versão impressa ISSN 0101-7438 / versão online ISSN 1678-5142

Eduardo Candido Xavier *
Flávio K. Miyazawa
Instituto de Computação
Universidade Estadual de Campinas (UNICAMP)
Campinas – SP
para@ic.unicamp.br
fkm@ic.unicamp.br

* *Corresponding author*/autor para quem as correspondências devem ser encaminhadas

Recebido em 07/2003; aceito em 06/2004
Received July 2003; accepted June 2004

Abstract

In this paper we consider an experimental study of approximation algorithms for scheduling problems in parallel machines minimizing the average weighted completion time. We implemented approximation algorithms for the following problems: $P|r_j|\Sigma C_j$, $P||\Sigma w_jC_j$, $P|r_j|\Sigma w_jC_j$, $R||\Sigma w_jC_j$ and $R|r_j|\Sigma w_jC_j$. We generated more than 1000 tests over more than 200 different instances and present some practical aspects of the implemented algorithms. We also made an experimental comparison on two lower bounds based on the formulations used by the algorithms. The first one is a semidefinite formulation for the problem $R||\Sigma w_jC_j$ and the other one is a linear formulation for the problem $R|r_j|\Sigma w_jC_j$. For all tests, the algorithms obtained very good results. We notice that algorithms using more refined techniques, when compared to algorithms with simple strategies, do not necessary lead to better results. We also present two heuristics, based on approximation algorithms, that generate solutions with better quality in almost all instances considered.

Keywords: approximation algorithms; practical analysis; scheduling.

Resumo

Neste artigo consideramos um estudo experimental de alguns algoritmos aproximados para problemas de escalonamento em máquinas paralelas onde se deve minimizar o tempo de término ponderado das tarefas. Foram implementados algoritmos aproximados para os seguintes problemas: $P|r_j|\Sigma C_j$, $P||\Sigma w_jC_j$, $P|r_j|\Sigma w_jC_j$, $R||\Sigma w_jC_j$ and $R|r_j|\Sigma w_jC_j$. Foram gerados mais de 1000 testes sobre mais de 200 instâncias diferentes e com isso apresentamos aspectos práticos dos algoritmos implementados. Também fizemos um estudo experimental sobre dois limitantes inferiores baseados em formulações usadas pelos algoritmos. A primeira é uma formulação semidefinida para o problema $R||\Sigma w_jC_j$ e a outra é uma formulação linear para o problema $R|r_j|\Sigma w_jC_j$. Em todos os testes os algoritmos obtiveram resultados muito bons. Notamos que algoritmos usando técnicas mais refinadas, quando comparados com algoritmos que usam estratégias simples, não necessariamente geram soluções melhores. Também apresentamos duas heurísticas, baseadas nos algoritmos aproximados, que geram soluções de melhor qualidade em quase todas as instâncias consideradas.

Palavras-chave: algoritmos de aproximação; análise prática; escalonamento.

1. Introduction

In this paper, we consider an experimental study of approximation algorithms for scheduling problems. For all problems considered, a set of jobs must be scheduled, under some restrictions, in a set of machines minimizing the average weighted completion time. All these problems are NP-hard [S97] and we consider polynomial time approximation algorithms. We have implemented some approximation algorithms to schedule jobs on parallel machines and study their computational performance.

Given a polynomial time algorithm A and an instance I for a minimization problem, we denote by $A(I)$ the value of the solution returned by A when applied to the instance I, and we denote by $OPT(I)$ the value of an optimal solution to I. We say that an algorithm A has an approximation factor α, or is an α-approximation, if $A(I)/OPT(I) \leq \alpha$, for all instances I. When the algorithm A is probabilistic and the inequality $E[A(I)]/OPT(I) \leq \alpha$ is valid, where $E[A(I)]$ is the expected value of the solution returned by algorithm, we say that A is a probabilistic α-approximation algorithm.

Given a polynomial time algorithm A_ε, for fixed $\varepsilon > 0$, and an instance I for some problem P, we say that A_ε is a polynomial time approximation scheme (PTAS) for a minimization problem if for any $\varepsilon>0$ and any instance I we have $A_\varepsilon(I) \leq (1+\varepsilon)OPT(I)$. If the algorithm is also polynomial time in $1/\varepsilon$ we say that A_ε is a fully polynomial time approximation scheme (FPTAS).

For all problems considered, we denote by $J = \{1,\dots,n\}$ the set of jobs and $M=\{1,\dots m\}$ the set of machines. For the case where the machines are unrelated, we denote by p_{ij} the processing time of the job j when executed on machine i. When all machines are identical, we denote this processing time by p_j. For some problems, there is a release date r_j, for each job j, which is a time where the job j cannot be scheduled before. The value w_j is the importance weight of finishing the job j earlier and the completion time of the job is denoted by C_j.

Since we consider several scheduling problems, we use the notation $\alpha|\beta|\gamma$, introduced by Graham, Lawler, Lenstra & Rinnooy Kan [GLLR79], to denote each problem. In the following, we detail the terms used in this paper under this notation. The term α corresponds to the machine environment, P for identical machines or R for unrelated machines. The term β tell us some restrictions about jobs, if they have release dates, r_j, if the schedule is preemptive (i.e., jobs can be interrupted and continued later), ***pmtn***, etc. Finally the term γ indicates the objective function we want to minimize.

All problems we consider are non-preemptive, although algorithms for preemptive problems are used to find intermediate solutions.

There are many papers describing approximation algorithms for scheduling problems, but few consider practical performance analysis. In [HS01], Hepner & Stein presented an implementation of a PTAS for the problem $1|r_j|\Sigma C_j$. Savelsbergh *et al.* [SUW98] also presented an experimental study of approximation algorithms for the problem $1|r_j|\Sigma w_jC_j$ and a variant of this problem when the average weighted flow time is minimized, i.e. problem $1|r_j|\Sigma w_j(C_j-r_j)$. Recently, Vredeveld & Hurkens [VH02] presented an experimental comparison of approximation algorithms for the problem $R||\Sigma w_jC_j$ and some dominance relations between linear and quadratic formulations for this problem. Baev *et al.* [BME02] presented a practical comparison for the problem $P|prec|\Sigma w_jC_j$ where the jobs have precedence constraints. They also show how algorithms for this problem can be used in the scheduling phase in profile-based program compilation. They used some instances extracted from the SPECint95 compiler benchmark and showed that the best solutions are within 5.7% of optimal.

We implemented algorithms for the following problems: $P|r_j|\Sigma C_j$, $P||\Sigma w_jC_j$, $P|r_j|\Sigma w_jC_j$, $R||\Sigma w_jC_j$ and $R|r_j|\Sigma w_jC_j$. For the problem $P|r_j|\Sigma C_j$ we implemented the algorithm developed by Phillips *et al.* [PSW98]. This algorithm is combinatorial and is based on a heuristic for the preemptive case. For the problem $P||\Sigma w_jC_j$ we implemented the algorithm of Kawaguchi & Kyan [KK86], that is based on a list scheduling heuristic. For the problems $P|r_j|\Sigma w_jC_j$ and $R|r_j|\Sigma w_jC_j$ we implemented algorithms of Schulz & Skutella [SS02]. The algorithm for the first problem is combinatorial and the algorithm for the second problem is based on a solution of a linear program. Both algorithms are probabilistic. Finally, for the problem $R||\Sigma w_jC_j$ we implemented the algorithm developed by Skutella [S98] that is based on a solution of a semidefinite program.

We chose to implement these algorithms because they are well known approximation algorithms, with good time complexity and good approximation factors. Also, the set of algorithms chosen, treat problems that have common cases and this permits to compare them. Some problems we consider are particular cases of others. So, implemented algorithms for more general problems are also compared with algorithms for more restricted problems. There are other approximation algorithms for some of these problems like the polynomial time approximation schemes for parallel machines presented by Afrati *et al.* [Afrati *et al.*, 1999]. These schemes appear to have only theoretical interest, since their running times are given by high degree polynomials. In fact, most of these schemes require an enumeration step that is intolerable in practice.

To our knowledge, this paper is the first to consider a practical comparison of approximation algorithms for scheduling problems with parallel machines and release dates (problems $P|r_j|\Sigma C_j$ and $R|r_j|\Sigma w_jC_j$). We also consider a practical study of two formulations that provide lower bounds for the problem $R||\Sigma w_jC_j$. Notice that Vredeveld & Hurkens [VH02], also studied these formulations presenting dominance relations among them, but they considered an exponential size linear formulation. In this paper we consider the same formulation with a small modification which leads to a formulation of polynomial size.

All algorithms are implemented in C. For the algorithms that require solutions of linear or quadratic programs we use the Xpress-MP library, of Dash Optimization [D02]. Based on the practical results, we propose a simple modification on the algorithm presented by Schulz & Skutella [SS02] for the problem $R|r_j|\Sigma w_jC_j$ and on the algorithm of Kawaguchi & Kyan [KK86]. In the tests we considered, we show that these heuristics obtain solutions with better quality.

The paper is organized as follows. In section 2 we describe the implemented algorithms and give some insight of how they work. In section 3 we compare two lower bounds for the problem $R||\Sigma w_jC_j$ with different number of machines. In section 4 we present the computational results of the implemented algorithms.

2. Algorithms

In this section we describe the algorithms and the way they are implemented. We do not show how their approximation factors are obtained. The interested reader can find more details about the approximation results of these algorithms in the references.

2.1 Algorithm PSW for the problem $P|r_j|\Sigma C_j$

The algorithm of this section, which we denote by PSW, was developed by Phillips *et al.* [PSW98]. The algorithm PSW finds a solution in two phases. In the first phase, it obtains an approximate solution for the preemptive version of this problem and in the second phase it uses an algorithm that converts the preemptive schedule to a non-preemptive one. The preemptive version of this problem is already *NP*-hard, and a solution is generated by a 2-approximation algorithm. The algorithm that converts the preemptive schedule to a non-preemptive one, produces a new schedule that is at most three times worse than the preemptive schedule. This leads to a 6-approximation algorithm for the problem $P|r_j|\Sigma C_j$ (see [PSW98]).

The algorithm for the preemptive schedule is based on the following idea: at any time, execute m jobs with the shortest remaining amount of work. The time complexity of the implemented algorithm, which we denote by *Preemptive*, is $O(n(\log n + m))$.

Once this preemptive schedule is generated, the algorithm generates a list M_i, for each machine i, of jobs ordered by their preemptive completion times. For each machine i, the algorithm PSW generates a non-preemptive schedule with jobs in the order specified by M_i, under the condition that no job starts before its release date. The time complexity of the implemented algorithm is $O(n\log n + m)$ plus the time complexity to generate the preemptive schedule.

2.2 Algorithm KK for the problem $P||\Sigma w_jC_j$

The algorithm of this section is an extension of the optimal algorithm for the problem $1||\Sigma w_jC_j$. The problem $1||\Sigma w_jC_j$ can be solved optimally with the following algorithm developed by Smith [S56]: order jobs in non-decreasing order of p_j/w_j and schedule the jobs in this order. The approximation algorithm for the parallel machine case is an extension: order jobs in non-decreasing order of p_j/w_j and schedule jobs in this order every time a machine becomes free. Kawaguchi & Kyan [KK86] have shown that this algorithm generates schedules with a factor of $(\sqrt{2}+1)/2$ of the optimal. The implemented algorithm, which we denote by KK, has time complexity $O(n\log n + n\log m)$.

