Int J Adv Manuf Technol (2009) 43:161–169
DOI 10.1007/s00170-008-1692-1

ORIGINAL ARTICLE

Hybrid metaheuristics for unrelated parallel machine scheduling with sequence-dependent setup times

Chun-Lung Chen · Chuen-Lung Chen

Received: 26 January 2007 / Accepted: 30 July 2008 / Published online: 26 August 2008

Abstract This paper proposes several hybrid metaheuristics for the unrelated parallel-machine scheduling problem with sequence-dependent setup times given the objective of minimizing the weighted number of tardy jobs. The metaheuristics begin with effective initial solution generators to generate initial feasible solutions; then, they improve the initial solutions by an approach, which integrates the principles of the variable neighborhood descent approach and tabu search. Four reduced-size neighborhood structures and two search strategies are proposed in the metaheuristics to enhance their effectiveness and efficiency. Five factors are used to design 32 experimental conditions, and ten test problems are generated for each condition. Computational results show that the proposed hybrid metaheuristics are significantly superior to several basic tabu search heuristics under all the experimental conditions.

Keywords Weighted number of tardy jobs · Unrelated parallel machine · Sequence-dependent setup · Variable neighborhood descent · Tabu search

C.-L. Chen (✉)
Department of MIS, National Chengchi University,
ZhiNan Rd., Wenshan District,
Taipei City 11605 Taiwan, Republic of China
e-mail: chencl@mis.nccu.edu.tw

C.-L. Chen
Department of Accounting Information,
Takming University of Science and Technology,
No. 56, Sec. 1, Huanshan Road, Neihu District,
Taipei City, Taiwan, Republic of China

1 Introduction

Unrelated parallel machine scheduling is widely applied in manufacturing environments such as the drilling operations for printed circuit board fabrication [1, 2] and the dicing operations for semiconductor wafer manufacturing [3]. This paper studies the unrelated parallel machine scheduling problem with sequence-dependent setup times given the objective of minimizing the weighted number of tardy jobs.

Much research work has been done to date on the development of solutions for unrelated parallel machine scheduling problems tackling a variety of objectives. Piserma and Van Dijk [4] developed a local search heuristic and a tabu search (TS) algorithm, while Ghirardi and Potts [5] developed recovering beam search algorithms to solve the problems with makespan as their objective. Furthermore, Chen [6] proposed a heuristic for the problems with job setup times. Weng et al. [7] presented seven heuristics to achieve a reduction to the weighted completion time.

There are a few papers that dealt with due date-related measures. Suresh and Chaudhuri [8] developed a GAP–EDD algorithm minimizing the maximum tardiness, while Chen [9] presented a TS heuristic with the same objective considering problems with job setup times. With respect to the objective of minimizing the mean tardiness, Guinet [10] proposed a simulated annealing heuristic, while Randhawa and Kuo [11] examined the factors that may affect the performance of a parallel machine system and presented several heuristics to achieve the same objective. In addition, Kim et al. [3] proposed a simulated annealing heuristic for the problems with job sequence-dependent setup times. Considering the objective of minimizing the number of tardy jobs, Ho and Chang [12] proposed several heuristics derived from Moore's algorithm [13] to solve problem with identical parallel machine, while Ruiz-Torres et al. [14]

proposed an integer programming formulation and several heuristics extended from Ho and Chang's heuristics [12] to solve problems with uniform parallel machine. M'Hallah and Bulfin's [15] research was the only one considering unrelated parallel machine; they developed branch and bound algorithms to optimally minimize the weighted and unweighted number of tardy jobs for the identical and unrelated parallel machine cases.

The review of available literature leads us to conclude that the candidate problem in this research, the unrelated parallel-machine scheduling problem with job sequence-dependent setup times and with the weighted number of tardy jobs as the objective, has not been studied; also, local search methods, TS and simulated annealing, are effective tools to achieve the objective of minimizing the maximum tardiness and mean tardiness for unrelated parallel machine scheduling with job sequence-dependent setup times. Therefore, in this research, we develop hybrid metaheuristics integrating variable neighborhood descent (VND) and TS principles to solve the candidate problem. Four reduced-size neighborhood structures and two search strategies are proposed in the hybrid metaheuristics to enhance their effectiveness and efficiency. Five factors are used to design 32 production scenarios, and ten test problems are generated for each scenario. Computational results show that the reduced-size neighborhood structures and the search strategies do benefit the effectiveness and efficiency of the proposed metaheuristics, and the performance of the metaheuristics is significantly superior to several heuristic rules and a basic TS under all the production scenarios. Eighty small-sized test problems, each comprised of eight jobs and two machines, are also generated under different production scenarios to test the capability of the proposed metaheuristics for finding the optimal solutions. The conclusion is also very promising.

