Bicriteria Heuristic for Correlated Parallel Machine Scheduling Problem

[1]Yang-Kuei, Lin and [2]Tzu-Yueh, Yin

Abstract—We consider the problem of scheduling correlated parallel machines with release times to minimize number of tardy jobs and total weighted completion time. We consider different levels and combination of machine correlations and job correlations in the processing times. We first present a mix integer programming (MIP) model that can find optimal solutions for the studied problem. Next, we propose a bi-criteria heuristic that can find non-dominated solutions for the studied problem efficiently. Computational results show that the proposed heuristic is computationally efficient and provides solutions of reasonable quality.

Keywords—Bicriteria, correlation, heuristic, parallel machines, scheduling

I. INTRODUCTION

In reality, multiple but conflicting objectives are usually considered when a manager plans the production scheduling. Hence, managers need to consider multiple objectives and trying to find a set of good solutions that satisfies all the considerations. Moreover, we examine correlated parallel machine scheduling problems since they better reflect real world manufacturing environments [1]. In this research, we consider the problem of scheduling *n* jobs on *m* correlated parallel machines to minimize number of tardy jobs and total weighted completion time (TWC) with release dates. Each job *j* has a release date (r_j), a correlated processing time (p_{ij}) on machine *i*, a due date (d_j) and a weight (w_j). Job preemptions are not allowed. The number of tardy jobs ($\sum_{j=1}^{n} U_j$) is defined as U_j=1 if C_j>d_j; 0, otherwise. It is often an objective in practice as it is a measure that can be recorded very easily where C_j is the completion time of job *j*. The total weighted completion time ($\sum_{j=1}^{n} w_j C_j$) gives an indication of the total holding or inventory costs incurred by the schedule. Following the three-field notation of [1], we refer to this problem as $\mathrm{R} \mid r_j\,, correlation \mid \sum U_j\,, \sum w_j C_j$.

This research considered correlated parallel machine scheduling problems. The studied correlated parallel machine scheduling problem is based on [2]. Reference [2] defined nine cases that considered different levels and combinations of machine correlations and job correlations for parallel machine environments. They also proposed processing time generation schemes for the nine cases. Reference [1] applied mathematical models on the nine cases defined by [2] to examine whether some problems are more difficult to solve than others. Reference [1] first applied a mathematical model to solve correlated parallel machine scheduling problem to minimize makespan. Next, they used another mathematical model to solve correlated parallel machine scheduling problem to minimize total weighted tardiness with release time. Reference [1] concluded that as the machine and job correlations increase, the problem instances become more difficult for mathematical models to solve. This implies that branch-and-bound based algorithms might have more difficulty solving parallel machine scheduling problems with correlations than without correlations.

In the last few decades, more and more researchers have studied bicriteria parallel machine scheduling problems. Reference [3] presented a heuristic based on simulated annealing (SA) and a neighborhood search for scheduling identical parallel machines to minimize the average flow time and the number of tardy jobs. Reference [4] suggested a model for a regular parallel machine scheduling problem to minimize the sum of machine holding cost and job tardiness cost. They proposed a heuristic based on a tabu search algorithm to search optimal or near optimal solutions. Reference [5] proposed a heuristic based on simulated annealing (SA) to minimize the total flow-time and the total number of tardy jobs on identical parallel machines. Reference [6] considered unrelated parallel machine scheduling problem with non-identical due dates, ready times, sequence-dependent and machine-dependent setup times. They proposed a genetic algorithm (GA) to solve this bi-objective problem to minimize the number of tardy jobs and the total completion time. Reference [7] considered a sequence-dependent setup with release times on unrelated parallel machine scheduling problem. They proposed a heuristic based on tabu search algorithm (TSA) to minimize the total weighted completion times and the total weighted tardiness. The results showed that the proposed TSA can find near optimal solutions in a short time. Reference [8] studied unrelated parallel machine scheduling problem. They first proposed a heuristic (LP-ATC) to minimize the makespan and the total weighted tardiness for unrelated parallel machine scheduling problem. Next, they proposed another heuristic (ATC-bi) to minimize the total weighted completion time and total weighted tardiness for unrelated machines scheduling problem. Moreover, [8] proposed a GA to minimize makespan, total weighted completion time and total weighted tardiness for unrelated machines scheduling problem. Reference [9] proposed a heuristic and a tabu search algorithm to find non-dominated solutions for bicriteria unrelated parallel machine scheduling problems with release dates. They

