academicJournals

Vol. 7(16), pp. 1553-1559, 28 April, 2013
DOI: 10.5897/AJBM2013.1593
ISSN 1993-8233 © 2013 Academic Journals
http://www.academicjournals.org/AJBM

African Journal of Business Management

Full Length Research Paper

Strategies for scheduling jobs in an identical parallel machine production environment

Maciel Manoel Queiroz* and André Bergsten Mendes

Department of Naval Architecture and Ocean Engineering - Escola Politécnica da Universidade de São Paulo. Av. Prof. Mello Moraes, 2231, Cidade Universitária, 05508-030, São Paulo, SP - Brazil.

Accepted 11 April, 2013

This paper addresses an identical parallel machine environment present in industry. In such an environment, no machine can execute more than one job at the same time. Each job is characterised by a release date that reflects the instant that the processing is able to start. There are no compatibility constraints between job and machine, so that each machine can perform any job. There are no precedence constraints or setup times between jobs. Whenever a job is scheduled after its due date, a penalty is incurred, reflecting the postponement. The scheduling of identical parallel machines has been proven to be important from both theoretical and practical points of view. In this paper, a metaheuristic based on GRASP with path relinking using a multi-threading approach is addressed. Computational experiments were conducted to compare metaheuristic solutions with lower bounds provided by a branch and bound algorithm. The proposed method is shown to be a competitive and effective solution strategy for production environments.

Key words: GRASP, path relinking, parallel machine, jobs, schedule, metaheuristic.

INTRODUCTION

Scheduling is a challenge in various production environments and always involves a number of activities that need to be processed using some available resources over a certain period of time (Morton and Pentico, 1993). The scheduling of activities is a decision process that plays an important role in most manufacturing systems as well as production and information-processing environments and transportation, distribution and other types of industries (Pinedo, 2002).

This paper considers the identical parallel machine scheduling problem with release dates in which the total weighted tardiness has to be minimised. Each job j has a release date denoted by r_j, a processing time p_j, a due date d_j and a weight w_j. Pre-emption is not allowed, and a job cannot start before its release date. No machine can execute more than one job at the same time. In a schedule where job j is completed at time C_j, its tardiness is expressed by $T_j = max(0, C_j - d_j)$. This problem can be written, utilising the classical notation, as $P_m|r_j|\sum w_j T_j$. The problem is NP-hard (Lenstra et al. 1977). The remainder of this paper is organised as follows. In the next section, a review of the relevant literature is provided. The proposed solution methods are then detailed. Computational experiments are presented in sequence. The last section concludes the paper and discuss possible extensions.

LITERATURE REVIEW

The parallel machine environment has been studied for several years because of its importance to academia and industry. Horowitz and Sahni (1976) presented exact and approximate algorithms for parallel machines with the

*Corresponding author. E-mail: maciel.queiroz@usp.br. Tel: +55 11 9 9778 8893.

objective of minimising the completion time of jobs and the weighted mean flow time.

Fuller (1978) made a comparison between optimal solutions and good solutions, analysing the effectiveness of heuristics in the decision-making process. The author observed that heuristics simplify the process for the decision maker, allowing decisions to be made quickly. The advantage of using heuristics is that the methods limit the search by reducing the number of alternatives.

Total weighted tardiness is one of the hardest criteria in the parallel machine environment. Liaw et al. (2003) addressed the problem of minimising the total weighted tardiness of unrelated parallel machines, showed the properties of an optimal schedule and also proposed a branch and bound algorithm. Alidaee and Rosa (1997) highlighted a case of identical parallel machines in minimising the total weighted tardiness using the modified due date (MDD) heuristic.

Xing and Zhang (2000) examined an identical parallel machine environment with independent job setup times and a makespan objective function and applied an ML heuristic that worked for their worst-case performance ratio within 7/4-1/m(m≥2). ML heuristics try to convert the original problem into a new problem by using the estimated maximum completion time.

Mokotoff (2004) analysed an identical parallel machine problem involving makespan minimisation with linear programming relaxations and constructive rules. Dominance rules are important in developing insights and new approaches. Jouglet and Carlier (2011) highlighted the importance of dominance rules in reducing the search space in combinatorial optimisation. Jouglet and Savourey (2011) and Yalaoui and Chu (2002) identified some dominance properties related to minimising total tardiness in identical parallel machine scheduling problems.