The rest of the paper is organized as follows: In Sections 2 and 3, we describe the candidate problem considered in this study and the proposed hybrid metaheuristics, respectively. We proceed to detail the generation test data and analyze the experimental results in Section 4. Finally, Section 5 summarizes the major findings of this paper and provides suggested directions for future research in this area.

2 Problem statement

The unrelated parallel machine scheduling problem with sequence-dependent setup times can be described as follows: A set of independent jobs $J_i(i \in J = \{1, 2, ..., n\})$ has to be scheduled on a set of unrelated parallel machines $M_k(k \in M = \{1, 2, ... m\})$. Each job J_i has a positive processing time p_{ik}, a positive setup time s_{ijk}, and a positive weight w_i, wherein p_{ik} is determined by the assigned machine k and s_{ijk} is determined by job J_j, the job scheduled subsequent to job J_i on machine k. In addition, the number of machines (m) is at least two, and the number of jobs (n) is greater than or equal to the number of machines (m). In addition, we assume that all the jobs are available for processing at time zero, there is no machine breakdown, machines can process only one job at a time, and that jobs cannot be preempted. The objective is to find a schedule that minimizes the weighted number of tardy jobs and is defined as follows: Objective $=$ Minimize $\sum_{i=1}^{n} w_i \times U_i$, where w_i is the weight of job J_i and U_i equals 1 if job J_i is tardy; otherwise, U_i equals 0.

3 The proposed hybrid metaheuristics

Two types of generators are used to generate initial solutions in this research, and the proposed metaheuristics, integrating the principles of the VND approach and TS, are applied to improve the initial solutions. Four reduced-size neighborhood structures and two search strategies are incorporated in the hybrid metaheuristics to enhance their effectiveness and efficiency. The main components of the proposed metaheuristics are described below.

3.1 Variable neighborhood descent heuristic

Variable neighborhood search (VNS) heuristic was developed by Mladenović and Hansen [16]. This heuristic searches the solution space with a set of predefined neighborhood structures and escapes from local optima by systematically changing the use of the neighborhood structures. The VNS heuristic has been successfully applied in many areas such as the traveling salesman problem [16], p median problem [17], degree-constrained minimum spanning tree problem [18], median cycle problem [19], vehicle routing problem [20, 21], maximum clique problem [22], resource-constrained project scheduling problem [23], and the multiprocessor scheduling problem with communication delays [24]. Hansen and Mladenović [17] modified the VNS heuristic and proposed the VND heuristic. The major difference between the VNS and VND is that, within each neighborhood structure, VNS starts the search with a randomly generated neighbor solution; on the other hand, VND starts the search from the best solution generated in the previous iteration.

3.2 Tabu search

TS was developed by Glover [25]. It is an iterative process that explores the solution space by repeatedly making moves from one solution, x, to another solution, x', located on the neighborhood, $N(x)$, of x. These moves are performed with an ultimate goal of reaching a good solution by evaluating some objective function $f(x)$.

However, unlike other search methods, in TS, the objective value, $f(x')$, need not be better than $f(x)$ in every iteration. One of the main ideas of TS, as its name depicts, is its use of a flexible memory (tabu list) to tabu certain moves for a number of iterations. In every iteration of TS, a move will instantly be assigned to the tabu list when the move is chosen to lead the search from the current solution to its neighbor solution. This move will no longer be chosen for a number of immediately succeeding iterations. This number of iterations is denoted as tabu list size, and the list size is limited to a certain length. When the list has reached its specified length, the move that was assigned to the list earliest is released from the list, and the most current is placed on top of the list. With an appropriate design of the tabu list, TS is able to prevent the cycling of the search and guide the search to the solution regions that have not been examined and approach good solutions in the solution space. A great number of successful applications of TS for scheduling problems have been reported and can be found in the reviewed literature [2, 11, 26–28].

Several factors will affect the performance of TS: initial solution, type of move, neighborhood size, tabu list size, aspiration criterion, and stopping criterion. We will first discuss the last four factors here and detail the first two factors in the next sections. Neighborhood size is the number of neighbor solutions to be evaluated in each iteration. The two most commonly used neighborhood sizes are considered in this paper. The first type evaluates all possible neighbor solutions and selects the best non-tabued solution in each iteration. This type of neighborhood size is denoted as whole (WHL) size in this paper. The second type evaluates the neighbor solutions in certain order and selects the first neighbor solution that is better than the current solution and that is not tabued in each iteration. This type of neighborhood size is denoted as random (RAN) size. Glover [25] indicated that regardless of problem size, seven is a magic number for tabu list size, so seven is chosen for the purposes of this research. Aspiration criterion is the criterion used to override the tabu status of a move. The most common aspiration criterion is that a tabued move can be aspired if the move can provide a better solution than the incumbent solution. A TS application can choose to use or not to use the aspiration criterion. The stopping criterion is the criterion used to terminate the search process. The number of iterations with no improvement in the incumbent solution is adopted as the stopping criterion in this paper.

