Engineering Journal of the University of Qatar, Vol. 15, 2002, pp. 193-210

UNRELATED MACHINES SCHEDULING WITH MACHINE ELIGIBILITY RESTRICTIONS

Ameer Al-Salem
Mechanical Engineering Department
University of Qatar
P.O. Box 2731
Doha, Qatar
Email: ameer@qu.edu.qa

Robert L. Armacost
Department of Industrial Engineering and Management Systems
University of Central Florida
P.O Box 162450
Orlando, FL 32816-2450

ABSTRACT

In this paper we present a new heuristic algorithm to minimize the makespan for scheduling jobs on unrelated parallel machines with machine eligibility restrictions ($R_m / M_j / C_{\max}$). To the best of our knowledge, the problem has not been addressed previously in the literature. The multi-phase heuristic algorithm incorporates new concepts from the multi-depot vehicle routing in the constructive heuristic. A computational study includes problems with two or four machines, up to 105 jobs, and three levels of a machine selection parameter. The heuristic algorithm solution values are compared to optimal solution values. The results show that the heuristic algorithm can yield solutions within a few percent of the optimal solutions with performance improving as the number of jobs to be scheduled increases.

KEY WORDS: Scheduling, makespan, unrelated machines, machine eligibility.

NOMENCLATURE

m	total number of machines
R_m	m unrelated machines in parallel
M_j	the set of machines that can process job j
m_j	the number of machines that belong to M_j
P_{jk}	processing time of job j on machine k
$C_{partial(k)}$	the partial makespan to date on machine k
$C^*_{Partial}$	the maximum partial makespan and equal to $\max_{k=1,\ldots,m} \{ C_{partial(k)} \}$
PJ	the set of jobs that have been considered for assignment and not yet assigned (pending jobs)
A_k	the set of jobs assigned to machine k
ρ_j	machine selection ratio for job j
α	machine selection parameter
$\min_k^{(1)}(P_{jk})$	the minimum machine processing time for job j on an eligible machine, $k \in M_j$
$\min_k^{(2)}(P_{jk})$	the second shortest machine processing time for job j on an eligible machine, $k \in M_j$

1. INTRODUCTION

Scheduling jobs on unrelated parallel machines is a common problem in scheduling. As a company expands, it frequently requires new machines that typically process jobs faster. The old machines are retained and operated in parallel with the newer machines. The jobs to be processed may have different processing times on the different machines. Additionally, some jobs may have to be processed on the newer machines, some may be able to be processed only on the old machines, and others may be processed on all machines. This limitation is called machine eligibility restrictions. If the objective to be achieved is to minimize the completion time of the last job to be processed, i.e., to minimize the makespan, it is necessary to schedule the jobs to particular machines to be processed rather than a random assignment. In this paper, we use a manufacturing concept (machines and jobs) but the problem has a broader industrial engineering approach (e.g., clerks, tasks).

The problem of scheduling jobs on unrelated machines in parallel is a generalization of both the uniform and the identical parallel machine scheduling problems. The differences are dependent on processing times. The processing time of job j on machine k is equal to P_{jk} for unrelated machines, P_j/V_k for uniform machines where V_k is the speed factor for machine k, and P_j for identical machines. The specific problem that is considered in this paper involves scheduling unrelated machines when machine eligibility restrictions are present. In particular, there are n jobs to be scheduled without preemption on m unrelated machines in parallel. Each job is to be assigned to a machine and each machine can process at most one job at any time. However, not all of the m machines are capable of processing each job. For each job j, there is a set of machines M_j ($j = 1,\ldots,n$) capable of processing that job. This is the machine eligibility restriction. It is assumed that job j becomes available for processing at time zero and requires a positive integer processing time P_{jk} if it is assigned to machine k ($k = 1,\ldots,m$). The objective is to schedule the jobs so that the maximum completion time, $C_{\max}$, is minimized.

