic problem definitions of PSD and LE are introduced, this study explains the core of the research which takes the PSD and LE into considerate. The scheduling problem is defined in the following manner. A set of n independent jobs to be processed on a continuously available single machine. The machine can process only one job at a time and job splitting and inserting idle times are not permitted. All the jobs are available at time zero. Each job has a normal processing time p_r, $(r = 1, 2, ..., n)$. The processing time of a job occupying position r in the sequence is given by

$$p^A_{[r]} = p_{[r]} * r^a, \quad r = 1, 2, ..., n \tag{3}$$

where $a \leq 0$ is a constant learning index. Let $s_{[r]}$ be the setup time of a job occupying position r in the sequence, and $s_{[r]}$ is defined as

$$\begin{aligned} s_{[1]} &= 0 \\ s_{[r]} &= b * \sum_{j=1}^{r-1} p^A_{[j]} \quad r = 2, 3, ..., n \end{aligned} \tag{4}$$

where $b \geq 0$ is a normalizing constant. In the above Eq. (4), the actual length of the setup time depends on the value of b and learning index a. Let C_r denote the completion time of job r in a sequence. It is shown in [3] that the well-known shortest processing time (SPT) sequence is optimal for both the problems Problem. (i) $(1/LE, s_{psd}/C_{max})$ and problem. (ii) $(1/LE, s_{psd}/TC)$.

3.4 PSD and LE for the Objectives TC and $TADC$

Section 3.3 already defines the processing time when we consider the PSD and LE into together. Since the bi-objectives studied by this project are the total completion time (TC) and total absolute difference in completion times $(TADC)$, the following sections show the formulation of the two objectives. TC is explained and then the second objective $TADC$ is shown later.

3.4.1 First Objective: $1/LE, s_{psd}/TC$:

In this section, we consider the single-machine scheduling problem with the objective of minimizing the total completion time (TC). The TC of the $1/LE, s_{psd}/TC$ scheduling problem is defined as:

$$\begin{aligned} TC &= \sum_{r=1}^{n}(n-r+1)(s_{[r]}+p^A_{[r]}) \\ &= \sum_{r=1}^{n}\left\{(n-r+1)+\frac{b(n-r)(n-r+1)}{2}\right\}r^a p_{[r]} \\ &= \sum_{r=1}^{n}\left\{(n-r+1)(1+\frac{b(n-r)}{2})\right\}r^a p_{[r]} \end{aligned} \quad (5)$$

As mentioned in Koulamas and Kyparisis [8], Eq.(5) can be viewed as scalar product of two vectors. One vector is $p_{[r]}$ the vector of processing time of jobs. The other vector is v_r known as positional weights vector given as

$$v_r = (n-r+1)(1+\frac{b(n-r)}{2})r^a, \quad r=2,3,...,n \quad (6)$$

In the above Eq.(6), the value of $v_1 = 0$ because $s_{[1]}$ is zero. It is well-known from Hardy [7] that Eq. (5) is minimized by arranging the elements of $p_{[r]}$ in in increasing order and the elements of other vector v_r in non-increasing order. This is known as shortest processing time (SPT) sequence. Hence, for a given vector of $p_{[r]}$, using SPT, the optimal sequence for the $1/s_{psd}/TC$ problem can be obtained in $O(n\ log\ n)$ time. It can be seen that the optimal sequence depends on the value of SPT.

3.4.2 Second Objective: $1/LE, s_{psd}/TADC$:

When we calculate the single-machine scheduling problem with the objective of minimizing the total absolute difference in completion times (TADC), it is denoted as the $1/LE, s_{psd}/TADC$ scheduling problem given in [3].

$$\begin{aligned} TADC &= \sum_{i=1}^{n}\sum_{j=i}^{n}|C_j - C_i| \\ &= \sum_{r=1}^{n}(r-1)(n-r+1)\ (s_{[r]}+p^A_{[r]}) \\ &= \sum_{r=1}^{n}\{(r-1)(n-r+1) \\ &\quad +b*\sum_{j=r+1}^{n}(j-1)(n-j+1)\}r^a p_{[r]} \end{aligned} \quad (7)$$

The above Eq. (7) can be viewed as the scalar product of two vectors. One vector is $p_{[r]}$ the vector of processing time of jobs. The other vector is v_r known as positional weights vector given as

$$\begin{aligned} v_r &= \{(r-1)(n-r+1)\ + \\ &\quad b*\sum_{j=r+1}^{n}(j-1)(n-j+1)\}r^a \\ &\quad r=2,3,...,n \end{aligned} \quad (8)$$

In the above positional weights vector Eq. (8), the value of $v_1 = 0$ because $s_{[1]}$ is zero ($\because\ s_{[r]} = b\sum_{j=1}^{r-1}P^A_{[j]}$ and v_1 is an initial weight, see also [3]). It is well-known from [7] that Eq. (7) is minimized by arranging the vectors v_r and $p_{[r]}$ in opposite orders. This is also given in [9] as Lemma.1. Hence, for a given value of b and a learning index a, the optimal sequence for the $1/s_{psd}/TADC$ problem can be obtained in $O(n\ log\ n)$ time. It can be seen that the optimal sequence depends on the values of both b and a.

3.5 Proposed Algorithm

Due to we studied a bi-criteria scheduling problem in this research, we need to find a set of solutions that minimized both TC and $TADC$. Now, we will show how the bi-criterion problem in single machine scheduling with PSD and LE, can be solved to obtain the minimum set of optimal schedules ($MSOS$) using the concept of Aneja-Nair approach. This approach recursively searches for a new Pareto solution between a pair of existing solutions.

To accomplish this task, the one to one correspondence of variables between his bi-criterion problem

and bi-criterion single machine scheduling problem with a learning effect should be calculated. The one to one correspondence between the two problems are given in Table .1 of Appendix .1 Once this is known, the concept of Aneja-Nair method could be used to generate the $MSOS$. The proposed algorithm was shown as follows:

Notation:

- $Weight_1$: Initial weight vector for TC
- $Weight_2$: Initial weight vector for $TADC$
- X_1: A solution obtained by matching the $Weight_1$
- X_2: A solution obtained by matching the $Weight_2$
- Z_1: The objective values of the solution X_1
- Z_2: The objective values of the solution X_2
- Ω: A set of Pareto solutions
- δ: A set of new solutions
- X_i: A new solution found by the weight match algorithm

Algorithm:

1: $Weight_1 \leftarrow$ Calculate the Weights of TC
2: $Weight_2 \leftarrow$ Calculate the Weights of $TADC$
3: $X_1 \leftarrow$ MatchAlgorithm($Weight_1$)
4: $X_2 \leftarrow$ MatchAlgorithm($Weight_2$)
5: $\Omega \leftarrow \{X_1, X_2\}$
6: $CombinedWeight \leftarrow$ UpdateWeight(Z^{X_1}, Z^{X_2})
7: $\delta \leftarrow$ MatchAlgorithm($CombinedWeight$)
8: **while** δ is not empty **do**
9: Selected a Solution (X) and a Solution (Y) from Ω and δ, respectively
10: $CombinedWeight \leftarrow$ UpdateWeight(Z^X, Z^Y)
11: $X_i \leftarrow$ MatchAlgorithm($Weight$)
12: isNew $\leftarrow$ RedundantSolution(X_i)
13: **if** isNew is true **then**
14: $\delta \leftarrow \{X_i\}$
15: **end if**
16: Add Y to Ω if Y is compared with all solutions in Ω, and remove Y in δ
17: **end while**
18: Output Ω

Line 1 and Line 2 calculated the weight vector based of the two objectives. Once they are obtained, we used the weight matching algorithm to obtain two sequences according to the weight vector in Line 3 and Line 4. The two solutions are collected into the set Ω. It represents the Pareto set founded by the proposed algorithm. In addition, the objective values of the two solutions are computed in Line 6. Later on, we could find out the intermediate solution between X_1 and X_2 and then have a new solution in Line 7. During Line 8 to Line 17, it is an iterative procedure to find out whether there is any intermediate solution between each pair of solutions in Ω and δ. Once a solution in δ is compared with all the solutions in Ω, this solution is moved to Ω instead of staying in δ. As a result, when there is no solutions in δ which could be searched with Ω, the iterative process is terminated and the Pareto set Ω is output in Line 18. We present a numerical example for illustration.

Numerical Example: We now use the same 4 job problem given in Bagchi [3], with a learning effect ($\alpha = -0.152$). We show how to apply Aneja-Nair method and obtain the minimum set of optimal schedules. The normal processing time of these jobs are $p_1 = 1$, $p_2 = 2$, $p_3 = 3$, and $p_4 = 4$. The parameter b is set as 0.25.

Following Aneja [1], we first obtain point 1 in the objective space. This is obtained by minimizing z_1 the first objective; i.e., the total completion time (TC). The positional weights and the optimal sequence obtained are shown in Table.1. The optimal sequence obtained is $\{1\ 2\ 3\ 4\}$. The value of z_1 for this sequence is 21.2019. The value of z_2 the second objective; i.e., the total absolute differences in completion times ($TADC$), is 32.8285. Hence, $z_1^{(1)} = 21.2019$, and $z_2^{(1)} = 32.8285$.

Table 1: Matching algorithm for minimization of TC: $\alpha = -0.152, b = 0.25$

Position-r	1	2	3	4
$w_r^{1,\alpha}$	5.5000	3.3750	1.9040	0.8100
Sequence*	**1**	**2**	**3**	**4**

We now obtain point 2 in the objective space. This is obtained by minimizing z_2 the second objective; i.e., the total absolute differences in completion times ($TADC$). The positional weights and the optimal sequence obtained are shown in Table.2. The optimal

sequence obtained is $\{4\ 2\ 1\ 3\}$. The values of z_1 and z_2 for this sequence ($\{3\ 2\ 1\ 4\}$) are 28.3940 and 29.7896 respectively. Hence, $z_1^{(2)} = 28.3940$, and $z_2^{(2)} = 29.7896$.