3.3 Initial solution generators

Two types of initial solution generators are used to generate initial solutions for TS and the proposed metaheuristics: the job based and the machine based. The job-based generator first arranges jobs in an ascending order based on their characteristics such as weighted processing time or weighted due date; the weighted processing time of a job is the ratio of its processing time to its weight, and the weighted due date of a job is the ratio of its due date to its weight. Then, each job is assigned to a machine based on the determined job sequence, enabling the completion of the job at the earliest time. If there is a tie, the job is assigned to the machine with the earliest available time. After all the jobs are scheduled, Moore's algorithm is applied to each machine separately. Thus, the two job-based initial solution generators are referred to as job-based shortest weighted processing time (JB-SWPT) and job-based earliest weighted due date (JB-EWDD). On the other hand, the machine-based initial solution generator first assigns each job to the machine that processes the job with the shortest processing time; if there is a tie, break the tie arbitrarily. After all jobs are assigned to the machines, Moore's algorithm is applied to each machine separately. This machine-based generator is denoted as MB.

3.4 Reduced-size neighborhood structures

Of the local search methods for scheduling problems, SWAP and INSERT are the two most commonly used ways to define neighbor solutions for a given solution. Given a job sequence, SWAP finds a neighbor solution by swapping two selected jobs, and INSERT finds a neighbor solution by identifying a job and placing the job directly before the first job, between every two consecutive jobs, and directly subsequent to the last job. Given a parallel machine schedule with n jobs and m machines, a direct manner of applying SWAP to generate its neighborhood structure starts with swapping every two jobs on the first machine then swapping each of the jobs on the first machine with the jobs on all other machines; the same operation is performed for the second machine, the third machine, and so on until the m-th machine. The neighborhood size generated by this procedure is denoted as $\mathrm{NS}_{\mathrm{SWAP}}$ in this research. The same procedure can be applied to generate a neighborhood structure by using INSERT, and the size generated by this procedure is denoted as $\mathrm{NS}_{\mathrm{INSERT}}$.

As the objective of the candidate problem is to minimize the total weighted number of tardy jobs, we propose four reduced-size neighborhood structures to enhance both efficiency and effectiveness of the proposed metaheuristics. The first two neighborhood structures are constructed by first choosing the machine with the largest total weighted number of tardy jobs. Then, the neighbor solutions in the first neighborhood structure (NS_1) are generated by applying SWAP to swap each of the tardy jobs on the chosen machine with the jobs on the same machine and with the jobs on the other machines. The neighbor solutions in the second neighborhood structure (NS_2) are generated by applying INSERT to insert each of the tardy jobs on the

Table 1 Experimental design used in random problem generation

Factors	Values used	Total values
Number of jobs	40, 80	2
Number of machines	4, 8	2
Processing times	DU[50, 70], DU[20, 100]	2
Setup times	DU[0.2, 0.4]×60, DU[0.6, 0.8] ×60	2
Tardiness factors and due-date range (*T*, *R*)	(0.4, 0.8), (0.5, 0.8)	2
	Total parameter combinations	32
	Number of problems/ combinations	10
	Total problems	320

chosen machine and on the other machines. As SWAP or INSERT is applied to only the tardy jobs on the chosen machine, the size of NS_1 is obviously much smaller than the size of NS_{SWAP}, and the size of NS_2 is much smaller than the size of NS_{INSERT}. Furthermore, as the chosen machine has the largest total weighted number of tardy jobs, there is a higher possibility that applying SWAP or INSERT to the tardy jobs on the machine will generate improved solutions. The third and the fourth neighborhood structures consider all the machines except the machine with the largest total weighted number of tardy jobs. In these machines, the third neighborhood structure (NS_3) is constructed by applying SWAP to swap each of the tardy jobs with the other jobs on the machines, and the fourth neighborhood structure (NS_4) is constructed by applying INSERT to insert each of the tardy jobs on all the possible positions on the machines. It is clear that the size of NS_3 is larger than the size of NS_1, but it is still much smaller than the size of NS_{SWAP}. Similarly, the size of NS_4 is larger than that size of NS_2, but it is still much smaller than the size of NS_{INSERT}. In addition, as SWAP or INSERT is only applied to the tardy jobs on the machines, it may still provide a good possibility to generate improved solutions.

3.5 The proposed hybrid metaheuristics (VND-TS)

As mentioned, VND searches the solution space with a set of predefined neighborhood structures and escapes from local optima by systematically changing the use of neighborhood structures. VND can apply different search algorithms to search the solution space. In this research, we integrate TS into VND; applying TS to search the solution space with the pre-defined neighborhood structures: NS_1, NS_2, NS_3, and NS_4. Given an initial solution, the proposed metaheuristics implement the following four steps to search the solution space:

Step 1: Apply TS with NS_1, which defines the neighborhood structure to search the solution space. When the stopping criterion is satisfied, go to step 2 with the incumbent solution produced in step 1 as the initial solution.