The scheduling problem is easily formulated as a zero-one integer program. Specifically, the problem is to

$$\text{Minimize} \quad C_{\max} \qquad \text{[A]}$$

Subject to:

$$C_{\max} - \sum_{j=1}^{n} P_{jk} X_{jk} \geq 0 \qquad \forall\, k \in M_j$$

$$\sum_{k \in M_j} X_{jk} = 1 \qquad 1 \leq j \leq n$$

$X_{jk} = 1$ if job j is assigned to machine k;

0 otherwise $\quad 1 \leq j \leq n$ and $k \in M_j$

When problem sizes are very small, a manual solution is possible. However, for any practical problems, a computer-based algorithm is required. When the problem sizes are "small enough," an optimization algorithm can be used to obtain an optimal solution to problem A. Many of real world problems exceed this capacity and require the use of computer-based heuristic algorithm to obtain a "good" solution.

The purpose of this paper is to present a new heuristic algorithm to find a good, quick solution to the unrelated parallel machine problem with machine eligibility restrictions where the objective is to minimize the makespan. The new heuristic algorithm that is used to solve the problem uses a constructive heuristic to assign jobs to machines followed by an improvement heuristic to modify and improve the solution. The paper is organized as follows: The second includes a brief discussion of existing scheduling approaches that bear on the problem of unrelated machine scheduling with machine eligibility restrictions. The third section includes a description of the new multi-phase heuristic algorithm and the fourth section illustrates computational studies to evaluate the performance of the new heuristic algorithm. The final section includes conclusions and directions for further research.

2. RESEARCH PERTANING TO UNRELATED MACHINE SCHEDULING TO MINIMIZE MAKESPAN

There is an extensive literature on parallel machine scheduling that was reviewed by Cheng and Sin [3]. Many authors assume that machines are identical so that the processing time of a job dose not depend on the machine to which it is assigned. Garey and Johnson [6] showed that the problem of scheduling identical parallel machines to minimize makespan is NP-hard even for two machines. Clearly, the problem considered in this paper is also NP-hard. Therefore, the existence of a polynomial time algorithm is highly unlikely. Consequently, for parallel machine scheduling problems of this type, most researchers have studied heuristic methods which provide an approximate solution.

Many of the research results address the unrelated parallel machine scheduling problem without machine eligibility restrictions ($R_m // C_{\max}$). Horowitz and Sahni [8] developed a heuristic that is similar to Sahni's [13] for identical machines. The earliest completion time heuristic (ECT) was presented by Ibarra and Kim [9] who also developed another four heuristics that are based on ECT. The earliest completion time heuristic (ECT) is similar to the Longest Processing Time first (LPT) rule that is considered a good heuristic to minimize the makespan on identical parallel machines. The LPT rule orders the jobs by decreasing order of processing time and assigns them to machine that results in lowest partial makespan. The ECT heuristic, however, assigns at $t = 0$ the m largest average processing time (over the machine) jobs to m machines. After that, whenever a machine is free, the unscheduled largest average processing time job is put on that machine. This heuristic tries to place the shortest average processing time jobs toward the end of the schedule where they can be used for balancing the loads. For more approximation algorithms for unrelated parallel machines without machine eligibility restrictions see Davis and Jaffe [4], De and Morton [5], Potts [12], and Hariri and Potts [7]. For exact and approximation algorithms see ven de Velde [15] and Martello, Soumis and Toth [10].

In general, there has been little treatment of machine eligibility restrictions in the literature. When machine eligibility restrictions (M_j) are involved, Pinedo [11] showed that the least flexible job first (LFJ) rule is optimal for (P_m / $P_j = 1$, M_j / C_{max}) when the M_j sets are *nested.* Centeno and Armacost [2] considered machine eligibility restrictions when machines are identical and proposed an algorithm to minimize maximum lateness on identical parallel machines with release dates and machine eligibility restrictions for the special case where due dates are equal to release date plus a constant. Centeno [1] introduced various heuristic algorithms to minimize makespan and minimize maximum lateness on identical parallel machines with machine eligibility restrictions. Lately, Weng et al. [16] addressed the unrelated parallel machine scheduling problem with setup times where they introduced seven heuristic algorithms for the problem, but the objective was to reduce the jobs' weighted mean completion times.