Table 2: Matching algorithm for minimization of $TADC$: $\alpha = -0.152$

Position-r	1	2	3	4
$w_r^{2,\alpha}$	2.5000	4.2750	4.0195	2.4300
Sequence*	**3**	**2**	**1**	**4**

As mentioned in [1], we now use the points 1 and 2, and obtain the values of $a_1^{(1,2)} = |z_2^{(2)} - z_2^{(1)}| = 3.0389$, and $a_2^{(1,2)} = |z_1^{(2)} - z_1^{(1)}| = 7.1921$. The new single machine scheduling problem is formulated as to find the sequence of jobs (σ) that minimizes

$$f(\sigma) = \sum_{r=1}^{n} a_1^{(1,2)} * (n-r+1) r^{\alpha} p_{[r]} + \sum_{r=1}^{n} a_2^{(1,2)} (r-1)(n-r+1) r^{\alpha} p_{[r]} \quad (9)$$

The positional weights are $w_r^{c,\alpha} = a_1^{(1,2)} * (n-r+1) r^{\alpha} p_{[r]} + a_2^{(1,2)} (r-1)(n-r+1) r^{\alpha} p_{[r]}$. The positional weights ($w_r^{c,\alpha}$) for the above combined problem are: $w_1^{c,\alpha} = 34.6942$, $w_2^{c,\alpha} = 41.0026$, $w_3^{c,\alpha} = 34.6946$, and $w_4^{c,\alpha} = 19.9384$. The positional weights and the optimal sequence obtained are shown in Table.3. The optimal sequence obtained with these weights is $\{3\ 1\ 2\ 4\}$. The values of z_1 and z_2 for this sequence ($\{3\ 1\ 2\ 4\}$) are 26.9230 and 29.5340 respectively. We call this as point 3 in the objective space and $z_1^{(3)} = 26.9230$, and $z_2^{(3)} = 29.5340$.

Table 3: Matching algorithm for minimization of Eq. (9) : $\alpha = -0.152, b = 0.25$

Position-r	1	2	3	4
$w_r^{1,\alpha}$	34.6942	41.0026	34.6946	19.9384
Sequence*	**3**	**1**	**2**	**4**

We now use the points 1 and 3, and obtain the values of $a_1^{(1,3)} = |z_2^{(3)} - z_2^{(1)}| = 3.2945$, and $a_2^{(1,3)} = |z_1^{(3)} - z_1^{(1)}| = 5.7211$. The new single machine scheduling problem is formulated as to find the sequence of jobs (σ) that minimizes

$$f(\sigma) = \sum_{r=1}^{n} a_1^{(1,3)} * (n-r+1) r^{\alpha} p_{[r]} + \sum_{r=1}^{n} a_2^{(1,3)} (r-1)(n-r+1) r^{\alpha} p_{[r]} \quad (10)$$

The positional weights are $w_r^{c,\alpha} = a_1^{(1,3)} * (n-r+1) r^{\alpha} p_{[r]} + a_2^{(1,3)} (r-1)(n-r+1) r^{\alpha} p_{[r]}$. The positional weights ($w_r^{c,\alpha}$) for the above combined problem are: $w_1^{c,\alpha} = 32.4225$, $w_2^{c,\alpha} = 35.5767$, $w_3^{c,\alpha} = 29.2685$, and $w_4^{c,\alpha} = 16.5709$. The positional weights and the optimal sequence obtained are shown in Table.4. The optimal sequence obtained with these weights is $\{2\ 1\ 3\ 4\}$. The values of z_1 and z_2 for this sequence ($\{2\ 1\ 3\ 4\}$) are 23.3269 and 31.0535 respectively. We call this as point 4 in the objective space and $z_1^{(4)} = 23.3269$, and $z_2^{(4)} = 31.0535$.

Table 4: Matching algorithm for minimization of Eq. (10) : $\alpha = -0.152, b = 0.25$

Position-r	1	2	3	4
$w_r^{1,\alpha}$	32.4225	35.5767	29.2685	16.5709
Sequence*	**2**	**1**	**3**	**4**

We now use the points 2 and 3, and obtain the values of $a_1^{(2,3)} = |z_2^{(3)} - z_2^{(2)}| = 0.2556$, and $a_2^{(2,3)} = |z_1^{(3)} - z_1^{(2)}| = 1.4710$. The new single machine scheduling problem is formulated as to find the sequence of jobs (σ) that minimizes

$$f(\sigma) = \sum_{r=1}^{n} a_1^{(2,3)} * (n-r+1) r^{\alpha} p_{[r]} + \sum_{r=1}^{n} a_2^{(2,3)} (r-1)(n-r+1) r^{\alpha} p_{[r]} \quad (11)$$

The positional weights are $w_r^{c,\alpha} = a_1^{(2,3)} * (n-r+1) r^{\alpha} p_{[r]} + a_2^{(2,3)} (r-1)(n-r+1) r^{\alpha} p_{[r]}$. The positional weights ($w_r^{c,\alpha}$) for the above combined problem are: $w_1^{c,\alpha} = 5.0833$, $w_2^{c,\alpha} = 7.1512$, $w_3^{c,\alpha} = 6.3993$, and $w_4^{c,\alpha} = 3.7816$. The positional weights and the optimal sequence obtained are shown in Table.5. The optimal sequence is obtained with these weights is $\{3\ 1\ 2\ 4\}$.

The values of z_1 and z_2 for this sequence ($\{3\ 1\ 2\ 4\}$) are 26.9230 and 29.5340 respectively. We call this as point 5 in the objective space and $z_1^{(5)} = 26.9230$, and $z_2^{(5)} = 29.5340$.

Table 5: Matching algorithm for minimization of Eq. (11) : $\alpha = -0.152, b = 0.25$

Position-r	1	2	3	4
$w_r^{1,\alpha}$	5.0833	7.1512	6.3993	3.7816
Sequence*	**3**	**1**	**2**	**4**

When we use the points 1 and 4, we obtain the same optimal sequence $\{1\ 2\ 3\ 4\}$. When, we use the points 3 and 4, we obtain the same optimal sequence obtained is $\{3\ 1\ 2\ 4\}$. When we use the points 3 and 5, we obtain the same optimal sequence $\{3\ 1\ 2\ 4\}$. When we use the points 2 and 5, we obtain the same optimal sequence $\{3\ 2\ 1\ 4\}$. There are no other optimal sequences and so the algorithm terminates.

Based on the above, the minimum set of optimal sequences to this problem is: $\{1\ 2\ 3\ 4\}$, $\{3\ 2\ 1\ 4\}$, $\{3\ 1\ 2\ 4\}$, and $\{2\ 1\ 3\ 4\}$.

It is shown in Bagchi [3] that the cardinality of the set $MSOS$ is n, when the learning effect is not considered; i.e., $\alpha = 0$. Then in the author's previous research, it indicates that the cardinality of the set $MSOS$ is more than n when the learning effect is included; i.e., $\alpha \neq 0$. When it comes to the PSD and LE are considered together, the $MSOS$ is n. Thus, there are some interesting characteristic should be studied.

To summarize the Aneja-Nair method to solve this single machine scheduling problem, the first step is to obtain the optimal solutions (Pareto) or good sequence of each objective. After that, we will search the weight combination of two Pareto solutions and then to sort the new weights resulted in a new sequence. Then we iteratively search the new sequence between the pairwise solutions. By using Aneja-Nair method, we can find the $MSOS$ efficiently since each operation needs $nlogn$ to do the sorting. So it is quite efficient to obtain a set of Pareto solutions. When we have to obtain $CSOS$, we need to apply MOEAs to do global search. In addition, we can compare the MOEAs with Aneja-Nair method so that we can evaluate the Aneja-Nair method. The next section presents the approaches of MOEAs.

3.6 MOEA/D

In this project, we propose an implementation of MOEA/D for the bi-criteria single machine scheduling problems. We discuss the choice of decomposition methods in MOEA/D first, which is the setting of reference point and the way of updating neighboring solutions in Section 3.6.1. Later on, Section 3.6.2 presents the general framework of MOEA/D in Section 3.6.3 and the revised version for this scheduling problem is shown in Section 3.6.3.

3.6.1 Decomposition of Multi-Objective Optimization

The multi-objective optimization problem (MOP) can be stated as follows:

$$\text{Minimize } F(x) = (f_1(x), \ldots, f_m(x))^T$$
$$\text{Subject to } x \in \Omega \tag{12}$$

There are several approaches for converting the problem of approximation of the PF into a number of scalar optimization problems and they can be found in the literature (e.g., [12]). The most popular ones among them include the weighted sum approach and Tchebycheff approach which are introduced in the following:

Weighted Sum Approach [12]

This approach considers a convex combination of the different objectives. Let $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)^T$ be a weight vector, i.e., $\lambda_i \geq 0$ for all $i = 1, \ldots, m$ and $\sum_{i=1}^m \lambda_i = 1$. Then, the optimal solution to the following scalar optimization problem:

$$\text{Minimize } g^{ws}(x|\lambda) = \sum_{i=1}^m \lambda_i f_i$$
$$\text{Subject to } x \in \Omega \tag{13}$$

is a Pareto optimal point to 13, where we use $g^{ws}(x|\lambda)$ to emphasize that λ is a coefficient vector in this objective function, while x is the variables to be optimized. To generate a set of different Pareto optimal vectors, one can use different weight vectors x in the above scalar optimization problem. If PF is convex, this approach would work well. However, not every Pareto optimal vector can be obtained by this approach in the case of non-convex PFs [12]. To overcome these shortcomings, Tchebysheff approach is suggested.

Tchebycheff Approach [12]

In this approach, the scalar optimization problem is in the form

$$\text{Minimize } g^{te}(x|\lambda, z^*) = \max_{1\le i\le m}(\lambda_i|f_i(x) - z_i^*|)$$

$$\text{Subject to } x \in \Omega \tag{14}$$

where $z^* = (z_1^*, \ldots, z_m^*)^T$is the reference point, i.e., $z^* = \min_{1\le i\le m}(f_i(x)|x \in \Omega)$ for each $i = 1, \ldots, m$. For each x^* Pareto optimal point there exists a weight vector λ such that x^* is the optimal solution of Eq. 14 and each optimal solution of Eq. 14 is a Pareto optimal solution of Eq. 12. Therefore, one is able to obtain different Pareto optimal solutions by altering the weight vector. One weakness with this approach is that its aggregation function is not smooth for a continuous MOP. However, since this work aims to solve scheduling problems which is a type of discrete problem, it still can be used in the EA framework in this work.

3.6.2 Introduction of MOEA/D

Multi-objective evolutionary algorithm based on decomposition (MOEA/D) needs to decompose the MOP under consideration. Any decomposition approaches can serve this purpose. In the following description, we suppose that the Tchebycheff approach is employed. It is very trivial to modify the following MOEA/D when other decomposition methods are used.