Step 2: Apply TS with NS_2 to search the solution space. When the stopping criterion is satisfied, evaluate the incumbent solution produced in step 2. If the incumbent solution produced in step 2 is better than that produced in step 1, return to step 1 and empty the tabu list produced in step 1; otherwise, go to step 3 and use the incumbent solution produced in step 2 as the initial solution.

Step 3: Apply TS with NS_3 to search the solution space. When the stopping criterion is satisfied, evaluate the incumbent solution produced in step 3. If the incumbent solution produced in step 3 is better than that produced in step 1, return to step 1 and empty the tabu lists produced in step 1 and step 2; otherwise, go to step 4 and use the incumbent solution produced in step 3 as the initial solution.

Step 4: Apply TS with NS_4 to search the solution space. When the stopping criterion is satisfied, evaluate the incumbent solution produced in step 4. If the incumbent solution produced in step 4 is better than that produced in step 1, return to step 1 and empty the tabu lists produced in step 1, step 2, and step 3; otherwise, stop.

The idea behind the procedure is to efficiently guide the search of TS to better solution regions by searching a small number of neighbor solutions in step 1 and step 2. If the search traps in a local optimum, the number of neighbor solutions searched is then increased in step 3 and step 4. It is expected that increasing the neighborhood size while maintaining a good possibility for improving the solutions may help the search escape from the local optimum.

Table 2 ANOVA table for testing the significance of the initial solution generators and the heuristics (α=0.01)

Source	Sum of squared error	Degree of freedom	Mean squared error	*F-ratio*
Initial solution generators	24.1132	2	12.0566	419.6826
Heuristics	235.0055	4	58.7514	2045.1010
Number of jobs	4.1875	1	4.1875	145.7647
Number of machines	6.3515	1	6.3515	221.0929
Processing times	3.1705	1	3.1705	110.3642
Setup times	0.0002	1	0.0002	0.0085
Due-date tightness	0.2796	1	0.2796	9.7326
Error	137.5490	4788	0.0287	
Total	410.6571	4799		

Table 3 Results of Duncan's multiple range test for the initial solution generators (α=0.01)

Initial solution generators	Results (groups)	Average RDI
MB	A	0.2313
JB-EWDD	B	0.3526
JB-SWPT	C	0.3995

Furthermore, note that two ways of searching the generated neighbor solutions, WHL and RAN, are considered in TS in this research, so they are also used in the TS in each step in the proposed metaheuristics. However, as the design of the reduced-size neighborhood structures and the four-step search strategy, WHL and RAN may contribute differently to the performance of the metaheuristics than that of the basic tabu search. In this research, we define the search strategy using WHL as VND/WHL and define the search strategy using RAN as VND/RAN. In addition, the metaheuristic using the search strategy VND/WHL is defined as VND-TS/WHL and as VND-TS/RAN when the search strategy VND/RAN is used.

4 Computational experiments

A series of computational experiments have been conducted to evaluate the performance of the proposed metaheuristics. Table 1 summarizes the experimental factors used in generating different test problems: number of jobs, number of machines, range of job processing times, range of setup times, and tightness of job due dates. The number of jobs has two levels, with values set at 40 and 80, and the number of machines has two levels, with values set at 4 and 8. The processing times of a job on different machines are generated from a discrete uniform distribution; two ranges, [50, 70] and [20, 100], with equal mean, but unequal variations are considered for uniform distribution. The due dates of jobs are generated from a discrete uniform distribution U(L(1-T-R/2), L(1-T + R/2)), where L is a lower bound of makespan and T and R are the tardiness factor and due date range, respectively. A straight lower bound for makespan of the candidate problem is $L = \sum_{i=1}^{n}\left(p_i^{\min} + s_i^{\min}\right)/m$, where $p_i^{\min}$ is the minimum processing time of job i on the machines and $s_i^{\min}$ is the minimum setup time of job i [3, 29]. This lower bound is used in this paper with two levels of [T, R], [0.4, 0.8] and [0.5, 0.8] considered in generating the due date. Two ranges, [12, 24] and [24, 36], are used in generating job setup times. These two ranges are 20% to 40% and 40% to 60% of the mean, which is 60, of the distributions for generating job processing times. The setup time matrices are asymmetric and satisfy the triangle inequality (for details, see Rios-Mercado [30]). Finally, the weight of job i, w_j is randomly generated from U[1, 10]. With the five two-level factors considered, there are a total of 32 experimental combinations, and ten test problems are generated for each combination in the experiment.