[1]Yang-Kuei, Lin, Department of Industrial Engineering and Systems Management, Feng Chia University, Taiwan, ROC,

[2]Tzu-Yueh, Yin, Department of Industrial Engineering and Systems Management, Feng Chia University, Taiwan, ROC, E

https://doi.org/10.17758/EIRAI2.F0418116

compared the proposed algorithm with other existing algorithms. They concluded that the proposed tabu search algorithm outperformed other algorithms in terms of the number of non-dominated solutions and the quality of its solutions. Reference [10] developed a heuristic to solve a batch scheduling problem on unrelated parallel machines with the objective of minimizing a linear combination of total weighted tardiness and the total weighted completion time.

To the best of our knowledge, no research has yet been published that develops a heuristic to minimize the number of tardy jobs and total weighted completion time on correlated parallel machines with release times. The $R \mid r_j, correlation \mid \sum U_j, \sum w_j C_j$ problem is strongly NP-hard since the task of minimize number of tardy jobs on a single machine with release time ($1 \mid r_j \mid \sum U_j$) is already NP-hard in the strong sense [11]. Since the studied problem is strongly NP-hard, it is impractical to solve it by using an exact algorithm. We present a heuristic to find non-dominated solutions for the $R \mid r_j, correlation \mid \sum U_j, \sum w_j C_j$ problem.

Correlated parallel machines environment

According to [1], when both machine and job correlation factors got involved in generating processing times, the problems have become difficult for the mathematical models to solve. This research mainly focusses on Case 9 defined by [2]. Case 9 is one of the most difficult cases for mathematical models to solve [1]. This implies that branch-and-bound based algorithms are sensitive to the distribution of the processing times. Moreover, [1] mentioned that the newly defined correlated parallel machine scheduling problems by [2] better fit real world manufacturing conditions and can thus provide results that better reflect real world scheduling problems.

Reference [2] used parameters Γ and Δ to control the relatedness of the generated processing times for the machine-correlated and job-correlated environments, respectively. Γ is inversely proportional to the relative dispersion of processing times between machines, and Δ is inversely proportional to the relative dispersion of processing times between jobs. The smaller the Γ or Δ value, the larger the variation it represents. The Γand Δ values are passed into the generation scheme and the scheme generates a problem instance. Case 9: $\Gamma \in (0 , 0.8) < \Delta \in (0.2 , 1)$; job correlation is greater than machine correlation. Figure 1 shows examples with 3 jobs on 3 machines for Case 9. The variation of processing times for each job across the different machines is smaller than for the three jobs on the same machine. We use $\Gamma=0.2$; $\Delta=0.4$ in Fig 1 as an example to explain Case 9. Assuming jobs are processed manually by human workers, it is virtually impossible that each worker requires exactly the same processing time on each job. Similarly, it is also unlikely that each machine will operate at a constant rate. In fact, small variations are likely to occur in processing times among individual machines. If the standard man-hour for processing job 3 is 60 minutes, operator A (M1) might spend 64 minutes processing job 3 as opposed to 65 minutes for operator B (M2) and 60 minutes for operator C (M3). Hence, it would be reasonable to assume that a job will result in a small dispersion of processing times with different machines and/or operators.

II. Methodology

A. Integer Programming (IP) Model

The studied problem can be written as an IP model based on the time-indexed formulation. The IP model that can generate the entire set of non-dominated solutions for the studied problem. The set of solutions provided by the IP serves as a reference set.

