Maximizing the Percentage of *On-Time* Jobs with Sequence Dependent Deteriorating Process Times

Alex J. Ruiz-Torres, Facultad de Administración de Empresas, Universidad de Puerto Rico – Rio Piedras, San Juan, PR, USA

Giuseppe Paletta, Dipartimento di Economia e Statistica, Universita` della Calabria, Cosenza, Italy

Eduardo Perez-Roman, Ingram School of Engineering, Texas State University, San Marcos, TX, USA

ABSTRACT

The paper addresses the problem of maximizing the percentage of on-time jobs in a parallel machine environment with sequence dependent deterioration. The deterioration of each machine (and therefore of the job processing times) is a function of the sequence of jobs that have been processed by the machine. Two machine loading strategies are combined with a set of list scheduling algorithms to solve the identical and unrelated machine versions of the problem. The proposed solutions approaches are tested using a large set of problem instances that consider various levels of the number of jobs and machines, the due date tightness, and the deterioration effect. The results indicate that the approach based on loading considering all machines simultaneously and assigns jobs by due date is the most effective.

Keywords: *Identical Machines, Job Deterioration, List Scheduling Heuristics, Machine Deterioration, Number Of Late Jobs, On-Time Jobs, Parallel Machine Scheduling, Unrelated Machines*

1. INTRODUCTION

This paper addresses the problem of maximizing the percentage of *on-time* jobs, which is equivalent to the *traditional* problem of minimizing the number of tardy jobs, when the job's processing times increase due to machine deterioration in an environment of parallel machines. In this problem, the deterioration of the machine depends on the job sequence. The problem is relevant for two reasons: first, it considers an environment observed in industrial and service settings and second, it focuses on schedules that take customer requirements as a priority. Several examples are described in Cheng et al. (2004) and Hsu et al. (2011).

The key element of the deteriorating jobs problem is that the processing time of

DOI: 10.4018/IJORIS.2015070101

the jobs is a function of their start time or the number of jobs since the start of the schedule (or a maintenance activity). In this paper, we propose a different version of the job deterioration problem where the deterioration of the job processing time depends on the specific jobs that have been previously processed by the machine. This perspective is in line with Yang (2011) and Yang et al. (2012), where the jobs are not per se deteriorating, but instead the machines are the ones deteriorating. This paper considers that the deterioration of the machines (and therefore, the job's processing times) is a function of the sequence of jobs that have been processed by the a machine, an approach first presented in Ruiz-Torres et al. (2013). This view is distinct from the two addressed in the literature as the specific job sequence is relevant in the deterioration level of the machine resource.

An example is used to illustrate the problem. Two teams have to complete a set of ten independent, non divisible/preemtable non sequence dependent jobs available at the start of the schedule. Each job has a baseline duration (if for example, performed first), a deterioration effect, considered the "wear and tear" in the team's performance level, and a due date for the job. Figure 1 provides the case data and two possible schedules. The schedule for each team presents the jobs' sequence, and for each job the team's performance level (above) and total time at the end of that job (below). The research assumes a performance model equal to: performance after a job = performance before job × (1 – deterioration effect of just completed job). For example, schedule 1 for team 1 has three assigned ordered jobs; 6, 1, and 3. A 100% performance level is assumed at the start of the schedule thus after completing job 6, the team is at 96.32% performance (= 100% × (1 – deterioration effect of job 6)), after completing job 1 the team performance would be 92.5% (= 96.32% × (1 – deterioration effect of job 1)), and by the end of job 3 the team would have a 89.2% performance level (= 92.5% × (1 – deterioration effect of job 3)). The effect on the time to complete jobs of the reduction on performance can be easily noted by comparing the sum of the baseline times for these three jobs (233 time units) versus the total time required under this sequence to complete them (242.8 time units) given the deterioration effects.

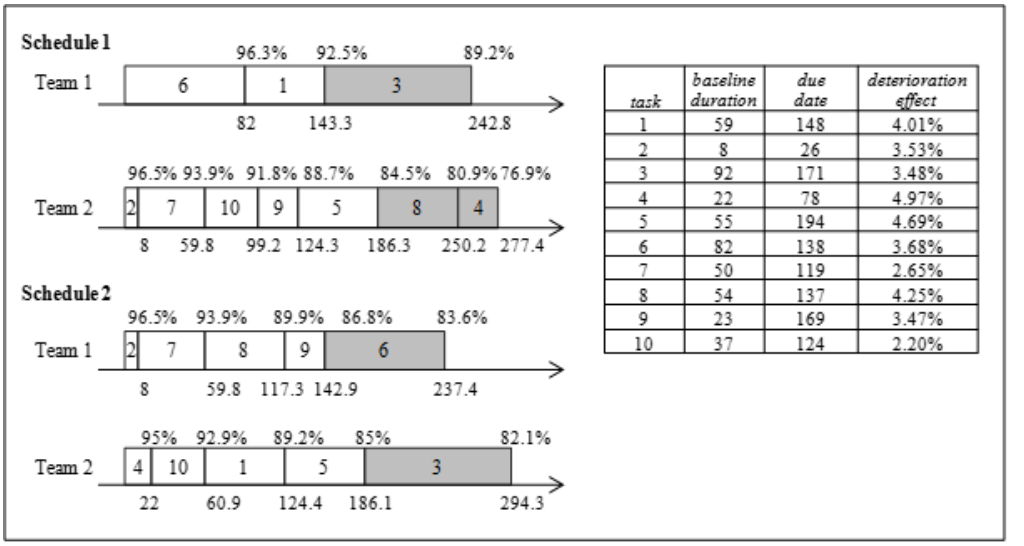
The two schedules have different values for the objective function under consideration; schedule 1 results in seven tasks completed *on-time* (70% *on-time*) while schedule 2 results in eight tasks completed *on-time* (80% *on-time*). Similarly, the schedules provide different values for other criteria of relevance to decision makers. For example, both the makespan (the completion time of the last performed task) and the sum of the machine completion times is lower in schedule 1 (which has fewer on time jobs). Thus a tradeoff exists when considering these other criteria versus *on-time* percentage. While in this research the makespan or the total sum of machine completion times are not considered in the evaluation of schedule performance, these criteria are frequently considered in the deteriorating jobs literature (Kang and Ng, 2007, Ji and Cheng, 2008, Yang et al., 2012), and thus the possibility of tradeoff solutions between these two criteria is relevant for future research.

The rest of the paper is organized as follows. Recent literature on deteriorating jobs and the relevant literature on parallel machine with late jobs objectives are discussed in Section 2. In Section 3, we formulate the problem, and Section 4 presents the solution approaches. An experimental analysis is presented in Section 5. Section 6 concludes the paper and provides suggestions for future research.

2. LITERATURE REVIEW

The problem of deteriorating jobs has received significant attention since the work by Gupta and Gupta (1988) and Browne and Yechiali (1990). Reviews of the literature for deteriorating job problems have been completed by Alidaee and Womer (1999) and by Cheng et. al. (2004). Similarly, problems related to the minimization of late jobs have received the attention of many

Figure 1. Illustrative example of the problem



task	*baseline duration*	*due date*	*deterioration effect*
1	59	148	4.01%
2	8	26	3.53%
3	92	171	3.48%
4	22	78	4.97%
5	55	194	4.69%
6	82	138	3.68%
7	50	119	2.65%
8	54	137	4.25%
9	23	169	3.47%
10	37	124	2.20%

researchers (Wang, 2007, Lai and Lee, 2011, Steiner and Zhang, 2011, Xu et al., 2010b). Reviews for scheduling problems with due date focused criteria has been completed by Biskup and Herrmann (2008), Sterna (2011) and Xu et. al. (2010a). This section focuses on recent papers that consider the parallel machine environment and either the deteriorating job condition or the minimization of the number of late jobs objective. It is noted here that only one research paper Wang (2007) was found that dealt with deteriorating jobs and the number of late jobs criterion.

