Submitted to *INFORMS Journal on Computing*
manuscript (Please, provide the manuscript number!)

An Improved Branch-Cut-and-Price Algorithm for Parallel Machine Scheduling Problems

Daniel Oliveira, Artur Pessoa
Departamento de Engenharia de Produção, Universidade Federal Fluminense, Rua Passo da Pátria 156, Niterói, RJ, Brazil, 24210-240, daniel_oliveira@id.uff.br, artur@producao.uff.br

This work presents an improved Branch-Cut-and-Price algorithm for the identical parallel machine scheduling problem minimizing a generic function of the job completion times. A new family of cuts is proposed to strengthen the arc-time-indexed formulation, along with an efficient separation algorithm. Also, the projection of the arc-time-indexed into a time-indexed formulation is introduced to take advantage of the variable fixations performed in the larger variable space. The improved algorithm was capable of solving 143 out of the 150 literature instances, being 9 solved for the first time. Also, the running time for the 134 previously solved instances decreased by 84.1% on the average.

Key words: parallel machine scheduling; integer programming; column generation; branch-cut-and-price; time-indexed formulations

1. Introduction

In the scheduling problem denoted by $P||\sum f_j(C_j)$, a set J of n jobs have to be processed by a set of m identical parallel machines. Each job has a processing time p_j, and is associated with a cost function $f_j(C_j)$ based on its completion time, C_j. Each machine can only process one job at a time and each job has to be processed by a single machine, with no pauses or preemption. The goal is to find a schedule that minimizes the sum of individual costs. A classical special case of this problem is the weighted tardiness variant $(P||\sum w_jT_j)$, where each job has a weight w_j, a due date d_j, and the cost function is $f_j(C_j) = w_jT_j$, being T_j the job tardiness, calculated as $\max\{0, C_j - d_j\}$.

Potts and Wassenhove (1985) proposed a branch-and-bound algorithm for the single machine total weighted tardiness problem $(1||\sum w_jT_j)$ capable of solving instances of up to 40 jobs. Its lower bound was based on Lagrangian relaxation, solved by a multiplier adjustment method. The superiority of the algorithm at the time was highlighted by a review of 6 exact algorithms for the $1||\sum w_jT_j$ problem, presented by Abdul-Razaq et al. (1990), which included computational tests.

Tanaka et al. (2009) presented an efficient algorithm for solving the $1||\sum f_j(C_j)$ when machine idle times are not permitted. Their algorithm is based on the work of Ibaraki and Nakamura (1994), which used the Successive Sublimation Dynamic Programming (SSDP) to solve $1||\sum f_j(C_j)$.

The SSDP, proposed by Ibaraki (1987), is based on a set of dynamic programming relaxations for a problem. It consists of solving a sequence of dynamic programming relaxations, stepping from the weaker to the stronger, tightening the gap in the process. Also, when stepping from one model to the next, states that can be proved not to be part of an optimal solution are eliminated to alleviate memory use and speed up computations.

The dynamic programming relaxations can be derived in many different ways. The ones used by Ibaraki and Nakamura (1994) were the state-space relaxations presented by Abdul-Razaq and Potts (1988).

It is interesting to note that the SSDP method uses dynamic programming in a different way than most exact algorithms, which usually employ the Lagrangian bounds on a Branch and Bound framework. As Tanaka et al. (2009) stated, their algorithm is based fully on dynamic programming, and they tried a Branch and Bound algorithm first (Tanaka and Araki 2006), not obtaining the same efficiency as the SSDP method.

Recently, Tanaka and Fujikuma (2012) extended Tanaka et al. (2009) to permit machine idle times. Following the same SSDP framework, Tanaka and Araki (2013) presented an algorithm for the single machine total weighted tardiness problem with sequence-dependent setup times, $1|s_{i,j}|\sum w_jT_j$.