In the time-indexed formulation, the planning horizon is discretized into the periods 1, 2,…, T, where period t starts at time t-1 and ends at time t. All jobs have to be completed by time T. We assume $T \geq \max_{1 \leq j \leq n} r_j + \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij}$. The time index variable, x_{ijt}, is equal to 1 if job j starts on machine i at time t and is otherwise equal to zero. The IP model for the studied problem is described below.

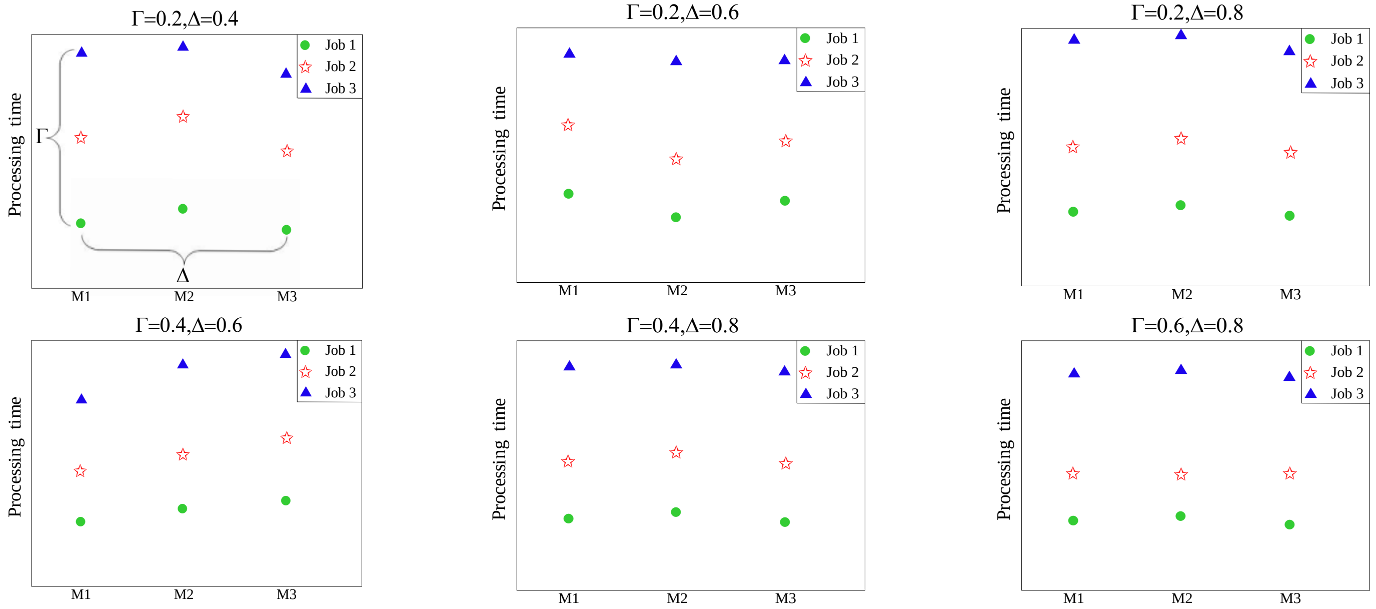

Fig. 1 Example of Case 9 forms

Objective: $min(1-\mathrm{A})\sum w_jC_j + A\sum U_j$ (1)

Subject to:

$$\sum_{i=1}^{m}\sum_{t=r_j}^{T-p_{ij}+1} x_{ijt} = 1\,, j = 1,2,\dots,n \quad (2)$$

$$\sum_{j=1}^{n}\sum_{s=\max(r_j,t-p_{ij}+1)}^{t} x_{ijs} \le 1, i = 1,2,\dots,m$$

$$t = 0,1,\dots,T \quad (3)$$

$$\sum_{i=1}^{m}\sum_{t=0}^{r_j} x_{ijt} = 0, j = 1,2,\dots,n \quad (4)$$

$$\sum_{i=1}^{m}\sum_{t=0}^{T-p_{ij}+1} x_{ijt}(t-1+p_{ij}) = C_j, j-1,2,\dots,n \quad (5)$$