Let $\lambda^1, \lambda^2, \ldots, \lambda^n$ be a set of even spread weight vectors and z^* be the reference point. As shown in Section I, the problem of approximation of the PF of 12 can be decomposed into scalar optimization subproblems by using the Tchebycheff approach and the objective function of the subproblem is

$$\text{Minimize } g^{te}(x|\lambda^j, z^*) = \max_{1\le i\le m}\{\lambda_i^j|f_i(x) - z_i^*|\} \tag{15}$$

where $\lambda^j = (\lambda^1, \lambda^2, \ldots, \lambda^n)^T$. MOEA/D minimizes all these objective functions simultaneously in a single run.

Note that g^{te} is continuous of $g^{te}(x|\lambda^i, z^*)$, the optimal solution of $g^{te}(x|\lambda^j, z^*)$ should be close to that of if λ^i and λ^j are close to each other. Therefore, any information about these g^{te}'s with weight vectors close to should be helpful for optimizing $g^{te}(x|\lambda^i, z^*)$. This is a major motivation behind MOEA/D.

In MOEA/D, a neighborhood of weight vector λ^i is defined as a set of its several closest weight vectors in $\{\lambda^1, \lambda^2, \ldots, \lambda^n\}$.

The neighborhood of the *i*th subproblem consists of all the subproblems with the weight vectors from the neighborhood of λ^i. The population is composed of the best solution found so far for each subproblem. Only the current solutions to its neighboring subproblems are exploited for optimizing a subproblem in MOEA/D.

At each generation, MOEA/D with the Tchebycheff approach maintains:

1. A population of *n* points $x^1, \ldots, x^n \in \Omega$, where x^i is the current solution of the *i*th subproblem.
2. $FV^1, \ldots, \mathrm{FV}^n$, where FV^i is the F-value of x^i, i.e., FV^i for each $i = 1, \ldots, n$;
3. $z = (z_1, \ldots, z_n)^T$, where z_i is the best value found so far for objective f_i;
4. An external population (EP), which is used to store nondominated solutions found during the search.

Consequently, the general framework of MOEA/D can be stated as follows:

Input:
MOP: Eq. 12;
a stopping criterion;
n: the number of the subproblems considered in MOEA/D;
A uniform spread of weight vectors: $\lambda^1, \lambda^2, \ldots, \lambda^n$;
T: the number of the weight vectors in the neighborhood of each weight vector.
Output: EP.
Step 1) Initialization:
Step 1.1) Set EP $= \phi$.
Step 1.2) Compute the Euclidean distances between any two weight vectors and then work out the *T* closest weight vectors to each weight vector. For each $i = 1, \ldots, n$; set $\mathrm{B(i)} = \{\lambda_1, \ldots, \lambda_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the closest weight vectors to λ^i.
Step 1.3) Generate an initial population $x^1, \ldots, x^n$ randomly or by a problem-specific method. Set FV^i.
Step 1.4) Evaluate $z = (z_1, \ldots, z_n)^T$ by a problem-specific method.

Step 2) Update:
For $i = 1, \ldots, n$, do
Step 2.1) Reproduction: Randomly select two indexes *k*, *l* from *B*(*i*), and then generate a new solution y from x^k and x^l by using genetic operators.
Step 2.2) Improvement: Apply a problem-specific repair/improvement heuristic on y to produce $y^{'}$.
Step 2.3) Update of z: For each $j = 1, \ldots, m$, if $z_j < f_j(y^{'})$, then set $z_j = f_j(y^{'})$
Step 2.4) Update of Neighboring Solutions: For each index $j \in B(i)$, if $g^{te}(y'|\lambda^j, z) \leq g^{te}(x^j|\lambda^j, z)$, then set $x_j = y^{'}$ and $FV^j = f_j(y')$.
Step 2.5) Update of EP:
Remove from EP all the vectors dominated by $F(y^{'})$.
Add $F(y^{'})$ to EP if no vectors in EP dominate $F(y^{'})$.
Step 3) Stopping Criteria: If stopping criteria is satisfied, then stop and output EP. Otherwise, go to **Step 2**.

In initialization, $\mathrm{B(i)}$ contains the indexes of the *T* closest vectors of λ^i. We use the Euclidean distance to measure the closeness between any two weight vectors. Therefore, λ^i's closest vector is itself, and then $i \in B(i)$. If $j \in B(i)$, the subproblem can be regarded as a neighbor of the *i*th subproblem.

In the *i*th pass of the loop in Step 2, the *T* neighboring subproblems of the *i*th subproblem are considered. Since x^k and x^l in Step 2.1 are the current best solutions to neighbors of the *i*th subproblem, their offspring y should hopefully be a good solution to the *i*th subproblem. In Step 2.2, a problem-specific heuristic is used to repair/improve y in the case when y invalidates any constraints, and/or optimize the *i*th g^{te}. Therefore, the resultant solution $y^{'}$ is feasible and very likely to have a lower function value for the neighbors of *i*th subproblem. Step 2.4 considers all the neighbors of the *i*th subproblem, it replaces x^j with $y^{'}$ if $y^{'}$ performs better than x^j with regard to the *j*th subproblem. FV^j is needed in computing the value of $g^{te}(x^j|\lambda^j, z)$ in Step 2.4.

Since it is often very time-consuming to find the exact reference point $z*$, we use z, which is initialized in Step 1.4 by a problem-specific method and updated in Step 2.3, as a substitute $z*$ for g^{te} in Step 2.4. The external population EP, initialized in Step 1.1, is updated by the new generated solution $y^{'}$ in Step 2.5. In the case when the goal in 12 is to minimize $F(x)$, the inequality in Step 2.3 should be reversed.

3.6.3 MOEA/D Modifications for Single Machine Scheduling problems

To further improve the performance of MOEA/D, some procedures of MOEA/D are modified. First of all, it is arguable that which decomposition method can be used in the MOEA/D. Miettinen [12] argued that the weighted-sum approach is good at convex problem while Tchebysheff approach is useful when the problem is non-convex. As a result, although our previous work showed that Tchebysheff approach outperformed the weighted sum approach, it is still not sufficient enough to conclude that Tchebysheff approach will perform better for the flowshop benchmarks.

Secondly, the z index is applied as a substitute of z^*. Hence, the value of z is apparently larger than or equal to that of z^* in the minimization problem and z doesn't guarantee a good lower reference value when the decomposition method normalizes the objective values. To provide a good approximation of the$z$, a parameter α is introduced and each z_i is multiplied by α if z_i is improved. The setting of α is configured by Design-of-Experiment.

Finally, once a good solution is found in the MOEA/D, the algorithm will replace its neighborhood solutions immediately. When the solution is very good, this new solution inevitably replaces all neighborhood solutions. This procedure enhances the convergence of the algorithm; however, it causes the problem of degrading the diversity of the population abruptly. Therefore, the genetic operators are not able to generate different offsprings since all the solutions are identical in the *T* neighbors. Therefore, this paper sets the maximum number of replaced neighborhood solution is 1 rather than be able to replace all neighbors.

There are many crossover and mutation methods. We utilize the two-point crossover and moving position mutation for the Crossover procedure and the Mutation procedure, respectively, because [15] found both of them were the better approaches for these two objectives. It is noted that there is no improvement or repair procedure applied in this multiobjective scheduling study.

Finally, we present the extensive results of MOEA/D and the proposed method to solve the single machine scheduling with LE and PSD. It is introduced in the next section.

3.7 Experimental Results

This project conducted extensive computational experiments on single machine scheduling problems to validate the performance of the proposed method on learning effect and past-sequence-dependent effect. To test the algorithms, there are numerous data sets published in the literature [16] for the single machine scheduling problems, including 20, 30, 40, 50, 60, and 90 jobs. We employed their instances of sks225a, sks325a, sks425a, sks525a, sks625a, and sks925a. In addition, we set the α as -0.152 and b is as 0.25 in our experiments. The stopping criterion of MOEA/D is the number of total examined evaluations. The following table (See Table 6) shows the parameter used by the MOEA/D, which were determined by Design-of-Experiment [13].

Table 6: Parameter settings of MOEA/D

Factor	Level
Population Size	200
Crossover Operator	Two-Point Central Crossover
Mutation Operator	Swap Mutation
Crossover Rate	0.9
Mutation Rate	0.5
Examined Solution	$n * 200$

The replication of each algorithm on the each instance is 10 times on a computer with Intel i5 650 CPU (3.2 GHZ). In addition, we aggregated the Pareto solutions obtained by MOEA/D and the proposed algorithm and then them into the reference set. The dominated solutions are removed from the aggregated list.

In Fig. 1, we firstly examine the solution quality for the MOEA/D and the proposed algorithm on small instances, including job 20, 30, and 40. We could find out MOEA/D could converge to reference. However, our proposed algorithm which utilized Aneja-Nair's approach may provide a better spread to the reference. Moreover, the convergence of these solutions fitted to the reference set. So our proposed method could outperform MOEA/D in the small-size instances.

When it comes to the larger-size instances, we demonstrated the results on sks525, sks625, and sks925 in Fig. 2. It clearly demonstrated that our proposed algorithm also provided a diversified Pareto solutions than MOEA/D. The number of solutions obtained by our proposed algorithms were more than the ones obtained by MOEA/D. Finally, the proposed algorithm also closed to the reference set. As a result, the proposed algorithm outperformed the MOEA/D in the single machine scheduling problems with learning effect and PSD.