Shim and Kim (2007) addressed an identical parallel machine problem with a total tardiness objective function. The authors presented some dominance properties and other techniques such as branch and bound. Leung and Li (2008) addressed a parallel scheduling problem with processing set restrictions applied in a parallel machine environment. The authors considered some performance criteria, such as C_{max} and L_{max}.

Nessah et al. (2008) addressed an identical parallel machine problem with release dates with a total weighted completion time objective function. The authors also developed some dominance properties. Tanaka and Araki (2008) developed a branch and bound algorithm to solve identical parallel machine problems with total tardiness objective functions. A Lagrangian relaxation was developed for a lower bound.

Some authors have explored approaches involving the use of metaheuristics. Min and Cheng (1999) applied a genetic algorithm to an identical parallel machine problem with a makespan objective function. Resende and Werneck (2004) presented the evolutionary path relinking (EvPR) approach. Chiang et al. (2010) applied a memetic algorithm to the solution of a total weighted tardiness objective function for an identical parallel batch machine scheduling problem with incompatible job families whose jobs arrived dynamically.

Bilge et al. (2007) applied a tabu metaheuristic to a single machine, total weighted tardiness problem with due dates. Koulamas (1997) presented a polynomial decomposition of the identical parallel machine environment with a total tardiness objective function in which metaheuristic simulated annealing was employed. Cao et al. (2005) addressed a problem of simultaneously selecting and scheduling parallel machines to minimise machine costs and tardiness cost susinga tabu metaheuristic approach.

Anghinolfi and Paolucci (2007) addressed a parallel machine problem with a total tardiness objective function using a hybrid metaheuristic (HMH) that consisted of integrated features of tabu search, simulated annealing and variable neighbourhood search. Biskup et al. (2008) addressed minimisation of total tardiness for an identical parallel machine problem and discussed some heuristics such as the traffic priority index (TPI) heuristic, the modified due date (MDD) heuristic, and KPM. New solution approaches were developed based on these heuristics.

Metaheuristics methods have been proven to be excellent techniques for solving other types of complex problems. Villegas et al. (2011) addressed a truck and trailer routing problem (TTRP) with a heterogeneous fleet. The authors reported that all of the GRASP versions proposed, including pure GRASP, GRASP/VNS and GRASP/path relinking, achieved better results than previous methods. Luis et al. (2011) applied a reactive GRASP method to a classical problem (the capacitated multi-source Weber problem).

Boudia et al. (2007) applied a reactive GRASP method with path relinking to a combined production–distribution problem with a total cost minimisation objective function taken from production setups and distribution and inventory levels. Reactive GRASP and GRASP with path relinking outperform the pure GRASP version.

Ribeiro and Rosseti (2007) proposed a parallel cooperative strategy with GRASP and GRASP with path relinking. The authors highlighted the single-walk and multiple-walk parallelisations. Piñana et al. (2004) applied GRASP with path relinking to a matrix bandwidth minimisation. Resende et al. (2010) applied GRASP and path relinking to the max–min diversity problem (MMDP) using evolutionary path relinking.

Delorme et al. (2004) applied a GRASP metaheuristic to a set packing problem. In the improvement phase, reactive GRASP and path relinking were used. Nascimento et al. (2010) addressed a multi-plant capacitated lot sizing problem using a GRASP metaheuristic with path relinking. Villegas et al. (2010) addressed a GRASP/VND and evolutionary local search for a single truck and trailer routing problem with satellite depots (STTRPSD). Aiex et al. (2003) applied a parallel GRASP method with path relinking to job shop scheduling, highlighting independent and cooperative parallelisation strategies.

Ho and Gendreau (2006) applied a tabu search with path relinking to a vehicle routing problem (VRP). Resende and Ribeiro (2011) applied GRASP with path relinking using restart strategies. Chaovalitwongse et al. (2011) described a revised GRASP approach with path relinking for a linear ordering problem (LOP). Armentano et al. (2011) applied a tabu search with path relinking to an integrated production–distribution problem. Additional research concerning parallel machine environments is summarised in the review of the state of the art of research on parallel machines by Cheng and Sin (1990).

GRASP METAHEURISTIC

GRASP metaheuristics have been applied to several types of problems. Feo and Resende (1989, 1995), developed a meta-heuristic GRASP (greedy randomized adaptive search procedure) that works with multiple starts. In each iteration, two steps are undertaken: a construction phase and a local search phase. In the first phase, a feasible solution is constructed using a greedy random algorithm. In the second phase, the neighbourhood of the solution generated is explored until a local optimum is reached.