$$C_j - MU_j \le d_j, \mathrm{j} = 1,2,\dots,\mathrm{n} \quad (6)$$

$$x_{ijt} \in \{0,1\},$$

$$i = 1,2,\dots,m; j = 1,2,\dots n; t = 0,1,\dots,T \quad (7)$$

$$U_j = \{0,1\}, j = 1,2,\dots n \quad (8)$$

Equation (1) indicates that the objective is to minimize $\sum U_j$ and TWC where A ∈ [0,1] controls the weight of objectives. Constraint (2) requires that each job can start only at one exact particular time on only one machine. Constraint (3) ensures that at any given time on each machine only one job at most can be processed. Constraint (4) demands that each job cannot be processed before it is released. Using the time-indexed variables, the completion time of a job j can be written as constraint (5). Constraint (6) specifies the tardy jobs where M is a big number. Constraints (7-8) state the non-negativity and integrality restrictions.

B. Bicriteria Heuristic

The WCT-NEH is originally developed for solving unrelated parallel machines with release time to minimize total weighted completion time [12]. The WCT-NEH combines the ideas of weighted completion time (WCT) and the Nawaz-Enscore-Ham (NEH) procedure. The NEH procedure was proposed by [13] to minimize makespan for the flow shop scheduling problem. We propose a bicriteria heuristic based on WCT-NEH and earliest due date (EDD) rule, so it can solve bicriteria problem. The proposed bicriteria heuristic is described below.

Bicriteria heuristic WCT_EDDR

1st phase: minimum completion time first

Step 1. Let V denote the set of unscheduled jobs; let t_i denote the total processing times of the jobs that have already been scheduled on machine i; let S_0 denote the set of scheduled jobs arranged by 1st phase. Initially, set V={1,...,n}, t_i=0 for i=1,...,m, S_0={null}.

Step 2. Determine the unscheduled job j^* on machine i^* such that $I_{i^*j^*} = \max_{1\le i\le m, j\in U} I_{ij}$, where $I_{ij} = \left(\frac{w_j}{\max(r_j,t_i)+p_{ij}}\right)$

Step 3. Schedule job j^* in the next available position on machine i^*. update $t_{i^*} = \max(r_{j^*}, t_{i^*}) + p_{i^*j^*}$. Set $V = V\backslash\{j^*\}$.

Step 4. Repeat Steps 2 to 3 until V={null}. The complete schedule is saved as S_0. Add S_0 into the solution set by setting solution set = solution set ∪ {num, $\sum U_j$, TWC} where num=0.

Step 5. Pick the lth (l=1,...,n) job from the job sequence in S_0 and find m+n-2 schedules by placing it at all possible m+n-2 positions in schedule S_0 without changing the relative positions of the remaining n-1 jobs. The number of enumerations in this step equals n(m+n-2).

Step 6: Add the generated n(m+n-2) solutions into the solution set by setting solution set = solution set ∪ {num, $\sum U_j$, TWC} where num=1,…, n(m+n-2).

Step 7. Find the schedule from the solution set that has minimum TWC and save its schedule to S_1.

2nd phase: EDD rule combines machine speed

Step 8. Let S denote the set of jobs scheduled by the 2nd phase; let A be the set of unscheduled jobs sorted in EDD rule; let B=(B_1,B_2,…,B_m) denote the set of machines sorted by nondecreasing order of sum of processing time of all jobs on machine i , ($\sum_{j=1}^{n} p_{ij}, \forall i$). Initially, set S_2={null}, t_i=0, for i=1,…,m, A=(A_1,A_2,…,A_n), and k=1.

Step 9. Increase num by 1.

Step 10. Based on jobs order in A, scan jobs from left to right, and find the first job p that assigns to the next available position on machine k without being tardy. If job p exists, then go to Step 13; otherwise, go to Step 11.

Step 11. If k <m, set k=k+1 and return to Step 10; otherwise, go to Step 12

Step 12. Select first job p from A. Find machine k that assigns job p to the next available position on machine k can led to minimum TWC.