2.3 Algorithm SZSK for the problem $P|r_j|\Sigma w_jC_j$

The algorithm SZSK is a probabilistic 2-approximation algorithm and was developed by Schulz & Skutella [SS02]. For each instance, the algorithm SZSK is executed 100 times and the best generated schedule is returned. In our experiments, we observed that more executions leads to very small improvements. The algorithm is related to the linear formulation for a single machine problem presented below. We have variables y_{jt}, for each job j and for each time interval $(t, t+1]$ that a job can run. We also have variables C_j, that represent the finishing time of job $\boldsymbol{j}$. The constant $\boldsymbol{T}$ is an upper bound for the completion time of any job. The relaxed linear program, denoted by *LPS*, is the following:

$$\text{Min} \sum_{j \in J} w_j C_j$$

$$
\begin{array}{lll}
(LPS) & \sum_{t=rj}^{T} y_{jt} = p_j & \forall j \in J, \\
& \sum_{j \in J} y_{jt} \le 1 & t = 0,\dots,T, \\
& C_j = p_j/2 + 1/p_j \sum_{t=rj}^{T} y_{jt}(t+1/2) & \forall j \in J, \\
& y_{jt} = 0 & \forall j \in J \text{ and } t = 0,\dots,r_j-1, \\
& y_{jt} \ge 0 & \forall j \in J \text{ and } t = r_j,\dots,T.
\end{array}
$$

The linear program (LPS) can be solved using a combinatorial algorithm [SS02]. Suppose we have only one machine that is $\boldsymbol{m}$ times faster than the machines considered. Consider the processing times of the jobs to be $\boldsymbol{m}$ times smaller. Construct a preemptive schedule for this single machine with the new processing times using the following rule: at any time, generate a preemptive schedule on the new single machine by scheduling, among the available jobs, the one with the smallest ratio $\boldsymbol{p_j/w_j}$. The resulting schedule corresponds to an optimal solution for the formulation. Each variable $\boldsymbol{y_{jt}}$ receives value 1 if job $\boldsymbol{j}$ is processed during time $\boldsymbol{[t-1,t)}$ in the generated schedule.

Notice that the algorithm *Preemptive* is easily modified to solve this formulation and can be implemented to run in $O(n\log n)$. After this, we construct a schedule based on probabilistic assignments. We choose for each job $\boldsymbol{j}$, a variable α_j uniformly distributed from the interval [0,1]. Then, we consider the probabilistic finishing time, i.e., the first time in the schedule where the total amount of work done is $\boldsymbol{p_j \alpha_j}$. We denote this value by $\boldsymbol{C_j(\alpha_j)}$. The algorithm SZSK attributes each job $\boldsymbol{j}$ uniformly and independently to one of the $\boldsymbol{m}$ machines. For each machine the algorithm schedules jobs in nondecreasing order of values $\boldsymbol{C_j(\alpha_j)}$. The time complexity of the algorithm SZSK is $O(n\log n + m)$.

2.4 Algorithm SK for the problem $R||\sum w_jC_j$

The algorithm of this section, which we denote by SK, is a probabilistic 2-approximation algorithm based on a semidefinite formulation. The algorithm was presented by Skutella [S98] and uses a quadratic program. This program has binary variables $\boldsymbol{a_{ij}}$, such that a job $\boldsymbol{j}$ is to be processed in machine $\boldsymbol{i}$, if and only if, $\boldsymbol{a_{ij}} = 1$, and variables $\boldsymbol{C_j}$ that represent the finishing time of job $\boldsymbol{j}$. We also have a function $\langle_i$ that specifies the execution order of a job pair $\boldsymbol{j,k}$ in machine $\boldsymbol{i}$. The job $\boldsymbol{j}$ must be processed before $\boldsymbol{k}$ in machine $\boldsymbol{i}$ if $\boldsymbol{w_j/p_{ij} \ge w_k/p_{ik}}$. The quadratic program is the following:

$$\text{Min} \sum_{j \in J} w_j C_j$$

$$
\begin{array}{ll}
C_j = \sum_{i=1}^{m} a_{ij}(p_{ij} + \sum_{k \langle j} a_{ik} p_{ik}) & \forall j \in J \\
a_{ij} \in \{0,1\} & \forall i \in M \quad \forall j \in J.
\end{array}
$$

Skutella have shown that this formulation is equivalent to the following quadratic formulation:

$$\begin{aligned} & Min\ c^T a + ½\, a^T D a \\ & \textstyle\sum_{i=1}^{m} a_{ij} = 1 \qquad \forall j \in J, \\ & a \geq 0, \end{aligned}$$

where $\boldsymbol{a} \in \Re^{mn}$ is a vector of all variables $\boldsymbol{a_{ij}}$ lexicographically ordered with respect to the natural order $\boldsymbol{1,2,\ldots,m}$ of the machines, and then for each machine $\boldsymbol{i}$, the jobs are ordered according to $\prec_i$. The vector $\boldsymbol{c} \in \Re^{mn}$ is given by $\boldsymbol{c_{ij} = w_j p_{ij}}$ and $\boldsymbol{D = (d_{(ij)(hk)})}$ is a symmetric $(mn \times mn)$-matrix given by: *(i)* 0 if $\boldsymbol{i \neq j}$ or $\boldsymbol{j = k}$; *(ii)* $\boldsymbol{w_j p_{ik}}$ if $\boldsymbol{i = h}$ and $\boldsymbol{k \prec_i j}$; *(iii)* $\boldsymbol{w_k p_{ij}}$ if $\boldsymbol{i = h}$ and $\boldsymbol{j \prec_i k}$.

This problem can be solved in polynomial time if, and only if, matrix $\boldsymbol{D}$ is positive semidefinite. This motivates the construction of a new formulation, which we call *QSP*:

$$(QSP) \quad \begin{aligned} & Min\ ½\, c^T a + ½\, a^T (D + diag(c)) a \\ & \textstyle\sum_{i=1}^{m} a_{ij} = 1 \qquad \forall j \in J, \\ & a \geq 0, \end{aligned}$$

where $\boldsymbol{(D + diag(c))}$ is positive semidefinite and $\boldsymbol{diag(c)}$ is a diagonal matrix with the vector $\boldsymbol{c}$.

Given a solution for *QSP*, each job $\boldsymbol{j}$ is assigned to machine $\boldsymbol{i}$ with probability $\boldsymbol{a_{ij}}$ and in each machine $\boldsymbol{i}$ the execution order is given by the function $\prec_i$. In our implementation, this assignment is performed 100 times and the algorithm returns the best generated schedule. For the special case of identical parallel machines, the optimal solution of the above formulation is given by $\boldsymbol{a_{ij} = 1/m}$. In this case, we implemented a combinatorial algorithm attributing each job to a machine with probability $\boldsymbol{1/m}$. This combinatorial algorithm is denoted by SK-C. The time complexity of the algorithm is $O(n \log n + m)$ plus the time complexity to solve the semidefinite program *QSP*.

2.5 Algorithm SZSK2 for $R|r_j|\Sigma w_j C_j$

The algorithm for the problem $\boldsymbol{R|r_j|\Sigma w_j C_j}$ is also a probabilistic algorithm, and was presented by Schulz and Skutella [SS02]. The algorithm, denoted by SZSK2, is based on the solution of a linear formulation and is a generalization of the algorithm SZSK. The formulation uses an upper bound $\boldsymbol{T}$ on the completion time of any job and uses variables $\boldsymbol{C_j}$, representing the finishing time of each job $\boldsymbol{j}$, and variables $\boldsymbol{y_{ijt}}$ that indicates if job $\boldsymbol{j}$ is being executed in machine $\boldsymbol{i}$ at time interval $\boldsymbol{(t,t+1]}$ for each time interval. The formulation has exponential size, but it can be made of polynomial size with a small loss in the objective function, using interval times that increase exponentially in their size. In this case, we have binary variables $\boldsymbol{y_{ijl}}$ indicating the execution of job $\boldsymbol{j}$ in machine $\boldsymbol{i}$ at interval $\boldsymbol{I_l = ((1+\beta)^{l-1},(1+\beta)^l]}$. The size of an interval $\boldsymbol{I_l}$ is denoted by $|\boldsymbol{I_l}|$. For simplicity, we denote $\boldsymbol{(1+\beta)^l}$ by $\boldsymbol{\beta_l}$. The relaxed formulation, denoted by *LPSS*, is the following:

$$
\text{(LPSS)}\quad
\begin{array}{lll}
Min \sum_{j=1}^{n} w_j C_j & & \\
\sum_{i=1}^{m} \sum_{l=0}^{L} (y_{ijl}|I_l|)/p_{ij} = 1 & & \forall j \in J, \\
\sum_{j\in J} y_{ijl} \le 1 & & \forall i \in M \ \ and \ \ l=0,\ldots,L, \\
C_j = \sum_{i=1}^{m} \sum_{l=0}^{L} \left((y_{ijl}|I_l|/p_{ij})\beta_{l-1} + \tfrac{1}{2} y_{ijl}|I_l| \right) & & \forall j \in L, \\
y_{ijl} = 0 & & \forall i \in M, \ \forall j \in J, \ \beta_l \le r_{ij}-1, \\
y_{ijl} \ge 0 & & \forall i \in M, \ \forall j \in J, \ l=0,\ldots,L.
\end{array}
$$

The algorithm solves the linear program *LPSS* and assign each job ***j*** to a machine-interval pair (i, I_l) at random with probability $(y_{ijl}|I_l|)/p_{ij}$. The jobs assigned to a machine ***i*** are scheduled in non-decreasing order of intervals assignment. If there is more than one job assigned to the same pair (i, I_l), the algorithm schedules them in the order of their values ***j***. For a given $\varepsilon>0$, setting $\beta=\varepsilon/2$ this algorithm has a probabilistic $(2+\varepsilon)$-approximation factor. As in the algorithm SK, the probabilistic assignment step is executed 100 times and the best generated schedule is returned. The time complexity of this algorithm is $O(nm\log_{(1+\varepsilon)}T + n\log n)$ plus the time complexity to solve the linear program *LPSS*. Since this algorithm is executed with different values of ε, we denote by SZSK2$_\varepsilon$ the algorithm SZSK2 with the given value of ε. That is, the algorithm SZSK2$_{0.1}$ is the algorithm SZSK2 with value of $\varepsilon = 0.1$.

2.6 Two Heuristic Algorithms

In this section we present a new algorithm denoted by HE1 for the problem $R|r_j|\Sigma w_jC_j$. It is a simple modification of the algorithm SZSK2. We also present an extended heuristic of the algorithm KK for the problem $P|r_j|\Sigma w_jC_j$, denoted by HE2.

In [HP83], Hariri and Potts presented a simple heuristic algorithm for problem $1|r_j|\Sigma w_jC_j$ used to find an upper bound for a branch and bound algorithm. The algorithm is as follows:

1. Let S be the set of all (unsequenced) jobs, $H=0$ and $k=0$ and find $T=\min_{j\in S}\{r_j\}$.
2. Let the set $S' = \{j \mid j\in S, r_j \le T\}$ and find a job $i\in S'$ such that $w_i/p_i = max_{j\in S'}\{w_j/p_j\}$.
3. Let $k = k+1$ and sequence job i at position k; let $T = T + p_i$, $H = H + w_iT$ and $S = S-\{i\}$.
4. If $S=\emptyset$, then stop with the sequence generated having H as its cost. Otherwise let $T=\max\{T, \min_{j\in S}\{r_j\}\}$ and go to step 2.

In the algorithm SZSK2, the jobs are assigned to pairs machine-interval and them executed in each machine by the order of interval assignments. In the algorithm HE1, the assignment step is performed as in the algorithm SZSK2, but the jobs assigned to a machine ***i*** are scheduled using the algorithm of Hariri and Potts.