The basic TS (BTS) in this research uses SWAP to generate neighbor solutions and uses WHL and RAN as neighborhood sizes, respectively. The BTS using WHL is

Table 4 Results of Duncan's multiple range test for the heuristics (α=0.01)

Heuristics	Results (groups)	Average RDI
VND-TS/RAN	A	0.1726
VND-TS/WHL	B	0.2150
BTS/RAN	C	0.2393
BTS/WHL	C	0.2447
NONE	D	0.7674

Table 5 ANOVA table for testing the significance of the heuristics combining the initial solution generators and the heuristics (α=0.01)

Source	Sum of squared error	Degrees of freedom	Mean squared error	F-ratio
Heuristics	261.0713	14	18.6479	657.3712
Number of jobs	4.1875	1	4.1875	147.6166
Number of machines	6.3515	1	6.3515	223.9018
Processing times	3.1705	1	3.1705	111.7663
Setup times	0.0002	1	0.0002	0.0087
Due-date tightness	0.2796	1	0.2796	9.8563
Error	135.5964	4,780	0.0284	
Total	410.6571	4,799		

Table 6 Results of Duncan's multiple range test for the heuristics combining the three initial solution generators and the five heuristics (α=0.01)

Heuristics	Results (groups)	Average RDI
MB/VND-TS/RAN	A	0.0529
MB/VND-TS/WHL	B	0.0874
MB/BTS/RAN	C	0.1548
MB/BTS/WHL	C	0.1608
JB-EWDD/VND-TS/RAN	D	0.2161
JB-SWPT/VND-TS/RAN	DE	0.2489
JB-EWDD/VND-TS/WHL	EF	0.2628
JB-EWDD/BTS/RAN	EF	0.2661
JB-EWDD/BTS/WHL	EFG	0.2693
JB-SWPT/VND-TS/WHL	FG	0.2948
JB-SWPT/BTS/AN	FG	0.2971
JB-SWPT/BTS/WHL	G	0.3038
MB	H	0.7005
JB-EWDD	I	0.7486
JB-SWPT	J	0.8531

Table 7 Performance comparisons of the heuristics combining the three initial solution generators and the five heuristics (in RDI and NBS[a])

	RDI					NBS[a]				
	NONE	BTS/WHL	BTS/RAN	VND-TS/WHL	VND-TS/RAN	NONE	BTS/WHL	BTS/RAN	VND-TS/WHL	VND-TS/RAN
JB-SWPT	0.8531	0.3038	0.2971	0.2948	0.2489	0	15	18	14	26
JB-EWDD	0.7486	0.2693	0.2661	0.2628	0.2161	0	15	17	13	28
MB	0.7005	0.1608	0.1548	0.0874	0.0529	0	37	45	89	162

[a] NBS found in a total of 320 problems.

denoted as BTS/WHL, and the BTS using RAN is denoted as BTS/RAN. It uses seven as the tabu list size and 50 iterations with no improvement as the stopping criterion and chooses to use aspiration criterion. As in the VND-TS/WHL and VND-TS/RAN, the TS uses the same tabu list size and aspiration criterion, but uses a smaller number of iterations as the stopping criterion. Note that the VND-TS/WHL and VND-TS/RAN procedures will be terminated when the stopping criterion of the TS in all four stages are satisfied. If we use 50 iterations with no improvement as the stopping criterion in the TS in each stage, the stopping criterion of the procedure becomes 200 iterations with no improvement. To make the stopping criterion of the procedure in line with that of the BTS, we use 15 iterations with no improvement as the stopping criterion for the TS in each stage of the procedure.

The relative deviation index (RDI) and the number of best solutions (NBS) are the criteria used to evaluate the performance of the heuristics. The RDI was used by Lee et al. [31] and Choi et al. [32] and is defined as:

$$RDI = \begin{cases} \frac{S_a - S_b}{S_w - S_b} & \text{if } (S_w - S_b) \neq 0, \\ 0 & \text{otherwise.} \end{cases}$$

S_a is the solution value obtained by method a, and S_b and S_w are, respectively, the best and the worst solution values among those obtained by the methods included in the comparison. If S_w equals S_b in a problem, then RDIs of all algorithms are set to 0.