We first describe the diverse approaches used by researchers to deal with the effect of job/machine deterioration. One perspective is to characterize processing times as a function of the job's start time. Let us define p'_j, p_j, e_j, and b_j as the actual processing time, 'baseline processing time', deterioration factor, and start time of job j respectively. Based on these definitions, the processing time has been modeled by $p'_j = p_j + b_j e_j$: a linear function of the start time. Based on this linear formulation, Kang and Ng (2007) proposed a fully polynomial approximation scheme for the parallel machine problem with makespan objective. Kuo and Yang (2008) and Toksari and Güner (2010) address a modification of the linear version where the increasing (or decreasing) rate is identical for all the jobs (thus $b_j = b$ for all jobs). Kuo and Yang (2008) consider the sum of the job completion times and the sum of the machine completion times as measures of performance, and show that the problems are polynomially solvable for two linear functions. Mazdeh et al. (2010) consider the parallel machine problem with job deterioration of the linear form concurrently with the cost of machine deterioration based on the allocation of jobs to the different machines. The authors consider the joint minimization of the total tardiness and the machine deterioration cost. Given the problem is NP-Hard, the authors propose a heuristic algorithm and test its effectiveness.

Another research stream addresses the parallel machine problem when $p'_j = b_j e_j$, in other words, eliminates the constant 'baseline' portion of the processing time. Ren and Kang (2007) present polynomial approximation algorithms for the makespan minimization problem and provide the complexity of the two machine case. Ji and Cheng (2008) solve the sum of job completion times problem, while Ji and Cheng (2009) address the makespan and sum

of machine completion times criteria, proposing approximation algorithms. Cheng et al. (2009) also address the makespan, but also consider the maximization of the minimum machine completion time. Given both problems are NP-hard the authors propose heuristic algorithms and evaluate their performance. Huang and Wang (2011) address two uncommon objectives: total absolute differences in completion times and the total absolute differences in waiting times. They demonstrate these problems are solvable by polynomial algorithms.

Researchers also have characterized the processing time as a function of the job's position in the machine sequence. Let's define p'_{jrh} as the processing time of job j if processed in the r^{th} position of machine h. The papers by Yang (2011) and Yang et al. (2012) consider the parallel machine problem where the processing time is defined by one of two models $p'_{jrh} = p_{jh} + r \times b_{jh}$ and $p'_{jrh} = p_{jh} \times r^{b}_{jh}$, where b_{jh} is the deterioration effect of job j on machine h, and the position r depends on the number of jobs after a maintenance event. Both papers address the minimization of the total machine load taking into consideration the joint decisions of maintenance frequency and timing, and the assignment and sequence of the jobs on the machines. The article by Yang (2011) deals with the identical parallel machine case, thus no difference in base processing times or deterioration effects between machines, while Yang et al. (2012) deals with the unrelated machines (a more general case). In both papers the authors demonstrate that all versions addressed with a given job frequency can be solved in polynomial time. Mosheiov (2012) addresses the problem where p'_{jrh} is defined as a non-decreasing function in r and the processing time could be unique to each machine, therefore possibly requiring a n^2m input matrix of processing times (where n is the number of jobs and m the number of machines). For this problem the author provides a polynomial time algorithm and describes several extensions. Hsu et al. (2011) consider the problem of unrelated parallel machines with rate modifying activities to minimize the total completion time, where at most one rate modifying activity can occur per machine. The authors propose an algorithm that can solve the problem in $O(n^{m+3})$ if the rate modifying activities are less than 1 (and greater than 0) and in $O(n^{2m+2})$ if the rate modifying activities are larger than 1.

The scheduling literature that deals with due date criteria is extensive, thus we first focus on relevant papers that deal with the number of late jobs in parallel machine and then on due date related problems for the condition of deteriorating jobs. The problem of minimizing the number of late jobs in parallel machines was addressed by Ho and Chang (1995). The authors propose and analyze the performance of several heuristics derived from the optimal single machine algorithm from Moore (1968). Lin and Jeng (2004) address the case of batch scheduling in parallel machines with two objectives: the minimization of the maximum lateness and the number of tardy jobs. A dynamic programming approach and some heuristics are proposed to solve the problem. M'Hallah and Bulfin (2005) develop a branch and bound algorithm to minimize the weighted and un-weighted number of tardy jobs for the identical and unrelated parallel machine cases. Bornstein et al. (2005) present a polynomial algorithm for identical parallel machine case with identical processing times and number of tardy jobs objective. The researchers formulate the problem as a maximum flow network model and provide an optimality proof. Ruiz-Torres et. al. (2007) examine the dual resource problem of scheduling jobs on a set of parallel machines with the objective of minimizing the number of late jobs. In this research the speed of the machines depends on the allocation of a secondary resource, which is fixed in quantity and is allocated to the machines at the start of the schedule. The paper proposes an integer programming formulation to solve the case where the jobs are pre-assigned to the machines and proposes a set of heuristics for the more general case.

Research papers dealing with deterioration processing times and due date related criteria are few. Wang and Xia (2005) address the single machine problem with decreasing linear

deterioration and develop optimal algorithms for several objectives including the minimization of the number of late jobs. Toksari and Güner (2010) address the problem where the processing time model assumes position based learning with linear and non-linear deterioration with the objective of minimizing the earliness/tardiness with a common due date. Both papers provide mathematical formulations and procedures to address large problems. Lee et al. (2011) consider the single machine environment with deteriorating jobs and setup times as to minimize the number of late jobs. The paper presents dominance properties, a lower bound, and an initial upper bound by using a heuristic. The presented algorithm is able to solve large instances in reasonable amounts of computational time. Lee and Lu (2012) consider the single-machine problem of minimizing the total weighted number of late jobs with deteriorating jobs and setup times. The authors developed a branch-and-bound algorithm and present several dominance properties and a lower bound to solve the problem optimally. Computational results show that the proposed algorithm can solve relatively large instances.

3. THE PROBLEM

The problem addressed in this paper is based on the model proposed by Ruiz-Torres et al. (2013) who considered sequence dependent deterioration on parallel machines with a makespan minimization objective. There are n independent non-divisible jobs $N=\{1,\ldots,j,\ldots n\}$ and all jobs are available for processing at time zero and preemption is not allowed. There are m parallel machines $M = \{1,\ldots,k,\ldots m\}$ and each machine can process one job at a time and cannot stand idle until the last job assigned to it is completed.. The baseline processing time of job j on machine k is p_{jk} and each job has a due date d_j. Let e_{jk} be the deteriorating effect of job j on machine k and $0 \leq e_{jk} < 1$ for all $j \in N$ and $k \in M$. Both the general unrelated parallel machines case and the identical machines case, where $p_{jk} = p_j$ and $e_{jk} = e_j$ for all machines k, are considered in this work.

There are g possible positions in each machine, $g = n$, and let G be the set of positions. Let $x[h, k]$ be the job assigned to position h of machine k. Let q_{kh} indicate the performance level of machine k for the job in position h and let q_{kh} be defined by $q_{kh} = (1 - e_{x[h-1,\,k]k}) \times q_{k(h-1)}$ for each machine $k \in M$ and each position h greater than 1. It is assumed that the machines start with no deterioration, thus $q_{k1} = 1$ for all $k \in M$. The actual processing time of the job $x[h, k]$ on machine k is equal to $p'_{x[h,k]k} = p_{x[h,k]k} / q_{kh}$. This approach to model the deterioration effect was first proposed in Ruiz-Torres et al. (2013).

The problem under consideration is the assignment and sequencing of jobs to machines to maximize the percentage of *on-time* jobs. This problem is equivalent to the problem of minimizing the number of tardy jobs. The complexity of this problem with $m > 1$ is clearly NP-hard given the problem that assumes identical machines and no deterioration ($P||U_{sum}$) is known to be NP-Hard (Ho and Chang, 1991).