To date, there have not been any reported results that consider the problem of minimizing makespan on unrelated parallel machines with machine eligibility restrictions ($R_m / M_j / C_{\max}$). The heuristic algorithm described in the following section extends the multi-phase heuristic for unrelated parallel machine scheduling that was developed by the authors to accommodate machine eligibility restrictions.

3. MULTI-PHASE HEURISITC ALGORITHM

The multi-phase heuristic algorithm uses a constructive heuristic followed by an improvement heuristic to solve the problem of minimizing makespan on unrelated parallel machines with machine eligibility restrictions. The phase 1 constructive heuristic assigns jobs to machines and the phase 2 improvement heuristic improves the previous solution. The phase 1 constructive heuristic to assign jobs to machines is modeled based on a concept used by Salhi and Sari [14] to solve the multi-depot vehicle routing problem (MDVRP) by making an initial assignment of selected jobs and then assigning the remaining pending jobs.

Phase 1—Constructive Heuristic

The constructive assignment heuristic proceeds in two stages. All assignment are made in order to satisfy the machine eligibility restrictions, generally assigning jobs in increasing order of the values of m_j. The first stage involves making the "obvious" assignment for jobs that can be processed on more than one machine. Specifically, for each job, the machine with the shortest processing time and the machine with second shortest processing time for that job are identified. If the ratio of the shortest time to the second shortest time is small enough, the job is assigned to the machine with the shortest processing time. Otherwise the job is considered "pending" and is assigned in the following step.

In stage 2, pending jobs are ordered by decreasing order of average processing time and assigned to machines so that the partial makespan is minimized over all machines. At the conclusion of stage 2, all jobs are assigned to machines. This approach follows proven concepts of scheduling the most restrictive jobs first (e.g., LPT).

The two stages of the constructive heuristic to assign jobs to machines are described as follows.

Stage 1—Initial job assignment to machine.
If a job can be processed by only one machine, the job is assigned to that machine. For each job j ($j = 1,\ldots,n$), the ratio (ρ_j) of the minimum machine processing time to the

second shortest machine processing time is computed. The ratio is compared with a machine selection parameter α ($0 \leq \alpha \leq 1$). If ρ_j is smaller than α, the job is assigned to the machine with shortest processing time. If ρ_j is larger than α, the job is considered pending and is assigned to a machine in stage 2. The following procedure is used to determine the pending jobs and assign the remaining jobs:

0. $PJ = \varnothing$; $C_{partial(k)} = 0$, $k = 1,\ldots,m$
1. For $j = 1$ to n DO:
2. If $|M_j| = 1$, let the corresponding $k \in M_j$ be $k(j)$. Assign job j to machine $k(j)$.
   Update $C_{partial(k(j))} = C_{partial(k(j))} + P_{jk(j)}$
3. If $|M_j| > 1$, compute $\rho_j = \dfrac{\min\limits_{k \in M_j}{}^{(1)} (P_{jk})}{\min\limits_{k \in M_j}{}^{(2)} (P_{jk})}$

   ~~(a)~~(i) If $\rho_j \leq \alpha$, let the k corresponding $\min^{(1)}$ be $k(j)$. Assign job j to machine $k(j)$.
   Update $C_{partial(k(j))} = C_{partial(k(j))} + P_{jk(j)}$

   ~~(b)~~(ii) If $\rho_j > \alpha$, job j is considered pending. Update $PJ = PJ \cup \{j\}$

END

Step 2--Assignment of pending jobs to machines:

Pending jobs are arranged into groups of jobs that can be processed on the same number of machines. The groups are ordered by increasing order of the number of machines that can process the jobs in each group. The jobs in each group are then ordered by decreasing values of the average (over the machines) of the processing time. Following that order, the pending jobs are assigned based on minimizing the partial makespan on each machine. The following steps describe the stage 2 algorithm.