At the start of the application of the metaheuristic, a list of jobs, ordered according to some criteria, is required. Then, a restricted candidate list (RCL) is built up to store the best elements that may be part of the partially generated solution, which will be used to draw the next job that will be incorporated into the partial solution. Once drawn, the list is rebuilt and the process continues until all jobs are drawn.

Some authors have developed strategies for determining the optimum size of the RCL. Prais and Ribeiro (2000), for example, developed an approach known as reactive GRASP, in which the goal is to find the best size for the RCL, instead of working with the same size in every iteration. Boudia et al. (2007) demonstrated the effectiveness of the reactive approach in comparison to using a list of fixed size in a coupled problem of production and distribution.

PATH RELINKING

The path relinking approach described by Glover (1996) consists of exploring the path or paths between two solutions, called the initial solution and the guide solution (the worst and the best solution, respectively). In each iteration, a move is made in the initial solutionto transform the initial solution into a guide solution. This movement consists of introducing attributes that are found in the guide solution into the initial solution until both solutions are equal. This trajectory can be understood as a process of intensification, and it is important to explore solution that allows rapprochement between the two solutions. The objective is to reach a new local optimum, where the initial solution is given by x^1 and the guide solution by x^2.

There are some techniques to implement path relinking. Ho and Gendreau (2006) highlight some of the critical components in the implementation of path relinking, such as the form in which the set of elite references (the pool) is built and the criteria for choosing the initial solution and guide solution. All these choices affect the path that the algorithm will explore.

Resende et al. (2010) discuss an important component in GRASP/path relinking implementation. The static update in the elite set is filled out and updated in each iteration of the GRASP application. In the end, path relinking is applied between the elite solutions to explore the space that exists between each pair of elite solutions. In dynamic updating, for each solution generated by GRASP is chosen randomly among a set solution of the elite, and the path relinking is applied. With both strategies (static or dynamic), the resulting solution is always sent to a local search.

Resende and Ribeiro (2003) highlight other important variations on the relinking process:

- Forward relinking: the path is constructed from the initial solution x^1 to the guide solution x^2;
- Backward relinking: this strategy is the reverse of the previous one;
- Mixed relinking: two paths are explored simultaneously.

Ribeiro and Rossetti (2007) and Ribeiro et al. (2009) examined the idea of a time-to-target solution, which determines the probability of the algorithm finding a solution less than or equal to a given solution (called the target solution) within a specified time. The authors also address implementation strategies in parallel meta-heuristics to accelerate the search and solve complex problems. They presented two approaches: a parallel independent approach, in which threads do not exchange any information, and a cooperative parallelisation approach, in which information is shared and used by other threads.

MATHEMATICAL FORMULATION

Consider a set of jobs $J = \{1,2,3,\dots,n\}$ to be scheduled for a set of identical machines $M = \{1,2,3,\dots,m\}$. Each job j has a release date denoted by r_j, a processing time p_j, a due date d_j and a weight w_j. Preemption is not allowed, and a job cannot start before its release date. No machine can execute more than one job at the same time. In a schedule where job j is completed at time C_j, the tardiness of the job is expressed by $T_j = max(0, C_j - d_j)$. The problem representation is $P_m|r_j|\sum w_j T_j$.

Sets

J	set of jobs	$j = 1, \ldots, n$
M	set of machines	$i = 1, \ldots, m$

Parameters

T	planning horizon = T_t
r_j	release time of job$j$
v_i	release time of machine$i$
p_j	execution time of job j
c_{jt}	cost associated with j job, which starts at time$t$

Decision variables

$x_{ij}^t = 1$, if machine i starts job $j \in J$ at time$t$; 0, otherwise.

Model

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{t=\max(r_j+1, v_i+1)}^{T-p_j^i+1} c_{jt}\, x_{ij}^t \qquad (1)$$

s.t.

$$\sum_{i=1}^{m} \sum_{t=\max(r_j+1, v_i+1)}^{T-p_j^i+1} x_{ij}^t = 1 \qquad \forall j \qquad (2)$$

$$\sum_{\substack{j=1:\\ r_j<t}}^{n} \sum_{s=\max(r_j+1,\ t-p_j^i+1)}^{\min(t, T-p_j^i+1)} x_{ij}^s \leq 1 \qquad \forall\, i, t: t > v_i \qquad (3)$$

$$x_{ij}^t \in \{0,1\} \qquad \forall\, j, t, i \qquad (4)$$

Equation (1) is the objective function of the problem and includes the sum of the weighted tardiness values of all jobs. Constraint (2) ensures that all demand is met. Constraint (3) ensures that, on each date, the capacity of each machine is not violated. Equation (4) requires that the variables are binary.