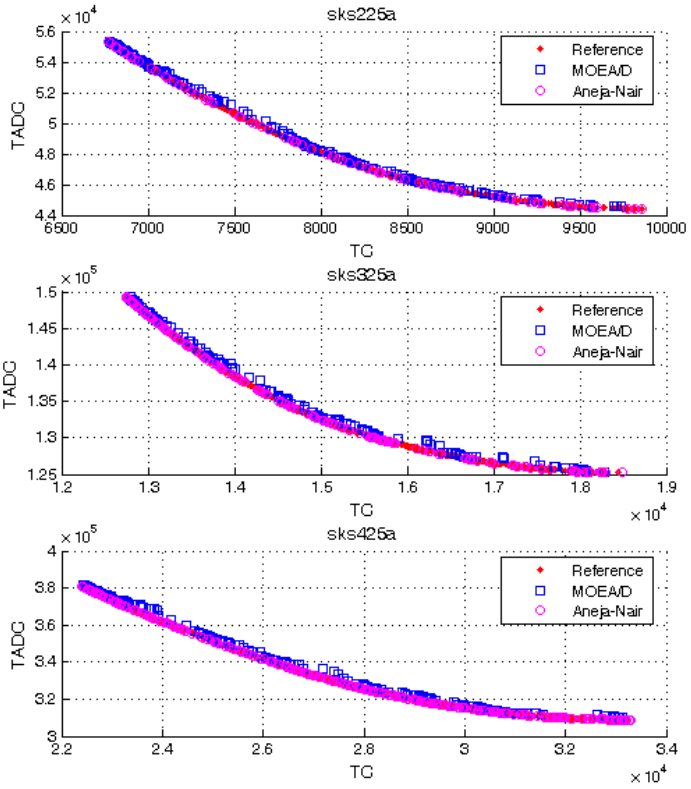

Figure 1: Reference set, MOEA/D and Aneja-Nair Method on sks225, sks325, and sks425

3.8 Conclusions

This project studied the single machine scheduling problems with learning effect (LE) and past-sequence-dependent (PSD) effect. Both effects has received increasing attentions. We considered two objectives involved in this problem, including the TC and $TADC$. Because the researcher did the parameter analysis of the learning effect (LE) and past-sequence-dependent (PSD), we proposed an algorithm which employed this technique. In addition, the proposed algorithm also utilized Aneja-Nair approach to find a new Pareto solution between two existing solutions. Our proposed algorithm could search for the Pareto solutions in an effective way instead of doing blind search. When our algorithm was compared with the state-of-art MO algorithm, MOEA/D, our proposed algorithm shown the convergence and the spread outperformed that of MOEA/D. Thus, the proposed algorithm was promising when we solved the single machine scheduling with LE and PSD.

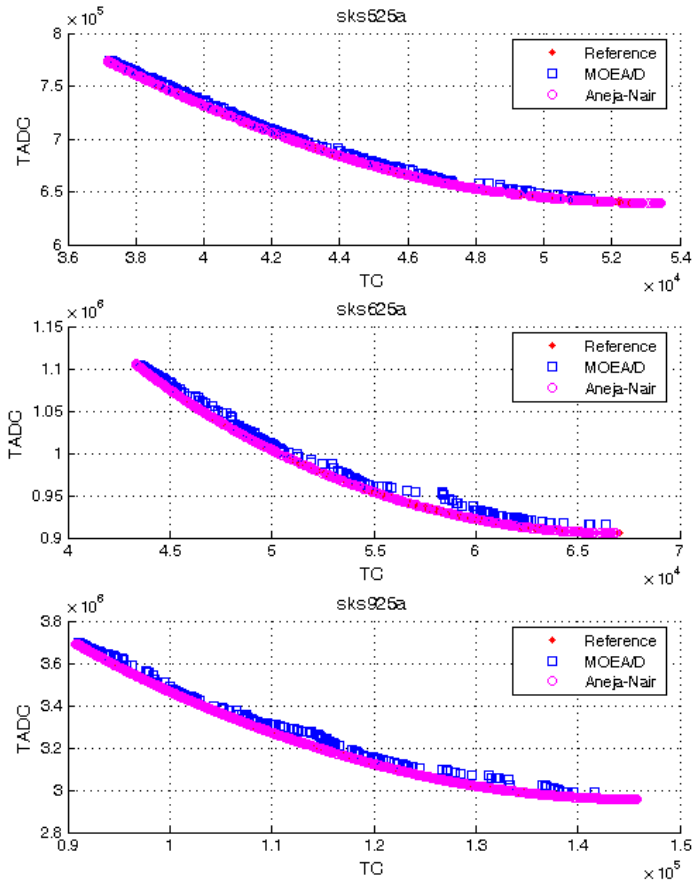

Figure 2: Reference set, MOEA/D and Aneja-Nair Method on sks525, sks625, and sks925

References

[1] Y. Aneja and K. Nair, "Scheduling jobs with position-dependent processing times," *Management Science*, vol. 25, pp. 73–78, 1979.

[2] A. Bachman and A. Janiak, "Scheduling jobs with position-dependent processing times," *Journal of the Operational Research Society*, vol. 55, no. 3, pp. 257–264, 2004.

[3] U. Bagchi, "Simultaneous minimization of mean and variation of flow time and waiting time in single machine systems," *Operations Research*, vol. 37, no. 1, pp. 118–125, 1989.

[4] D. Biskup, "Single-machine scheduling with learning considerations," *European Journal of Operational Research*, vol. 115, no. 1, pp. 173–178, 1999.

[5] D. Biskup and J. Herrmann, "Single-machine scheduling against due dates with past-sequence-dependent setup times," *European Journal of Operational Research*, vol. 191, no. 2, pp. 586–591, 2008.

[6] D. Biskup and D. Simons, "Common due date scheduling with autonomous and induced learning," *European Journal of Operational Research*, vol. 159, no. 3, pp. 606–616, 2004.

[7] G. Hardy, J. Littlewood, and G. Pólya, *Inequalities*. Cambridge University Press, 1988.

[8] C. Koulamas and G. J. Kyparisis, "Single-machine scheduling problems with past-sequence-dependent setup times," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1045–1049, 2008.

[9] W. H. Kuo and D. L. Yang, "Single machine scheduling with past-sequence-dependent setup times and learning effects," *Information Processing Letters*, vol. 102, no. 1, pp. 22–26, 2007.

[10] W. Kuo and D. Yang, "Minimizing the makespan in a single machine scheduling problem with a time-based learning effect," *Information Processing Letters*, vol. 97, no. 2, pp. 64–67, 2006.

[11] W. Lee, "A note on single-machine scheduling with general learning effect and past-sequence-dependent setup time," *Computers & Mathematics with Applications*, vol. 64, no. 4, pp. 2095–2100, 2011.

[12] K. Miettinen, *Nonlinear Multiobjective Optimization*. Springer, 1999.

[13] D. Montgomery, *Design and analysis of experiments*. John Wiley & Sons Inc, 2008.

[14] G. Mosheiov and J. Sidney, "Note on scheduling with general learning curves to minimize the number of tardy jobs," *Journal of the Operational Research Society*, vol. 56, no. 1, pp. 110–112, 2005.

[15] T. Murata and H. Ishibuchi, "Performance evaluation of genetic algorithms for flowshop scheduling problems," in *Proceedings of the First IEEE World Congress on Computational Intelligence*. IEEE, 1994, pp. 812–817.

[16] F. Sourd and S. Kedad-Sidhoum, "The One-Machine Problem with Earliness and Tardiness Penalties," *Journal of Scheduling*, vol. 6, no. 6, pp. 533–549, 2003.

[17] J. Wang and Z. Xia, "Flow-shop scheduling with a learning effect," *Journal of the Operational Research Society*, vol. 56, pp. 1325–1330, 2005.

[18] T. Wright, "Factors affecting the cost of airplanes," *Journal of Aeronautical Sciences*, vol. 3, no. 4, pp. 122–128, 1936.

.1 One to one correspondence between the TC and $TADC$

Table 7: One to one correspondence between the two problems

Bi-criterion Transportation Problem	Bi-criterion Single machine scheduling Problem
z_1	TC
z_2	$TADC$
c_{ij}	$(n-r+1)(1+\frac{b(n-r)}{2})r^{\alpha} = w_r^{1,\alpha}$
d_{ij}	$\{(r-1)(n-r+1) + b*\sum_{j=r+1}^{n}(j-1)(n-j+1)\}r^{a} = w_r^{2,\alpha}$
extreme point	optimal sequence

行政院國家科學委員會補助國內專家學者出席國際學術會議報告

年　　月　　日

附件三

報告人姓名	陳世興	服務機構及職稱	南華大學電子商務管理學系 助理教授
會議時間地點	June 5-8, 2009 Trondheim, Norway during	本會核定補助文號	Bi-criterion Single machine scheduling problem with a past-sequence-dependent setup times and learning effect
會議名稱	(中文) 2011 IEEE 演化式演算法研討會 (英文) 2011 IEEE Congress on Evolutionary Computation (IEEE CEC 2011)		
發表論文題目	(中文) 雙變數人造解基因演算法解決具整備時間之單機排程問題 (英文) Bi-Variate Artificial Chromosomes with Genetic Algorithm for Single Machine Scheduling Problems with Sequence-Dependent Setup Times		

報告內容應包括下列各項：

一、參加會議經過

It is my third time to join this conference because it is the best one in the field of evolutionary computation. Due to I'm going to familiar with the participants who are well-known in the area, there is no gap to discuss research with them. First of all, I met J. Lozano and J. McCall again who are well-known in Estimation of Distribution Algorithm (EDAs). They are interested in the bi-variate model used in our EDAs. In addition, because Prof. Lozano is going to re-submit an article in the reviewing paper of EDAs, it is quite excited to know his works in advance. Except for the EDAs, I focus on two topics related to multi-objective optimization and super computing on GPGPU. I learnt a lot from the masters and the presenters. I'm sure they are quite beneficial to doing the research together.

二、與會心得

When I demonstrated the research, I felt it draws the attentions of others. The reason is the results are fruitful and my presentation content is quite clear to the audiences even if they didn't learn the scheduling problems before. Finally, I also contact with a master who does the supercomputing in France. He is very kind to invite me to contact with him because he might be very interested in the supercomputing in the cloud. So we could start to work together since now.

三、考察參觀活動(無是項活動者省略)

N/A

四、建議

This conference is of good quality of papers and many world-class researchers are inivited. So I sincerely suggest that no matter for faculties or PhD students join this international conference since we can share the research experiences with the famous researchers and to get something back. (I mean the research inspirations and ideas.)

五、攜回資料名稱及內容

Except for a lot of research ideas, I brought the proceedings in DVDs. It is helpful to go through some interesting papers after I discussed with the participants.

六、其他

Thanks the NSC supports for the plane ticket, the partial living cost, and registration fee. I like to thanks NHU who may support the living cost.

AUBURN
UNIVERSITY

SAMUEL GINN COLLEGE OF ENGINEERING
INDUSTRIAL AND SYSTEMS ENGINEERING

May 23, 2011

Shih-Hsin Chen, PhD
Department of Electronic Commerce Management
Nanhua University
No. 55, Sec. 1, Nanhua Rd., Zhongkeng, Dalin Township
62248 Chiayi County Taiwan (R.O.C.)