We applied the analysis of variance (ANOVA) and Duncan's multiple range test to analyze the output (RDI) of the test problems. Table 2 presents the results of the ANOVA, which show that both initial solution generators and heuristics significantly affect the output (RDI) of the test problems. The initial solution generators include JB-SWPT, JB-EWDD, and MB, and the heuristics include BTS/WHL, BTS/RAN, VND-TS/WHL, VND-TS/RAN, and NONE, which represents making no modifications on the solutions produced by the initial solution generators. We further applied Duncan's multiple range test to measure if the performance of any two of the initial solution generators or any two of the heuristics is significantly different. Table 3 presents the results of Duncan's test on the initial solution generators. Note that the initial solution generators are sequenced in ascending order based on the average RDI. If two generators have the same letter in the column of results (groups), the performance of the two generators is not considered to be significantly different. As all the generators in Table 3 have different letters, the performance of every pair of generators is significantly different with α= 0.01. The average RDIs of MB, JB-EWDD, and JB-SWPT are 0.2313, 0.3526, and 0.3995, respectively. These data show that the machine-based generator significantly dominates job-based generators. Table 4 presents the results of Duncan's test on the heuristics. The average RDIs of VND-TS/RAN, VND-TS/WHL, BTS/RAN, BTS/WHL, and NONE are 0.1726, 0.2150, 0.2393, 0.2447, and 0.7674, respectively. The test results show that both the VND-TS heuristics, VND-TS/RAN and VND-TS/WHL, significantly dominate the two BTS heuristics, BTS/WHL and BTS/RAN. The performance of BTS/RAN and BTS/WHL is not significantly different; however, VND-TS/RAN significantly dominates VND-TS/WHL. These results demonstrate the effectiveness of the proposed metaheuristics and confirm our expectation that WHL and RAN contribute significant-

Table 8 Results of the heuristics under different experimental factors (in RDI)

Heuristics	Number of Jobs		Number of machines		Processing time		Setup time		Due date tightness	
	Low	High	Low	High	Low	High	Low	High	Low	High
MB	0.7106	0.6905	0.6009	0.8002	0.8187	0.5823	0.7031	0.698	0.7212	0.6799
MB/BTS-WHL	0.1687	0.153	0.1512	0.1704	0.1926	0.129	0.1662	0.1555	0.1827	0.1389
MB/BTS-VAN	0.155	0.1547	0.1462	0.1634	0.1906	0.119	0.1581	0.1515	0.1757	0.134
MB/VND-TS/WHL	0.1053	0.0696	0.0814	0.0935	0.0921	0.0828	0.0927	0.0822	0.0783	0.0966
MB/VND-TS/RAN	0.0755	0.0303	0.0457	0.06	0.0565	0.0493	0.0527	0.0531	0.0506	0.0552

Table 9 Average computation time required for the heuristics (seconds)

Heuristics	Job sizes * Machine sizes				Average
	40×4	40×8	80×4	80×8	
MB	<0.01	<0.01	<0.01	<0.01	<0.01
MB/BTS/WHL	6.98	6.82	38.98	40.41	23.30
MB/BTS/VAN	7.39	7.21	40.19	40.71	20.58
MB/VND-TS/WHL	2.27	3.17	12.05	15.39	8.22
MB/VND-TS/RAN	3.09	3.79	18.85	23.15	12.22

ly differently to the performance of the metaheuristics when compared to that of the basic tabu search. The contribution of the proposed search strategies to the proposed metaheuristics can be further investigated into the average effects of the search strategies on the performance of the metaheuristics. The average effect of the proposed strategy, VND/RAN, on the performance of VND-TS/RAN can be estimated by the difference between the average RDIs of BTS/RAN and VND-TS/RAN (0.2393−0.1726=0.0667). The average effect of the search strategy, VND/WHL, on the performance of VND-TS/WHL can be estimated by the difference between the average RDIs of BTS/WHL and VND-TS/WHL (0.2447−0.2150=0.0297). These average effects reveal that the search strategies make the metaheuristics dominate the BTS heuristics. This is a noteworthy point because local search methods, such as TS, simulated annealing and genetic algorithms, are commonly used methods for the candidate problem and other scheduling problems. Therefore, the proposed metaheuristic approach deserves to be studied for solving other scheduling problems or to integrate with other local search methods. In addition, the average effects show that RAN makes the search strategies more effective. This may be resulted from the fact that RAN guides the search more randomly and may increase the explorative capability of the search; however, this deserves to be further studied.

Furthermore, we investigate the performance of the heuristics, which combine the three initial solution generators and the five heuristics; the ANOVA and Duncan's test are applied to measure the performance of the 15 heuristics. Tables 5 and 6 present the results of the ANOVA and the Duncan test. The results in Table 6 show that the performance of MB/VND–TS/RAN, the heuristic combining MB and VND-TS/RAN, significantly dominates all the other heuristics. Table 7 summarizes the average RDI and NBS for the 15 heuristics. The average RDI and NBS of MB/VND–TS/RAN are 0.0529 and 162, respectively. That is MB/VND–TS/RAN generated 162 best solutions out of 320 test problems; also, although it does not produce the best solutions in all of the test examples, on average, it deviates from the best solutions by only about 5%. Therefore, we choose MB/VND–TS/RAN as the best heuristic for the candidate problem and further study the robustness of its performance. Table 8 summarizes the average RDI for the heuristics using MB as the initial solution generator at each level of the experimental factors. The results show that MB/VND–TS/RAN consistently dominates all the other heuristics under different experimental conditions. Furthermore, the data in the table show that, except the number of jobs, the average RDIs of MB/VND–TS/RAN under all the factors, regardless of low level or high level, are all around 0.05. Therefore, we may conclude that the performance of MB/VND–TS/RAN is quite robust to the experimental conditions. In addition, the average RDI of MB/VND–TS/RAN decreases to 0.0303 at a high level of jobs, which are 80-job problems. This may be resulted to the effectiveness of MB/VND–TS/RAN for more complicated problems.