SOLUTION METHODS

In the present study, the GRASP algorithm was implemented and the mechanism of path relinking (forward and backward movement) was subsequently added to explore two paths simultaneously shown as follows:

```
input data
for k=1 to max iterations or max time do
    update configuration k
    for j=1 to n do
        update RCL
        randomly choose a job in RCL
        insert job in the best position
    end
    local search( )
    path relinking( )
end
return best solution
```

In each iteration, a particular configuration of the algorithm (*k*) is updated. This configuration consists of three elements: the size (*l*) of the restricted candidates list, the sorting rule to be applied to jobs, and the updatingof the seed of the random number generator. The size of the RCL is given by the following series: 2, 3, 4, 5, 6 and (*n*). These sizes will work with approximately greedy rules (when *l*=2) or a completely random search (when *l*=n).The larger the size of the RCL is, the more random the choice task is in rendering the solution. Thus, we decided to test some various RCL sizes to check that they could choose to provide the best solution.

Instead of using a dynamic sorting rule, after manyof the remaining jobs were reclassified, we chose to use a fixed ordering rule. Thus, when a job belonging to the RCL was chosen, the next job to make the RCL would already be known. Three rules for sorting the jobs were tested, and each rule had two sorting criteria. In case of a tie, after applying the first criterion, the second criterion was used to break ties. If the tie persisted, then the jobs were sorted by their index values. The rules developed were named as follows:

(1) Major weight and less release date;
(2) Less due date and major weight;
(3) Less release date and less due date.

As for the seed, for each combination described previously (size of the RCL and sorting rule), 100 different seeds were tested, for a total of 6 × 3 × 100 = 1800 iterations or 1800 s (the stop criterion) (for Jouglet and Savourey (2011) instances). The seed represents the number of replications to guarantee randomness in obtaining different solutions.

In each iteration of the search, the movement that provides the best reduction of the value of the objective function was carried out. When no improvement was found, a local optimal solution was considered to have been found, and the algorithm terminated.

Suppose the following set of jobs is to be scheduled: $\{j_1, j_7, j_9, j_{10}, j_5\}$. The RCL works in the following way:

Iteration 1
RCL = {j1, j7, j9} Partial solution {j7}
Job randomly chosen = j7

Iteration 2
RCL = {j1, j9, j10} Partial solution {j7, j10}
Job randomly chosen = j10

Iteration 3
RCL = {j1, j9, j5} Partial solution {j1, j7, j10}
Job randomly chosen = j1

Iteration 4
RCL = {j9, j5} Partial solution {j9, j1, j7, j10}
Job randomly chosen = j9

Iteration 5
RCL = {j5} Complete solution {j9, j1, j5, j7, j10}
Job randomly chosen = j5

When we have a complete solution, local search is employed. This consists of removing a job from its current position and inserting it in the best position. This routine is applied for all of the jobs. The local search stops when no more improvements are achieved. Suppose a complete solution is {j9, j1, j5, j7, j10}; a local search can make the following movement { j1, j5, j7, j9, j10}: job 9 in the first phase

Table 1. Comparison between the proposed algorithm and the exact method.

Method	Jobs × Machines [time in mS]		
	10 × 2	15 × 3	20 × 3
Branch and Bound (Jouglet and Savourey)	39.075	10,099.58	291,236.38
GRASP with Path Relinking (Backward)	31.549	54.005	52.782
GRASP with Path Relinking (Multi-threading)	18.997	31.033	32.098

(the complete solution) was scheduled in the first position, but in the local search, the best position is the fourth. The best position represents the position of a particular job that results in a lower cost for the schedule.

The path relinking routine begins by setting the reference solutions (initial and guide). In the case of forward relinking, the search is from the initial solution to the guide solution. Path relinking performs the movements that transfer jobs that are outside their corresponding sequences (in the guide solution) to the initial solution. This can be performed either by insertion (relocation) or by exchange (a swap) between two jobs. After all possible moves are evaluated, the best move is executed.