The mathematical formulation for this problem is presented next. The decision variable x_{jkh}, $j \in N$, $k \in M$, $h \in G$, is a binary variable that is equal to 1 if job j is assigned to machine k in position h, 0 otherwise. The decision variable u_{kh}, $k \in W$, $h \in G$, is a binary variable that is equal to 1 if the job assigned to worker k in position h is late, 0 otherwise. Let B be a big number.

$$\text{Minimize } z = \sum_{h \in G, k \in M} u_{kh} \tag{1}$$

$$\sum_{j \in N} x_{jkh} \leq 1 \;\forall\; h \in G,\; k \in M \tag{2}$$

$$\sum_{h \in G, k \in M} x_{jkh} = 1 \;\forall\; j \in N \tag{3}$$

$$t_{kh} = \sum_{j \in N, l = 1..h} p_{jk}/q_{kl} \times x_{jkl} - \sum_{j \in N} d_j \times x_{jkh} \;\forall\; k \in M,\; h \in G \tag{4}$$

$$t_{kh} \leq u_{kh} \times B\,(1 - \sum_{j \in N} x_{jkh}) \;\forall\, k \in M,\ h \in G \tag{5}$$

$$x_{jkh} \leq \sum_{l \in N} x_{lk(h-1)} \;\forall\, j \in N,\ k \in M,\ h \in G \setminus \{1\} \tag{6}$$

$$q_{kh} = \sum_{j \in N} (1 - e_{jk}) \times q_{k(h-1)} \times x_{jk(h-1)} \;\forall\, h \in G \setminus \{1\},\ k \in M \tag{7}$$

$$q_{k1} = 1 \;\forall\, k \in M \tag{8}$$

$$x_{jkh} \in \{0, 1\} \;\forall\, j \in N,\ k \in M,\ h \in G \tag{9}$$

$$u_{kh} \in \{0, 1\} \;k \in M,\ h \in G \tag{10}$$

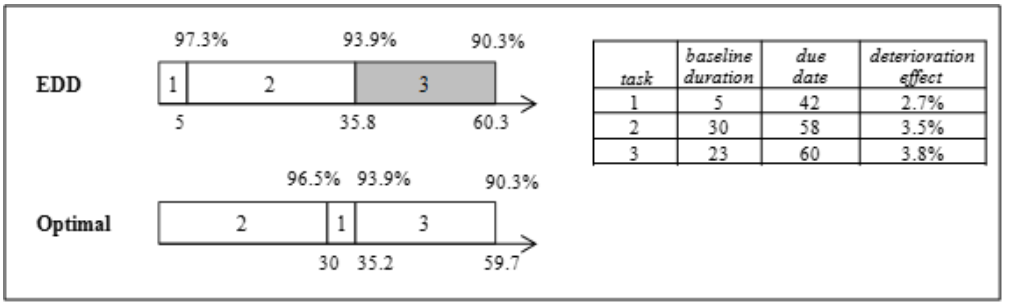
In the mathematical model, equation (1) is the objective function which is defined as the number of tardy jobs. Given the total number of jobs (i.e. n) is fixed, minimizing the number of tardy jobs maximizes the percentage of *on-time* jobs ($[n - z]/n$). Equation (2) states that to each position in each machine can be assigned at most one job, while equation (3) states that each job must be assigned just once to one position in one machine. In Equation (4), if a job is assigned to machine k at position h (i.e. $\sum_{j \in N} x_{jkh}=1$), t_{kh} represents the tardiness of the job at that position. In this case, a value of t_{kh} equal or less than 0 indicates an *on-time* job, while a positive value indicates the job in that position/ machine is late. Whereas if no job is assigned to that position (i.e. $\sum_{j \in N} x_{jkh} = 0$), t_{kh} represents the workload of machine k. Equation (5) is used to set the binary variable to 1 if the job in that position/ machine is late, 0 for *on-time* (or unused positions). Equation (6) guarantees continuous assignments. Equations (7-8) define the performance level of each machine for each job position and equations (9-10) set up the binary variables. The problem has $nm(n+1)$ binary variables, making it "computationally" challenging as n and m increase.

The single machine case is an important building block in the development and analysis of parallel machine problems. The well-known algorithm by Moore (1968) generates an optimal schedule (minimum number of late jobs) in the case of no processing time deterioration. However, Moore's algorithm does not generate an optimal solution for the proposed problem as job processing times depend on the sequence, and ordering jobs by the earliest due date (EDD) rule, (which is part of Moore's algorithm) can result in actual processing times that result in late jobs. For example, consider the three job case in Figure 2. If the three jobs are ordered by using the EDD rule, job 3 is late, while if ordered in the sequence 2-1-3 (the optimal sequence), all jobs are *on-time*.

4. SOLUTION METHODS

This section presents the proposed solution approaches for the problem of maximizing the percentage of *on-time* jobs in parallel machines with sequence dependent deterioration. The percentage of *on-time* jobs is O_t and equal to $(n - \sum_{j \in N} \{u_j\})/n$. As in Ho and Chang (1995), who addressed the minimization of the number of late jobs in identical parallel machines, the loading of jobs in the machines will follow one of two strategies: loading one machine at a time or loading in all the machines simultaneously. Given the effect of deterioration on the sequence, a job being added to a machine will be considered for all possible positions. Of all the assignments that result in jobs being *on-time*, the assignment that either minimizes the load in the machine or the deterioration of the machine will be selected. This approach builds on the general scheme for the special case of the single machine problem discussed in Section 3. We first present some general notation followed by the details for the two heuristics. Since this research considers both the unrelated and identical parallel machine cases, particulars about each implementation are discussed.

Figure 2. Example of the relationship between due date ordering and deterioration



task	baseline duration	due date	deterioration effect
1	5	42	2.7%
2	30	58	3.5%
3	23	60	3.8%

4.1. Notation

M' Current empty machines.
Z Set of jobs not yet assigned to the machines.
Q_k Performance level in machine k if it performs all the not yet assigned jobs.
P_k Total processing time of the not yet assigned jobs if they are performed in machine k.
C_k Current load in machine k.
q_k Current performance level in machine k (at end of the last job in the machine) .
n_k Number of *on-time* jobs currently assigned to machine k.
L Set of late jobs (or temporarily un-assignable jobs).
α Job under consideration.
γ Machine being currently loaded.
V_k A set of temporary job sequences for machine k.
$C(v)$ The completion time of the temporary sequence $v \in V_k$.
$\Delta(v)$ Gap between the current load in machine k and the completion time of the temporary sequence $v \in V_k$ ($\Delta(v) = C_k - C(v)$).
$q(v)$ Performance level of machine k after performing the sequence $v \in V_k$.

4.2. Heuristic SM

The objective of heuristic *SM* (*Single Machine* loading) is to load one machine at a time achieving as many *on-time* jobs as possible. The first consideration of the implementation of this approach is what machine to select first when machines are not identical. Several approaches were tested in pilot experiments. Among these approaches a rule based on deterioration provided the best performance and therefore was the one implemented (remark: a second approach considered consisted on selecting the machine with the highest cumulative processing time for all jobs yet to be scheduled). When machines are not identical, the approach is to select the machine that has the largest cumulative deterioration, with the logic that this is the most problematic machine and it is intuitive to use it as efficiently as possible early on the plan. The second consideration is the order in which jobs are considered for loading. Four priority rules (ω) are used to create a ordered list:

- ($\omega = p$) listing the jobs in non-decreasing order of processing times $p_{j\gamma}$,
- ($\omega = d$) listing the jobs in non-decreasing order of due dates d_j,
- ($\omega = e$) listing the jobs in non-decreasing order of deterioration effects $e_{j\gamma}$, and
- ($\omega = r$) listing the jobs in non-increasing order of processing time-deterioration effect ratios $r_{j\gamma} = p_{j\gamma}/e_{j\gamma}$,

where γ is the machine being currently loaded. There are four versions of heuristic *SM* (*SM-p, SM-d, SM-e, SM-r*) both for the identical and unrelated machine problem, depending on the rule ω used to order the unscheduled jobs.

Step 0: Let $L = \varnothing$. Let $M' = M$. Let Z the set of all jobs in N in any order.
Step 1: Let $P_k = \sum_{j\in Z} p_{jk}$ for all k from M' and $Q_k = \prod_{j\in Z} (1- e_{jk})$ for all k from M'.
Step 2: If processing times are identical for all machines, select any γ from M', else let γ be the machine with $\boldsymbol{min}_{k\in M'} \{Q_k\}$. Remove γ from M' and set $n_\gamma = 0$. Order Z with respect the priority rule ω.
Step 3: Remove the first job from Z. This is job α.
Step 4: Let V_γ be the set of $n_\gamma + 1$ temporary machine sequences generated by a) inserting job α ahead of each of the n_γ jobs in the *original* machine sequence and b) adding job α at the end of the *original* machine sequence.
Step 5: Let v^* be the machine sequence from V_γ with all jobs *on-time* and least completion time. If v^* is found, assign it to machine γ and set $n_\gamma = n_\gamma + 1$, else $L = L \cup \alpha$.
Step 6: If $Z \neq \varnothing$ go to Step 3.
Step 7: If $M' \neq \varnothing$ and $L \neq \varnothing$ then $Z = L$ and go to Step 1, else End.