Dear Dr. Chen:

This letter is to invite you to attend the Congress on Evolutionary Computation 2011 (CEC 2011) held from June 5 to 8, 2011 in New Orleans, U.S.A. Complete information about the conference can be found at http://cec2011.org. We understand that your passport number is .

We look forward to your presence at this conference. I can be reached on smithae@auburn.edu or 1-334-844-1400.

Sincerely,

Alice E. Smith

Alice E. Smith, Ph.D., P.E.
Professor and Chair
CEC 2011 Conference Co-Chair

SHELBY CENTER FOR
ENGINEERING TECHNOLOGY
SUITE 3301
AUBURN, AL 36849-5346

TELEPHONE
334-844-4340

FAX:
334-844-1381

www.auburn.edu

The Big EC

2011 IEEE Congress on Evolutionary Computation
June 5-8, 2011, New Orleans, USA

The paper submission deadline has been extended to January 28, 2011!

IEEE CEC 2011: Paper C-0180 Confirmation

Please print this confirmation page for future reference. You should receive an e-mail confirmation of your submission. **If you do not receive an e-mail notification within 24 hours, please contact Alice Smith <cec2011@ieee-cis.org>.**

Dear Author(s),

Your paper was submitted successfully to IEEE CEC 2011 and assigned number C-0180. Please use this number in all your correspondence. In case of any problem with your PDF file you will be notified and asked to resubmit a corrected file.

Your submission was recorded as follows:

Title: Bi-Variate Artificial Chromosomes with Genetic Algorithm for Single Machine Scheduling Problems with Sequence-Dependent Setup Times
Author(s): Shih-Hsin Chen and Min-Chih Chen
Affiliation(s):
Nanhua University, Taiwan
National Cheng Kung University, Taiwan
Email(s): shihhsin@mail.nhu.edu.tw, cthunter@mail.wfc.edu.tw
Preferred form of presentation: Oral

Abstract:
Artificial chromosomes with genetic algorithm (ACGA) is one of the latest Estimation of Distribution Algorithms (EDAs). This algorithm has been used to solve different kinds of scheduling problems successfully. However, due to its probabilistic model does not consider the variable interactions, ACGA may not perform well in some scheduling problems, particularly the sequence-dependent setup times are considered because a former job influences the processing time of next job. It is not sufficient that probabilistic model just captures the ordinal information from parental distribution. As a result, this paper proposes a bi-variate probabilistic model added into the ACGA. The new algorithm is named extended artificial chromosomes with genetic algorithm (eACGA) and it is used to solve single machine scheduling problem with sequence-dependent setup times in a common due-date environment. Some heuristics are also employed with eACGA. The results indicate that the average error ratio of eACGA is one-half of the ACGA. In addition, when eACGA works with other heuristics, the hybrid algorithm achieves the best solution quality when it is compared with other algorithms in literature. Thus, the proposed algorithms are effective for solving this scheduling problem with setup consideration.

Paper Topics:
1f. Estimation of distribution algorithms
1i. Genetic algorithms
1k. Heuristics, metaheuristics and hyper-heuristics

Student Paper: No

For the latest news and announcements, please visit the conference's home page:

http://www.cec2011.org

You will be informed of the status of your submission via email by March 15, 2011. If you have any questions, please send an e-mail to Alice Smith <cec2011@ieee-cis.org>.

Thank you for your submission.

Sincerely,
Alice Smith <cec2011@ieee-cis.org>

Home

Processed: 2011-01-28 02:42:52 EST.

Bi-Variate Artificial Chromosomes with Genetic Algorithm for Single Machine Scheduling Problems with Sequence-Dependent Setup Times

Shih-Hsin Chen
Department of Electronic Commerce Management
Nanhua University
Chiayi 62248, Taiwan (R.O.C.)
Email: shihhsin@mail.nhu.edu.tw

Min-Chih Chen
Institute of Manufacturing Engineering
National Cheng Kung University
Tainan, Taiwan (R.O.C.)
Email: cthunter@mail.wfc.edu.tw

***Abstract*—Artificial chromosomes with genetic algorithm (ACGA) is one of the latest Estimation of Distribution Algorithms (EDAs). This algorithm has been used to solve different kinds of scheduling problems successfully. However, due to its probabilistic model does not consider the variable interactions, ACGA may not perform well in some scheduling problems, particularly the sequence-dependent setup times are considered because a former job influences the processing time of next job. It is not sufficient that probabilistic model just captures the ordinal information from parental distribution. As a result, this paper proposes a bi-variate probabilistic model added into the ACGA. The new algorithm is named extended artificial chromosomes with genetic algorithm (eACGA) and it is used to solve single machine scheduling problem with sequence-dependent setup times in a common due-date environment. Some heuristics are also employed with eACGA. The results indicate that the average error ratio of eACGA is one-half of the ACGA. In addition, when eACGA works with other heuristics, the hybrid algorithm achieves the best solution quality when it is compared with other algorithms in literature. Thus, the proposed algorithms are effective for solving this scheduling problem with setup consideration.**

Keywords: ACGA, Bi-Variate EDAs, Scheduling Problems, Sequence-Dependent Setup Times, Common Due-Date

I. Introduction

In our previous studies [6], [7], [8], [9], ACGA has been applied for solving several scheduling problems. The main characteristic of ACGA is to alternate the EDAs and genetic operators in each generation. Because other EDAs [3], [14], [19], [29], [27] do not use genetic operators, this feature distinguishes ACGA with others. This approach is beneficial for EDAs to have a diversified population [12] where GA-EDA [23] used similar framework.

ACGA utilizes an univariate probabilistic model which extracts parental distribution from previous search when EDAs operator is responsible for generating offsprings. After that, the univariate probabilistic model is used to sample new solutions called the artificial chromosomes. From these remarkable prior results, ACGA is able to provide a satisfactory results.

The univariate probabilistic model of ACGA assumed that there are no dependencies between/among variables. However, some researches pointed out when variable interactions exist, EDAs may employ the bi-variate or even the multivariate probabilistic models [5], [4], [14], [22]. Most important of all, this research is going to study the single machine scheduling problems with sequence-dependent setup time in a common due date environment [28]. Because a prior job influences the processing time of the next job, there exists interactions between the jobs. Once ACGA is used to solve this scheduling, we may not have a satisfactory result. A new bi-variate probabilistic model together with the univariate probabilistic model is adopted into the proposed algorithm. The new algorithm is named extended artificial chromosomes with genetic algorithm (eACGA). eACGA could have better chance to capture more accurate parental distribution from the two probabilistic models so that it could produce better offsprings.

The organization of the paper is shown as follows: The Section II indicates the importance of the studied scheduling problems and its problem definition. We start to introduce the details of eACGA in Section III. Section IV shows the extensive comparisons with other algorithm in literature when they are used to solve the studied scheduling problems. Finally, we draw the conclusions in Section V.

II. The Background of the Studied Scheduling Problems

This research discusses the single machine scheduling problems with sequence-dependent setup time in a common due date environment. In addition, the objective is to minimize the total earliness and tardiness cost. To point out the importance of this studied scheduling problems, Section II-A presents the literature survey and the problem statement. In Section II-B, we further explain the model of the scheduling problem.

A. Review and Problem Statements

Single-machine scheduling problems are one of the well-studied problems by many researchers. The application of single machine scheduling with setups can be found in minimizing the cycle time for pick and place (PAP) operations in Printed Circuit Board manufacturing company [17]; in a steel

wire factory in China [15] and a sequencing problem in the weaving industry [1]. The results developed in the literature not only provide the insights into the single machine problem but also for more complicated environment such as flow shop or job shop.

The problem considered in this paper is to schedule a set of n jobs $\{j_1, j_2, \cdots, j_n\}$ on a single machine that is capable of processing only one job at a time without preemption. As explained in [2], [28], all jobs are available at time zero, and a job j requires a processing time P_j. Job j belongs to a group $g_j \in \{1, \ldots, q\}$ (with $q \leq n$). Setup or changeover times, which are given as two $q \times q$ matrices, are associated to these groups. This means that in a schedule where j_j is processed immediately after j_i where $i, j \in \{1, 2, \cdots, n\}$, there must be a setup time of at least S_{ij} time units between the completion time of j_i, denoted by C_i, and the start time of j_j , which is $C_j - P_j$. During this setup period, no other task can be performed by the machine and we assume that the cost of the setup operation is $c\,(g_i; g_j) \geq 0$ and let it be equal to machine setup time S_{ij} which is included as sequence dependent.

Apart from the sequence-dependent setup times, the objective is to complete all the jobs as close as possible to a large, common due date d. To accomplish this objective, the summation of earliness and tardiness is minimized. The earliness of job j is denoted as $E_j = \max(0, d - C_j)$ and its tardiness as $T_j = \max(C_j - d, 0)$, where C_j is the completion time of job j. Earliness and tardiness penalties for job j are weighted equally. The objective function is given by:

$$\min Z = \sum_{j=1}^{n} (E_j + T_j) = \sum_{j=1}^{n} |d - C_j| \tag{1}$$

The inclusion of both earliness and tardiness costs in the objective function is compatible with the philosophy of just-in-time production, which emphasizes producing goods only when they are needed. The early cost may represent the cost of completing a product early, the deterioration cost for a perishable goods or a holding (stock) cost for finished goods. The tardy cost can represent rush shipping costs, lost sales and loss of goodwill. Some specific examples of production settings with these characteristics are provided by [21]. The set of jobs is assumed to be ready for processing at the beginning which is a characteristic of the deterministic problem. The set of jobs is assumed to be ready for processing at the beginning which is a characteristic of the deterministic problem. As a generalization of weighted tardiness scheduling, the problem is strongly NP-hard in [18]. Consequently, the early/tardy problem is also a strong NP-hard problem. It is the reason why this work attempts to use eACGA to conquer this NP-hard problem in a reasonable time.

B. Formulations of the Scheduling Model

The common due date model corresponds; for instance, to an assembly system in which the components of the product should be ready at the same time, or to a shop where several jobs constitute a single customer's order [13]. It is shown in [16] that an optimal sequence in which the b-th job is completed at the due-date. The value of b is given by:

$$b = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (n+1)/2 & \text{if } n \text{ is odd} \end{cases}, \tag{2}$$

The common due-date (k^*) is the sum of processing times of jobs in the first b positions in the sequence; i.e.,

$$k^* = C_b \tag{3}$$

As soon as the common due date is assigned, see Fig. 1, jobs can be classified into two groups that are early and tardy which are at position from 1 to b and b+1 to n respectively. The following notations are employed in the latter formulations.