Step 13. Set S_2=S_2 ∪ $\{p\}$. Assign job p to the next available position on machine k, and update $t_k = \max(r_p, t_k) + p_{kp}$. Update A=A\\{$p$}.

Step 14. Set $S_3 = S_1\backslash S_2$, and $S = S_2 \cup S_3$, preserving their ordering.

Step 15. Add the generated solution S into the solution set by setting solution set = solution set ∪ {num, $\sum U_j$, TWC }.

Step 16. If A={null}, go to Step 17; otherwise, return to Step 9.

Step 17. Find the schedule from the solution set that has minimum $\sum U_j$ and save its schedule to S. If there are more than two solutions that has minimum $\sum U_j$, then chose the one that has smaller TWC. The number of enumerations in this phase equals to n. So far, we have num=n(m+n-2)+n=(m-1)n+n^2. Assuming x=(m-1)n+n^2.

3rd phase: Reassign the tardy jobs

Step 18. Pick the lth (l=1,...,n) job from the job sequence in S and find m+n-2 schedules by placing it at all possible m+n-2 positions in schedule S without changing the relative positions of the remaining n-1 jobs. The number of enumerations in this step equals n(m+n-2).

Step 19. Add the generated n(m+n-2) solutions into the solution set by setting solution set = solution set ∪ {num, $\sum U_j$, TWC}, for num=x+1,..., x+ n(m+n-2). So, now we have num=(2m-3)n+2n^2.

Step 20. Let C be the set of tardy jobs and let π be the schedule in solution set{ x, $\sum U_j$, TWC}. Initially, set h=0, C={null}.

Step 21. Increase num by 1 and set h=h+1.

https://doi.org/10.17758/EIRAI2.F0418116

Step 22. Removes tardy jobs from schedule π, $\pi \in$ solution set$\{x+h, \sum U_j$, TWC$\}$ and add those tardy jobs into C. Sort jobs in C by the EDD rule. Update schedule π and C_j for all remaining jobs in π.

Step 23. Select the first job p from C. For schedule π, find machine k that assigns job p to the next available position on machine k can led to minimum TWC. Assign job p to the next available position on machine k, and update $t_k = \max(r_p, t_k) + p_{kp}$. Update C=C\\$\{p\}$.

Step24. If C=$\{null\}$, add the generated solution π into the solution set by setting solution set = solution set $\cup$ $\{num, \sum U_j$, TWC$\}$, go to Step 25; otherwise, return to Step 23.

Step25. If h=n(m+n-2), find all non-dominated solutions from solution set and terminate the procedure; otherwise, return to Step 21.

III. Computational results

In this section, we present several computational results regarding the performance of IP model and the WCT_EDDR. The IP model was coded in AMPL and implemented in CPLEX 11.2. The WCT_EDDR was implemented in Visual C++ and run on a computer with a 2.5 GHz Pentium Dual-Core E5200 CPU with 4GB memory. The correlated processing times p_{ij} were generated based on [2]. The value of w_j for each j was chosen randomly from the uniform distribution [1, 10]. Release dates and due dates were generated in a manner similar to that of [14]. We first generated release dates r_j from the uniform distribution $\left[0, \frac{\alpha}{m}\frac{\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}}{m}\right]$. In the next step, we generated slack times between due dates and earliest completion times from a uniform distribution $\left[0, \frac{\beta}{m}\frac{\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}}{m}\right]$. The earliest completion times of job j were estimated by $r_j + \bar{p}_j$ where $\bar{p}_j = \sum_{i=1}^{m} p_{ij}/m$. The due date of job j was then generated from $d_j = (r_j + \bar{p}_j) + \left[0, \frac{\beta}{m}\frac{\sum_{i=1}^{m}\sum_{j=1}^{n} p_{ij}}{m}\right]$. α controlled the range of release dates, and β controlled the range of due dates. High values of α tend to produce widely separated release dates, while high values of β tend to produce loose due dates. In this research, α and β were set at 0.25 and 0.50, and 0.75. The machine correlation factor Γ was set to 0.2, 0.4, and 0.6; the job correlation factor Δ was set to 0.4, 0.6, and 0.8. Since solving IP model can be very time consuming, we only tested the IP model and our heuristic on 4 machines with 20 jobs. For each combination of Γ, Δ, α and β, 10 problem instances were randomly generated.