1. Order jobs by increasing values of m_j, $\forall j \in$ PJ
2. Compute $\bar{p}_j = \dfrac{1}{m_j} \sum\limits_{k \in M_j} (P_{jk})$, $\forall j \in$ PJ

3. For l= 2 to m in the ordered set in step 1, DO:

   Order jobs by decreasing values of $\bar{p}_j$ for $m_j = l,\ \forall j \in$ PJ

4. Following this nested order, for job j

   Find $\min_{k \in M_j} \{ C_{partial(k)} + P_{jk} \}$. Let the corresponding k be $k(j)$

   Assign job j to machine $k(j)$. Update $C_{partial(k(j))} = C_{partial(k(j))} + P_{jk(j)}$

END

Phase 2—Improvement Heuristic

The result of phase 1 is an assignment of all jobs to machines. The makespan is the largest of the partial makespans developed in phase 1. Specifically,

$$C_{\max} = \max_k (C_{partial(k)}),\ \ k = 1,\dots,m$$

In order to reduce the makespan, an improvement heuristic can be applied to the phase 1 schedule. In this application, the composite exchange heuristic developed by Harriri and Potts [7] is modified to account for machine eligibility restrictions and applied to the phase 1 schedule. The modified composite exchange heuristic consists of two stages. In the first stage, a job is removed from the machine that produces the largest makespan and is assigned to a machine that is capable of processing that job and produces the lowest makespan. All jobs and all possible assignments are considered. This is called a one to zero exchange. In the second stage, two jobs are exchanged, one from the machine that produces the largest makespan and one from the machine that produces the lowest makespan. This is called a one to one exchange and all jobs are considered for the two machines. The first stage is applied first using the constructive heuristic schedule as input. Then using the resulting improved schedule as input to the second stage, a further reduction in makespan is attempted. The procedure continues by repeatedly applying the first stage and the second stage until no further reduction in makespan is possible.

The one-to-zero exchange proceeds as follows:

Stage 1—One-to-zero exchange

1. Select any machine k for which $C_{partial(k)} = C^*_{partial}$.
2. Search for job j, where $j \in A_k$ and $l \in M_j$, such that $C_{partial(l)} + P_{jl} < C^*_{partial}$ for some machine l ($l \neq k$).
3. If no such job j and machine l are found, then $C^*_{partial}$ is the final solution.
4. If j and l are found, then job j is assigned to machine l.
5. Update $A_l = A_l \cup \{j\}$, $C_{parial(l)} = C_{parial(l)} + P_{jl}$ and $A_k = A_k - \{j\}$.

The entire procedure is repeated searching for all one to zero exchanges until no further reduction in the maximum completion time is possible.

Stage 2—One-to-one exchange

1. Select any machine k such that $C_{partial(k)} = C^*_{partial}$.
2. Search for jobs j and i where $j \in A_k$, $i \in A_l$, and $l \in M_j$ for some machine l ($l \neq k$), for which $C_{partial(k)} - P_{jk} + P_{ik} < C^*_{partial}$ and $C_{partial(l)} - P_{il} + P_{jl} < C^*_{partial}$.
3. If jobs j and i and machine l cannot be found, then $C^*_{partial}$ is the final solution.
4. If jobs j and i and machine l are found, then j and i are interchanged.
5. Update $A_l = A_l \cup \{j\} - \{i\}$, $C_{partial(l)} = C_{prtial(l)} + P_{jl} - P_{il}$, $A_k = A_k \cup \{i\}$-$\{j\}$, $C_{parial(k)} = C_{partial(k)} + P_{ik} - P_{jk}$, and $C^*_{partial} = \max\{C_{partial(k)}, C_{partial(l)}\}$.