The algorithm HE2 is an extended heuristic of algorithm KK: every time a machine becomes free, execute among the available jobs, the one with smallest ratio p_j/w_j. Notice that without the presence of release dates, this algorithm is essentially algorithm KK.

Notice that we cannot guarantee approximation factors for these two heuristics. Since we changed the way the schedules are generated, some properties of the schedule are lost. These properties are essential in the analysis of their approximation factor. To prove that these heuristics have approximation factors is not a trivial step and an entire paper can be devoted to this subject.

3. Study of Two Lower Bounds

In this section we present an experimental comparison of two formulations that provide lower bounds for the implemented algorithms. The first formulation is the semidefinite formulation *QSP* used in the algorithm SK, and the second is a linear formulation *LPSS* used in the algorithm SZSK2. For problems that consider jobs with release dates we used the lower bounds provided by the linear program *LPSS*. For problems without release dates we performed a computational study to determine which formulation gives lower bounds with better quality. We notice that Vredeveld & Hurkens [VH02] proved that the formulation *LPSS* with unit time interval gives better lower bounds than the formulation *QSP*. But in this case, the formulation *LPSS* has exponential size and the time required to solve the instances may be very high. We performed tests with $LPSS_{\varepsilon}$ and formulation *QSP* for $\varepsilon \in \{0.3, 0.1\}$. In this case, where $\varepsilon > 0$ in the formulation *LPSS*, it is not true that *LPSS* gives better bounds. For the most generic problem $R||\Sigma w_j C_j$, we consider three cases: $R2||\Sigma w_j C_j$, $R5||\Sigma w_j C_j$ and $R7||\Sigma w_j C_j$. We also tried to study the case $R10||\Sigma w_j C_j$ but we could not solve integer instances of this problem in a reasonable amount of time (two hours). We performed five tests with 100 jobs for each case. The processing times of jobs were taken uniformly from the interval [1,100] and w_j was uniformly chosen from the interval [1,10]. We notice that the quality of the lower bound increases using $\varepsilon = 0.1$ when compared with the solutions with $\varepsilon = 0.3$, but *QSP* provides better lower bounds. We tried to solve the instances with the formulation *LPSS* with smaller values of ε, but when $\varepsilon \to 0$, the number of time intervals increases in such a way that is better to consider unit time intervals. The use of the $\varepsilon > 0$ in the formulation *LPSS* is justified since we are comparing lower bounds obtained in polynomial time. We present the results obtained in at most two hours.

The lower bounds of these two formulations are compared with the value of an integer solution, which we obtained from the integral solutions of program *QSP*. The results of these tests can be seen in Table 1.

Table 1 – Comparison between formulations QSP and LPSS.

Problem	Integer Optimal	*QSP*		*LPSS* $\epsilon = 0.3$		*LPSS* $\epsilon = 0.1$	
$R2\|\| \sum w_j C_j$		Value	Ratio	Value	Ratio	Value	Ratio
	163066	163052.92	0.999	152588.46	0.935	159313.82	0.976
	228766	228732.05	0.999	214004.87	0.935	223453.15	0.976
	223714	223673.35	0.999	209223.01	0.935	218505.52	0.976
	174802	174764.49	0.999	163503.50	0.935	170744.85	0.976
	180367	180337.23	0.999	168738.85	0.935	176189.91	0.976
$R5\|\| \sum w_j C_j$		Value	Ratio	Value	Ratio	Value	Ratio
	36767	36690.74	0.997	34506.58	0.938	35914.83	0.978
	33675	33636.31	0.998	31603.44	0.938	32925.71	0.977
	44130	44043.36	0.998	41379.21	0.937	43097.67	0.976
	36168	36104.74	0.998	33948.00	0.938	35340.76	0.977
	37343	37251.78	0.997	35028.43	0.938	36457.91	0.976
$R7\|\| \sum w_j C_j$		Value	Ratio	Value	Ratio	Value	Ratio
	26542	26473.41	0.997	24920.74	0.938	25924.81	0.976
	22429	22361.25	0.996	21074.51	0.939	21915.70	0.977
	26919	26857.20	0.997	25279.07	0.939	26302.32	0.977
	29093	29017.66	0.997	27314.76	0.938	28415.94	0.976
	25543	25440.19	0.995	23967.79	0.938	24928.01	0.975

We also performed computational tests to compare the lower bounds for the problem ***$P||\Sigma w_jC_j$***. In this case, we could solve only instances up to 20 jobs with 2 machines, and 15 jobs with 5 machines. The next theorem, proved by Skutella [S98], helps us to understand the hardness to obtain integer solutions for instances of this problem.

Theorem 3.1 *For instances of $Pm||\Sigma w_jC_j$, an optimal vector solution **a** of the quadratic program QSP is $a_{ij}=1/m$ for all **i,j**. This optimum solution is unique if all ratios p_j/w_j, are different and positive.*

In all instances, the solution of the quadratic program is exactly the one provided in the theorem. Since the Xpress solver finds the optimal integer solution using a branch and bound tree, the number of nodes is exponential. We could not solve these kind of problems even if we use an upper bound provided by our approximation algorithms. We could solve only instances with 20 jobs for the problem ***$P2||\Sigma w_jC_j$*** and instances with 15 jobs for the problem ***$P5||\Sigma w_jC_j$***. The results of our tests are presented in Table 2.

Table 2 – Comparison between formulations QSP and LPSS.

Problem	Integer Optimal	*QSP*		*LPSS* $\epsilon = 0.3$		*LPSS* $\epsilon = 0.1$	
		Value	Ratio	Value	Ratio	Value	Ratio
$P2\|\|\sum w_jC_j$	19615	19546.75	0.996	18413.86	0.938	19142.58	0.975
	19214	19164.00	0.997	18082.13	0.941	18773.20	0.977
	17199	17135.75	0.996	16158.01	0.939	16785.03	0.975
	16398	16322.50	0.995	15385.34	0.938	15992.72	0.975
	16415	16365.00	0.996	15451.82	0.941	16033.15	0.976
		Value	Ratio	Value	Ratio	Value	Ratio
$P5\|\|\sum w_jC_j$	5121	4913.60	0.959	4712.12	0.920	4842.30	0.945
	5467	5257.60	0.961	5035.09	0.920	5177.84	0.947
	6536	6312.40	0.965	6039.39	0.924	6215.11	0.950
	4875	4651.60	0.954	4468.68	0.916	4590.74	0.941
	5105	4895.80	0.959	4694.55	0.919	4824.64	0.945

In all generated tests, the lower bounds provided by the formulation *QSP* are better than the lower bounds provided by the formulation *LPSS*. Also notice that when ***ε=0.1***, the difference is not so large. We do not use smaller values of ε since the increase in the computational time to solve such formulations is high (more than two hours of computational processing).

4. Practical Analysis of the Implemented Algorithms

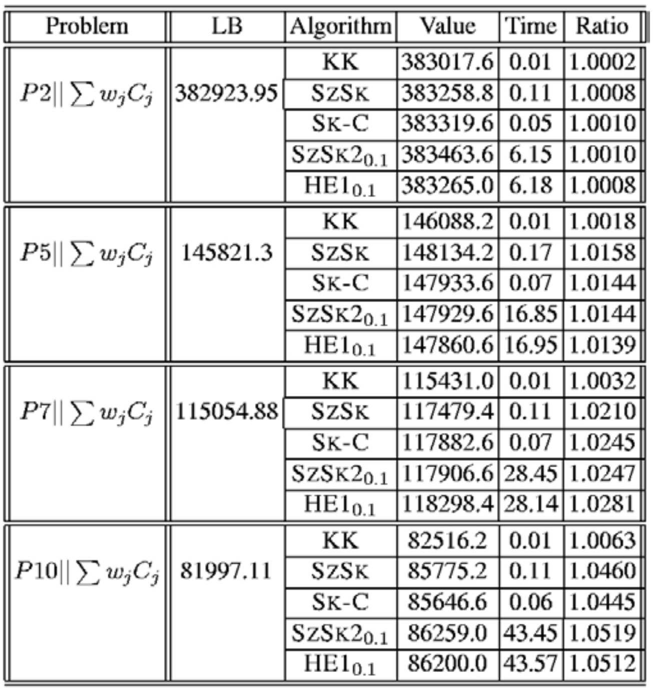
In this section we present the results of our tests. Since some problems are particular cases of others, we performed several different tests. Each subsection is reserved for one case. Before presenting the computational results for each problem, we describe the procedure to generate each test. For each test, we generate 100 jobs with processing times uniformly chosen from the interval [1,100] and w_j chosen from the interval [1,10]. When the problem require release dates, the data is generated using the same approach used by Hariri & Potts [HP83]. The release dates are uniformly chosen from the interval [0,*E[p]n*γ]. This simulates the arrival of ***n*** jobs from a stable queue according to a Poisson process with parameter γ [HS01]. The time in all tables is given in seconds. The ratio in the table corresponds to ***V/LB***, where ***V*** is the value found by the algorithm and ***LB*** is a lower bound for the optimal solution. We performed tests with 2, 5, 7 and 10 machines. As was done in [HS01], we generated five different instances for each test problem, so the results in each line of the tables corresponds to the mean of five tests. The algorithms were tested on an AMD Athlon 1.2GHz with 800 MB of RAM under Linux 2.4.2-2 kernel.

4.1 Tests for the problem $P||\Sigma w_jC_j$

In this problem we used the algorithms KK, SZSK, SK-C, SZSK2 and HE1. We do not use the algorithm HE2 here because without the presence of release dates this algorithm generates the same solutions of the algorithm KK. The Table 3 presents the results of these tests. The LB column corresponds to the optimal fractional solution of the quadratic formulation *QSP*.

The algorithms obtained very good results for all tested instances. The algorithm KK is the most simple and obtained the best results generating solutions with values less than 0.7% of the lower bounds, besides the other algorithms use more advanced ideas. As we can see, the ratio grows when we use more machines. For algorithm KK the increase is very small. For the other ones the growth is more representative. We believe that with more jobs per machine the ratios obtained tends to decrease. This can be seen in graphics 1, 2, 3 and 4. We will describe more about this behavior in the next subsection.

Table 3 – Comparison for the problem $P||\Sigma w_jC_j$.



Problem	LB	Algorithm	Value	Time	Ratio
$P2\|\|\sum w_jC_j$	382923.95	KK	383017.6	0.01	1.0002
		SZSK	383258.8	0.11	1.0008
		SK-C	383319.6	0.05	1.0010
		$\text{SZSK2}_{0.1}$	383463.6	6.15	1.0010
		$\text{HE1}_{0.1}$	383265.0	6.18	1.0008
$P5\|\|\sum w_jC_j$	145821.3	KK	146088.2	0.01	1.0018
		SZSK	148134.2	0.17	1.0158
		SK-C	147933.6	0.07	1.0144
		$\text{SZSK2}_{0.1}$	147929.6	16.85	1.0144
		$\text{HE1}_{0.1}$	147860.6	16.95	1.0139
$P7\|\|\sum w_jC_j$	115054.88	KK	115431.0	0.01	1.0032
		SZSK	117479.4	0.11	1.0210
		SK-C	117882.6	0.07	1.0245
		$\text{SZSK2}_{0.1}$	117906.6	28.45	1.0247
		$\text{HE1}_{0.1}$	118298.4	28.14	1.0281
$P10\|\|\sum w_jC_j$	81997.11	KK	82516.2	0.01	1.0063
		SZSK	85775.2	0.11	1.0460
		SK-C	85646.6	0.06	1.0445
		$\text{SZSK2}_{0.1}$	86259.0	43.45	1.0519
		$\text{HE1}_{0.1}$	86200.0	43.57	1.0512

4.2 Tests for the problem $P|r_j|\Sigma C_j$

To solve this problem we used the algorithms PSW, SZSK, SZSK2, HE1 and HE2. Although the algorithm SZSK is the combinatorial version of the algorithm SZSK2 for identical machines, we also included the algorithm SZSK2 in the comparisons. The algorithms SZSK2 and HE1 were executed with parameter ε=0.3 and ε=0.1. We perform different tests using different values of γ to generate the release dates. We used γ=0.2, γ=0.4 and γ=0.6. The LB column has the values of the optimal solutions of the linear program *LPSS*, with ε=0.1. It is interesting to notice that this lower bound may be far away from the optimum, since the value of an optimal integer solution for the program *LPSS* is already a lower bound for the original problem $P|r_j|\Sigma C_j$. The Tables 4, 5 and 6 present the results obtained for these tests.