Another aspect to be noted is that when a job is transferred to a new sequence, it is positioned in the best insertion position. However, after all assignments have been made to sequences matching the guide solution, some jobs may not be in the same positions as in the reference sequences. In this case, a procedure for inserting or removing the intrinsic sequence is used to reposition the jobs one by one until the initial solution and guide solution are equal.

After executing the path relinking routine, the set of elite solutions is updated. A solution will only be accepted in the elite set if its objective function value is less than that of the worst solution belonging to the set. The determination of which solution to exclude is made by deciding to replace the solution with an objective function value greater than that of the candidate solution that is the shortest distance to this solution.

This will cause the nearest or most similar solution to be excluded, thus contributing to increased diversity among the elite solutions. The measure of distance between two solutions is given by the number of times that a job has a successor job different from its successor job for another solution.

COMPUTATIONAL RESULTS

To evaluate the performance of the proposed meta-heuristics, results obtained using a branch and bound algorithm provided by Jouglet and Savourey (2011) were used. The instances were the same as those used by Jouglet and Savourey (2011), in which there are two parameters,α and β, to control the hardness problem. The processing times have a uniform distribution $[1, 100]$, as do the weights$[1, 10]$. The release dates are distributed according to $[0, \alpha \sum p_i]$, and the due dates $d_i - r_i + p_i$ are distributed according to $[0, \beta \sum p_i]$. The α and β parameters assume the following values: $\{0, 0.5, 1, 1.5\}$ and $\{0.05, 0.25, 0.5\}$. There are 120 instances for each combination (n, m); when n, m, α and β are fixed, 10 instances are created. Tests were conducted using $2m\ x\ 10n, 3m\ x\ 15n\ and\ 3m\ x\ 20n$. The time limit fixed was set to 1800 s. The method implemented was a GRASP with path relinking (forward and backward movement) utilising a multithreading approach to explore two paths simultaneously.

DISCUSSION

The algorithms were coded in the C++ language and processed on a 1.6-GHz Pentium computer with MS Windows XP OS. In Table 1, the instances of 10 jobs and 2 machines for which the metaheuristic achieved an optimal solution in a competitive amount of time, when compared with the exact approach, are presented. For the cases of 15 jobs and 3 machines and 20 jobs and 3 machines, the performance of the metaheuristic was similar.

Using the proposed metaheuristics greatly reduces the CPU time required because, in the exact method (Jouglet and Savourey), a branch and bound procedure enumerates a search space to explore, based on dominance rules, whereas the heuristic explores only a limited part of the problem based on particular criteria, and after a complete solution is obtained, the region around it is explored to enhance the solution. Because of this randomness, the optimal solution is not guaranteed, but in most cases, a good solution is achieved in a competitive amount of time, compared to other methods, such as enumeration procedures. Table 1 show the time employed by the branch and bound method (with an enumeration procedure), backward path relinking and finally multi-threading path relinking.

To enhance the efficiency of multi-threading in relation to the processing of a single trajectory, further tests were carried out using the following instances described by Pessoa et al. (2010): 50 jobs and 2 machines and 100 jobs and 2 machines, with each size having 120 instances and 10 replications being performed for each instance. Each instance was processed until the optimal (opt) solution was achieved or the best known value provided by the authors was reached, with 180 s set as the maximum time and Gap = 100 × (GRASP - Opt)/Opt. Table 2 shows a comparison between the proposed methods, GRASP multi-threading and GRASP without threading. GRASP multi-threading is much faster than without threading, and the average gap, when compared

Table 2. Average GAP and time between GRASP multithreading × pure GRASP.

n × m	% Gap GR_PR_ST × Opt	Average time (s)
50 × 2	3.44	289.633
100 × 2	3.27	291.756
n × m	**% Gap GR_PR × Opt**	**Average time (s)**
50 × 2	0.96	94.493
100 × 2	0.91	89.442

Table 3. Improvement cost obtained by GRASP path relinking.

n × m	% Gap GR_PR_ST × Opt	% Gap GR_PR × Opt	Improvement cost GR_PR × PR_ST
50 × 2	3.44	0.96	-0.721
100 × 2	3.27	0.91	-0.722

Table 4. Improvement time obtained by GRASP path relinking.

n × m	% Gap GR_PR_ST × Opt	% Gap GR_PR × Opt	Improvement time GR_PR × PR_ST
50 × 2	289.633	94.493	-0.674
100 × 2	291.756	89.442	-0.693

with the optimal/best solutions, is less than 1%, whereas without threading, the gap is greater than 3.30% on average.