The entire procedure is repeated searching for all possible one to one exchanges until no further reduction in the maximum completion time is possible. If any exchanges are made in stage 2, stage 1 is repeated. If additional exchanges are made in stage 1, stage 2 is repeated. The process continues until no further exchanges are realized in one stage.

4. COMPUTATIONAL STUDIES

The new heuristic is based on concepts that have worked well in other settings (e.g., MDVRP, LPT). A computational evaluation is designed to assess the practicality and usefulness of the new approach in this expanded context for unrelated machine scheduling with machine eligibility restrictions. The new heuristic algorithm was coded in Borland C++ and implemented on a Pentium III 500 personal computer. The purpose of this evaluation is to determine how well the heuristic performs with respect to obtaining an optimal solution. The factors considered are the number of jobs, the number of machines and the machine selection parameter (α) that was used in stage 1 of the phase 1 constructive heuristic. The computational study examines the effect of these three factors on the heuristic performance.

In order to measure the performance of the heuristic, the value of the heuristic solution (makespan) is compared to the value of the optimal solution for each problem instance. Optimal solutions are obtained by using AMPLE software as a modeling language and CPLEX 6.0 as a solver.

Let Z_{opt} denote the percent deviation of the heuristic solution value above the optimal solution value. Then the measure of performance for each problem instance is

$$Z_{opt} = \frac{Heuristic\ Solution\ Makespan - Optimal\ Solution\ Makespan}{Optimal\ Solution\ Makespan} \times 100$$

Experimental Design

Three factors are considered. The evaluation was conducted as a full factorial (7x2x3) experiment. The factors and their levels are as follows:

Number of jobs (n)	15, 30, 45, 60, 75, 90, 105
Number of machines (m)	2, 4
Machine selection parameters (α)	0.5, 0.7, 0.9

Fifteen replications were conducted for each factor combination. A total of 630 problem instances were examined.

To account for machine eligibility restrictions, jobs were assigned equally to each possible grouping of machines. For the 2 machine problem, possible groupings are machine 1, machine 2, and machines 1 and 2. With 15 jobs, five would be randomly assigned to each group and the machine eligibility sets created. For the 4 machine problem, there are 15 groups: four singletons, six pairs, four triples, and one group with all four machines. When there are 15 jobs, one is assigned to each group, when there are 30 jobs, 2 are assigned to each group and so on.

The processing times for each job on each machine are uniformly distributed on the interval [50, 100].

Computational Results

The summery results for the experiment included in Table 1 show that the overall average of the percentage deviation between the heuristic solution value and the optimal solution value is equal to 1.027, 0.833, and 0.640 when $\alpha = 0.5$, 0.7, and 0.9 respectively. These compare with 24.48% for the random assignment comparison (see Table 1). The random assignment solutions are obtained by assigning jobs to machines randomly. One random assignment is used for each problem instance. This comparison clearly indicates that the new heuristic algorithm provides solutions that are much better than a random assignment and are close to the optimal value. The data in Table 1 are averages for the 15 replications for each factor combination.

Table 1. Computational results: Average percentage deviation above optimal

			Heuristic algorithm results			*Random assignment results*
Test #	*Machines*	*Jobs*	$\alpha = 0.5$	$\alpha = 0.7$	$\alpha = 0.9$	
1	2	15	0.046	0.046	0.046	18.82
2	2	30	0.187	0.082	0.425	12.51
3	2	45	0.091	0.143	0.072	10.03
4	2	60	0.209	0.214	0.077	12.66
5	2	75	0.096	0.027	0.070	7.63
6	2	90	0.077	0.159	0.113	8.33
7	2	105	0.098	0.111	0.087	8.76
8	4	15	3.324	2.518	2.703	62.19
9	4	30	3.176	2.402	1.691	42.43
10	4	45	1.452	1.261	1.309	36.59
11	4	60	1.654	1.196	0.601	32.33
12	4	75	1.296	1.378	0.687	30.69
13	4	90	1.391	1.110	0.630	29.97
14	4	105	1.276	1.010	0.447	29.91
	Average		1.027	0.833	0.640	24.48

The results in Table 1 suggest that the average performance (over the 14 condition combinations) improves as the value of the machine selection parameter increases. The average performance data are plotted in Figure 1.