[j]: job in position j

A: the job set of tardy jobs

B: the job set of early jobs

$S_{[j][j+1]}$: Setup time of a job in position [j+1] follows a job in position [j]

$AP_{[j][j+1]}$: Adjusted processing time for the job in position j followed by the job in position [j+1]

b: the median position

$AP_{[j][j+1]}$ is actually the processing time of job $j+1$ with setup time. Thus, the original form of $AP_{[j][j+1]}$ is $S_{[j][j+1]} + P_{j+1}$.

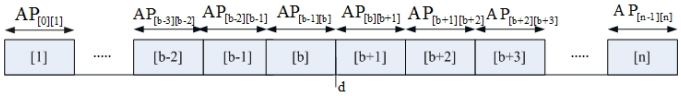

Fig. 1. The total earliness and total tardiness for a pre-assigned due-date d

Our objective is to minimize the total earliness/tardiness cost. The formulation is given below.

$$Minimize\ f(x) = \sum_{j=1}^{n} (E_j + T_j) = TT + TE \tag{4}$$

where

TT: Total tardiness for a job sequence

TE: Total earliness for a job sequence

TT and TE can be transformed into the following equations based on the pre-defined adjusted processing time.

$$TT = \sum_{j=b}^{n-1} (n - j) AP_{[j][j+1]} \tag{5}$$

$$TE = \sum_{j=1}^{b} (j - 1) AP_{[j-1][j]} \tag{6}$$

III. METHODOLOGY

In order to capture the information of variable interactions, we take the bi-variate probability model into considerate as well. That is, eACGA extends from the ACGA, which not only collect the order information of the job in the sequence but also the variable interaction of the jobs. As a result, eACGA could extract the parental information by using the univariate probability model and bi-variate probability model.

The following Section III-A explains the proposed algorithm in detail. Because the univariate and bi-variate are the core of EDAs, they are further explained in Section III-B.

A. The procedures of eACGA

The major procedures of eACGA include population initialization, selection of better chromosomes, and then to decide whether EDAs or genetic operators is ran. After that, a replacement strategy is used and then to test the stopping criterion. The framework of eACGA is depicted in Fig. 2.

Population: A set of solutions
Generations: The maximum number of generations
$OP(t)$: Univariate Probabilistic model
$DP(t)$: Bi-variate Probabilistic model
t : Generation index
k : The execution of the EDA interval
$startingGen$: The generation at which the EDA will start to run

1: Initialize *Population*
2: $t \leftarrow 0$
3: Initialize $OP(t)$ and $DP(t)$
4: **while** $t < Generations$ **do**
5: Selection / Elitism(*Population*)
6: **if** $g \ \% \ k == 0 \ and \ t > startingGen$ **then**
7: $OP(t+1) \leftarrow$BuildUnivariateProbabilityModel(*Selected Chromosomes*)
8: $DP(t+1) \leftarrow$BuildBiVariateProbabilityModel(*Selected Chromosomes*)
9: Learning
10: Sampling new solutions into Population
11: **else**
12: Crossover()
13: Mutation()
14: **end if**
15: EvaluateFitness (*Population*)
16: Replacement()
17: $t \leftarrow t + 1$
18: **end while**

Fig. 2. Algorithm1: MainProcedure of eACGA()

We describe the methods in the following steps.
Step 1: Initialization

We initialize the population which consists a number of chromosomes in Line 1. A chromosome represents a processing sequence for the scheduling problem. Each chromosome is generated randomly. In order to improve results, some researches use heuristic to generate better initial solutions in this step. For example, dominance properties (DP) in [10] for this studied scheduling problem is considered to generate a single chromosome in eACGA. Shortest Adjusted Processing Time first(SAPT) is the other heuristic could initialize good initial solutions. The two heuristics are employed to work with eACGA and they are named eACGA$_{DP}$ and eACGA$_{SAPT}$, respectively.

On the other hand, we also have to initialize the probabilistic models used in eACGA in Line 3. No matter the univariate or the bi-variate probabilistic models, each $P_{ij}(t)$ is initialized to be $\frac{1}{n}$, where n is the number of jobs and $P_{ij}(t)$ is the probability of job i in position j in a promising solution.
Step 2: Selection and Elitism Strategy (Line 5)

Evolutionary algorithms attempt to select fitter solutions corresponding to their objective values. The selection operator chooses better chromosomes to be survived. For the purpose of simplicity, the binary tournament operator is employed, which selects the better chromosomes with lower objective values in this minimization problem. In order to preserve elites in the population, a proportion of better chromosomes are stored in an external archive and to be copied into the mating pool during the selection stage.
Step 3: Decision (Line 6)

There are two parameters control whether the EDAs or GAs is ran, which are $startingGen$ and $interval$. The first parameter $startingGen$ is to determine the starting time of generating artificial chromosomes. The main reason is that the probabilistic model should be only applied to generate better chromosomes when the searching process reaches a more stable state.

The other important parameter $interval$ sets the period of artificial chromosomes generated. $g \ \% \ k$ represents that g mod k. When the resultant value is 0, it means the current generation reaches the k interval. As a result, the algorithm alternates EDAs and genetic operators in the whole evolutionary progress. When we like to execute the EDAs, constructing the probabilistic models, the learning of parental distribution, and then samples new offsprings from probabilistic models. They are described in Step 4.1.1 to Step 4.1.3. On the other hand, genetic operators contain the crossover and mutation operator which are Step 4.2.1 and Step 4.2.2, respectively.
Step 4: Variations
Step 4.1: eACGA Segment
Step 4.1.1: Modeling (Line 7 and Line 8)

The univariate probabilistic model and the bi-variate probabilistic models are built while we run the EDAs. The former one represents all jobs at different positions referring to the frequency count of a job at the positions. The bi-variate probabilistic model is similar to a container of interaction counters which are blocking out the similar jobs in the sequences. Fuller step by step detail will be presented in Section. III-B.
Step 4.1.2: Learning (Line 9)

As in PBIL [5], we update the two probabilistic models in an incremental learning way. In addition, the learning rate determines the importance of the current and historical

probability information. The probability learning models of the univariate and bi-variate probabilistic models are shown in Eq. 14 and Eq. 15, respectively.

Step 4.1.3: Sampling (Line 10)

Now that the two probabilistic models have been established, the actual procedure implemented in the optimization algorithm needs to be specified. The goal is to devise a strategy to form the offspring populations which reflect the two probabilistic models. For each position in the sequence of a new individual, first we select a job randomly as the first position, then according to the multiplication of two probabilistic models, proportional selection fill out the other sequence of a new individual.

Step 4.2.1: Crossover (Line 12)

This study applies the two-point central crossover operator [20] to mate two chromosomes which are randomly selected. Crossover rate (P_c) decides whether the chromosome is mated with others.

Step 4.2.2: Mutation (Line 13)

A chromosome is decided to be mutated if a random probability value is lower than the mutation rate (P_m). The swap mutation operator is used in our experiments. When we decide to do the mutation, the genes of the two random positions are swapped.

Step 5: Replacement (Line 16)

In order to improve the population quality and keep the population diversity, an individual replaces with the worst one in the parent population when the offspring is better. Furthermore, the offspring must be different from any one of the parent population.

B. Establishing probabilistic models

We explain how to establish univariate probabilistic model first and then the bi-variate probabilistic model. Suppose a population has M chromosomes $X^1, X^2, ..., X^M$ at the current generation t, which is denoted as Population(t). Then, X_{ij}^k is a binary variable in chromosome k, which is shown in Eq. 7.

$$X_{ij}^k = \begin{cases} 1 & \text{if job } i \text{ is assigned to position } j \\ 0 & \text{Otherwise} \end{cases} \tag{7}$$

where $i = 1, \ldots, n; j = 1, \ldots, n$.

If job i exists at position j, the number of occurrence of X_{ij} is incremented by 1. There are m chromosomes and the order information of job i on position j (f_{ij}) will be calculated as follows:

$$f_{ij}(t) = \sum_{k=1}^{m} X_{ij}^k, i = 1, \ldots, n,\ j = 1, \ldots, n \tag{8}$$

The univariate probability model presents the occurrence possibility of these jobs in the sequence at different positions. In order to combine the univariate probabilistic model with bi-variate probabilistic model, the univariate probabilistic model Op_{ij} will be the total number of times of appearance of job i before or in the position j at current generation t. Thus, the ordinal probability is to accumulate the distribution Eq. 9 in position j, which is as follows:

$$Op_{ij}(t) = \sum_{l=1}^{j} (\sum_{k=1}^{m} X_{ij}^k)^l, i = 1, \ldots, n,\ j = 1, \ldots, n \tag{9}$$

The ordinal probabilistic matrix of all jobs at different positions are written as the Eq. 10.

$$Op(t) = \begin{pmatrix} Op_{11}(t) & \cdots & Op_{1n}(t) \\ \vdots & \ddots & \vdots \\ Op_{n1}(t) & \ldots & Op_{nn}(t) \end{pmatrix} \tag{10}$$

Moreover, we explain how to establish a dependency probabilistic model. A dependence (v_{ij}) means a job i neighbors with another job j. Suppose a population has M strings $v^1, v^2, ..., v^M$ at current generation t. Then, v_{ij}^k is a binary variable in chromosome k, which is shown in Eq. 11.

$$v_{ij}^k = \begin{cases} 1 & \text{if job } j \text{ connect to job } i \\ 0 & \text{Otherwise} \end{cases} \tag{11}$$

where $i = 1, \ldots, n; j = 1, \ldots, n$.