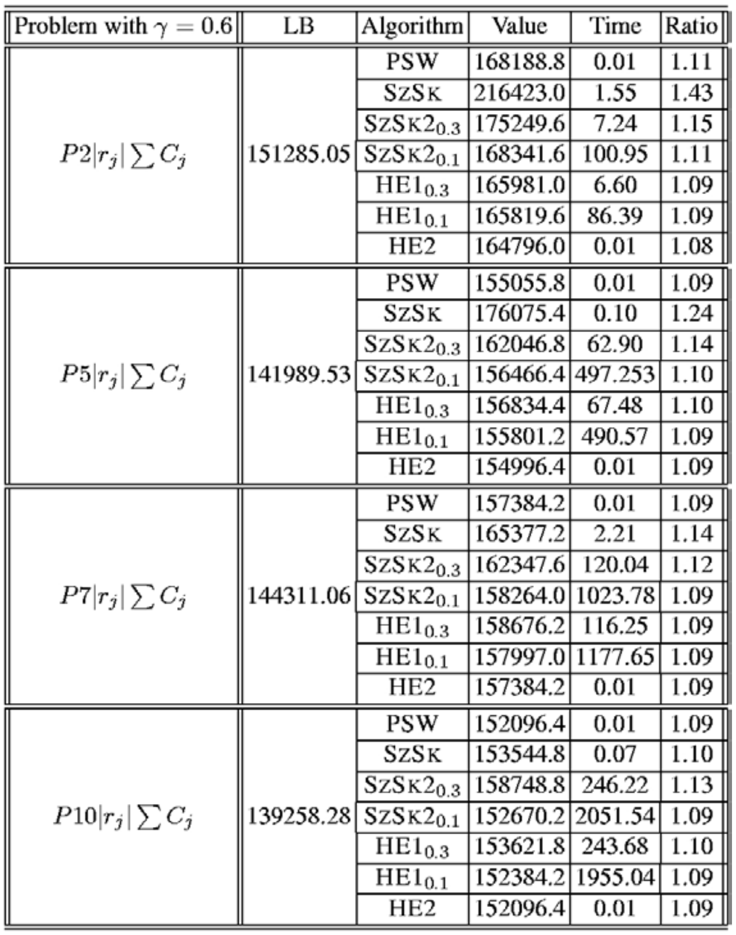
Table 4 – Comparison for the problem $P|r_j|\Sigma C_j$ with γ =0.2.

Problem with $\gamma = 0.2$	LB	Algorithm	Value	Time	Ratio
$P2\|r_j\|\sum C_j$	100161.56	PSW	109136.8	0.01	1.08
		SzSk	124153.6	0.01	1.23
		$\text{SzSk2}_{0.3}$	113298.6	5.08	1.13
		$\text{SzSk2}_{0.1}$	108391.2	78.45	1.08
		$\text{HE1}_{0.3}$	105280.4	5.79	1.05
		$\text{HE1}_{0.1}$	105276.4	79.72	1.05
		HE2	104981.4	0.01	1.04
$P5\|r_j\|\sum C_j$	52569.48	PSW	60768.4	0.01	1.15
		SzSk	75553.8	0.25	1.43
		$\text{SzSk2}_{0.3}$	63130.6	58.19	1.20
		$\text{SzSk2}_{0.1}$	62362.6	425.89	1.18
		$\text{HE1}_{0.3}$	60418.6	52.95	1.14
		$\text{HE1}_{0.1}$	60651.0	431.40	1.15
		HE2	57960.8	0.01	1.10
$P7\|r_j\|\sum C_j$	51345.58	PSW	57953.6	0.01	1.12
		SzSk	68479.4	0.01	1.33
		$\text{SzSk2}_{0.3}$	59530.8	90.13	1.15
		$\text{SzSk2}_{0.1}$	59246.2	807.20	1.15
		$\text{HE1}_{0.3}$	58486.2	90.17	1.13
		$\text{HE1}_{0.1}$	58983.2	819.64	1.14
		HE2	57204.4	0.01	1.11
$P10\|r_j\|\sum C_j$	47731.29	PSW	53526.4	0.01	1.12
		SzSk	62120.2	0.24	1.30
		$\text{SzSk2}_{0.3}$	55645.2	171.29	1.16
		$\text{SzSk2}_{0.1}$	54684.6	1584.29	1.14
		$\text{HE1}_{0.3}$	54845.2	183.71	1.14
		$\text{HE1}_{0.1}$	54618.4	1611.62	1.14
		HE2	53494.6	0.01	1.12

Table 5 – Comparison for the problem $P|r_j|\Sigma C_j$ with $\gamma = 0.4$.

Problem with $\gamma = 0.4$	LB	Algorithm	Value	Time	Ratio
$P2\|r_j\|\sum C_j$	113585.05	PSW	126569.0	0.01	1.11
		SzSk	162040.8	1.73	1.42
		SzSk2$_{0.3}$	133894.2	6.52	1.17
		SzSk2$_{0.1}$	126938.2	86.24	1.11
		HE1$_{0.3}$	121682.4	6.11	1.07
		HE1$_{0.1}$	121691.0	95.55	1.07
		HE2	121034.2	0.01	1.06
$P5\|r_j\|\sum C_j$	94406.70	PSW	104561.0	0.01	1.10
		SzSk	124759.4	0.26	1.32
		SzSk2$_{0.3}$	107531.0	62.90	1.13
		SzSk2$_{0.1}$	104917.8	495.95	1.11
		HE1$_{0.3}$	105297.2	59.61	1.11
		HE1$_{0.1}$	104759.8	503.41	1.10
		HE2	103638.0	0.01	1.09
$P7\|r_j\|\sum C_j$	98514.48	PSW	108252.4	0.01	1.09
		SzSk	119628.8	2.15	1.21
		SzSk2$_{0.3}$	111726.6	112.95	1.13
		SzSk2$_{0.1}$	109100.8	952.18	1.10
		HE1$_{0.3}$	109449.6	107.21	1.11
		HE1$_{0.1}$	108890.2	967.13	1.10
		HE2	108235.0	0.01	1.09
$P10\|r_j\|\sum C_j$	94268.95	PSW	103936.8	0.01	1.10
		SzSk	107757.0	0.17	1.14
		SzSk2$_{0.3}$	107522.0	218.68	1.14
		SzSk2$_{0.1}$	104450.2	1912.04	1.10
		HE1$_{0.3}$	105247.4	221.41	1.11
		HE1$_{0.1}$	104419.4	1917.11	1.10
		HE2	103936.8	0.01	1.10

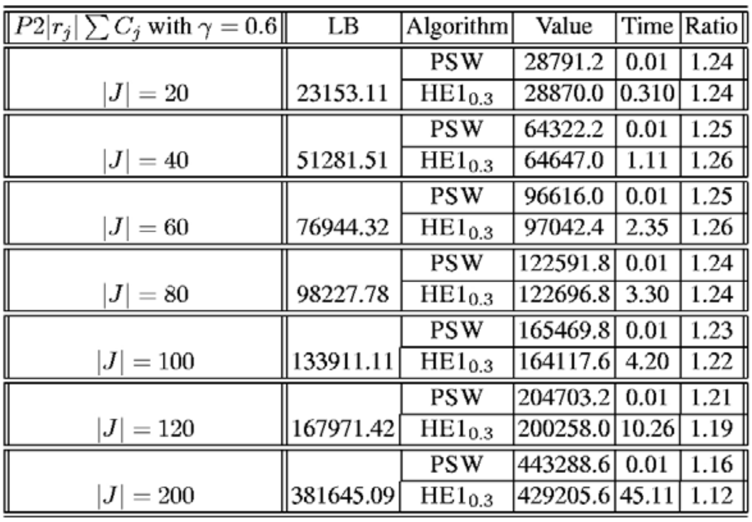
Table 6 – Comparison for the problem P|r_j|ΣC_j with γ = 0.6.



Problem with $\gamma = 0.6$	LB	Algorithm	Value	Time	Ratio
$P2\|r_j\|\sum C_j$	151285.05	PSW	168188.8	0.01	1.11
		SzSk	216423.0	1.55	1.43
		SzSk2$_{0.3}$	175249.6	7.24	1.15
		SzSk2$_{0.1}$	168341.6	100.95	1.11
		HE1$_{0.3}$	165981.0	6.60	1.09
		HE1$_{0.1}$	165819.6	86.39	1.09
		HE2	164796.0	0.01	1.08
$P5\|r_j\|\sum C_j$	141989.53	PSW	155055.8	0.01	1.09
		SzSk	176075.4	0.10	1.24
		SzSk2$_{0.3}$	162046.8	62.90	1.14
		SzSk2$_{0.1}$	156466.4	497.253	1.10
		HE1$_{0.3}$	156834.4	67.48	1.10
		HE1$_{0.1}$	155801.2	490.57	1.09
		HE2	154996.4	0.01	1.09
$P7\|r_j\|\sum C_j$	144311.06	PSW	157384.2	0.01	1.09
		SzSk	165377.2	2.21	1.14
		SzSk2$_{0.3}$	162347.6	120.04	1.12
		SzSk2$_{0.1}$	158264.0	1023.78	1.09
		HE1$_{0.3}$	158676.2	116.25	1.09
		HE1$_{0.1}$	157997.0	1177.65	1.09
		HE2	157384.2	0.01	1.09
$P10\|r_j\|\sum C_j$	139258.28	PSW	152096.4	0.01	1.09
		SzSk	153544.8	0.07	1.10
		SzSk2$_{0.3}$	158748.8	246.22	1.13
		SzSk2$_{0.1}$	152670.2	2051.54	1.09
		HE1$_{0.3}$	153621.8	243.68	1.10
		HE1$_{0.1}$	152384.2	1955.04	1.09
		HE2	152096.4	0.01	1.09