Pessoa et al. (2010) were the first to propose an exact algorithm for the same class of parallel machine scheduling problems addressed in this paper, but focusing on the weighted tardiness objective function. The explicit use of the arc-time-indexed formulation for a scheduling problem, where each variable is indexed by a pair of jobs and a time, was a novelty of their work. To handle the huge number of variables of such formulation, column generation was used along with a series of advanced techniques, composing a Branch-Cut-and-Price (BCP) algorithm. Some important highlights of their work were the use of a strong family of cuts, variable fixation by Lagrangean bounds, dual stabilization, and the use of a commercial MIP solver to finish the optimization when the number of remaining variables is sufficiently small. Throughout this paper, their algorithm is referred to as BCP-PMWT, where PMWT stands for parallel machine with weighted tardiness.

In this work, the following techniques were developed in order to improve the BCP-PMWT.

- A new family of cuts, called Overload Elimination Cuts (OECs), for the arc-time-indexed formulation, that improves the quality of the relaxation used by the BCP-PMWT. As the number of constraints potentially generated by this family is exponential, they are also added by demand.
- A genetic algorithm to separate OECs during the BCP execution.
- The use of job-time indexed, or simply time-indexed for shortness, formulations instead of arc-time-indexed to finish the optimization when the number of remaining variables is small enough. Experiments using four alternative formulations are presented. The time-indexed formulations are more compact, allowing for the commercial solver to expand the Branch-and-Bound tree much faster. However, the relaxations based on these formulation are weaker, reducing the algorithm ability to prune the tree.
- The addition of cuts to the time-indexed formulation used in the previous item. These cuts are derived from the arc-time-indexed formulation taking into account the fixed variables.

The improved method is called as BCP-PMWT-OTI, where OTI stands for overload, referring to the new family of cuts, and time-indexed, due to the formulation used to finish the optimization. As a result of the inclusion of these techniques, the BCP-PMWT-OTI was capable of solving 143 out of the 150 literature instances, being 9 solved for the first time, and greatly improved the execution time for the instances that were already solved.

The rest of the paper is organized as follows. Section 2 reviews the main characteristics of the BCP-PMWT. Section 3 presents the proposed improvements that composes the BCP-PMWT-OTI. Section 4 details how computational experiments were performed and analyses its results. Section 5 summarizes our conclusions, highlighting our main results, and points a few possible directions for future research.

2. The Original Algorithm

In this section, we give an overview of the BCP-PMWT algorithm, including all details necessary to explain the proposed improvements. For further details, we refer to Pessoa et al. (2010).

2.1. The Arc-Time-indexed Formulation

The formulation used in the BCP-PMWT is the arc-time-indexed formulation (ATIF), which uses a binary variable for each job pair (i,j) and time period t to indicate that job i finishes and job j starts at time t, on the same machine. Considering T as the latest time a job can finish in an optimal schedule, defining $J_0 = J \cup \{0\}$ and $p_0 = 0$, the formulation follows:

$$\text{Minimize} \quad \sum_{i \in J_0} \sum_{j \in J \setminus \{i\}} \sum_{t=p_i}^{T-p_j} f_j\,(t+p_j)\, x_{ij}^t \tag{1a}$$

$$\text{Subject to} \quad \sum_{i \in J_0 \setminus \{j\}} \sum_{t=p_i}^{T-p_j} x_{ij}^t = 1 \quad (j \in J) \tag{1b}$$

$$\sum_{\substack{j \in J_0 \setminus \{i\},\\ t-p_j \geq 0}} x_{ji}^t - \sum_{\substack{j \in J_0 \setminus \{i\},\\ t+p_i+p_j \leq T}} x_{ij}^{t+p_i} = 0 \quad (\forall i \in J;\, t = 0, \ldots, T-p_i) \tag{1c}$$

$$\sum_{\substack{j \in J_0,\\ t-p_j \geq 0}} x_{j0}^t - \sum_{\substack{j \in J_0,\\ t+p_j+1 \leq T}} x_{0j}^{t+1} = 0 \quad (t = 0, \ldots, T-1) \tag{1d}$$

$$\sum_{j \in J_0} x_{0j}^0 = m \tag{1e}$$

$$x_{ij}^t \in Z^+ \quad (\forall\, i \in J_0;\, \forall\, j \in J_0 \setminus \{i\};\, t = p_i, \ldots, T-p_j) \tag{1f}$$

$$x_{00}^t \in Z^+ \quad (t = 0, \ldots, T-1). \tag{1g}$$

The objective function (1a) is defined as the sum of the completion costs for all jobs. Constraints (1b) impose that every job j starts exactly once. Constraints (1c) and (1d) are flow conservation constraints. Constraint (1e) requires that m flows, one per machine schedule, start on time 0. Constraints (1f) and (1g) enforce integrality.