In order to measure the results, we used number of non-dominated solutions, computation times, and a modified μ_d distance based on [15]. The $\boldsymbol{\mu_d}$ value is used to evaluate the distance between two non-dominated fronts. Let F_1 and F_2 be two non-dominated fronts obtained by a different method. Let n_1 be the number of solutions in the front F_1 and let n_2 be the number of solutions in the front F_2. The modified μ-distance is defined as :

$$\mu = \sum_{i=1}^{n_2} d_i$$

where di presents the minimum distance of every point i of F_2 to the point k (k=1,...,n_1) of F1, i.e.,

$$\boldsymbol{d_i = \min_{k=1,\dots,n_1}(\sqrt{(f_{1i}-f_{1k})^2+(f_{2i}-f_{2k})^2}\quad i = 1,\dots,n_2} \tag{9}$$

f_{1i} represents the first objective value of point i; f_{2i} represents the second objective value of point i. Thus the di takes the value zero if, for all objectives, point i reaches the value of point k. Then $\boldsymbol{\mu_d}$ is defined as in [9]:

$$\boldsymbol{\mu_d = \frac{\frac{1}{n_2}\sum_{i=0}^{n_2} d_i}{\sqrt{(f_{1max}-f_{1min})^2+(f_{2max}-f_{2min})^2}}} \tag{10}$$

where f_{1max}=max$\{f_1(x) \mid x \in F_1 \cup F_2\}$, f_{1min}=min$\{f_1(x) \mid x \in F_1 \cup F_2\}$, f_{2max}=max$\{f_2(x) \mid x \in F_1 \cup F_2\}$, and f_{2min}=min$\{f_2(x) \mid x \in F_1 \cup F_2\}$. The smaller the μ_d value is, the better the quality of its corresponding result is.

Table I shows the performance of heuristic WCT_EDDR and IP model for different combination of machine correlation and job correlation. Table I shows that the number of non-dominated solutions generated by the IP used is about 5.1 on average, the number of non-dominated solutions generated by the WCT_EDDR used is about 3.8. The average distance from solutions generated by the IP to the solutions generated by the WCT_EDDR is about 0.133. The average computation time of the IP is about 177.97 seconds and the average computation time of the WCT_EDDR is about 0.006 seconds. As the machine correlation Γ or job correlation Δ increasing, the computation time of IP increasing.

TABLE I
The performance of the heuristic WCT_EDDR for different combination of machine correlation and job correlation

α=β=0.5		Non-dominated solutions		Computational time (seconds)		μ-distance
Γ	Δ	Opt.	WCT_EDDR	Opt.	WCT_EDDR	
0.2	0.4	5.0	3.7	157.87	0.005	0.129
	0.6	5.0	3.6	163.02	0.008	0.163
	0.8	5.1	3.9	200.26	0.005	0.096
0.4	0.6	4.7	3.8	167.28	0.005	0.123
	0.8	5.0	3.7	186.43	0.005	0.128
0.6	0.8	5.6	3.8	192.97	0.005	0.159
Avg.		**5.1**	**3.8**	**177.97**	**0.006**	**0.133**

Table II shows the performance of heuristic WCT_EDDR and IP model for different release time factor and due date factor. Table II shows that the number of non-dominated solutions generated by the IP used is about 4.62 on average, the number of non-dominated solutions generated by the WCT_EDDR used is about 3.63. The average distance from

https://doi.org/10.17758/EIRAI2.F0418116

solutions generated by the IP to the solutions generated by the WCT_EDDR is about 0.154. The average computation time of the IP is about 155.23 seconds and the average computation time of the WCT_EDDR is about 0.006 seconds. As the release time factor α increases, the average distance from solutions generated by the IP to the solutions generated by the WCT_EDDR increases. Similarly, as the due date factor β decreases, the average distance from solutions generated by the IP to the solutions generated by the WCT_EDDR increases.