The efficiency of the proposed heuristics is further discussed. All the heuristics are coded using the C++ language, and all the experiments are performed on a PC with AMD 1.8 GHz and 448 MB RAM. Table 9 displays the average computation time (in seconds) required for the heuristics using MB as the initial solution generator. The results indicate that the proposed metaheuristics not only dominate the BTS, but is more efficient than the BTS. It reduces the CPU times by about 50%. This efficiency should be resulted from the effect of the reduced-size neighborhood structures. In addition, the average CPU time for MB/VND–TS/RAN to solve an 80-job, eight-machine problem is 23.15 s, making it fast enough to be used in practice.

Finally, we study the capability of MB/VND–TS/RAN for finding optimal solutions. As no research has been conducted for finding optimal solutions for the candidate problem, a set of small-size (eight-job and two-machine) test problems are generated based on the combinations of the remaining three experimental factors: range of job processing times, range of setup times, and tightness of job due dates, and ten test problems are generated for each combination. The enumerative method is applied to find the optimal solutions for all the 80-test problems. Table 10 presents the average RDI and the NBS produced by applying MB, MB/BTS/RAN, and MB/VND–TS/RAN to solve the test problems. The resulting data show that MB/VND–TS/RAN is able to find optimal solutions for 62 out of the 80 test problems, and its average RDI is as low as

Table 10 Performance comparisons of MB/VND-TS/RAN vs. optimal solutions

	RDI	NBS[a]	CPU times
Optimal Solutions	0	80	496
MB	0.3425	14	0.0006
MB/BTS/RAN	0.0592	51	0.0109
MB/VND-TS/RAN	0.0239	62	0.0062

[a] NBS found in a total of 80 problems.

0.0239. These results once again confirm that MB/VND-TS/RAN is a promising heuristic for the candidate problem.

5 Conclusions

This paper develops hybrid metaheuristics to minimize the total weighted number of tardy jobs for the unrelated parallel machine problem with sequence-dependent setup times. Taking the characteristic of the objective into consideration, the hybrid metaheuristics propose four reduced-size neighborhood structures and two search strategies; the computational results show that the reduced-size neighborhood structures and the search strategies significantly enhance the effectiveness and efficiency of the proposed metaheuristics such that the proposed metaheuristics significantly outperform the BTS. As local search methods, such as TS, simulated annealing, and genetic algorithms, are commonly used methods for scheduling problems, applying the hybrid metaheuristic approach to other scheduling problems with the neighborhood structures and search strategies considering the characteristics of the scheduling problems is a valuable research subject. Furthermore, combining VND with other local search methods or path-relinking procedures can be another interesting subject.

Acknowledgements This paper was supported in part by the National Science Council, Taiwan, ROC, under the contract NSC 97-2218-E-147-001. The authors are grateful to the anonymous referees for their constructive comments that have greatly improved the presentation of this paper.