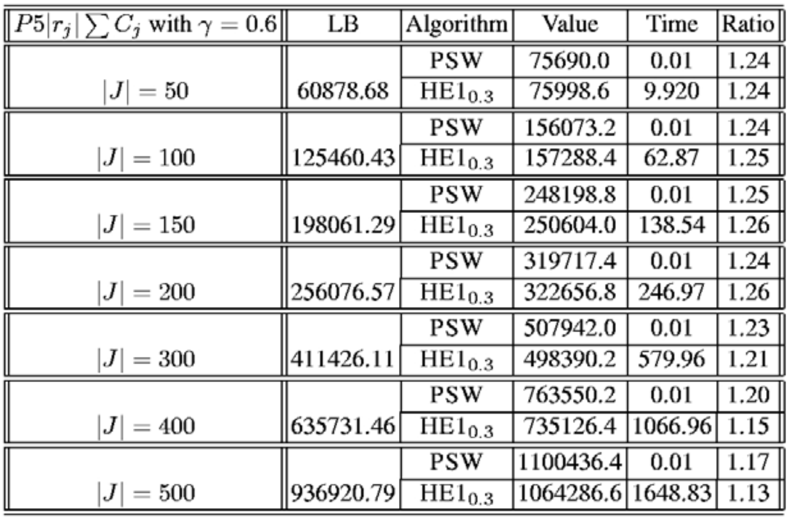
The algorithm HE2 generates the best schedules in all tests. Notice that the algorithm HE1 obtain better results when we have few machines and small values of γ. The algorithms PSW and HE1 are the second best in all cases. For all tests, the algorithm PSW generates solutions that are at most 12% of the lower bound although its approximation factor is 6. The algorithm SZSK2 obtained better results than the algorithm SZSK for all cases, except when we have big values of γ and more machines, as we can see in Table 6. Analyzing the fractional solution of the linear program used by the algorithm SZSK2, we observed that the solver obtained solutions where almost all variables for some machines have null values. Consequently, the generated schedule have some machines that are almost unused. The algorithm SZSK is the combinatorial version of SZSK2, but the jobs are attributed to all machines uniformly. This also explains why the algorithm HE1 when compared to the algorithm PSW, get better results using two machines than 7 and 10 machines. Based on this observation we try to solve the linear program *LPSS* under the algorithm HE1 with an increase in the number of jobs per machine. Notice that the algorithm HE1 is based on the algorithm SZSK2 and we can expect the same behavior in both algorithms. We performed several tests that can be seen in Tables 7, 8, 9 and 10. The interesting point to note is that when we get a ratio of approximately 60 jobs per machine, the algorithm HE1 produces better schedules. The solution of the linear program has a better attribution when this happens. We also present some graphics (Figures 1, 2, 3 and 4) that summarize these results. As we mentioned in the previous subsection, the algorithms get better results when we use more jobs per machine. This can be easily verified in these graphics. But it is important to note that when we compare the execution time, the algorithm PSW have a much better performance since it is a combinatorial algorithm, and the algorithm HE1 have to solve large linear programs. Notice that we could not solve all instances of the problem with a given ***ε*=0.3** in algorithm HE1. For example, in the tests with ten machines we used ***ε*=0.8** and the time to solve the corresponding linear program *LPSS* is very high. With such values, the lower bound provided by the linear program becomes worse and the ratios obtained for these tests are worse than the ones for the previous tests. We believe that the solutions obtained are closer to the optimum and better ratios could be obtained with better lower bounds.

Table 7 – Comparison between PSW and HE1 with 2 machines.



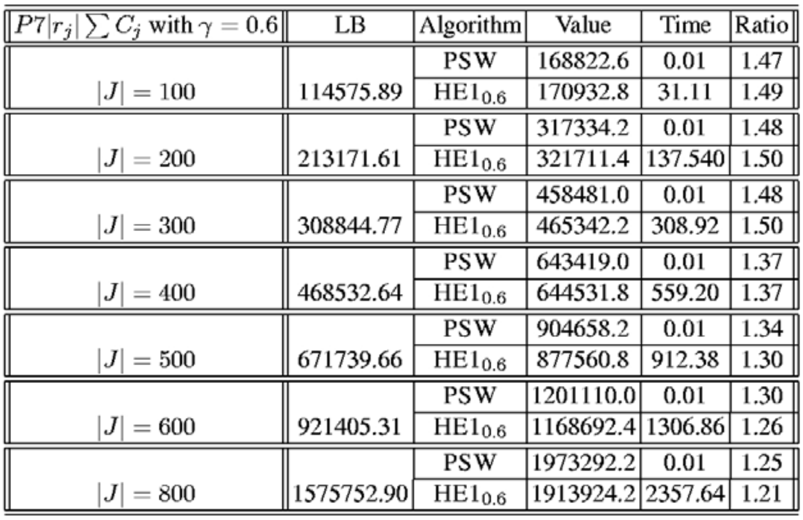
$P2\|r_j\|\sum C_j$ with $\gamma = 0.6$	LB	Algorithm	Value	Time	Ratio
$\|J\| = 20$	23153.11	PSW	28791.2	0.01	1.24
		$HE1_{0.3}$	28870.0	0.310	1.24
$\|J\| = 40$	51281.51	PSW	64322.2	0.01	1.25
		$HE1_{0.3}$	64647.0	1.11	1.26
$\|J\| = 60$	76944.32	PSW	96616.0	0.01	1.25
		$HE1_{0.3}$	97042.4	2.35	1.26
$\|J\| = 80$	98227.78	PSW	122591.8	0.01	1.24
		$HE1_{0.3}$	122696.8	3.30	1.24
$\|J\| = 100$	133911.11	PSW	165469.8	0.01	1.23
		$HE1_{0.3}$	164117.6	4.20	1.22
$\|J\| = 120$	167971.42	PSW	204703.2	0.01	1.21
		$HE1_{0.3}$	200258.0	10.26	1.19
$\|J\| = 200$	381645.09	PSW	443288.6	0.01	1.16
		$HE1_{0.3}$	429205.6	45.11	1.12

Table 8 – Comparison between PSW and HE1 with 5 machines.



$P5\|r_j\|\sum C_j$ with $\gamma = 0.6$	LB	Algorithm	Value	Time	Ratio
$\|J\| = 50$	60878.68	PSW	75690.0	0.01	1.24
		$HE1_{0.3}$	75998.6	9.920	1.24
$\|J\| = 100$	125460.43	PSW	156073.2	0.01	1.24
		$HE1_{0.3}$	157288.4	62.87	1.25
$\|J\| = 150$	198061.29	PSW	248198.8	0.01	1.25
		$HE1_{0.3}$	250604.0	138.54	1.26
$\|J\| = 200$	256076.57	PSW	319717.4	0.01	1.24
		$HE1_{0.3}$	322656.8	246.97	1.26
$\|J\| = 300$	411426.11	PSW	507942.0	0.01	1.23
		$HE1_{0.3}$	498390.2	579.96	1.21
$\|J\| = 400$	635731.46	PSW	763550.2	0.01	1.20
		$HE1_{0.3}$	735126.4	1066.96	1.15
$\|J\| = 500$	936920.79	PSW	1100436.4	0.01	1.17
		$HE1_{0.3}$	1064286.6	1648.83	1.13

Table 9 – Comparison between PSW and HE1 with 7 machines.



$P7\|r_j\|\sum C_j$ with $\gamma = 0.6$	LB	Algorithm	Value	Time	Ratio
$\|J\| = 100$	114575.89	PSW	168822.6	0.01	1.47
		$HE1_{0.6}$	170932.8	31.11	1.49
$\|J\| = 200$	213171.61	PSW	317334.2	0.01	1.48
		$HE1_{0.6}$	321711.4	137.540	1.50
$\|J\| = 300$	308844.77	PSW	458481.0	0.01	1.48
		$HE1_{0.6}$	465342.2	308.92	1.50
$\|J\| = 400$	468532.64	PSW	643419.0	0.01	1.37
		$HE1_{0.6}$	644531.8	559.20	1.37
$\|J\| = 500$	671739.66	PSW	904658.2	0.01	1.34
		$HE1_{0.6}$	877560.8	912.38	1.30
$\|J\| = 600$	921405.31	PSW	1201110.0	0.01	1.30
		$HE1_{0.6}$	1168692.4	1306.86	1.26
$\|J\| = 800$	1575752.90	PSW	1973292.2	0.01	1.25
		$HE1_{0.6}$	1913924.2	2357.64	1.21

Table 10 – Comparison between PSW and HE1 with 10 machines.

$P10\|r_j\|\sum C_j$ with $\gamma = 0.6$	LB	Algorithm	Value	Time	Ratio
$\|J\| = 100$	96960.46	PSW	160832.6	0.01	1.65
		$HE1_{0.8}$	162182.8	38.420	1.67
$\|J\| = 200$	187381.96	PSW	312235.6	0.01	1.66
		$HE1_{0.8}$	315302.8	174.60	1.68
$\|J\| = 400$	384035.05	PSW	635543.6	0.01	1.65
		$HE1_{0.8}$	644026.2	720.43	1.67
$\|J\| = 600$	677843.38	PSW	995554.0	0.01	1.46
		$HE1_{0.8}$	996502.2	1614.99	1.47
$\|J\| = 800$	1113753.2	PSW	1535404.0	0.01	1.37
		$HE1_{0.8}$	1494949.8	2901.76	1.34
$\|J\| = 1000$	1666006.6	PSW	2214162.8	0.01	1.32
		$HE1_{0.8}$	2168788.2	4648.66	1.30
$\|J\| = 1200$	2269898.8	PSW	2994767.4	0.01	1.31
		$HE1_{0.8}$	2920309.8	6843.41	1.28

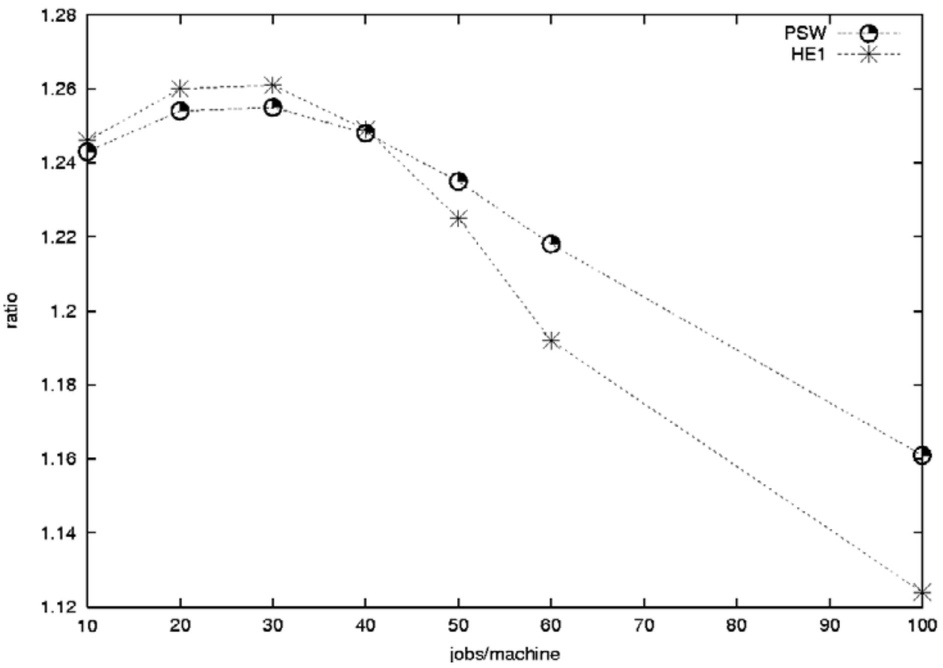


Figure 1 – Solution quality of the algorithms PSW and HE1 for 2 machines.