4.3. Heuristic AM

The objective of heuristic *AM* (*All Machines* loading) is to consider all machines when loading each of the jobs as to determine the "best" position to load each job. The first consideration on the implementation of this approach is the order in which jobs are loaded and analyzed. As in heuristic *SM*, the four job parameters p_{jk}, r_{jk}, d_j, and e_{jk} are used to create a priority list. When machines are unrelated, for each job there are m possible values (one for each machine) of processing time, deterioration, and processing time to deterioration ratio. Based on pilot experiments, we use four ω rules to create a priority list:

- ($\omega = p$) listing the jobs in non-decreasing order by smallest processing times, i.e by $p_j = \min_{k\in M} \{p_{jk}\}$,
- ($\omega = d$) listing the jobs in non-decreasing order by due dates d_j,
- ($\omega = e$) listing the jobs in non-decreasing order by smallest deterioration effects, i.e by $e_j = \min_{k\in M} \{e_{jk}\}$,
- ($\omega = r$) listing the jobs in non-increasing order by largest processing time-deterioration effect ratios, i.e by $r_j = \min_{k\in M} \{p_{jk}/e_{jk}\}$.

When machines are identical, the same rules are used, with the difference that $p_j = p_{jk}$, $e_j = e_{jk}$ and $r_j = p_{jk}/e_{jk}$, for each $k \in M$. The second consideration is the selection of the machine and position for loading a job if there are multiple options that will have the job completed *on-time*. Based on pilot experiments, the position that minimizes the increase in machine load is selected from those that result in the job being *on-time*. There are four versions of heuristic *AM* (*AM -p, AM -d, AM -e, AM -r*) for both the unrelated and identical machine problems depending on the rule ω used to order the jobs.

Step 0: Let $L = \varnothing$, $n_k = 0$ for each $k \in M$, and Z be an ordered list of jobs by ω rule.
Step 1: Remove the first job from Z. This is job α.
Step 2: For all $k \in M$ let V_k be the set of $n_k + 1$ temporary machine sequences generated by a) inserting job α ahead of each of the n_k jobs in the *original* machine sequence and b) adding job α at the end of the *original* machine sequence. Let ς be the combination of all machine sequence sets V_k, $\forall\, k \in M$.
Step 3: Let v^* be the machine sequence in machine k^* from ς with all jobs *on-time* and least completion time. Solve ties by minimum machine performance level q_k. If v^* is found, assign to machine k^*, set $n_{k^*} = n_{k^*} + 1$ and go to step 6.
Step 4: For all $k \in M$ let V_k be the set of n_k temporary machine sequences generated by exchanging each of the n_k jobs from machine k with job α . Let ς be the combination of all machine sequence sets V_k, $\forall\, k \in M$.
Step 5: Let v^* be the machine sequence in machine k^* from ς with job g^* exchanged with job α, all jobs *on-time*, and $\Delta(v^*) = \max_{k\in\varsigma} \{\Delta(v)\} > 0$. If v^* is found, assign it to machine k^*, let $L = L \cup g^*$ else let $L = L \cup \alpha$.
Step 6: If $Z \neq \varnothing$ go to Step 1, else End.

The two heuristic approaches are polynomial in nature with a complexity of $O(n^2 + nm)$.

5. EXPERIMENTS

This section describes the set of experiments used to evaluate the relative performance and robustness of the proposed heuristics. The experiments consider four main factors: number of machines, number of jobs, range of the machine deterioration effects, and congestion ratio. Two responses are used to evaluate the performance of the heuristics: the average *on-time* results and the average error versus the best solution.

In previous parallel machine research the assignment of due dates has considered the average processing time, the number of resources, and the congestion ratio (Ho and Chang, 1995, Ruiz-Torres et al., 2007) . The congestion ratio (*CR*) serves to establish the overall due date tightness; a higher *CR* would result in a larger number of late jobs. Let $D_{max} = p_{ave} \times n / (m \times CR)$, and the due date of a job be determined by $p_j^{min} + U(0, D_{max})$ where $p_j^{min} = min_{k \in M} \{p_{jk}\}$. This due date assignment method guarantees that each job can be *on-time* if at least placed in one of the machines at the start of the schedule, therefore there are no inherent late jobs.

The experiments consider the number of machines at three levels, 5, 10, and 20 while the number of jobs at two levels, 100 and 200. The experimental setup assumes that the processing times and deterioration effects are randomly generated by a uniform distribution. For the processing time, the range is 1 to 100, whereas for the deterioration effect only two levels are considered, first with range (1%, 5%) and second with range (5%, 10%). The deterioration effect range variable is called *e_range*. Finally, the congestion ratio has three levels. For identical machines, the *CR* levels are 0.5, 1, and 2. For unrelated machines, the *CR* levels are 2, 3.5, and 5. The *CR* levels were selected based on pilot experiments with the objective of providing relatively similar overall *on-time* job performance. For each machine setting (identical and unrelated) there are $3 \times 2 \times 2 \times 3$ experimental levels. Twenty five replications are evaluated by experimental level, giving a total of 1,800 instances. Instances are available at http://ruiz-torres.uprrp.edu/nt/.

5.1. Results for the Identical Machine Case

Table 1 provides the average *on-time* results for the identical machines experiments. While Table 2, presents the average error versus the best solution. The heuristic error for each instance is based on the best solution found by all heuristics, = (*best instance value – heuristic value*) / *best instance value*. Table 1 shows that when *CR* =0.5 there are multiple experimental points where the heuristics found schedules with all the jobs *on-time* for all the instances (i.e. 100% on Table 1 and 0.00 on Table 2). At *CR* = 0.5 there is not much difference between the heuristic's performance (in terms of *on-time* percentage) . From a managerial perspective this is intuitive since if there is a large amount of slack on the due dates. In such a case many methods/scheduling approaches can result in good/optimal schedules. An interesting result here is the experimental point with *e_range* = (5%, 10%), *m* = 5 and *n* = 200, where the *on-time* percentages are significantly lower than all other experimental combinations with *CR* = 0.5. This is the experimental combination with the highest *n/m* ratio (average number of jobs that will be scheduled in each machine) and has the highest deterioration range. In this case, a high deterioration of the machines is expected by the end of the schedule, therefore a larger percentage of late jobs even when due dates have large slacks.

For *CR* = 1 and *CR* = 2 the difference in performance across the heuristics is more evident, in particular between the due date based rules (*SM-d* and *AM-d*) and all others. When considering only *CR* = 1 and *CR* = 2, the average performance for *SM-d* and *AM-d* is 72.2% and 72.3% respectively, while for *SM-p*, *SM-e*, *SM-r*, *AM-p*, *AM-e*, and *AM-r* the averages are 71.7%, 64.5%, 53.4%, 67.8%, 70%, and 71.4% respectively. While *SM-p* is a relatively close