References

1. Yu L, Shih HM, Pfund M, Carlyle WM, Fowler JW (2002) Scheduling of unrelated parallel machines: an application to PWB manufacturing. IIE Trans 34:921–931
2. Hsieh JC, Chang PC, Hsu LC (2003) Scheduling of drilling operations in printed circuit board factory. Comp Industr Engin 44:461–473, doi:10.1016/S0360-8352(02)00231-0
3. Kim DW, Kim KH, Jang W, Chen FF (2002) Unrelated parallel machine scheduling with setup times using simulated annealing. Robo Comput-Integr Manuf 18:223–231, doi:10.1016/S0736-5845(02)00013-3
4. Piersma N, van Dijk W (1996) A local search heuristic for unrelated parallel machine scheduling with efficient neighborhood search. Math Comput Model 24:11–19, doi:10.1016/0895-7177(96)00150-1
5. Ghirardi M, Potts CN (2005) Makespan minimization for scheduling unrelated parallel machines: A recovering beam search approach. Eur J Oper Res 165:457–467, doi:10.1016/j.ejor.2004.04.015
6. Chen JF (2005) Unrelated parallel machine scheduling with secondary resource constraints. Int J Adv Manuf Technol 26:285–292, doi:10.1007/s00170-003-1622-1
7. Weng MX, Lu J, Ren H (2001) Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. Int J Prod Econ 70:215–226, doi:10.1016/S0925-5273(00)00066-9
8. Suresh V, Chaudhuri D (1994) Minimizing maximum tardiness for unrelated parallel machines. Int J Prod Econ 34:223–229, doi:10.1016/0925-5273(94)90038-8
9. Chen JF (2006) Minimization of maximum tardiness on unrelated parallel machines with process restrictions and setups. Int J Adv Manuf Technol 29:557–563
10. Guinet A (1995) Scheduling independent jobs on uniform parallel machines to minimize tardiness criteria. J Intell Manuf 6:95–103, doi:10.1007/BF00123681
11. Randhawa SU, Kuo CH (1997) Evaluating scheduling heuristics for non-identical parallel processors. Int J Prod Res 35:969–981, doi:10.1080/002075497195489
12. Ho JC, Chang YL (1995) Minimizing the number of tardy jobs for m-parallel machines. Eur J Oper Res 84:343–355, doi:10.1016/0377–2217(93)E0280-B
13. Lawler EL, Moore JM (1969) A functional equation and its application to resource allocation and sequencing problems. Manage Sci 16:77–84
14. Ruiz-Torres AJ, Lopez FJ, Ho JC (2007) Scheduling uniform parallel machines subject to a secondary resource to minimize the number of tardy jobs. Eur J Oper Res 179:302–315, doi:10.1016/j.ejor.2006.03.028
15. M'Hallah R, Bulfin RL (2005) Minimizing the weighted number of tardy jobs on parallel processors. Eur J Oper Res 160:471–484, doi:10.1016/j.ejor.2003.06.027
16. Mladenović N, Hansen P (1997) Variable neighborhood search. Comput Oper Res 24:1097–1100, doi:10.1016/S0305–0548(97)00031–2
17. Hansen P, Mladenović N (2001) Variable neighborhood search: principles and applications. Eur J Oper Res 130:449–467, doi:10.1016/S0377–2217(00)00100–4
18. Ribeiro CC, Souza MC (2002) Variable neighborhood search for the degree-constrained minimum spanning tree problem. Discrete Appl Math 118:43–54, doi:10.1016/S0166-218X(01)00255-4
19. Moreno-Pérez JA, Moreno-Vega JM, Martín IR (2003) Variable neighborhood tabu search and its application to the median cycle problem. Eur J Oper Res 151:365–378, doi:10.1016/S0377-2217(02)00831-7
20. Bräysy O (2003) A reactive variable neighborhood search for the vehicle-routing problem with time windows. INFORMS J Comput 15:347–368, doi:10.1287/ijoc.15.4.347.24896
21. Polacek M, Hartl RF, Doerner K (2004) A variable neighborhood search for the multi depot vehicle routing problem with time windows. J Heuristics 10:613–627, doi:10.1007/s10732–005–5432–5
22. Hansen P, Mladenović N, Urošević D (2004) Variable neighborhood search for the maximum clique. Discrete Appl Math 145:117–125, doi:10.1016/j.dam.2003.09.012
23. Fleszar K, Hindi KS (2004) Solving the resource-constrained project scheduling problem by a variable neighbourhood search. Eur J Oper Res 155:402–413, doi:10.1016/S0377-2217(02)00884-6
24. Davidović T, Hansen P, Mladenović N (2005) Permutation based genetic, tabu and variable neighborhood search heuristics for multiprocessor scheduling with communication delays. Asia-Pacific J Oper Res 22:297–326, doi:.1142/S021759590500056X
25. Glover F (1989) Tabu search, part I. ORSA J Comp 1:190–206
26. Armentano VA, Yamashita DS (2000) Tabu search for scheduling on identical parallel machines to minimize mean tardiness. J Intellig Manufact 11:453–460, doi:10.1023/A:1008918229511
27. Kim CO, Shin HJ (2003) Scheduling jobs on parallel machines: a restricted tabu search approach. Int J Adv Manuf Technol 22:278–287, doi:10.1007/s00170-002-1472-2
28. Bilge Ü, Kıraç F, Kurtulan M, Pekgün P (2004) A tabu search algorithm for parallel machine total tardiness problem. Comput Oper Res 31:397–414, doi:10.1016/S0305-0548(02)00198-3

29. Potts CN, Van Wassenhove LN (1982) A decomposition algorithm for the single machine total tardiness problem. Ops Res Lett 1:177–181, doi:10.1016/0167-6377(82)90035-9
30. Rios-Mercado RZ, Bard JF (1998) Heuristics for the flow line problem with setup costs. Eur J Oper Res 110:76–98, doi:10.1016/S0377-2217(97)00213-0
31. Lee GC, Kim YD, Choi SW (2004) Bottleneck-focused scheduling for a hybrid flowshop. Int J Prod Res 42:165–181, doi:10.1080/00207540310001602892
32. Choi SW, Kim YD, Lee GC (2005) Minimizing total tardiness of orders with reentrant lots in a hybrid flowshop. Int J Prod Res 43:2149–2167, doi:10.1080/00207540500050071