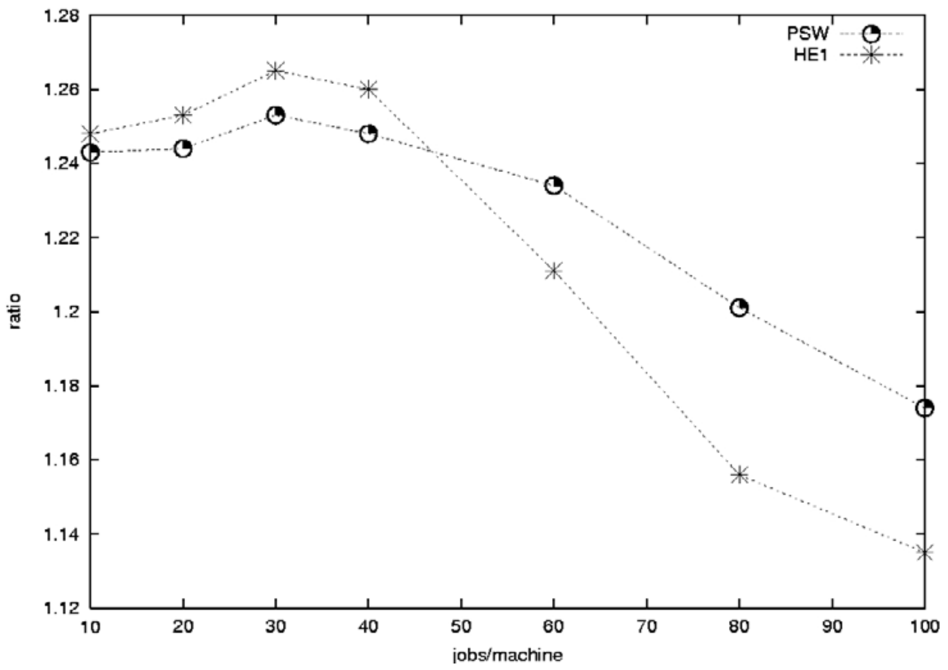


Figure 2 – Solution quality of the algorithms PSW and HE1 for 5 machines.

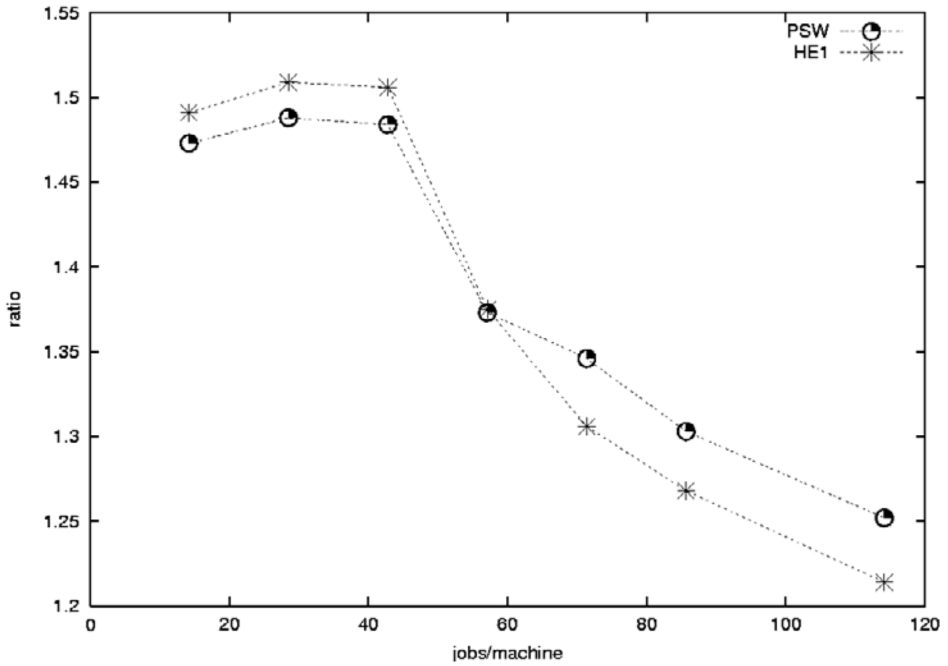


Figure 3 – Solution quality of the algorithms PSW and HE1 for 7 machines.

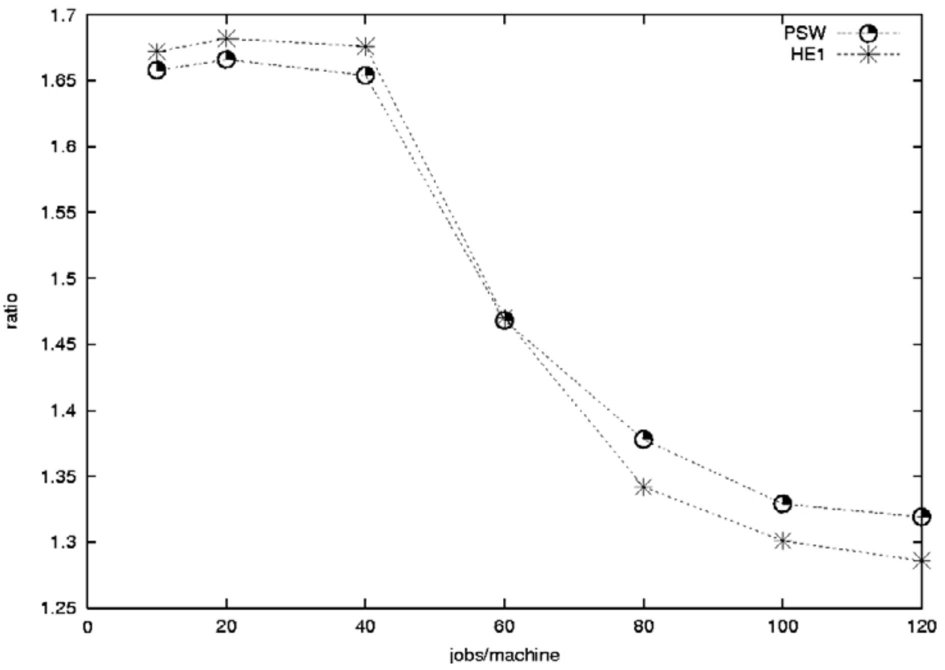


Figure 4 – Solution quality of the algorithms PSW and HE1 for 10 machines.

4.3 Tests for the problem $P|r_j|\Sigma w_jC_j$

For the problem $P|r_j|\Sigma w_jC$ we used the algorithms SZSK, SZSK2 and HE1. Remember that the algorithm SZSK is the combinatorial version of the algorithm SZSK2 for identical machines. The algorithms SZSK2 and HE1 were executed with parameter $\varepsilon\in\{0.1, 0.3\}$ and the tests were produced with release dates generated with parameter $\gamma\in\{0.2, 0.4, 0.6\}$. The Tables 11, 12 and 13 present the results obtained for these tests. The lower bounds (LB) were obtained from the optimal fractional solutions of the linear program of the algorithm SZSK2 with $\varepsilon=0.1$. Remember that this lower bound may be far away from the optimum, since the value of an optimal integer solution for the program *LPSS* is already a lower bound for the original problem $P|r_j|\Sigma w_jC$.

The behavior of the algorithms is essentially the same in all tests, except that algorithm SZSK has a bad quality performance in the tests with two and five machines. Algorithm SZSK have a worse quality performance than algorithm SZSK2, but it is much faster than it. The algorithm HE1 is the one that produces the best schedules.

Table 11 – Comparison for the problem $P|r_j|\Sigma w_jC_j$ with $\gamma = 0.2$.

Problem with $\gamma = 0.2$	LB	Algorithm	Value	Time	Ratio
$P2\|r_j\|\sum w_jC_j$	401484.96	SzSk	489182.6	2.4	1.21
		SzSk2$_{0.3}$	458123.0	6.5	1.14
		SzSk2$_{0.1}$	446167.3	73.2	1.11
		HE1$_{0.3}$	428054.3	6.0	1.06
		HE1$_{0.1}$	427810.6	71.9	1.06
$P5\|r_j\|\sum w_jC_j$	274727.29	SzSk	362339.0	2.2	1.31
		SzSk2$_{0.3}$	327884.3	54.8	1.19
		SzSk2$_{0.1}$	328976.6	408.5	1.19
		HE1$_{0.3}$	318633.0	52.8	1.15
		HE1$_{0.1}$	321541.3	401.9	1.17
$P7\|r_j\|\sum w_jC_j$	244972.5	SzSk	303367.6	2.2	1.23
		SzSk2$_{0.3}$	286762.3	94.5	1.17
		SzSk2$_{0.1}$	285346.0	761.6	1.16
		HE1$_{0.3}$	281960.6	92.5	1.15
		HE1$_{0.1}$	282133.3	751.5	1.15
$P10\|r_j\|\sum w_jC_j$	247130.08	SzSk	291006.3	1.9	1.17
		SzSk2$_{0.3}$	285885.3	190.3	1.15
		SzSk2$_{0.1}$	284247.6	1599.5	1.15
		HE1$_{0.3}$	281372.0	188.0	1.13
		HE1$_{0.1}$	283477.3	1513.4	1.14

Table 12 – Comparison for the problem $P|r_j|\Sigma w_jC_j$ with $\gamma = 0.4$.

Problem with $\gamma = 0.4$	LB	Algorithm	Value	Time	Ratio
$P2\|r_j\|\sum w_jC_j$	562054.4	SzSk	730586.3	2.2	1.29
		SzSk2$_{0.3}$	658244.6	7.0	1.17
		SzSk2$_{0.1}$	628967.0	75.0	1.11
		HE1$_{0.3}$	613529.0	6.6	1.09
		HE1$_{0.1}$	613979.6	82.0	1.09
$P5\|r_j\|\sum w_jC_j$	476777.12	SzSk	572827.6	2.0	1.20
		SzSk2$_{0.3}$	544158.6	58.1	1.14
		SzSk2$_{0.1}$	529113.0	48.5	1.10
		HE1$_{0.3}$	530403.0	61.6	1.11
		HE1$_{0.1}$	526752.3	468.1	1.10
$P7\|r_j\|\sum w_jC_j$	475822.01	SzSk	552136.0	1.8	1.16
		SzSk2$_{0.3}$	539034.0	113.3	1.13
		SzSk2$_{0.1}$	529724.3	926.2	1.11
		HE1$_{0.3}$	528659.0	111.2	1.11
		HE1$_{0.1}$	528034.6	898.9	1.10
$P10\|r_j\|\sum w_jC_j$	467883.30	SzSk	525903.0	1.8	1.12
		SzSk2$_{0.3}$	527858.3	221.3	1.12
		SzSk2$_{0.1}$	518648.0	1819.4	1.10
		HE1$_{0.3}$	519222.6	218.1	1.10
		HE1$_{0.1}$	518332.6	1767.1	1.10

Table 13 – Comparison for the problem $P|r_j|\Sigma w_j C_j$ with $\gamma = 0.6$.