Table 1. On time percentage results for the identical machine experiments

CR	e_range	m	n	SM-d	SM-p	SM-e	SM-r	AM-d	AM-p	AM-e	AM-r
0.5	(1%, 5%)	5	100	100%	98.2%	98.6%	98.7%	99.9%	91.3%	91.9%	92.9%
			200	100%	93.9%	95.6%	94.3%	99.7%	85.5%	86.6%	87.3%
		10	100	100%	100%	100%	100%	100%	95.4%	97.0%	98.5%
			200	100%	99.8%	100%	99.9%	100%	92.8%	93.4%	94.2%
		20	100	100%	100%	100%	100%	100%	98.6%	99.7%	100%
			200	100%	100%	100%	100%	100%	96.7%	97.9%	99.3%
	(5%, 10%)	5	100	95.7%	90.0%	91.9%	88.4%	94.1%	84.0%	86.1%	87.2%
			200	74.9%	71.3%	71.2%	59.4%	76.3%	67.9%	74.1%	75.2%
		10	100	100%	99.1%	99.6%	99.6%	100%	92.4%	94.0%	95.7%
			200	96.6%	91.9%	93.2%	89.9%	94.3%	84.4%	86.7%	87.8%
		20	100	100%	100%	100%	100%	100%	98.0%	99.2%	100%
			200	100%	99.9%	100%	100%	100%	93.2%	95.5%	96.8%
1	(1%, 5%)	5	100	85.6%	83.9%	80.4%	68.1%	86.2%	79.1%	79.7%	80.9%
			200	75.3%	74.0%	70.6%	57.9%	76.9%	70.1%	72.7%	73.0%
		10	100	92.9%	90.7%	89.1%	80.1%	91.8%	84.2%	85.8%	87.5%
			200	86.6%	85.4%	81.7%	69.7%	86.6%	79.0%	80.9%	82.0%
		20	100	98.3%	95.7%	97.1%	92.8%	95.4%	87.8%	91.5%	95.1%
			200	93.4%	92.0%	89.9%	80.8%	91.8%	85.2%	86.6%	88.1%
	(5%, 10%)	5	100	72.2%	71.2%	65.8%	52.7%	73.0%	67.9%	70.5%	71.5%
			200	59.0%	57.3%	52.5%	39.6%	60.4%	54.6%	58.9%	59.5%
		10	100	83.1%	82.0%	78.4%	66.4%	82.7%	77.0%	79.0%	81.1%
			200	72.9%	71.9%	65.8%	51.5%	73.0%	67.7%	71.0%	72.4%
		20	100	93.6%	92.0%	91.2%	84.4%	90.8%	84.8%	88.0%	91.2%
			200	84.3%	83.5%	79.1%	65.9%	82.9%	77.6%	80.6%	82.2%
2	(1%, 5%)	5	100	61.1%	62.1%	49.9%	38.5%	62.8%	59.9%	61.4%	61.7%
			200	57.5%	57.8%	46.5%	33.6%	59.3%	55.4%	57.9%	58.1%
		10	100	67.2%	68.2%	55.9%	46.2%	68.0%	65.6%	66.8%	68.0%
			200	62.8%	63.7%	50.5%	38.4%	63.9%	61.0%	62.8%	63.5%
		20	100	72.7%	73.5%	66.4%	57.8%	73.6%	71.6%	72.6%	74.9%
			200	66.8%	67.6%	55.1%	44.8%	67.1%	65.0%	66.5%	67.8%
	(5%, 10%)	5	100	54.9%	54.7%	43.4%	30.9%	55.3%	52.2%	54.5%	55.2%
			200	46.2%	45.6%	35.5%	24.9%	47.3%	43.3%	46.4%	46.8%
		10	100	60.6%	61.4%	50.2%	38.1%	60.6%	58.7%	60.2%	61.7%
			200	54.8%	54.7%	42.9%	30.5%	55.0%	52.3%	54.7%	55.2%
		20	100	68.7%	69.4%	60.9%	49.9%	68.5%	66.7%	68.6%	71.8%
			200	62.5%	63.2%	50.3%	38.2%	62.0%	60.2%	62.1%	63.6%
			overall	**80.56%**	**79.60%**	**74.97%**	**67.00%**	**80.53%**	**75.20%**	**77.27%**	**78.55%**

Table 2. Error results for the identical machine experiments

CR	e_range	m	n	SM-d	SM-p	SM-e	SM-r	AM-d	AM-p	AM-e	AM-r
0.5	(1%, 5%)	5	100	**0.00%**	1.84%	1.36%	1.28%	0.08%	8.72%	8.12%	7.08%
			200	**0.02%**	6.12%	4.36%	5.68%	0.26%	14.46%	13.36%	12.66%
		10	100	0.00%	0.04%	0.00%	0.00%	0.00%	4.64%	3.04%	1.52%
			200	**0.00%**	0.24%	0.04%	0.06%	0.02%	7.16%	6.62%	5.76%
		20	100	0.00%	0.00%	0.00%	0.00%	0.00%	1.36%	0.28%	0.00%
			200	0.00%	0.00%	0.00%	0.00%	0.00%	3.34%	2.08%	0.72%
	(5%, 10%)	5	100	**0.09%**	6.05%	4.04%	7.66%	1.74%	12.31%	10.09%	8.87%
			200	2.04%	6.63%	6.82%	22.20%	0.15%	11.15%	3.08%	1.59%
		10	100	0.00%	0.88%	0.36%	0.36%	0.00%	7.56%	6.04%	4.28%
			200	**0.06%**	4.96%	3.60%	6.98%	2.46%	12.62%	10.25%	9.15%
		20	100	0.00%	0.00%	0.00%	0.00%	0.00%	2.00%	0.84%	0.00%
			200	0.00%	0.08%	0.00%	0.00%	0.00%	6.78%	4.54%	3.22%
1	(1%, 5%)	5	100	0.79%	2.79%	6.88%	21.11%	0.14%	8.35%	7.65%	6.25%
			200	2.08%	3.67%	8.15%	24.67%	0.00%	8.79%	5.35%	5.04%
		10	100	**0.30%**	2.69%	4.39%	14.11%	1.44%	9.62%	7.91%	6.06%
			200	**0.28%**	1.69%	5.98%	19.82%	0.32%	9.05%	6.87%	5.58%
		20	100	**0.17%**	2.84%	1.43%	5.80%	3.09%	10.88%	7.06%	3.43%
			200	**0.13%**	1.59%	3.87%	13.63%	1.88%	8.89%	7.39%	5.80%
	(5%, 10%)	5	100	1.37%	2.73%	10.08%	27.87%	0.21%	7.18%	3.59%	2.27%
			200	2.36%	5.13%	13.20%	34.40%	0.00%	9.58%	2.54%	1.52%
		10	100	**0.52%**	1.79%	6.20%	20.55%	0.98%	7.76%	5.35%	2.90%
			200	0.44%	1.88%	10.21%	29.69%	0.35%	7.55%	3.13%	1.19%
		20	100	**0.55%**	2.16%	3.15%	10.43%	3.44%	9.89%	6.40%	3.00%
			200	**0.14%**	1.10%	6.34%	22.00%	1.84%	8.06%	4.51%	2.61%
2	(1%, 5%)	5	100	2.71%	1.14%	20.57%	38.63%	0.07%	4.62%	2.23%	1.77%
			200	3.00%	2.43%	21.51%	43.37%	0.03%	6.54%	2.33%	1.95%
		10	100	2.37%	0.95%	18.81%	32.85%	1.28%	4.68%	2.91%	1.22%
			200	2.07%	0.66%	21.22%	40.16%	0.43%	4.94%	2.06%	0.98%
		20	100	3.36%	2.28%	11.74%	23.07%	2.17%	4.87%	3.51%	0.48%
			200	1.83%	0.59%	19.00%	34.14%	1.38%	4.49%	2.33%	0.41%
	(5%, 10%)	5	100	1.42%	1.79%	22.12%	44.50%	0.64%	6.16%	2.06%	0.78%
			200	2.37%	3.67%	24.92%	47.38%	0.08%	8.48%	1.98%	1.05%
		10	100	2.36%	1.15%	19.16%	38.46%	2.31%	5.38%	2.96%	0.60%
			200	1.20%	1.30%	22.66%	45.04%	0.87%	5.73%	1.43%	0.43%
		20	100	**4.37%**	3.39%	15.19%	30.49%	4.61%	7.18%	4.55%	0.11%
			200	**2.10%**	1.10%	21.16%	40.10%	2.85%	5.68%	2.74%	0.40%
			overall	**1.13%**	**2.15%**	**9.40%**	**20.74%**	**0.98%**	**7.40%**	**4.64%**	**3.08%**

second to *SM-d*, *AM-p* is not the second best performer of the *AM* based rules; *AM-r* is the next best performing *AM* rule. This indicates that not all sequencing rules have the same relative performance level when combined with a machine rule (*SM* or *AM*), although clearly the due date based rule is the best performer. The results presented in Table 2 (error) confirm the difference in heuristic performance at the two higher levels of *CR*. For example, in multiple experimental combinations heuristics *SM-e*, *SM-r*, and *AM-p* have errors above 40%.

The rest of our analysis focuses on the two dominant heuristic combinations: *SM-d* and *AM-d*. These two heuristics provide a similar overall performance, having *on-time* averages values of 80.56% and 80.53% respectively for all 900 experiments. However, their performance across the experiments and experimental variables is not similar. While heuristic *AM-d* is the best overall performer, *SM-d* outperforms it in 14 out of the 36 experimental points (in 16 cases *AM-d* outperforms *SM-d*, and in 6 cases they tie). The values in bold in Table 2 indicate points where *SM-d* outperformed *AM-d*.

Figure 3 presents the error percentages for each of the experimental variables for heuristics *SM-d* and *AM-d*. Both rules share similar tendencies for the *CR* variable, although the effect on *SM-d* is higher. The overall error at $CR = 2$ for *SM-d* is 2.43%, while for *AM-d* the overall error is 1.39%. In terms of the *e_range* variable, both rules also show a similar trend, with the *SM-d* rule experiencing a smaller effect. The relationship between the number of machines and performance presents a situation of inverse effects; while as m increases the performance of *AM-d* worsens, the performance of *SM-d* tends to improve, in other words it generates most of the best solutions. The effect of variable n is relatively small and with a similar tendency for both rules; they decrease in error as n increases. In overall terms, the *SM-d* heuristic is more robust for the parameters *e_range*, m, and n (smaller slopes) than *AM-d*, while heuristic *AM-d* is more robust for the *CR* parameter.