Table 3 shows the contribution of GRASP multi-threading. A comparison of the average gap associated with the two approaches (multi-threading and without threading) shows that GRASP with path relinking by multi-threading improved the solution by 72% on average.

Table 4 presents the improvement time with GRASP multi-threading. On average, the improvement time was almost 70% greater than without multi-threading.

CONCLUDING REMARKS AND FURTHER RESEARCH

This paper presents an analysis of a real scheduling problem for identical parallel machines, modelled and solved by a GRASP path relinking using multi-threading techniques. To evaluate the results, we used the results for the branch and bound approach given by Jouglet and Savourey (2011) and Pessoa et al. (2010). The solutions provided were shown to be of good quality. To decision makers, it is very important to have alternatives available quickly. Metaheuristics provide, in several cases, the optimal solution to be implemented in a company. In all instances analysed in this study, GRASP with path relinking (forward-backward) multi-threading was, on average, more effective than the branch and bound approach. Further improvements to these results will be pursued by performing simultaneous processing of several variations of path relinking with multi-thread.

ACKNOWLEDGEMENTS

Financial support for this work was provided by ANP, CAPES and CNPq. This support is gratefully acknowledged. We also thank the anonymous referees for their useful references and constructive feedback.

REFERENCES

Aiex RM, Binato S, Resende MGC (2003). Parallel GRASP with path relinking for job shop scheduling, Parallel Comput. 29:393-430.
Alidaee B, Rosa D (1997). Scheduling parallel machines to minimize total weighted and unweighted tardiness, Comput. Oper. Res. 24:775-788.
Anghinolfi D, Paolucci M (2007). Parallel machine total tardiness scheduling with a new hybrid metaheuristic approach, Comput. Oper. Res. 34:3471-3490.
Armentano VA, Shiguemoto AL, Løkketangen A (2011). Tabu search with path relinking for an integrated production-distribution problem, Comput. Oper. Res. 38:1199-1209.
Bilge Ü, Kurtulan M, Klraç F (2007). A tabu search algorithm for the single machine total weighted tardiness problem, Eur. J. Oper. Res. 176:1423-1435.

Biskup D, Herrmann J, Gupta JND (2008). Scheduling identical parallel machines to minimize total tardiness, Int. J. Prod. Econ. 115:134-142.

Boudia M, Louly MAO, Prins C (2007). A reactive GRASP and path relinking for a combined production-distribution problem, Comput. Oper. Res. 34:3402-3419.

Cao D, Chen M, Wan G (2005). Parallel machine selection and job scheduling to minimize machine cost and job tardiness, Comput. Oper. Res. 32:1995-2012.

Chaovalitwongse WA, Oliveira CAS, Chiarini B, Pardalos PM, Resende MGC (2011). Revised GRASP with path-relinking for the linear ordering problem, J. Comb. Optim. 22:572-593.

Cheng TCE, Sin CCS (1990). A state-of-the-art review of parallel-machine scheduling research, Eur. J. Oper. Res. 47:271-292.

Chiang T-C, Cheng H-C, Fu L-C (2010). A memetic algorithm for minimizing total weighted tardiness on parallel batch machines with incompatible job families and dynamic job arrival, Comput. Oper. Res. 37:2257-2269.

Delorme X, Gandibleux X, Rodriguez J (2004). GRASP for set packing problems, Eur. J. Oper. Res. 153:564-580.

Feo TA, Resende MGC (1989). A probabilistic heuristic for a computationally difficult set covering problem. Oper. Res. Lett. 8: 67-71.

Feo TA, Resende MGC (1995). Greedy randomized adaptive search procedures. J. Global. Optim. 6:109-133.

Fuller JA (1978). Optimal solutions versus ‘good’ solutions: an analysis of heuristic decision making. Omega 6(6):479-484.

Glover F (1996). Tabu search and adaptive memory programming – advances, applications and challenges, In: Barr RS, Helgason RV, Kennington JL (Eds.), Interfaces in Computer Science and Operations Research, Kluwer Academic Publishers, Boston pp.1-75,

Ho SC, Gendreau M (2006). Path relinking for the vehicle routing problem, J. Heuristics 12:55-72.

Horowitz E, Sahni S (1976). Exact and approximations algorithms for scheduling non identical processors. J. ACM. 23(2):317-327.

Jouglet A, Carlier J (2011). Dominance rules in combinatorial optimization problems, Eur. J. Oper. Res. 212:433-444.