Each ATIF solution can be seen as a set of paths in a graph $G = (V, A)$, where $V = \{(i,t) \mid i \in J_0,\, t \in \{0, 1, 2 \ldots, T-1\}\} \cup \{(0, T)\}$ and each arc $a^t = ((i, t-p_i), (j, t)) \in A$ $(a = (i,j))$ corresponds to a x_{ij}^t variable. Figure 1 represents a possible solution, for an instance with 8 jobs and 2 machines, as such graph, where $p_1 = p_4 = p_6 = 6$, $p_2 = p_7 = 4$, $p_5 = 5$, and $p_8 = 8$, $C_1 = 6$, $C_2 = 10$, $C_3 = 13$, $C_4 = 19$, $C_5 = 5$, $C_6 = 11$, $C_7 = 15$, and $C_8 = 23$. In this figure, the variables x_{01}^0, x_{12}^6, x_{23}^{10}, x_{34}^{13}, x_{40}^{19}, x_{00}^{20}, x_{00}^{21}, x_{00}^{22}, x_{00}^{23}, x_{05}^0, x_{56}^5, x_{67}^{11}, x_{78}^{15}, and x_{80}^{23} are equal to one, and $x_{00}^{24} = 2$.

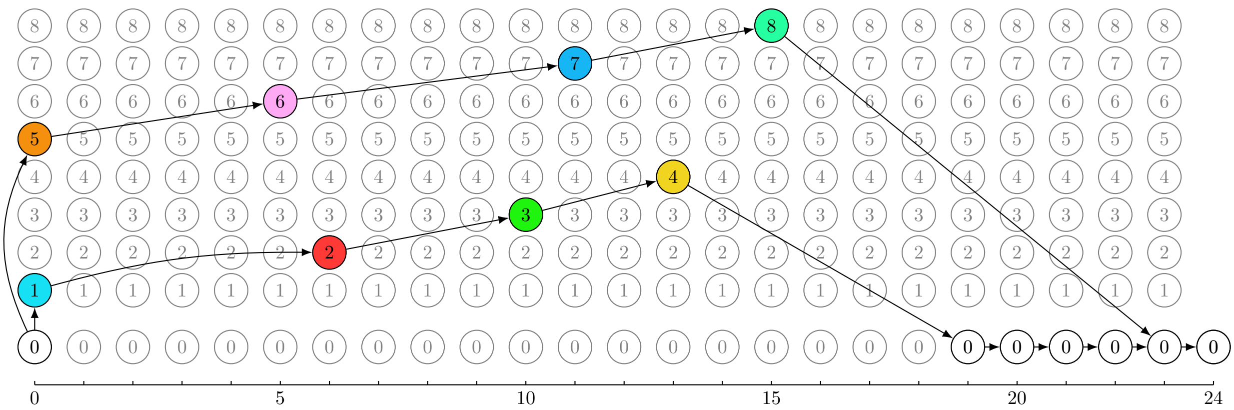

Figure 1 Graph of a valid solution

Following the idea represented in figure 1, a pseudo-schedule is defined as a path from $(0,0)$ to $(0,T)$ in G, possibly repeating jobs. Each pseudo-schedule represents the part of the schedule to be processed on one machine in a fractional solution to (1a)-(1e) (with $x_{ij}^t \geq 0$; $\forall i,j,t$).