Problem with $\gamma = 0.6$	LB	Algorithm	Value	Time	Ratio
$P2\|r_j\|\sum w_j C_j$	706573.38	SzSk	973967.3	2.1	1.37
		SzSk$2_{0.3}$	813596.6	7.4	1.15
		SzSk$2_{0.1}$	787153.3	88.6	1.11
		HE$1_{0.3}$	778875.6	7.0	1.10
		HE$1_{0.1}$	777878.6	86.2	1.10
$P5\|r_j\|\sum w_j C_j$	738399.47	SzSk	845729.3	1.9	1.14
		SzSk$2_{0.3}$	835492.6	65.5	1.13
		SzSk$2_{0.1}$	813745.6	521.4	1.10
		HE$1_{0.3}$	812734.0	66.6	1.10
		HE$1_{0.1}$	809256.3	512.9	1.09
$P7\|r_j\|\sum w_j C_j$	760000.39	SzSk	855123.6	1.9	1.12
		SzSk$2_{0.3}$	856218.6	121.8	1.12
		SzSk$2_{0.1}$	836550.3	987.0	1.10
		HE$1_{0.3}$	834758.0	120.4	1.09
		HE$1_{0.1}$	833242.0	973.6	1.09
$P10\|r_j\|\sum w_j C_j$	736998.73	SzSk	807679.0	1.4	1.09
		SzSk$2_{0.3}$	828656.0	246.8	1.12
		SzSk$2_{0.1}$	810024.0	1999.6	1.09
		HE$1_{0.3}$	809834.3	751.5	1.09
		HE$1_{0.1}$	807307.3	1971.3	1.09

4.4 Tests for the problem $R||\Sigma w_j C_j$

In this problem we use the algorithms SK, SZSK2 and HE1. For the tests in Table 14 we chose p_{ij} uniformly from the interval [1,100]. In the tests presented in Table 15 the processing times were chosen from different intervals to give the idea that we have machines with different speeds. Using two machines the processing times were chosen from the interval [1,50] for the first machine and from [50,100] for the second machine. Using five machines the processing times were chosen from intervals, [1,20],[20,40],…,[80,100]. Using seven machines the processing times were chosen from intervals, [1,15],[15,30],…,[90,100]. In the tests with ten machines, the processing times were chosen from intervals [1,10],[10,20],…,[90,100]. We use $\varepsilon=0.1$ and $\varepsilon=0.3$ in the algorithms SZSK2 and HE1. The LB column corresponds to the fractional solution found by the quadratic formulation *QSP* of the algorithm SK.

Table 14 – Comparison for problem R||Σ$w_j C_j$.

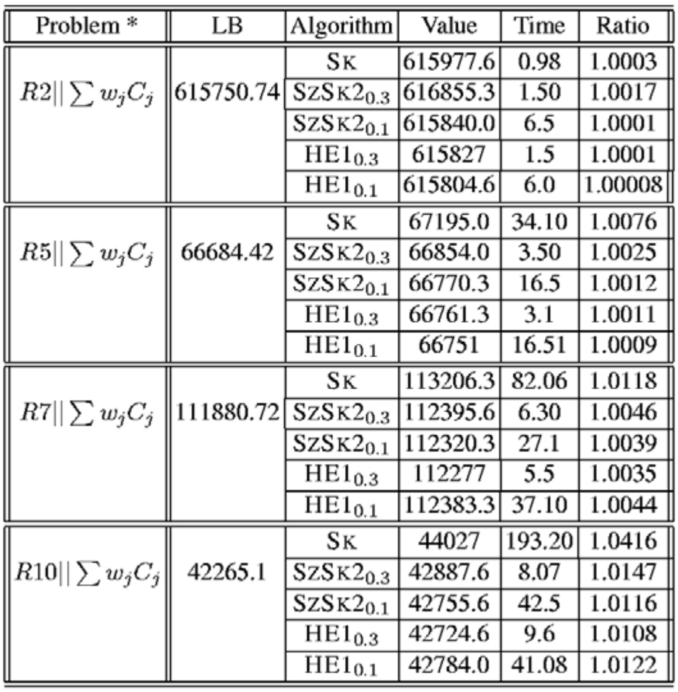
Problem	LB	Algorithm	Value	Time	Ratio
$R2\|\|\sum w_jC_j$	194112.01	Sk	194216.4	1.13	1.0005
		SzSk2$_{0.3}$	194149.8	1.40	1.0001
		SzSk2$_{0.1}$	194156.4	6.14	1.0002
		HE1$_{0.3}$	194143.8	1.21	1.0001
		HE1$_{0.1}$	194143.8	6.78	1.0001
$R5\|\|\sum w_jC_j$	37616.6	Sk	37763.0	38.31	1.0038
		SzSk2$_{0.3}$	37644.0	4.08	1.0007
		SzSk2$_{0.1}$	37635.8	21.31	1.0005
		HE1$_{0.3}$	37627.4	3.81	1.0002
		HE1$_{0.1}$	37625.8	21.99	1.0002
$R7\|\|\sum w_jC_j$	26049.94	Sk	26305.0	89.36	1.0097
		SzSk2$_{0.3}$	26154.2	5.15	1.0040
		SzSk2$_{0.1}$	26149.8	25.43	1.0038
		HE1$_{0.3}$	26140.2	5.06	1.0034
		HE1$_{0.1}$	26145.6	26.21	1.0036
$R10\|\|\sum w_jC_j$	11337.05	Sk	11666.2	200.79	1.0290
		SzSk2$_{0.3}$	11474.6	8.15	1.0121
		SzSk2$_{0.1}$	11463.0	40.96	1.0111
		HE1$_{0.3}$	11450.2	8.79	1.0099
		HE1$_{0.1}$	11465.2	39.56	1.0113

Table 15 – Comparison for problem R||Σ$w_j C_j$.

Problem *	LB	Algorithm	Value	Time	Ratio
$R2\|\|\sum w_jC_j$	246745.49	Sk	246873.6	0.97	1.0005
		SzSk2$_{0.3}$	246811.2	1.12	1.0002
		SzSk2$_{0.1}$	246818.6	6.24	1.0002
		HE1$_{0.3}$	246783.8	1.13	1.0001
		HE1$_{0.1}$	246783.8	6.78	1.0001
$R5\|\|\sum w_jC_j$	73513.03	Sk	73934.6	37.41	1.0057
		SzSk2$_{0.3}$	73670.0	3.80	1.0021
		SzSk2$_{0.1}$	73673.0	16.78	1.0022
		HE1$_{0.3}$	73659.6	3.71	1.0019
		HE1$_{0.1}$	73667.6	17.21	1.0021
$R7\|\|\sum w_jC_j$	51544.92	Sk	52211.6	98.06	1.0129
		SzSk2$_{0.3}$	51828.4	5.90	1.0054
		SzSk2$_{0.1}$	51808.0	26.31	1.0051
		HE1$_{0.3}$	51849.2	5.14	1.0059
		HE1$_{0.1}$	51834.2	26.71	1.0056
$R10\|\|\sum w_jC_j$	28453.73	Sk	30516.2	207.62	1.0724
		SzSk2$_{0.3}$	29472.2	8.07	1.0357
		SzSk2$_{0.1}$	29451.0	40.95	1.0350
		HE1$_{0.3}$	29415.8	8.16	1.0338
		HE1$_{0.1}$	29449.4	41.08	1.0349

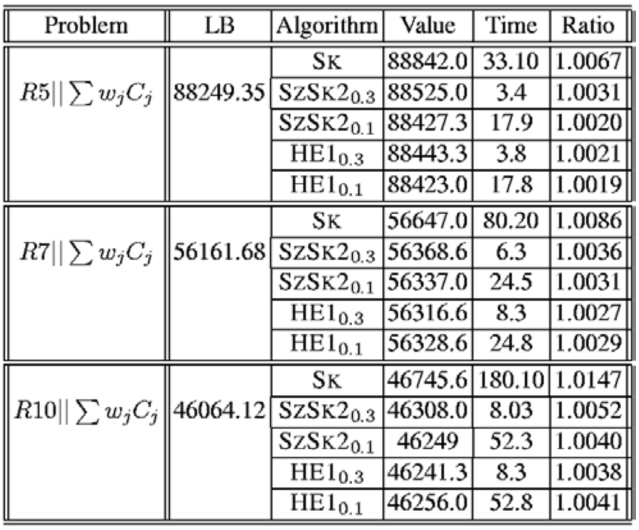
We also present another set of tests based on the approach of Vredeveld & Hurkens [VH02]. The instances of the test in Table 16 were generated to give a machine correlation different from the approach described for the tests in Table 15. The instances were generated with each processing time p_{ij} taken uniformly in $[\alpha_i, \alpha_i+10]$ where α_i is an integer from the uniform distribution in [1,100]. This approach is called *Machine Correlation*. The instances of the tests in Table 17 were made to give the idea that a job have two favorite machines to execute. For each job j, two machines i_{j1} and i_{j2} were randomly chosen, where the processing time of j in these two machines is uniformly chosen in $[\beta_j, \beta_j+4]$, where β_j is an integer from the uniform distribution in [15,25]. The processing times of j in the other machines were drawn from the uniform distribution in [60,90]. This approach is called *Favorite Machines*.

Table 16 – Instance set Machine Correlation for the problem $R||\Sigma w_jC_j$.



Problem *	LB	Algorithm	Value	Time	Ratio
$R2\|\| \sum w_jC_j$	615750.74	SK	615977.6	0.98	1.0003
		$\text{SzSK2}_{0.3}$	616855.3	1.50	1.0017
		$\text{SzSK2}_{0.1}$	615840.0	6.5	1.0001
		$\text{HE1}_{0.3}$	615827	1.5	1.0001
		$\text{HE1}_{0.1}$	615804.6	6.0	1.00008
$R5\|\| \sum w_jC_j$	66684.42	SK	67195.0	34.10	1.0076
		$\text{SzSK2}_{0.3}$	66854.0	3.50	1.0025
		$\text{SzSK2}_{0.1}$	66770.3	16.5	1.0012
		$\text{HE1}_{0.3}$	66761.3	3.1	1.0011
		$\text{HE1}_{0.1}$	66751	16.51	1.0009
$R7\|\| \sum w_jC_j$	111880.72	SK	113206.3	82.06	1.0118
		$\text{SzSK2}_{0.3}$	112395.6	6.30	1.0046
		$\text{SzSK2}_{0.1}$	112320.3	27.1	1.0039
		$\text{HE1}_{0.3}$	112277	5.5	1.0035
		$\text{HE1}_{0.1}$	112383.3	37.10	1.0044
$R10\|\| \sum w_jC_j$	42265.1	SK	44027	193.20	1.0416
		$\text{SzSK2}_{0.3}$	42887.6	8.07	1.0147
		$\text{SzSK2}_{0.1}$	42755.6	42.5	1.0116
		$\text{HE1}_{0.3}$	42724.6	9.6	1.0108
		$\text{HE1}_{0.1}$	42784.0	41.08	1.0122

Table 17 – Instance set Favorite Machines for the problem R||Σ$w_j C_j$.



Problem	LB	Algorithm	Value	Time	Ratio
$R5\|\|\sum w_jC_j$	88249.35	SK	88842.0	33.10	1.0067
		SZSK$2_{0.3}$	88525.0	3.4	1.0031
		SZSK$2_{0.1}$	88427.3	17.9	1.0020
		HE$1_{0.3}$	88443.3	3.8	1.0021
		HE$1_{0.1}$	88423.0	17.8	1.0019
$R7\|\|\sum w_jC_j$	56161.68	SK	56647.0	80.20	1.0086
		SZSK$2_{0.3}$	56368.6	6.3	1.0036
		SZSK$2_{0.1}$	56337.0	24.5	1.0031
		HE$1_{0.3}$	56316.6	8.3	1.0027
		HE$1_{0.1}$	56328.6	24.8	1.0029
$R10\|\|\sum w_jC_j$	46064.12	SK	46745.6	180.10	1.0147
		SZSK$2_{0.3}$	46308.0	8.03	1.0052
		SZSK$2_{0.1}$	46249	52.3	1.0040
		HE$1_{0.3}$	46241.3	8.3	1.0038
		HE$1_{0.1}$	46256.0	52.8	1.0041

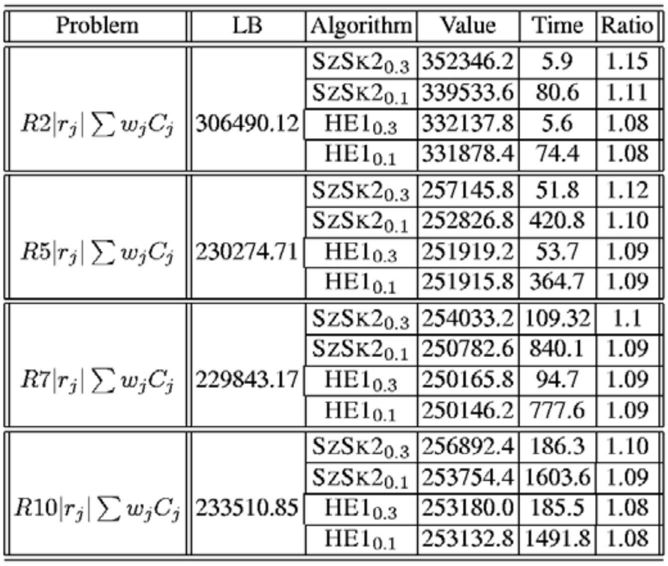
As we can see, all algorithms produces schedules very close to the optimal. For all tests, the algorithms produced solutions with values that are at most 3% of the lower bound except for the algorithm SK that generated a solution with value 7% of the lower bound. In general, the algorithm HE1 generates better schedules. Another point, is that although the semidefinite program *QSP* generates fractional solutions that are closer to the optimal, the algorithm SK generates the worst schedules even if compared with the algorithm SZSK$2_{0.3}$.