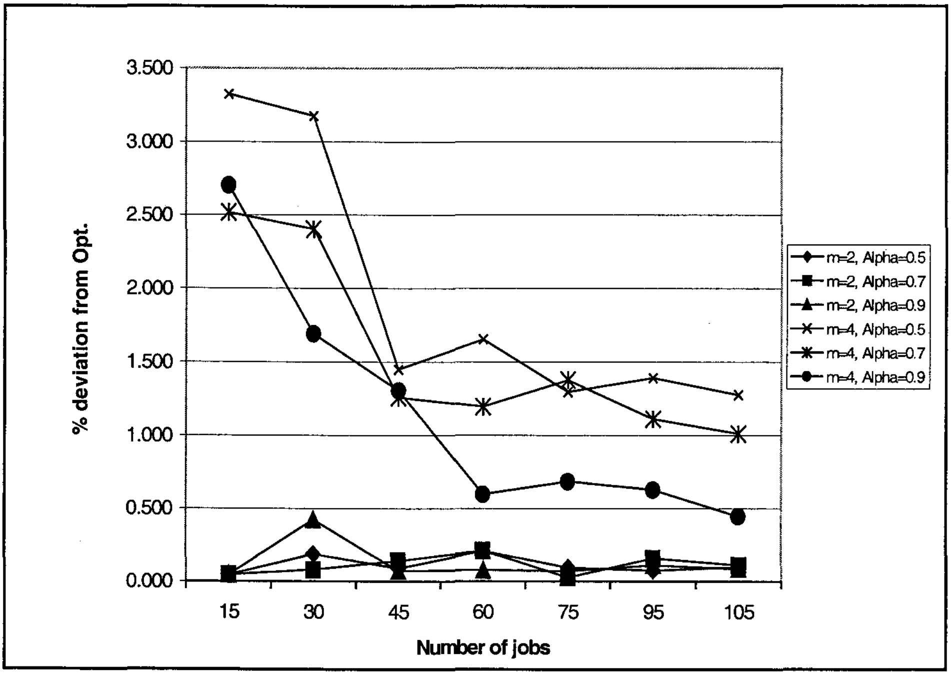

Fig. 1. Comparative performance-% deviation above optimal

It is obvious from Figure 1 that the performance improves as the number of jobs increases when the number of machines is at high level (*m*=4). When the number of machines is at low level (*m*=2), performance seems to be almost the same as the number of jobs increases. Figure 1 also illustrates the performance improvement as the machine selection parameter value increases when the number of machines is at the high level.

In this computational study, the new heuristic algorithm obtained optimal solution values 227 times out of 630 generated test problems, 35 percent of the cases (see Table

2). Table 2 shows that the overall best percent deviation is 0% and the overall worst percent deviation is 10.9%.

Table 2 Heuristic algorithm performance

Test #	Machines	Jobs	# problem instances	# optimal solutions	Best deviation	Worst deviation
1	2	15	45	41	0	0.69
2	2	30	45	37	0	1.99
3	2	45	45	30	0	0.84
4	2	60	45	30	0	1.26
5	2	75	45	30	0	0.82
6	2	90	45	18	0	0.57
7	2	105	45	18	0	0.47
8	4	15	45	17	0	10.9
9	4	30	45	2	0	8.53
10	4	45	45	3	0	5.31
11	4	60	45	1	0	3.72
12	4	75	45	0	0.08	3.51
13	4	90	45	0	0.13	3.56
14	4	105	45	0	0.22	2.13

Table 1 and Figure 1 suggest that the performance improves as the number of jobs increases when there are four machines, and that larger values of the machine selection parameter provide better solutions when there are four machines. An analysis of variance was conducted (up to two-way interactions) using the JMP statistical software. The ANOVA results along with the effect test results are included in Table 3. Figure 2 includes the pairwise interaction graphs.