Let P be the set of all possible pseudo-schedules. For every pseudo-schedule $p \in P$, define a variable λ_p and a set of constants $\{q_a^{tp} | a^t \in A\}$ to indicate if an arc a^t appears in p. Define $f_0(t)$ as zero for any t. The ATIF formulation can then be reformulated as

$$\text{Minimize} \quad \sum_{(i,j)^t \in A} f_j(t+p_j) x_{ij}^t \tag{2a}$$

$$\text{Subject to} \quad \sum_{p \in P} q_a^{tp} \lambda_p - x_a^t = 0 \qquad (\forall\, a^t \in A) \tag{2b}$$

$$\sum_{(j,i)^t \in A} x_{ji}^t = 1 \qquad (\forall\, i \in J) \tag{2c}$$

$$\sum_{(0,j)^0 \in A} x_{0j}^0 = m \tag{2d}$$

$$\lambda_p \geq 0 \qquad (\forall\, p \in P) \tag{2e}$$

$$x_a^t \in Z_+ \qquad (\forall\, a^t \in A). \tag{2f}$$

Formulation (2), containing both λ and x variables, is said to be in the explicit format, as defined by Marcus Poggi de Aragão (2003). Using the redundant equations (2b) to eliminate x, and relaxing the integrality, the Dantzig-Wolfe Master (DWM) LP is written as:

$$\text{Minimize} \quad \sum_{p \in P} \left(\sum_{(i,j)^t \in A} q_{ij}^{tp} f_j(t + p_j) \right) \lambda_p \tag{3a}$$

$$\text{Subject to} \quad \sum_{p \in P} \left(\sum_{(j,i)^t \in A} q_{ji}^{tp} \right) \lambda_p = 1 \qquad (\forall i \in J) \tag{3b}$$

$$\sum_{p \in P} \left(\sum_{(0,j)^0 \in A} q_{0j}^{0p} \right) \lambda_p = m \tag{3c}$$

$$\lambda_p \geq 0 \qquad (\forall p \in P). \tag{3d}$$

Any generic constraint l of format $\sum_{a^t \in A} \alpha_{al}^t x_a^t \geq b_l$ can be included in the DWM by converting the x variables to λ by using the same equation (2b). This is needed for all the cuts used, since they were defined in terms of x.

Since the number of λ variables is exponential on n, in order to solve DWM, these variables are generated on demand. For that, given an optimal solution to (3) restricted to a subset of the λ variables, and its dual, the pricing subproblem consists of finding the variable λ_p with minimum reduced cost. If such a reduced cost is negative, then λ_p is added to the restricted master problem and the process continues. Otherwise, the current solution is also optimal for the complete DWM.

To efficiently compute the optimal λ_p variable, its reduced cost is expressed as the sum of the reduced costs of the arcs of p, which are defined in the following.

Suppose that, at a given instant, there are $r+1$ constraints in the DWM. Let π_0 be the dual variable of constraint (3c), π_i be the dual variable of constraints (3b) for $i \in J$, and π_l, $n < l \leq r$, be the dual variable of any additional constraint. Being α_{al}^t the coefficient of variable x_a^t in constraint l, the reduced cost of arc a^t is defined using the α as:

$$\bar{c}_a^t = f_j(t + p_j) - \sum_{l=0}^{r} \alpha_{al}^t \pi_l. \tag{4}$$

2.2. Pricing and Fixing by Reduced Costs

The pricing subproblem in the BCP-PMWT algorithm consists of finding the pseudo-schedule p with the minimum reduced cost for its corresponding variable λ_p. This can be done by finding the shortest path from $(0,0)$ to $(0,T)$ in the previously defined graph G, setting the length of each arc a^t to $\bar{c}_a^t$.

Let $\bar{c}^*$ be the reduced cost of a shortest path from $(0,0)$ to $(0,T)$ in G, when the current objective value for the DWM is LB and the best known upper bound on the optimal solution cost is UB. If the shortest path from $(0,0)$ to $(0,T)$ that uses a given arc a^t has a reduced cost $\hat{c}_a^t$ such that $LB + (m-1)\bar{c}^* + \hat{c}_a^t > UB - 1$, then one can conclude that no solution better than UB may use this arc. Thus, it can be fixed to zero. In order to efficiently check the previous condition for all non-fixed arcs, the fixing procedure pre-computes two shortest path trees: one rooted at $(0,0)$ and following the arcs of G in the forward direction, and another one rooted at $(0,T)$ and following the backward direction. Then, it visits every arc a^t computing the corresponding value of $\hat{c}_a^t$ in $\Theta(1)$. For further details, we refer to Pessoa et al. (2010).