4.5 Comparison for the problem $R|r_j|\Sigma w_jC_j$

For the problem $R|r_j|\Sigma w_jC_j$, we used the algorithms SZSK2 and HE1. The results for these tests are presented in Table 18. The processing times were uniformly chosen from the interval [1,100] and the release dates were generated with $\gamma=0.2$. The LB is the optimal fractional solution of the linear program *LPSS* with $\varepsilon=0.1$. We emphasize that this lower bound may be far away from the optimal solution, since an optimal integer solution to *LPSS* is already a relaxation for the problem $R|r_j|\Sigma w_jC_j$.

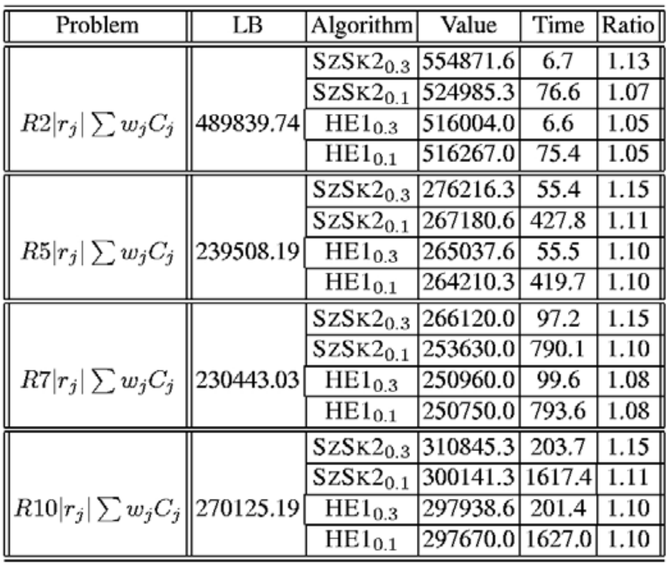
We also made another set of tests with instances of type *Machine Correlation* and instances of type *Favorite Machines*. The tests can be seen in Tables 19 and 20. The algorithm HE1 obtained the best results in all tests. The hardest instances are for the set *Machine Correlation* (Table 19), where the algorithm HE1 obtained schedules that are at most 10% of the lower bound and the algorithm SZSK2 obtained schedules that are at most 15% from the lower bound.

Table 18 – Comparisons for the problem $R|r_j|\Sigma w_j C_j$.



Problem	LB	Algorithm	Value	Time	Ratio
$R2\|r_j\|\sum w_j C_j$	306490.12	SzSk2$_{0.3}$	352346.2	5.9	1.15
		SzSk2$_{0.1}$	339533.6	80.6	1.11
		HE1$_{0.3}$	332137.8	5.6	1.08
		HE1$_{0.1}$	331878.4	74.4	1.08
$R5\|r_j\|\sum w_j C_j$	230274.71	SzSk2$_{0.3}$	257145.8	51.8	1.12
		SzSk2$_{0.1}$	252826.8	420.8	1.10
		HE1$_{0.3}$	251919.2	53.7	1.09
		HE1$_{0.1}$	251915.8	364.7	1.09
$R7\|r_j\|\sum w_j C_j$	229843.17	SzSk2$_{0.3}$	254033.2	109.32	1.1
		SzSk2$_{0.1}$	250782.6	840.1	1.09
		HE1$_{0.3}$	250165.8	94.7	1.09
		HE1$_{0.1}$	250146.2	777.6	1.09
$R10\|r_j\|\sum w_j C_j$	233510.85	SzSk2$_{0.3}$	256892.4	186.3	1.10
		SzSk2$_{0.1}$	253754.4	1603.6	1.09
		HE1$_{0.3}$	253180.0	185.5	1.08
		HE1$_{0.1}$	253132.8	1491.8	1.08

Table 19 – Instance set Machine Correlation for the problem $R|r_j|\Sigma w_j C_j$.



Problem	LB	Algorithm	Value	Time	Ratio
$R2\|r_j\|\sum w_j C_j$	489839.74	SzSk2$_{0.3}$	554871.6	6.7	1.13
		SzSk2$_{0.1}$	524985.3	76.6	1.07
		HE1$_{0.3}$	516004.0	6.6	1.05
		HE1$_{0.1}$	516267.0	75.4	1.05
$R5\|r_j\|\sum w_j C_j$	239508.19	SzSk2$_{0.3}$	276216.3	55.4	1.15
		SzSk2$_{0.1}$	267180.6	427.8	1.11
		HE1$_{0.3}$	265037.6	55.5	1.10
		HE1$_{0.1}$	264210.3	419.7	1.10
$R7\|r_j\|\sum w_j C_j$	230443.03	SzSk2$_{0.3}$	266120.0	97.2	1.15
		SzSk2$_{0.1}$	253630.0	790.1	1.10
		HE1$_{0.3}$	250960.0	99.6	1.08
		HE1$_{0.1}$	250750.0	793.6	1.08
$R10\|r_j\|\sum w_j C_j$	270125.19	SzSk2$_{0.3}$	310845.3	203.7	1.15
		SzSk2$_{0.1}$	300141.3	1617.4	1.11
		HE1$_{0.3}$	297938.6	201.4	1.10
		HE1$_{0.1}$	297670.0	1627.0	1.10

Table 20 – Instance set Favorite Machines for the problem R|r_j|Σ$w_j C_j$.

Problem	LB	Algorithm	Value	Time	Ratio
$R5\|r_j\|\sum w_j C_j$	239705.1	SzSk2$_{0.3}$	270809.3	55.3	1.12
		SzSk2$_{0.1}$	269149.3	420.8	1.12
		HE1$_{0.3}$	267289.3	53.2	1.11
		HE1$_{0.1}$	268175.0	414.5	1.11
$R7\|r_j\|\sum w_j C_j$	243361.23	SzSk2$_{0.3}$	273059.3	97.3	1.12
		SzSk2$_{0.1}$	267799.6	778.6	1.10
		HE1$_{0.3}$	267688.6	95.2	1.09
		HE1$_{0.1}$	267545.6	767.8	1.09
$R10\|r_j\|\sum w_j C_j$	253293.67	SzSk2$_{0.3}$	280551.6	198.2	1.10
		SzSk2$_{0.1}$	277072.6	1610.7	1.09
		HE1$_{0.3}$	277365.6	195.4	1.09
		HE1$_{0.1}$	276944.0	1586.5	1.09

5. Conclusion

We present computational results for some approximation algorithms for scheduling problems on parallel machines. As expected, the practical solutions yield ratios that are much better than the approximation factors of the presented algorithms. We also notice that algorithms with more refined techniques do not necessarily lead to better results. In fact, for the problems $\boldsymbol{P||\Sigma w_j C_j}$ and $\boldsymbol{P|r_j|\Sigma C_j}$ algorithms PSW and KK obtained the best results even when compared to algorithms with advanced ideas. We also notice that the solutions provided by the algorithm SK is worse than the solutions provided by the algorithm SZSK2 despite the semidefinite program generates fractional solutions with better quality. Finally, we present two heuristics that get better results in almost all cases considered, although for problem $\boldsymbol{P|r_j|\Sigma C_j}$ the processing time of algorithm HE1 is much bigger than the processing time of algorithm PSW.

Acknowledgements

This research was partially supported by FAPESP project 01/04412-4, CAPES and CNPq (Proc. 664107/97-4, 470608/01-3, 464114/00-4, 478818/03-3, 300301/98-7).

References

[Afrati *et al.*, 1999] Afrati, F.; Bampis, E.; Chekuri, C.; Karger, D.; Kenyon, C.; Khanna, S.; Milis, I.; Queyranne, M.; Skutella, M.; Sviridenko, M. & Stein, C. (1999). Approximation schemes for minimizing average weighted completion time with release dates. *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science (FOCS'99)*, 32-44.

[BME02] Baev, I.D.; Meleis, W.M. & Eichenberger, A. (2002). An Experimental Study of Algorithms for Weighted Completion Time Scheduling. *Algorithmica*, **33**, 34-51.

[D02] Dash Optimization (2002). Xpress-MP Release 13. Xpress-MP Manual.

[GLLR79] Graham, E.L.; Lawler, E.L.; Lenstra, J.K. & Rinnooy Kan, A.H.G. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, **5**, 287-326.

[HP83] Hariri, A.M.A. & Potts, C.N. (1983). An algorithm for single machine sequencing with release dates to minimize total weighted completion time. *Discrete Applied Mathmatics*, **5**, 99-109.

[HS01] Hepner, C. & Stein, C. (2001). Implementation of a PTAS for Scheduling with Release Dates. **In:** *3rd Workshop on Algorithm Engineering and Experiments (ALENEX 2001)*. Lecture Notes in Computer Sciense, **2513**, 202-215.

[KK86] Kawaguchi, T. & Kyan, S. (1986). Worst Case Bound of an LRF Schedule for the Mean Weighted FlowTime Problem. *SIAM J. Computing*, **15**(4), 1119-1129.

[PSW98] Phillips, C.; Stein, C. & Wein, J. (1998). Minimizing Average Completion time in the Presence of Release Dates. *Mathematical Programming B*, **82**, 199-223.

[SUW98] Savelsbergh, M.W.P.; Uma, R.N. & Wein, J.M. (1998). An experimental study of LP-based approximation algorithms for scheduling problems. *Proceedings of the 9th Annual ACM/SIAM Symposium on Discrete Algorithms*, 453-462.

[SS02] Schulz, A.S. & Skutella, M. (2002). Scheduling Unrelated Machines by Randomized Rounding. *SIAM Journal on Discrete Mathematics*, **15**(4), 450-469.

[S98] Skutella, M. (1998). Semidefinite Relaxations for Parallel Machine Scheduling. *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*, 472-481.

[S97] Sipser, M. (1997). *Introduction to the Theory of Computation*. PWS Publishing Company.

[S56] Smith, W.E. (1956). Various optimizers for single-stage production. *Naval Res. Logist. Quart.*, **3**, 58-66.

[VH02] Vredeveld, T. & Hurkens, C. (2002). Experimental Comparison of Approximation Algorithms for Scheduling Unrelated Paralell Machines. *INFORMS Journal on Computing*, **14**(2), 175-189.