TABLE II
THE PERFORMANCE OF THE HEURISTIC WCT_EDDR FOR DIFFERENT RELEASE TIME FACTOR AND DUE DATE FACTOR

		Non-dominated solutions		Computational time (seconds)		μ -distance
α	β	Opt.	WCT_EDDR	Opt.	WCT_EDDR	
0.25	0.25	4.58	3.55	177.59	0.006	0.184
	0.50	5.52	4.23	212.67	0.006	0.097
	0.75	5.68	4.28	189.46	0.005	0.099
0.50	0.25	4.50	3.37	162.76	0.006	0.190
	0.50	5.07	3.75	177.97	0.006	0.133
	0.75	4.58	3.82	136.45	0.005	0.114
0.75	0.25	4.00	3.23	135.00	0.006	0.226
	0.50	4.15	3.35	116.06	0.006	0.171
	0.75	3.48	3.12	89.16	0.006	0.168
Avg.		**4.62**	**3.63**	**155.23**	**0.006**	**0.154**

We also analyze the results graphically in the objective space, as shown in Figs. 2 and 3. [16] stated that a "good approximation (solutions) typically consists of a set of diverse solutions that are uniformly distributed along the efficient frontier, and which are also close to the efficient frontier." Figs. 2 and 3 both show that the WCT_EDDR can generate a set of diverse solutions that are uniformly distributed along the efficient frontier generated by the IP.

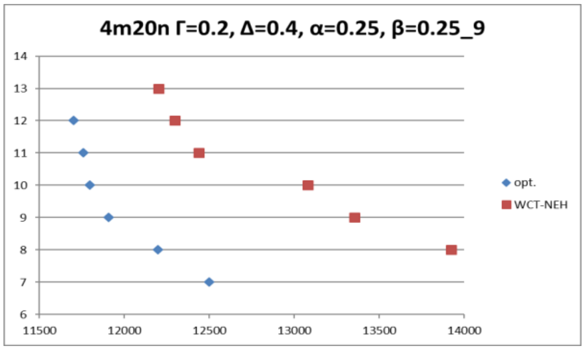

Fig. 2 Solutions in objective space: 4*m*20*n* with Γ=0.2, Δ=0.4, α=0.25, β=0.25

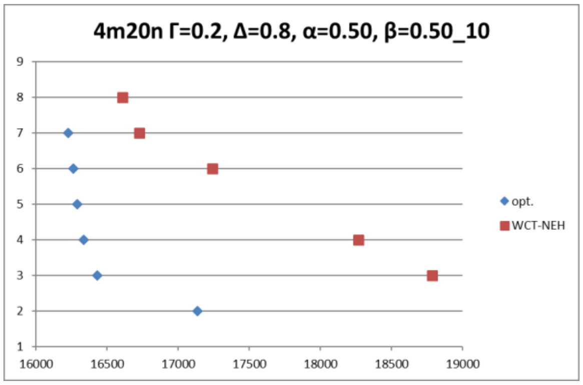

Fig. 3 Solutions in objective space: 4m20n with Γ=0.2, Δ=0.8, α=0.5, β=0.5

IV. CONCLUSIONS AND FUTURE WORK

This research proposes a heuristic WCT_EDDR to find non-dominated solutions for scheduling correlated parallel machine with release dates to minimize number of tardy jobs and total weighted completion time. The computational results show that the proposed heuristic is computationally efficient and provides solutions of reasonable quality. Future work can consider developing memetic algorithm by using WCT_EDDR to generate initial non-dominated solutions to tackle the studied problem.

REFERENCES

[1] Y.K. Lin. “Scheduling efficiency on correlated parallel machine scheduling problems”, Operational Research. Doi:1 /10.1007/s12351-017-0355-0 ,2017.