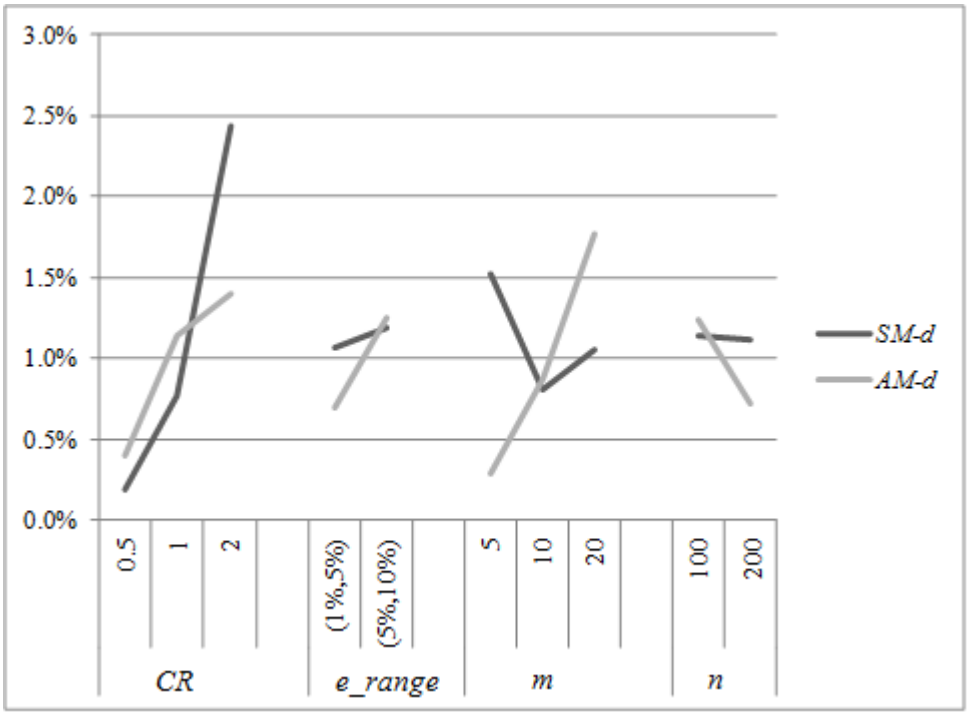
5.2. Results for the Unrelated Machine Case

The average *on-time* and error results are presented in Tables 3 and 4 respectively. The results for the unrelated machine experiment follow similar behavior to that observed for the identical machines case, although here heuristic *AM-d* is more dominant. The overall *on-time* performance of the two best performing heuristics, *AM-d* and *SM-d*, is 86.8% and 83.1% respectively. Only in one experimental point does *SM-d* outperform *AM-d* ($CR = 2$, *e_range* = (1%, 5%), $m = 20$, $n = 100$). Over the 900 experiments, heuristic *SM-d* generates a schedule with the best performance in 141 of the replications (16%), while heuristic *AM-d* in 833 of the replications (93%). Only in 3.7% of the replications does *SM-d* outperform *AM-d*. It is concluded that *AM-d* outperforms *SM-d* for the unrelated machine case regardless of the experimental conditions (this limited to the range of experiments conducted).

Heuristic *AM-d* is also robust based on the job sorting rule used. Regardless of the job sorting rules the overall average *on-time* performance is above 80% for the *AM* based heuristics, while for the *SM* based heuristics the performance varies significantly. While *SM-d* and *SM-p* have an overall *on-time* jobs percentage above 80%, heuristic *SM-e* has an overall *on-time* jobs percentage of 53.1% and *SM-r* has an overall value of 30.1%. This is an interesting result that demonstrates the value of testing multiple scheduling rules for a problem. For example, for one of the replications for experimental parameters CR=3.5, *e_range* = (5%, 10%), $m = 10$, and $n = 100$ the *SM-d* heuristic generated a schedule with 93% *on-time* jobs, while heuristic *SM-r* generated a schedule with only 22% *on-time* jobs.

Finally, in terms of computational time, the heuristics were able to generate the schedules in less than a few minutes for any of the instances (on Intel Core Duo processor at 2.2GHz and 4GB RAM). However as expected, computational time increased with problem size due to the larger search space.

Figure 3. Error performance of SM-d and AM-d versus the experimental variables (identical machines)



6. CONCLUSION

The nature of tasks and machines is that, as work is performed, their performance typically changes. One view relates to the learning effect, thus performance improves as work is performed. Another relates to a decrease in performance due to the wear and tear caused by completing the work, thus performance deterioration. This paper addresses the case where performance deteriorates, and in particular where the deterioration level depends on what particular tasks have been completed. Therefore the time to complete a task at time x depends on the specific tasks that were completed beforehand.

The particular problem addressed in this paper, maximizing the percentage of *on-time* jobs in parallel machines, is a challenging problem, made even more so by the consideration of machine deterioration based on the sequence of jobs processed. Problems that consider customer service type criteria such as the percentage of *on-time* jobs in complex production environments are highly relevant in today's customer focused world. In this paper we propose a group of heuristics that take into account job characteristics such as processing times, deterioration effect, and due dates. Two strategies are proposed, one that loads one machine at a time, and another that considers all machines simultaneously. In general terms the results show that the strategy of loading in all the machines simultaneously combined with due date based rules provided the best performance. Among the relevant areas of future work are the consideration of maintenance as Arnaout and Rabadi (2008) and Mungan et al. (2012), and the analysis of real cases where the model and solution approaches can be applied.

Table 3. On time percentage results for the unrelated machine experiments

CR	*e_range*	*m*	*n*	*SM-d*	*SM-p*	*SM-e*	*SM-r*	*AM-d*	*AM-p*	*AM-e*	*AM-r*
2	(1%, 5%)	5	100	88.1%	86.3%	57.8%	37.7%	94.1%	84.7%	83.3%	83.4%
			200	80.3%	79.4%	52.5%	32.3%	87.2%	78.5%	77.8%	78.3%
		10	100	98.3%	96.6%	69.4%	44.4%	99.3%	95.0%	94.2%	94.5%
			200	98.9%	95.7%	62.2%	36.7%	99.7%	93.4%	92.0%	91.9%
		20	100	99.8%	99.0%	89.8%	60.0%	99.7%	98.9%	99.0%	98.7%
			200	99.9%	99.5%	76.1%	45.0%	100%	98.4%	98.4%	98.4%
	(5%, 10%)	5	100	76.5%	75.4%	47.7%	29.8%	82.1%	74.9%	75.2%	75.2%
			200	65.2%	63.6%	41.3%	24.7%	70.7%	63.4%	65.7%	66.5%
		10	100	96.9%	93.7%	63.4%	36.9%	98.9%	92.9%	91.8%	92.0%
			200	89.7%	87.7%	52.6%	29.2%	95.7%	86.5%	86.1%	85.8%
		20	100	99.6%	98.7%	85.2%	48.6%	99.6%	98.6%	98.6%	98.7%
			200	99.8%	98.7%	68.7%	36.8%	99.9%	97.5%	97.5%	97.4%
3.5	(1%, 5%)	5	100	74.4%	74.2%	42.0%	25.0%	80.3%	74.4%	72.3%	72.1%
			200	68.1%	67.4%	35.3%	20.5%	73.5%	67.3%	66.9%	67.5%
		10	100	90.2%	88.1%	57.0%	33.1%	94.6%	87.4%	85.6%	86.5%
			200	87.7%	86.6%	43.7%	24.2%	93.6%	85.2%	82.8%	82.7%
		20	100	98.1%	96.4%	80.7%	50.9%	98.5%	96.2%	96.4%	96.4%
			200	98.8%	96.2%	62.7%	33.9%	99.3%	95.0%	94.7%	94.5%
	(5%, 10%)	5	100	63.9%	63.3%	36.0%	19.6%	68.1%	63.8%	62.7%	63.6%
			200	54.6%	53.8%	29.5%	16.4%	59.1%	53.6%	55.8%	56.0%
		10	100	85.4%	84.3%	51.7%	27.3%	90.6%	83.6%	82.3%	82.5%
			200	77.2%	76.3%	39.3%	19.8%	82.6%	75.9%	75.3%	76.0%
		20	100	97.4%	95.8%	76.7%	39.7%	98.7%	94.7%	95.1%	94.9%
			200	97.0%	94.3%	55.0%	28.0%	99.0%	92.8%	92.1%	92.2%
5	(1%, 5%)	5	100	63.6%	63.9%	34.4%	18.8%	68.3%	64.1%	62.9%	63.0%
			200	59.2%	59.8%	28.0%	15.3%	64.4%	59.7%	59.6%	60.4%
		10	100	82.2%	81.9%	50.9%	27.7%	87.4%	81.8%	79.5%	79.5%
			200	78.4%	77.8%	38.2%	19.3%	84.5%	77.7%	75.7%	75.8%
		20	100	94.4%	93.2%	74.8%	49.9%	96.5%	92.2%	92.3%	92.0%
			200	95.2%	93.4%	55.1%	27.7%	98.4%	92.1%	91.1%	90.5%
	(5%, 10%)	5	100	56.3%	57.2%	29.0%	15.9%	60.8%	57.6%	56.9%	57.2%
			200	48.7%	47.8%	22.9%	12.3%	52.6%	47.8%	49.3%	49.6%
		10	100	76.9%	76.4%	44.6%	20.6%	81.6%	76.5%	74.2%	74.2%
			200	69.5%	69.4%	34.1%	15.5%	75.2%	69.1%	68.8%	68.8%
		20	100	92.1%	90.6%	71.6%	38.0%	94.7%	89.8%	89.6%	90.0%
			200	90.1%	88.4%	51.7%	21.5%	94.5%	87.4%	86.2%	86.7%
			overall	**83.1%**	**82.0%**	**53.1%**	**30.1%**	**86.8%**	**81.3%**	**80.8%**	**80.9%**