The interaction information is collected from N best chromosomes where only paired interactions between the jobs are taken into account. Let $Dp_{ij}(t)$ (dependency probability) be the number of times of appearance of job i after the job j at current generation t. The $Dp_{ij}(t)$ is updated as follows:

$$Dp_{ij}(t) = \sum_{k=1}^{m} v_{ji}^k, i = 1, \ldots, n,\ j = 1, \ldots, n \tag{12}$$

For the dependency probabilistic matrix of paired interaction of all jobs, they are written as the Eq. 13.

$$Dp(t) = \begin{pmatrix} 0 & Dp_{12}(t) & \cdots & Dp_{1n}(t) \\ Dp_{21}(t) & 0 & \cdots & Dp_{2n}(t) \\ \vdots & \vdots & \ddots & \vdots \\ Dp_{n1}(t) & Dp_{n2}(t) & \ldots & 0 \end{pmatrix} \tag{13}$$

Furthermore, the above two probabilities with learning can continue to modify the search space to improve the performance, the equation as the Eq. 14 and Eq. 15. In this research, two parameters ordinal learning rate ($\lambda_{OP} \in (0, 1)$) and dependent learning rate ($\lambda_{DP} \in (0, 1)$) are decided by Design-of-Experiment (DOE), we discuss them in next session.

$$Op_{ij}(t) = Op_{ij}(t) \times (1.0 - \lambda_{Op}) + Op_{ij}(t-1) \times \lambda_{Op} \tag{14}$$

$$Dp_{ij}(t) = Dp_{ij}(t) \times (1.0 - \lambda_{Dp}) + Dp_{ij}(t-1) \times \lambda_{Dp} \tag{15}$$

As soon as the ordinal Op and dependent probability Dp are built, jobs are assigned onto each positions. As far as the diversification concerns, there are three ideas for creating

diversified artificial chromosomes. The first is the assignment sequence for each position assigned in random sequence. The second is proportional selection which is used to mitigate the probability of job i assigned to a position j. The third is zero value transformation which is used to transform 0 into $1/n$ in dependent probability when jobs do not have any interactions. The assignment procedure is determined as follows:

S: A set of shuffled sequence which determines the sequence of each position is assigned a job.
Ω: The set of un-arranged jobs.
J: The set of arranged jobs. J is empty in the beginning.
θ: A random probability is drawn from $U(0, 1)$.
i: A selected job by proportional selection
k: The element index of the set S

1: $S \leftarrow$ shuffled the job number $[1 \dots n]$
2: $J \leftarrow \Phi$
3: **while** $k \neq \Phi$ **do**
4: $\theta \leftarrow U(0, 1)$
5: Select a job i satisfies $\theta \leq OP_{ik} \times DP_{iJ(k-1)} / \sum OP(i,k) \times DP(i, J(k-1))$, where $i \in \Omega$
6: $J(k) \leftarrow i$
7: $\Omega \leftarrow \Omega \backslash i$
8: $S \leftarrow S \backslash k$
9: **end while**

IV. EXPERIMENT RESULTS

The testing instances are designed by [25] and the job size of each instance includes 10, 15, 20 and 25. The property of the processing time range contains low, median and high, which are based on Uniform(10, 60), Uniform(10, 110) and Uniform(10, 160), respectively. Each combination has 15 similar instances, the total number of instance is 180 (4*3*15) and each instance is replicated 30 times for our proposed algorithm and the compared Algorithms. In order to configure the parameter settings, we utilized the DOE to select the best setting of eACGA parameters. Table. I shows the parameters setting of the eACGA experiment. We code the algorithm in Java and to be ran on Windows 2003 server (Intel Xeon 3.2 GHZ).

TABLE I
eACGA PARAMETERS SETTING

Factor	Default
Crossover Rate	0.5
Mutation Rate	0.3
Starting generation	0
Interval	2
Ordinal probability learning rate	0.1
Dependent probability learning rate	0.5
Population Size	100
Generations	1000

In order to evaluate the performance of eACGA, they are compared with some algorithms in literature. We divide these algorithms into two groups which are stand-alone algorithms and hybrid algorithms. To test the efficiency of our proposed eACGA against the stand-alone algorithms, we compare the SGA [26], ACGA [11] and SAPT [24]. We also select some hybrid algorithms, such as SGA$_{DP}$ [10], ACGA$_{DP}$ [11] and SA$_{SAPT}$ [24]. In addition, when eACGA works with two heuristic algorithms DPs [10] and SAPT [24], they are called eACGA$_{DP}$ and eACGA$_{SAPT}$, respectively. Because DPs and SAPT generate good initial solutions, eACGA takes these solutions as the initial population. The brief information of these algorithms is discussed as follows:

- SGA [26]: A standard genetic algorithm with elitism strategy. The genetic operators include binary tournament selection, two-point central operator and swap mutation operator. During the selection stage, 10% of elites in the population are reserved to the next generation. We use the data from [10] for comparison.
- ACGA [11]: We mentioned that ACGA does not consider the dependencies between/among variables. When ACGA is used to solve the benchmark problems in this study, we could distinguish the performance difference when we employ the bi-variate probabilistic model together with the univariate probabilistic model.
- SGA$_{DP}$ [10]: Dominance properties (DPs) were developed by swapping the neighborhood jobs. The DPs are very efficient when combined with the GA. We take the results from [10] for comparison.
- ACGA$_{DP}$ [10]: According to the research of ACGA [11] and DPs [10], we combine both algorithms to solve the single machine scheduling problems with setup costs. DPs are utilized to generate a set of good initial solutions. ACGA takes the advantage of these good initial solutions and then continue the evolutionary progress. We expect ACGA$_{DP}$ could perform better than ACGA.
- SAPT [24]: Shortest Adjusted Processing Time first(SAPT) is based on a concept: Jobs with shorter adjusted processing times to be scheduled, which shall closer to the median position in an optimal job sequence. They also combined SAPT with simulated annealing (SA) algorithm, called SAPT-SA. Since the presented data of SAPT-SA [24] is not completed and termination condition is different from our experiments, we only used the SAPT data from [24] for comparison.

To compare the performance of these algorithms clearly, we employs the average relative error ratio. In literature, the average relative error ratio is often used to evaluate the performance of algorithms, whereby the error ratio (ER$_i$) of a solution (X_i) generated by an algorithm is calculated as follows:

$$ER_i = \frac{Obj_{avg}(X_i) - Opt_i}{Opt_i} * 100, \tag{16}$$

Where Opt_i is the objective value of the best known or optimal solution which is available by [28] who applied Branch-and-Bound algorithm to derive the solution. The Obj$_{avg}$(X$_i$) is the X_i average objective value. Table II showed the statistics

of the average ER values of all the algorithms on the test instances.

It can be seen from algorithms without hybridization in Table II, the eACGA outperforms SGA and ACGA. Particularly, the average error ration of eACGA is one-half of the ACGA. It is clearly that the effort of bi-variate probability model is quite beneficial for the sequence-dependent scheduling problems. Then, there is no difference between eACGA and SAPT because the difference is within the range of random error.

TABLE II
AVERAGE ERROR RATIO EVALUATION OF ALGORITHMS

Type	Size	Without hybridization				With hybridization			
		SGA	ACGA	SAPT	eACGA	SGA$_{DP}$	ACGA$_{DP}$	eACGA$_{DP}$	eACGA$_{SAPT}$
low	10	0.86	0.63	2.55	0.27	0.31	0.28	0.18	0.11
	15	4.21	4.05	3.59	1.85	3.13	2.93	1.40	0.77
	20	9.58	8.83	3.13	3.97	6.85	6.98	3.66	1.80
	25	13.18	11.81	3.21	6.10	9.44	9.55	5.15	2.06
med	10	1.89	1.29	2.80	0.18	0.93	0.94	0.26	0.36
	15	7.86	6.84	5.33	3.11	4.74	4.69	2.30	0.89
	20	13.93	12.53	3.93	5.74	10.06	9.86	4.90	2.25
	25	20.80	18.83	5.96	9.55	14.83	15.32	8.56	4.26
high	10	1.37	1.16	2.80	0.04	0.54	0.62	0.00	0.13
	15	9.70	9.35	8.22	3.42	6.38	5.66	2.44	0.69
	20	21.04	18.18	5.95	7.78	13.94	13.55	6.62	3.70
	25	27.88	24.74	6.83	13.09	20.22	19.78	10.88	4.84
	Avg	11.03	9.85	4.53	4.59	7.61	7.51	3.86	1.82

TABLE III
AVERAGE CPU TIME EVALUATION OF ALGORITHMS

Type	Size	Without hybridization				With hybridization			
		SGA	ACGA	SAPT	eACGA	SGA$_{DP}$	ACGA$_{DP}$	eACGA$_{DP}$	eACGA$_{SAPT}$
low	10	0.26	0.39	0.01	0.70	0.39	0.57	0.98	0.77
	15	0.31	0.47	0.03	1.18	0.46	0.71	1.42	1.25
	20	0.37	0.56	0.07	1.85	0.54	0.90	2.06	1.96
	25	0.43	0.65	0.18	2.60	0.63	1.09	2.79	2.79
med	10	0.26	0.39	0.02	0.72	0.38	0.57	0.98	0.77
	15	0.31	0.47	0.04	1.18	0.46	0.71	1.42	1.25
	20	0.37	0.56	0.06	1.86	0.55	0.89	2.06	1.96
	25	0.43	0.66	0.12	2.60	0.63	1.09	2.79	2.76
high	10	0.26	0.39	0.07	0.73	0.38	0.57	0.98	0.77
	15	0.31	0.46	0.11	1.18	0.46	0.72	1.42	1.25
	20	0.37	0.56	0.13	1.86	0.54	0.89	2.06	1.95
	25	0.43	0.65	0.18	2.60	0.63	1.09	2.79	2.75
	AVG	0.34	0.52	0.09	1.59	0.50	0.82	1.81	1.69

Apart from the stand-alone algorithms, two heuristics are further embedded in meta-heuristic algorithm to improve the efficiency and effectiveness of the global searching procedure. The results of the hybrid framework are also shown in Table II. both eACGA$_{DP}$ and eACGA$_{SAPT}$ were better than eACGA and SAPT in terms of average relative error ratio. For example, eACGA$_{SAPT}$ was 2.5 times lower than SAPT and eACGA$_{SAPT}$ algorithm was 4.2 times lower than SGA$_{DP}$. The empirical results show the domain-specific heuristics are useful for the studied scheduling problem and eACGA$_{SAPT}$ achieves the best solution quality when it is compared with others in literature.

We further compared the CPU time of these algorithms. The results are given in Table III. The SGA appears more efficient than eACGA and ACGA in terms of CPU time because it requires a linear time to create new solution, whereas artificial chromosome requires $O(n^2)$ time. ACGA is faster than eACGA because we execute the EDAs more frequently in eACGA than the one in ACGA. In addition, due to eACGA needs to construct a bi-variate probabilistic model, it takes longer time to do so. In comparison of eACGA$_{DP}$ and eACGA$_{SAPT}$, eACGA$_{SAPT}$ was superior to eACGA$_{DP}$ in terms of CPU time. Although DPs and SAPT both used general pair-wise interchange (GPI) for the sequence, SAPT only uses the neighborhood which is generated by every possible pairwise interchange, i.e. the neighborhood distance is 1. However, the neighborhood distance of DPs is 2 and 3. It is the reason why DPs may need longer computational time.