2.3. Extended Capacity Cuts

One family of cuts is separated to further to strengthen the continuous relaxation of the ATIF. Let $S \subseteq J$ be a set of jobs, define $p(S) = \sum_{j \in S} p_j$ as the total processing time of S, $\delta^-(S) = \{(i,j)^t \in A : i \notin S, j \in S\}$ and $\delta^+(S) = \{(i,j)^t \in A : i \in S, j \notin S\}$. Equation (5) below is valid.

$$\sum_{a^t \in \delta^+(S)} t x_a^t - \sum_{a^t \in \delta^-(S)} t x_a^t = p(S) \tag{5}$$

The Rounded Homogeneous Extended Capacity Cuts (RHECCs), inequality (6), is obtained by multiplying (5) by a value $r \in (0,1)$ and applying integer rounding.

$$\sum_{a^t \in \delta^+(S)} \lceil rt \rceil x_a^t - \sum_{a^t \in \delta^-(S)} \lfloor rt \rfloor x_a^t \geq \lceil r p(S) \rceil \tag{6}$$

To separate RHECCs, a specific heuristic procedure is used.

2.4. Branch-cut-and-price

The algorithm consists of a Branch-cut-and-price where the continuous relaxation of the ATIF strengthened by RHECCs is solved in each node, using column generation stabilized by the technique of Wentges (1997).

After every 5 column generation iterations, variable fixing by reduced costs is performed. This helps to reduce the number of arcs in A, therefore speeding the pricing and improving convergence. The relaxation bounds may also be improved.

Branching is performed by choosing a job $j \in J$ and partitioning the variables x_{ij}^t (those entering j) into two sets S_1 and S_2 such that $\sum_{(i,j,t)\in S_1} \bar{x}_{ij}^t$ and $\sum_{(i,j,t)\in S_2} \bar{x}_{ij}^t$ are close to 0.5. The variables in S_1 are fixed to zero in the left child node; variables in S_2 are fixed to zero in the right child node. Strong branching is performed by testing eight possible choices of sets before each branch.

When the number of arcs in A after solving the root node is below 200,000, additionally to the BCP, the current reduced ATIF is fed to a commercial MIP solver (CPLEX 11.1).

3. The Improved Algorithm

In this section we propose improvements to the BCP-PMWT algorithm, resulting in the algorithm referred to as the BCP-PMWT-OTI. We present a new family of cuts, along with its separation algorithm. Next, we propose that the existing procedure of feeding a residual model to a MIP solver uses a time-indexed formulation. We then study different time-indexed formulations to be used in such procedure, and show how they can be strengthened by information gathered from the arc-time-indexed formulation.

3.1. Overload Elimination Cuts

Define the aggregated variables v^t and u^t, for a given set $S \subseteq J$, as follows:

$$v^t = \sum_{a^t \in \delta^+(S)} x_a^t \quad (t = 1, \dots, T), \tag{7}$$

$$u^t = \sum_{a^t \in \delta^-(S)} x_a^t \quad (t = 0, \dots, T-1). \tag{8}$$

For $m \geq 2$, a subset of jobs $S \subseteq J$, and $t \in \left\{1, \dots, \left\lfloor \frac{p(S)-1}{m} \right\rfloor + 1\right\}$, we have the following inequality, which we call the overload elimination constraint (OEC):

$$\begin{aligned} &\sum_{q=t}^{t_1} v^q + \sum_{q=t_1+1}^{T} 2\, v^q - \sum_{\substack{q=\max\{t_1,\\ T-p(S)+m(t-1)+1\}}}^{T-1} u^q \geq 2, \\ &t_1 = p(S) - t - (m-2)(t-1). \end{aligned} \tag{9}$$