Table 4. Error results for the unrelated machines experiments

CR	e_range	m	n	SM-d	SM-p	SM-e	SM-r	AM-d	AM-p	AM-e	AM-r
2	(1%, 5%)	5	100	6.32%	8.26%	38.50%	59.89%	0.00%	9.93%	11.45%	11.39%
			200	7.94%	8.95%	39.73%	62.89%	0.00%	9.91%	10.75%	10.18%
		10	100	1.25%	2.98%	30.21%	55.39%	0.20%	4.59%	5.35%	5.06%
			200	0.84%	4.05%	37.66%	63.18%	0.06%	6.35%	7.82%	7.92%
		20	100	**0.12%**	0.92%	10.06%	39.97%	0.20%	1.00%	0.92%	1.16%
			200	0.10%	0.50%	23.90%	55.04%	0.04%	1.64%	1.62%	1.58%
	(5%, 10%)	5	100	6.81%	8.16%	41.79%	63.72%	0.00%	8.77%	8.31%	8.35%
			200	7.77%	10.06%	41.60%	65.10%	0.00%	10.42%	7.07%	6.03%
		10	100	2.26%	5.45%	36.07%	62.76%	0.20%	6.26%	7.43%	7.22%
			200	6.26%	8.38%	44.98%	69.46%	0.00%	9.61%	9.97%	10.32%
		20	100	0.24%	1.12%	14.63%	51.36%	0.20%	1.20%	1.28%	1.12%
			200	0.12%	1.22%	31.26%	63.22%	0.06%	2.44%	2.48%	2.56%
3.5	(1%, 5%)	5	100	7.41%	7.67%	47.62%	68.80%	0.00%	7.32%	9.94%	10.20%
			200	7.41%	8.36%	52.01%	72.16%	0.00%	8.43%	9.03%	8.15%
		10	100	4.61%	6.85%	39.69%	64.99%	0.00%	7.61%	9.46%	8.51%
			200	6.25%	7.44%	53.26%	74.17%	0.00%	8.92%	11.50%	11.56%
		20	100	0.93%	2.63%	18.48%	48.59%	0.52%	2.87%	2.66%	2.66%
			200	0.72%	3.32%	37.01%	65.90%	0.22%	4.54%	4.86%	5.06%
	(5%, 10%)	5	100	6.22%	7.05%	47.02%	71.17%	0.00%	6.40%	7.95%	6.57%
			200	7.66%	8.91%	50.18%	72.30%	0.00%	9.35%	5.56%	5.34%
		10	100	5.68%	6.96%	42.96%	69.82%	0.04%	7.75%	9.19%	8.97%
			200	6.52%	7.57%	52.41%	76.06%	0.00%	8.04%	8.83%	7.96%
		20	100	1.62%	3.27%	22.55%	59.92%	0.32%	4.37%	3.92%	4.13%
			200	2.10%	4.81%	44.42%	71.76%	0.04%	6.26%	6.99%	6.94%
5	(1%, 5%)	5	100	6.87%	6.45%	49.55%	72.40%	0.06%	6.13%	7.91%	7.79%
			200	8.12%	7.25%	56.60%	76.30%	0.00%	7.32%	7.46%	6.30%
		10	100	5.98%	6.30%	41.77%	68.31%	0.00%	6.43%	9.03%	9.01%
			200	7.30%	7.97%	54.76%	77.19%	0.00%	8.09%	10.43%	10.31%
		20	100	2.40%	3.67%	22.69%	48.37%	0.25%	4.67%	4.59%	4.84%
			200	3.27%	5.06%	44.00%	71.82%	0.00%	6.39%	7.36%	7.97%
	(5%, 10%)	5	100	7.36%	5.87%	52.13%	73.81%	0.08%	5.13%	6.41%	5.83%
			200	7.46%	9.15%	56.42%	76.61%	0.00%	9.12%	6.23%	5.77%
		10	100	5.77%	6.42%	45.37%	74.80%	0.05%	6.18%	8.99%	9.05%
			200	7.48%	7.63%	54.55%	79.31%	0.00%	8.03%	8.41%	8.42%
		20	100	2.87%	4.44%	24.47%	59.92%	0.17%	5.26%	5.58%	5.10%
			200	4.65%	6.41%	45.29%	77.25%	0.00%	7.45%	8.72%	8.25%
			overall	**4.63%**	**5.88%**	**40.16%**	**66.21%**	**0.08%**	**6.51%**	**7.10%**	**6.88%**

REFERENCES

Alidaee, B., & Womer, N. (1999). Scheduling with time dependent processing times: Review and extensions. *The Journal of the Operational Research Society*, *50*(7), 711–720. doi:10.1057/palgrave.jors.2600740

Arnaout, J. P., & Rabadi, G. (2008). Rescheduling of unrelated parallel machines under machine breakdowns. *Int. J. of Applied Management Science*, *1*(1), 75–89.

Biskup, D., & Herrmann, J. (2008). Single-machine scheduling against due dates with past-sequence-dependent setup times. *European Journal of Operational Research*, *191*(2), 587–592. doi:10.1016/j.ejor.2007.08.028

Bornstein, C. T., Alcoforado, L. F., & Maculan, N. (2005). A graph-oriented approach for the minimization of the number of late jobs for the parallel machines scheduling problem. *European Journal of Operational Research*, *165*(3), 649–656. doi:10.1016/j.ejor.2003.06.045

Browne, S., & Yechiali, U. (1990). Scheduling deteriorating jobs on a single processor. *Operations Research*, *38*(3), 495–498. doi:10.1287/opre.38.3.495

Cheng, T. C. E., Ding, Q., & Lin, B. M. T. (2004). A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research*, *152*(1), 1–13. doi:10.1016/S0377-2217(02)00909-8

Cheng, T. E., Lai, P. J., Wu, C. C., & Lee, W. C. (2009). Single-machine scheduling with sum-of-logarithm-processing-times-based learning considerations. *Information Sciences*, *179*(4), 3127–3135. doi:10.1016/j.ins.2009.05.002

Gupta, J. N. D., & Gupta, S. K. (1988). Single facility scheduling with nonlinear processing times. *Computers & Industrial Engineering*, *14*(4), 387–393. doi:10.1016/0360-8352(88)90041-1

Ho, J. C., & Chang, Y. L. (1995). Minimizing the number of tardy jobs for *m* parallel machines. *European Journal of Operational Research*, *84*(10), 343–355. doi:10.1016/0377-2217(93)E0280-B