Table 3 Statistical analysis for the proposed heuristic

Analysis of Variance

Source	*DF*	*Sum of Squares*	*Mean Square*	*F Ratio*	Prob>F
Model	29	517.9043	17.8588	13.9114	<.0001
Error	600	770.2484	1.2837		
C Total	629	1288.153			

Effect Test

Source	*DF*	*Sum of Squares*	*F Ratio*	*Prob>F*
Machines	1	321.362	250.3312	<.0001
Jobs	6	80.29262	10.4242	<.0001
Alpha	2	15.48498	6.0312	0.0026
Machines & Jobs	6	78.38573	10.1767	<.0001
Machines & Alpha	2	16.96857	6.609	0.0014
Jobs & Alpha	12	5.4104	0.3512	0.9788

Y= percent deviation from optimal

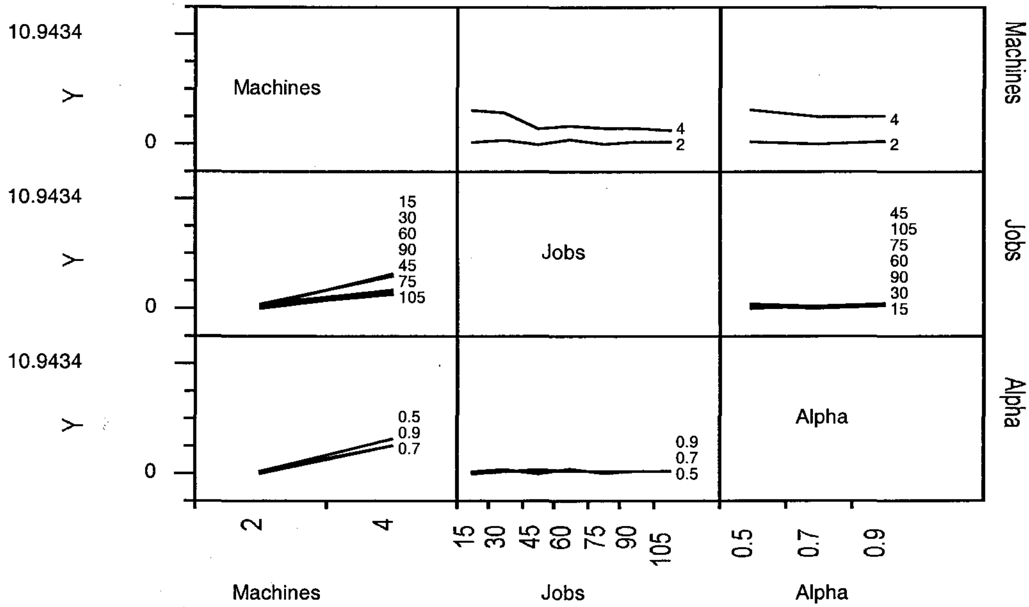

Fig. 2 Interaction plots

The effect tests show that the machines and jobs, and the machines and machine selection parameters pairwise interaction are significant, thereby precluding meaningful conclusions about the main effects. Nevertheless, the interaction plots suggest the observation in Figure 1 that larger value of the machine selection parameter seem to yield better solutions when there are four machines over all jobs levels. Figure 1 also suggests the significant performance for the two machine levels and for the number of jobs for the four-machine case.

5. CONCLUSIONS

Machine eligibility constraints are a common occurrence, but there has been little empirical work to evaluate heuristics for this domain. The new heuristic algorithm to minimize makespan on unrelated parallel machines with machine eligibility restrictions presented in the paper is one of the first empirical treatments of this subject. The algorithm uses a new approach adapted from a new multi-depot vehicle routing heuristic, integrates the approach with a proven improvement heuristic.