THEOREM 1. *The OEC is valid for the ATIF.*

I t is sufficient to prove that, given a feasible solution to the ATIF, (9) is satisfied. As m represents the number of machines available to process J, at least one and at most m paths traverse the set in this solution. Each path can enter and exit the set more than once. Let $T^-(i)$ and $T^+(i)$, $i = 1, \ldots, m$, be functions mapping the last time path i enters and exits the set S, respectively. We can assume, w.l.o.g., that $T^+(i-1) \le T^+(i)$, $i = 2, \ldots, m$. Note that, given this assumption and the choice of t, it is impossible to have $T^+(m) < t$. This is true because, since all jobs in S must be processed by the m machines, $\sum_{i=1}^{m} T^+(i) \ge p(S)$. First, considering solutions with $T^-(i) < \max\{t_1, T - p(S) + m(t-1) + 1\}$, for $i = 1, \ldots, m$, we have the following two cases:

Case 1: $T^+(m-1) < t$

In this case, $\sum_{i=1}^{m-1} T^+(i) \le (m-1)(t-1)$, so $T^+(m) \ge p(S) - (m-1)(t-1) = t_1 + 1$ and the cut is valid since $v^{T^+(m)}$ has coefficient 2.

Case 2: $T^+(m-1) \ge t$

In this case, the cut is valid since both $v^{T^+(m-1)}$ and $v^{T^+(m)}$ have coefficients greater than or equal to 1.

To complete the proof we need to show that the inequality still is not violated when there exists $i \in \{1, \ldots, m\}$ such that $T^-(i) \ge \max\{t_1, T - p(S) + m(t-1) + 1\}$. Again, w.l.o.g., we can assume that $T^-(i-1) \le T^-(i)$, $i = 2, \ldots, m$ (and not necessarily $T^+(i-1) \le T^+(i)$, $i = 2, \ldots, m$).

As $T^-(m) \ge t_1$, $T^+(m)$ will be strictly greater than t_1. And as $T^-(m) \ge T - p(S) + m(t-1) + 1$, since $T^+(m) \le T$, we have $T^+(m) - T^-(m) \le p(S) - m(t-1) - 1$, this means that at least $m(t-1)+1$ have to be processed by machines $1, \ldots, m-1$ or another part of machine's m path. By that, we can infer that at least one arc will exit S at time t or later, resulting in a left hand side of at least 2. Moreover, if paths other than m enter S at time greater than or equal to t_1, one more exit is required for each additional entrance, resulting in a net increase of 1 unit per entrance.

The term Overload Elimination comes from the fact that the inequality could only be violated by an integer solution if, hypothetically, some machine was overloaded to allow one or more jobs of S to finish prematurely before t. In light of fractional solutions, such premature finish occurs when a fraction of a job finishes before t.

3.1.1. Separation For the separation of OECs, we implemented a genetic algorithm. Here, the term solution and individual will be used interchangeably to denote a cut, being it violated or not. Each OEC is fully described by the elements of S and the t value. Given a subset S of jobs, the t value that leads to the best cut can always be found by testing all possibilities. Thus, the separation procedure described in the following focus on defining which subsets will be tested.

Define $\bar{G} = (\bar{V}, \bar{E})$ as an (undirected) support graph for the fractional solution $\bar{x}$, such as $\bar{V} = J$ and $\bar{E} = \{(i,j) : \exists\, t \mid \bar{x}^t_{ij} > 0 \text{ or } \bar{x}^t_{ji} > 0\}$. Also, define $\bar{C}$ as the average completion time for jobs $j \in J$, calculated as

$$\bar{C}_j = \sum_{(i,t)\mid \bar{x}^t_{ji} > 0} t\, \bar{x}^t_{ji}, \tag{10}$$

$\bar{F}(k)$ as the set of jobs with the k smallest values of $\bar{C}$, $j(k)$ as the job j with the k-th smallest $\bar{C}_j$, $\bar{G}(k)$ as the subgraph of $\bar{G}$ induced by $\bar{F}(k)$, and $\bar{S}(k)$ as the connected component of $\bar{G}(k)$ that includes $j(k)$. In a fractional schedule, the set $\bar{F}(k)$ can be interpreted as the first k jobs to finish, and $j(k)$ as the k-th job to finish.