Jouglet A, Savourey D (2011). Dominance rules for the parallel machine total weighted tardiness scheduling problem with release dates, Comput. Oper. Res. 38:1259-1266.

Koulamas C (1997). Decomposition and hybrid simulated annealing heuristics for the parallel-machine total tardiness problem, Naval Res. Logist. 44:109-125.

Lenstra J, Kan AR, Brucker P (1977). Complexity of machine scheduling problems. Ann. Discrete Math. 1:343-362.

Leung JY-T, Li C-L (2008). Scheduling with processing set restrictions: a survey, Int. J. Prod. Econ. 116:251-262.

Liaw C-F, Lin Y-K, Cheng C-Y, Chen M (2003). Scheduling unrelated parallel machines to minimize total weighted tardiness. Comput. Oper. Res. 30: 1777-1789.

Luis M, Salhi S, Nagy G (2011). A guided reactive GRASP for the capacitated multi-source weber problem, Comput. Oper. Res. 38:1014-1024.

Min L, Cheng W (1999). A genetic algorithm for minimizing the makespan in the case of scheduling identical parallel machines, Artif. Intell. Eng. 13:399-403.

Mokotoff E (2004). An exact algorithm for the identical parallel machine scheduling problem, Eur. J. Oper. Res. 152:758-769.

Morton TE, Pentico DW (1993). Heuristic scheduling systems: with applications to production systems and project management. New York: John Wiley & Sons.

Nascimento MCV, Resende MGC, Toledo FMB (2010). GRASP heuristic with path-relinking for the multi-plant capacitated lot sizing problem, Eur. J. Oper. Res. 200:747-754.

Nessah R, Yalaoui F, Chu C (2008). A branch-and-bound algorithm to minimize total weighted completion time on identical parallel machines with job release dates, Comput. Oper. Res. 35:1176-1190.

Pessoa A, Uchoa E, Aragão MP, Rodrigues R (2010). Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. Math. Prog. Comp. 2:259-290.

Piñana E, Plana I, Campos V, Martí R (2004). GRASP and path relinling for the matrix bandwidth minimization, Eur. J. Oper. Res. 153:200-210.

Pinedo M (2002). Scheduling: theory, algorithm, and systems. New Jersey: Prentice-Hall.

Prais M, Ribeiro CC (2000). Reactive GRASP: an application to a matrix decomposition problem in TDMA traffic assignment, Informs. J. Comput. 12:164-176.

Resende MGC, Ribeiro CC (2003). Grasp with path relinking: recent advances and applications. AT & T Labs Technical Report TD-5TU726: 1-24.

Resende MGC, Martí R, Gallego M, Duarte A (2010). Grasp and path relinking for the max-min diversity problem, Comput. Oper. Res. 37:498-508.

Resende MGC, Ribeiro CC (2011). Restart strategies for GRASP with path-relinking heuristics, Optim. Lett. 5:467-478.

Resende MGC, Werneck RF (2004). A hybrid heuristic for the p-median problem, J. Heuristics 10:59-88.

Ribeiro CC, Rosseti I (2007). Efficient parallel cooperative implementations of GRASP heuristics, Parallel. Comput. 33:21-35.

Ribeiro CC, Rosseti I, Vallejos R (2009). On the use of run time distributions to evaluate and compare stochastic local search algorithms In: Stützle T, Birattari M, Hoos HH (Eds.), Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg pp.16-30.

Shim S-O, Kim Y-D (2007). Scheduling on parallel identical machines to minimize total tardiness, Eur. J. Oper. Res. 177:135-146.

Tanaka S, Araki M (2008). A branch-and-bound algorithm with lagrangian relaxation to minimize total tardiness on identical parallel machines, Int. J. Prod. Econ. 113:446-458.

Villegas JG, Prins C, Prodhon C, Medaglia AL, Velasco N (2011). A GRASP with evolutionary path relinking for truck and trailer routing problem, Comput. Oper. Res. 38:1319-1334.

Villegas JG, Prins C, Prodhon C, Medaglia AL, Velasco N (2010). GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots, Eng. Appl. Artif. Intell. 23:780-794.

Xing W, Zhang J (2000). Parallel machine scheduling with splitting jobs, Discrete. Appl. Math. 103:259-269.

Yalaoui F, Chu C (2002). Parallel machine scheduling to minimize total tardiness, Int. J. Prod. Econ. 76:265-279.