Hsu, C. J., Cheng, T., & Yang, D. L. (2011). Unrelated parallel-machine scheduling with rate-modifying activities to minimize the total completion time. *Information Sciences*, *181*(12), 4799–4803. doi:10.1016/j.ins.2011.06.010

Huang, X., & Wang, M. Z. (2011). Parallel identical machines scheduling with deteriorating jobs and total absolute differences penalties. *Applied Mathematical Modelling*, *35*(3), 1349–1353. doi:10.1016/j.apm.2010.09.013

Ji, M., & Cheng, T. (2008). Parallel-machine scheduling with simple linear deterioration to minimize total completion time. *European Journal of Operational Research*, *188*(2), 342–347. doi:10.1016/j.ejor.2007.04.050

Ji, M., & Cheng, T. (2009). Parallel-machine scheduling of simple linear deteriorating jobs. *Theoretical Computer Science*, *410*(38), 3761–3768. doi:10.1016/j.tcs.2009.04.018

Kang, L., & Ng, C. (2007). A note on a fully polynomial-time approximation scheme for parallel-machine scheduling with deteriorating jobs. *International Journal of Production Economics*, *109*(1), 180–184. doi:10.1016/j.ijpe.2006.11.014

Kuo, W. H., & Yang, D. L. (2008). Parallel-machine scheduling with time dependent processing times. *Theoretical Computer Science*, *393*(1), 204–210. doi:10.1016/j.tcs.2007.12.004

Lai, P. J., & Lee, W. C. (2011). Single-machine scheduling with general sum-of-processing-time-based and position-based learning effects. *Omega*, *39*(5), 467–471. doi:10.1016/j.omega.2010.10.002

Lee, W. C., Lin, J. B., & Shiau, Y. R. (2011). Deteriorating job scheduling to minimize the number of late jobs with setup times. *Computers & Industrial Engineering*, *61*(3), 782–787. doi:10.1016/j.cie.2011.05.010

Lee, W. C., & Lu, Z. S. (2012). Group scheduling with deteriorating jobs to minimize the total weighted number of late jobs. *Applied Mathematics and Computation*, *218*(17), 8750–8757. doi:10.1016/j.amc.2012.02.033

Lin, B., & Jeng, A. (2004). Parallel-machine batch scheduling to minimize the maximum lateness and the number of tardy jobs. *International Journal of Production Economics*, *91*(2), 121–134. doi:10.1016/j.ijpe.2003.07.003

M'hallah, R., & Bulfin, R. (2005). Minimizing the weighted number of tardy jobs on parallel processors. *European Journal of Operational Research*, *160*(2), 471–484. doi:10.1016/j.ejor.2003.06.027

Mazdeh, M. M., Zaerpour, F., Zareei, A., & Hajinezhad, A. (2010). Parallel machines scheduling to minimize job tardiness and machine deteriorating cost with deteriorating jobs. *Applied Mathematical Modelling*, *34*(6), 1498–1510. doi:10.1016/j.apm.2009.08.023

Moore, J. M. (1968). An n job, one machine sequencing algorithm for minimizing the number of late jobs. *Management Science*, *15*(1), 102–109. doi:10.1287/mnsc.15.1.102

Mosheiov, G. (2012). A note: Multi-machine scheduling with general position-based deterioration to minimize total load. *International Journal of Production Economics*, *135*(1), 523–525. doi:10.1016/j.ijpe.2011.09.005

Mungan, D., Yu, J., Sarker, B., & Rahman, M. A. (2012). A Pareto-Optimal Solution for a Multi-Objective Scheduling Problem with Periodic Maintenance Requirement. *International Journal of Operations Research and Information Systems*, *3*(2), 24–45.

Ren, C., & Kang, L. (2007). An approximation algorithm for parallel machine scheduling with simple linear deterioration. *Journal of Shanghai University*, *11*(4), 351–354. doi:10.1007/s11741-007-0406-3

Ruiz-Torres, A. J., Lopez, F. J., & Ho, J. C. (2007). Scheduling uniform parallel machines subject to a secondary resource to minimize the number of tardy jobs. *European Journal of Operational Research*, *179*(2), 302–315. doi:10.1016/j.ejor.2006.03.028

Ruiz-Torres, A. J., Paletta, G., & Perez-Roman, E. (2013). Parallel machine scheduling to minimize the makespan with sequence dependent deteriorating effects. *Computers & Operations Research*, *40*(8), 2051–2061. doi:10.1016/j.cor.2013.02.018

Steiner, G., & Zhang, R. (2011). Minimizing the weighted number of tardy jobs with due date assignment and capacity-constrained deliveries. *Annals of Operations Research*, *191*(1), 171–181. doi:10.1007/s10479-011-1000-6

Sterna, M. (2011). A survey of scheduling problems with late work criteria. *Omega*, *39*(2), 120–129. doi:10.1016/j.omega.2010.06.006

Toksari, M. D., & Güner, E. (2010). Parallel machine scheduling problem to minimize the earliness/tardiness costs with learning effect and deteriorating jobs. *Journal of Intelligent Manufacturing*, *21*(6), 843–851. doi:10.1007/s10845-009-0260-3

Wang, J. B. (2007). Single-machine scheduling problems with the effects of learning and deterioration. *Omega*, *35*(4), 397–402. doi:10.1016/j.omega.2005.07.008

Wang, J. B., & Xia, Z. Q. (2005). Scheduling jobs under decreasing linear deterioration. *Information Processing Letters*, *94*(2), 63–69. doi:10.1016/j.ipl.2004.12.018

Xu, K., Feng, Z., & Jun, K. (2010a). A tabu-search algorithm for scheduling jobs with controllable processing times on a single machine to meet due-dates. *Computers & Operations Research*, *37*(11), 1924–1938. doi:10.1016/j.cor.2009.11.012

Xu, K., Feng, Z., & Ke, L. (2010b). A branch and bound algorithm for scheduling jobs with controllable processing times on a single machine to meet due dates. *Annals of Operations Research*, *181*(1), 303–324. doi:10.1007/s10479-010-0746-6

Yang, D. L., Cheng, T., Yang, S. J., & Hsu, C. J. (2012). Unrelated parallel-machine scheduling with aging effects and multi-maintenance activities. *Computers & Operations Research*, *39*(7), 1458–1464. doi:10.1016/j.cor.2011.08.017

Yang, S. J. (2011). Parallel machines scheduling with simultaneous considerations of position-dependent deterioration effects and maintenance activities. *Journal of the Chinese Institute of Industrial Engineers*, *28*(4), 270–280. doi:10.1080/10170669.2011.573006

Alex J. Ruiz-Torres is a full professor in the Facultad de Administración de Empresas, Universidad de Puerto Rico – Rio Piedras. Dr. Ruiz-Torres received Bachelor, Masters, and Ph.D. degrees in Industrial Engineering from Georgia Tech, Stanford, and Penn State respectively. His research interests include supply chain models, decision support systems, and production planning and control. His publications have appeared in International Journal of Production Research, European Journal of Operational Research, International Journal of Production Economics, Computers and Operations Research, Journal of the Operational Research Society, Computers and Industrial Engineering, and OMEGA.

Giuseppe Paletta is full professor of Operations Research at the University of Calabria, Italy. His research interests include stability analysis of feedback control systems, network design, distributed database optimization, scheduling problems, routing problems and integrated production-inventory-routing problems. He has published in European Journal of Operational Research, International Journal of Control, Mathematical Programming Studies, Annals of Discrete Mathematics, Transportation Science, Computers and Operations Research, Information Systems, Networks, Journal of Heuristics, Maritime Economics & Logistics, and SIAM Journal on Discrete Mathematics.

Eduardo Pérez is an AP at Texas State University, Ingram School of Engineering, San Marcos, Texas, USA. He obtained his Ph.D. in Systems and Industrial Engineering from the Texas A&M University and his B.S. in Industrial Engineering from the University of Puerto Rico, Mayaguez Campus. Dr. Pérez is the director of the Integrated Modeling and Optimization for Service Systems (iMOSS) research laboratory. His research interests include stochastic optimization and discrete-event simulation with application to healthcare systems, parallel machines, and renewable energy. Dr. Pérez has served as session chair for the IIE and INFORMS conferences and as a reviewer for journals such as Simulation, IIE Transactions, the European Operational Research, andOmega. His works have been published in multiple journals including IIE Transactions, IIE Transactions on Healthcare Systems Engineering, Simulation, Healthcare Management Sciences, and Computers and Operations Research.