[2] Y.K. Lin, M.E. Pfund, and J.W. Fowler, “Processing time generation schemes for parallel machine scheduling problems with various correlation structures”, Journal of Scheduling, vol.17, no.6, pp.569-586, 2014.
https://doi.org/10.1007/s10951-013-0347-8

[3] A.J. Ruiz-Torres, E.E. Enscore, and R.R. Barton, “Simulated annealing heuristics for the average flow-time and the number of tardy jobs bi-criteria identical parallel machine problem”, Computers & industrial engineering, vol.33, no.1-2, pp.257-260, 1997.
https://doi.org/10.1016/S0360-8352(97)00087-9

[4] D. Cao, M. Chen, and G. Wan, “Parallel machine selection and job scheduling to minimize machine cost and job tardiness”, Computers & operations research, vol.32, no.8, pp.1995-2012, 2005.
https://doi.org/10.1016/j.cor.2004.01.001

[5] J.N. Gupta, and A.J. Ruiz-Torres, “Generating efficient schedules for identical parallel machines involving flow-time and tardy jobs”, European Journal of Operational Research, vol.167, no.3, pp.679-695, 2005
https://doi.org/10.1016/j.ejor.2004.07.015

[6] R. Tavakkoli-Moghaddam, F. Taheri, M. Bazzazi, M. Izadi, and F. Sassani, “Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints”, Computers & Operations Research, vol.36,no.12, pp. 3224-3230, 2009.
https://doi.org/10.1016/j.cor.2009.02.012

[7] M.A. Bozorgirad, and R. Logendran, “Sequence-dependent group scheduling problem on unrelated-parallel machines”, Expert Systems with Applications, vol.39, no.10, pp.9021-9030, 2012.
https://doi.org/10.1016/j.eswa.2012.02.032

[8] Y.K. Lin, J.W. Fowler., and M.E. Pfund, “Multiple-objective heuristics for scheduling unrelated parallel machines”, European Journal of Operational Research, vol.227, no.2, pp.239-253, 2013.
https://doi.org/10.1016/j.ejor.2012.10.008

[9] Y.K. Lin, and H.C. Lin, “Bicriteria scheduling problem for unrelated parallel machines with release dates”, Computers & Operations Research, vol.64, pp.28-39, 2015
https://doi.org/10.1016/j.cor.2015.04.025.

[10] O. Shahvari, and R. Logendran, "An enhanced tabu search algorithm to minimize a bi-criteria objective in batching and scheduling problems on unrelated-parallel machines with desired lower bounds on batch sizes", Computers & Operations Research vol.77, pp.154-176, 2017. https://doi.org/10.1016/j.cor.2016.07.021

[11] J.K. Lenstra, A.R. Kan, P. Bricker, "Complexity of machine scheduling problems", Ann Discrete Math 1, pp.343-362, 1977. https://doi.org/10.1016/S0167-5060(08)70743-X

[12] Y.K. Lin, and C.W. Lin, "Dispatching rules for unrelated parallel machine scheduling with release dates", The International Journal of Advanced Manufacturing Technology, vol.67, pp.269-279, 2013. https://doi.org/10.1007/s00170-013-4773-8

[13] M. Nawaz., E.E. Enscore Jr, I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem", Omega vol.11, no.1, pp.91-95, 1983.. https://doi.org/10.1016/0305-0483(83)90088-9

[14] L. Mönch, H. Balasubramanian, J.W. Fowler, M.E. Pfund, "Heuristic scheduling of jobs on parallel batch machines with incompatible job families and unequal ready times", Computers and Operations Research, vol.32, pp.2731-2750, 2005 https://doi.org/10.1016/j.cor.2004.04.001

[15] F. Dugardin, F. Yalaoui, L. Amodeo, "New multi-objective method to solve reentrant hybrid flow shop scheduling problem", European Journal of Operational Research, vol.203, pp.22-31, 2010.https://doi.org/10.1016/j.ejor.2009.06.031

[16] W.M. Carlyle, J.W. Fowler, E.S. Gel, B. Kim, "Quantitative comparison of approximate solution sets for bicriteria optimization problems", Decision Sciences, vol.34,pp. 63-82, 2003. https://doi.org/10.1111/1540-5915.02254