V. Discussion and Conclusions

This work proposed the eACGA which enables the ACGA to deal with interactions between the variables. Both univariate and bi-variate probabilistic models are used in eACGA so that the eACGA could extract more accurate problem structure from parental distribution. eACGA and other algorithms are used to deal with the single machine scheduling problem with sequence-dependent setup times in a common due date environment. Due to the problem properties that former job impacts the processing time of the next job, there exists strong variable interactions in the problem. As a result, when eACGA is compared with ACGA, eACGA is superior to ACGA because the average error ratio of eACGA is just one-half of the ACGA. It is apparently that the bi-variate probabilistic model together with univariate probabilistic model could gain more information from previous search for the condition of sequence-setup times.

Except the bi-variate probabilistic model is useful, some domain-specific heuristics could further improve the solution quality of eACGA. When eACGA works with DP or SAPT, the two hybrid algorithms are further improved, particularly the eACGA$_{SAPT}$ is the state-of-art when it is compared with others in literature. In the future study, eACGA could be used to solve larger-size single machine sequence-dependent scheduling problems than existing 25-jobs benchmarks. The larger-size instances could distinguish the performance of the different algorithms. So we could understand the advanatage or disadvantage of the proposed algorithm.

References

[1] M. Al-Haboubi and S. Selim, "A sequencing problem in the weaving industry," *European Journal of Operational Research*, vol. 66, no. 1, pp. 65–71, 1993.

[2] K. Baker and G. Scudder, "Sequencing with earliness and tardiness penalties: a review," *Operations Research*, vol. 38, no. 1, pp. 22–36, 1990.

[3] S. Baluja, "An Empirical Comparison of Seven Iterative and Evolutionary Function Optimization Heuristics," *Technical report, Carnegie Mellon University*, 1995.

[4] S. Baluja and S. Davies, "Using Optimal Dependency-Trees for Combinational Optimization," *Proceedings of the Fourteenth International Conference on Machine Learning table of contents*, pp. 30–38, 1997.

[5] ——, "Fast probabilistic modeling for combinatorial optimization," *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence table of contents*, pp. 469–476, 1998.

[6] P. C. Chang, S. H. Chen, and C. Y. Fan, "Genetic algorithm integrated with artificial chromosomes for multi-objective flowshop scheduling problems," *Applied Mathematics and Computation*, vol. 205, no. 2, pp. 550–561, 2008.

[7] ——, "Injecting Artificial Chromosome for flowshop scheduling problems," *Applied Mathematics and Computation*, vol. 205, no. 2, pp. 550–561, 2008.

[8] ——, "Mining gene structures to inject artificial chromosomes for genetic algorithm in single machine scheduling problems," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 767–777, 2008.

[9] P. C. Chang, J. C. Hsieh, S. H. Chen, J. L. Lin, and W. H. Huang, "Artificial chromosomes embedded in genetic algorithm for a chip resistor scheduling problem in minimizing the makespan," *Expert Systems With Applications*, vol. 36, no. 3, Part 2, pp. 7135–7141, 2009.

[10] P. C. Chang, T. Lie, J. Y. Liu, and S. Chen, "A genetic algorithm enhanced by dominance properties for single machine scheduling problems with setup cost," *International Journal of Innovative Computing, Information and Control*, vol. 7, no. 5, pp. 1–21, 2011.

[11] P. Chang, S. H. Chen, and C. Y. Fan, "Generating Artificial Chromosomes by Mining Gene Structures with Diversity Preservation for Scheduling Problems," *Annals of Operations Research*, vol. 180, no. 1, pp. 197–211, 2010.

[12] S. Chen, M. Chen, P. Chang, Q. Zhang, and Y. Chen, "Guidelines for developing effective Estimation of Distribution Algorithms in solving single machine scheduling problems," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6441–6451, 2010.

[13] V. Gordon, J. Proth, and C. Chu, "A survey of the state-of-the-art of common due date assignment and scheduling research," *European Journal of Operational Research*, vol. 139, no. 1, pp. 1–25, 2002.

[14] G. Harik, F. Lobo, and D. Goldberg, "The compact genetic algorithm," *Evolutionary Computation, IEEE Transactions on*, vol. 3, no. 4, pp. 287–297, 1999.

[15] F. Jin, J. Gupta, S. Song, and C. Wu, "Single machine scheduling with sequence-dependent family setups to minimize maximum lateness," *Journal of the Operational Research Society*, vol. 61, no. 7, pp. 1181–1189, 2009.

[16] J. Kanet, "Minimizing the average deviation of job completion times about a common due date," *Naval Research Logistics Quarterly*, vol. 28, no. 4, pp. 643–651, 1981.

[17] O. Kulak, I. Yilmaz, and H. G
"unther, "PCB assembly scheduling for collect-and-place machines using genetic algorithms," *International Journal of Production Research*, vol. 45, no. 17, pp. 3949–3969, 2007.

[18] J. Lenstra, A. Kan, and P. Brucker, "Complexity of machine scheduling problems," *Stud. integer Program., Proc. Workshop Bonn*, 1975.

[19] H. Muhlenbein and G. Paaß, "From recombination of genes to the estimation of distributions I. Binary parameters," *Lecture Notes in Computer Science*, vol. 1141, pp. 178–187, 1996.

[20] T. Muruta and H. Ishibuchi, "Performance evaluation of genetic algorithms for flowshop scheduling problems," *Proceedings of First IEEE International Conference on Evolutionary Computation*, vol. 2, pp. 812–817, 1994.

[21] P. S. Ow and T. E. Morton, "The Single Machine Early/Tardy Problem," *Management Science*, vol. 35, no. 2, pp. 177–191, 1989.
[22] M. Pelikan, D. E. Goldberg, and E. Cantu-Paz, "BOA: The Bayesian optimization algorithm," *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, vol. 1, pp. 525–532, 1999.
[23] J. Peña, V. Robles, P. Larrañaga, V. Herves, F. Rosales, and M. Perez, "GA-EDA: Hybrid Evolutionary Algorithm Using Genetic and Estimation of Distribution Algorithms," *Innovations In Applied Artificial Intelligence: 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2004, Ottawa, Canada, May 17-20, 2004: Proceedings*, 2004.
[24] G. Rabadi, G. Anagnostopoulos, and M. Mollaghasemi, "A heuristic algorithm for the just-in-time single machine scheduling problem with setups: a comparison with simulated annealing," *The International Journal of Advanced Manufacturing Technology*, vol. 32, no. 3, pp. 326–335, 2007.
[25] G. Rabadi, M. Mollaghasemi, and G. Anagnostopoulos, "A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence-dependent setup time," *Computers & Operations Research*, vol. 31, no. 10, pp. 1727–1751, 2004.
[26] C. R. Reeves, "A genetic algorithm for flowshop sequencing," *Computers and Operations Research*, vol. 22, no. 1, pp. 5–13, 1995.
[27] R. Santana, P. Larrañaga, and J. Lozano, "Combining variable neighborhood search and estimation of distribution algorithms in the protein side chain placement problem," *Journal of Heuristics*, vol. 14, pp. 519–547, 2008.
[28] F. Sourd, "Earliness–tardiness scheduling with setup considerations," *Computers and operations research*, vol. 32, no. 7, pp. 1849–1865, 2005.
[29] Q. Zhang, J. Sun, and E. Tsang, "An Evolutionary Algorithm With Guided Mutation for the Maximum Clique Problem," *Evolutionary Computation, IEEE Transactions on*, vol. 9, no. 2, pp. 192–200, 2005.

國科會補助計畫衍生研發成果推廣資料表

日期:2011/10/12

國科會補助計畫	計畫名稱: 求解具有與過去順序相依準備時間以及學習效應之雙目標單機排程問題之研究(I)
	計畫主持人: 陳世興
	計畫編號: 99-2221-E-343-002- 學門領域: 生產系統規劃與管制

無研發成果推廣資料

99 年度專題研究計畫研究成果彙整表

計畫主持人：陳世興	計畫編號：99-2221-E-343-002-
計畫名稱：求解具有與過去順序相依準備時間以及學習效應之雙目標單機排程問題之研究(I)	

成果項目			量化			單位	備註（質化說明：如數個計畫共同成果、成果列為該期刊之封面故事...等）
			實際已達成數（被接受或已發表）	預期總達成數(含實際已達成數)	本計畫實際貢獻百分比		
國內	論文著作	期刊論文	0	0	100%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	0	0	100%		
		專書	0	0	100%		
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（本國籍）	碩士生	0	0	100%	人次	
		博士生	0	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		
國外	論文著作	期刊論文	1	1	100%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	0	0	100%		
		專書	0	0	100%	章/本	
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（外國籍）	碩士生	3	3	100%	人次	
		博士生	2	2	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		

其他成果 (無法以量化表達之成果如辦理學術活動、獲得獎項、重要國際合作、研究成果國際影響力及其他協助產業技術發展之具體效益事項等，請以文字敘述填列。)	無

	成果項目	量化	名稱或內容性質簡述
科教處計畫加填項目	測驗工具(含質性與量性)	0	
	課程/模組	0	
	電腦及網路系統或工具	0	
	教材	0	
	舉辦之活動/競賽	0	
	研討會/工作坊	0	
	電子報、網站	0	
	計畫成果推廣之參與（閱聽）人數	0	

國科會補助專題研究計畫成果報告自評表

請就研究內容與原計畫相符程度、達成預期目標情況、研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）、是否適合在學術期刊發表或申請專利、主要發現或其他有關價值等，作一綜合評估。

1. 請就研究內容與原計畫相符程度、達成預期目標情況作一綜合評估

■達成目標

□未達成目標（請說明，以 100 字為限）

□實驗失敗

□因故實驗中斷

□其他原因

說明：

2. 研究成果在學術期刊發表或申請專利等情形：

論文：■已發表 □未發表之文稿 □撰寫中 □無

專利：□已獲得 □申請中 ■無

技轉：□已技轉 □洽談中 ■無

其他：(以 100 字為限)

3. 請依學術成就、技術創新、社會影響等方面，評估研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）(以 500 字為限)

This research is the first research which consider the bi-objective single machine scheduling problem with the consideration of the learning effect and past-sequence-dependent effect. Because this kind of scheduling research have attracted many attentions, this paper is of interesting for the academic field.