The new heuristic was tested on problems with 2 and 4 machines and 15, 30, 45, 60, 75, 90, and 105 jobs. The performance of the heuristic algorithm, measured as the percentage deviation between heuristic solution values and optimal solution values, was very satisfactory, yielding solution makespan within a few percentage points of the optimal solution values. The overall average of the percentage deviation between the heuristic solution values and the optimal solution values is equal to 1.027%, 0.833%, and 0.640% for the three increasing values of the machine selection parameter.

The analysis for the computational results shows that algorithm performance improves as the number of jobs increases, and performance improves with larger values of the machine selection parameter as the number of machines increases. This result provides general guidance for parameter selection in a given implementation.

The new heuristic algorithm is effective in generating good "solutions" for unrelated machine scheduling problems with machine eligibility restrictions. The computational study involved problem sizes for which the optimal solutions could be obtained. The next step is to examine the algorithm performance on problems with a larger number of jobs and larger number of machines. Because these problems are not easily solvable to optimality, it well be necessary to develop reasonable bounds on performance and compare the actual performance with these bounds.

REFERENCES

1. **Centeno, G., 1998**. Parallel machine scheduling with specified release dates, due dates, and machine eligibility restrictions for minimizing the maximum lateness. Ph.D. Dissertation, University of Central Florida.

2. **Centeno, G.** and **Armacost, R., 1997**, "Parallel machine scheduling with release time and machine eligibility restrictions," Computers and Industrial Engineering, Vol. 33, pp. 273-276.

3. **Cheng, T.** and **Sin, C., 1990,** "A state-of-the-art review of parallel-machine scheduling research," European Journal of Operation Research, Vol. 47, pp. 271-292.

4. **Davis, E.** and **Jaffe, J.M., 1981**, "Algorithms for scheduling tasks on unrelated processors," Journal of the Association for Computing Machinery, Vol. 28, pp. 721-736.

5. **De, P.** and **Morton, T., 1980**, "Scheduling to minimize makespan on unrelated processors," Decision Sciences, Vol. 11, pp. 586-602.

6. **Garey, M. R.** and **Johnson, D. S, 1979,** Computers and intractability: A guide to the theory of NP-completeness, W. H. Freeman and Company.

7. **Hariri, A.M.A.** and **Potts, C.N., 1991,** "Heuristics for scheduling unrelated parallel machines," Computers and Operations Research, Vol. 18, pp. 313-321.

8. **Horowitz, E.** and **Sahni, S, 1976,** "Exact and approximate algorithms for scheduling nonidentical processors," Journal of the Association for Computing Machinery, Vol. 23, pp. 317-327.

9. **Ibarra, O.** and **Kim, C., 1977,** "Heuristic algorithms for scheduling independent tasks on nonidentical processors," Journal of the Association for Computing Machinery, Vol. 24, pp. 280-289.

10. **Martello, S., Soumis, F.,** and **Toth, P. 1997,** " Exact and approximation algorithms for makespan minimization on unrelated parallel machines," Discrete Applied Mathematics, Vol. 75, pp. 169-188.

11. **Pinedo, M., 1995,** Scheduling: Theory Algorithm, and Systems, Prentice-Hall, Englewood Cliffs, New Jersey.

12. **Potts, C.N., 1985,** "Analysis of a linear programming heuristic for scheduling unrelated parallel machines," Discrete Applied Mathematics, Vol. 10, pp. 155-164.

13. **Sahni, S., 1976,** "Algorithms for scheduling with independent tasks," Journal of the Association for Computing Machinery, Vol. 23, pp. 116-127.

14. **Salhi, S.** and **Sari, M., 1997,** "A multi-level composite heuristic for the multi-depot vehicle fleet mix problem," European Journal of Operation Research, Vol. 103, pp. 95-112.

15. **Ven de Velde, S, 1993,** " Duality-based algorithm for scheduling unrelated parallel machines," ORSA Journal of Computing, Vol. 5, pp. 192-205.

16. **Weng, M.X, Lu, J.,** and **Ren, H., 2001**, "Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective," International Journal of Production Economics, Vol. 70, pp. 215-226.