The genetic algorithm starts by generating an initial population of n solutions. Then, a number of crossover and selection operations are performed iteratively until the stopping criterion is met. For every new individual, being generated either by the initial population or the crossover operation, local search is performed to improved it. Algorithm 1 outlines the genetic algorithm, where the input parameters are the size of the population to be carried from one generation to the next (*nPopulation*), the crossover rate (*crossoverRate*), which is the percentage of the *nPopulation* to be used as the number of individuals generated at each iteration by crossover, and the criterion to stop the algorithm (*stopCriterion*).

As did Uchoa et al. (2006) for separating RHECCs, we require that the set S of every generated cut induces a connected subgraph in $\bar{G}$. However, we allow that induced subgraphs become disconnected during the local search. For shortness, in this section, we refer to sets with this property as connected S sets.

The *Selection* operator consists of eliminating the worst individuals so that the remaining population has exactly *nPopulation* individuals. However, a pool of violated cuts (including the ones found during a local search) is kept apart from the current population. No cut is removed from this pool.

The crossover operator consists in deriving a new solution (say child) from two solutions (say father and mother) randomly chosen from the population. To generate the child, we first select all elements that are in both parents. If this intersection is empty, we include

Algorithm 1: Separation Genetic Algorithm

```
input  : nPopulation, crossoverRate, stopCriterion( )
output: Violatedcuts
ViolatedCuts ← ∅
Population ← ∅
for k ← 1 to n do
    individual ← the most violated cut with S = S̄(k)
    individual ← localSearch(individual, ViolatedCuts)
    Population ← Population + {individual}
end
Population ← Selection(Population, nPopulation)
while not stopCriterion( ) do
    for i ← 1 to nPopulation × crossoverRate do
        father ← random individual from Population
        mother ← random individual from (Population \ {father})
        child ← crossover(father, mother)
        child ← localSearch(child, ViolatedCuts)
        Population ← Population ∪ {child}
    end
    Population ← Selection(Population, nPopulation)
end
return ViolatedCuts
```

one random element from each parent, along with the smallest subset of nodes that yields a connected S. Then, each element contained in the parents is included in S at random, with probability 0.5. If the resulting S is not connected, we select a random subcomponent of S and discard the remaining nodes.

The local search operator, consists of performing every single element insertion and deletion on S until no movement yields a better cut. For every change, the violation is calculated for every valid value of t and we pick the one generating the greater violation, which is considered as minus the constraint slack if it is not violated.

No mutation operator is used. Based on preliminary tests we set *nPopulation* to 20, *crossoverRate* to 100%, and *stopCriterion* to the execution of 100 generations.

3.2. Triangle Clique Cuts

Following the approach of Pessoa et al. (2009), the triangle clique cuts are used here to further strengthen the ATIF.

Given a set $S \subset J$, with exactly three elements, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the compatibility graph where each vertex in $\mathcal{V}$ represents an arc $a^t = (i,j)^t \in A$, i, $j \in S$ and each edge $e = (a_1^{t_1}, a_2^{t_2})$ belongs to $\mathcal{E}$ if and only if $a_1^{t_1}$ and $a_2^{t_2}$ are compatible. For each $i, j, k \in S$, there are 4 compatibility cases:

Case 1: if $e = ((i,j)^{t_1}, (i,k)^{t_2})$, then $e \notin \mathcal{E}$;

Case 2: if $e = ((i,j)^{t_1}, (k,j)^{t_2})$, then $e \notin \mathcal{E}$;

Case 3: if $e = ((i,j)^{t_1}, (j,k)^{t_2})$ and $t_2 \neq t_1 + p_j$, then $e \notin \mathcal{E}$;

Case 4: if $e = ((i,j)^{t_1}, (j,k)^{t_2})$ and $t_2 = t_1 + p_j$, then $e \in \mathcal{E}$.

For any independent set $I \subset \mathcal{V}$, a triangle clique constraint (TCC) is defined by the valid inequality

$$\sum_{a^t \in I} x_a^t \leq 1. \tag{11}$$

Given $\bar{x}$, the separation algorithm consists in building the compatibility graph $\mathcal{G}$ for every triple of jobs (i,j,k), and finding the maximum-weight independent set. As noted by Pessoa et al. (2009), $\mathcal{G}$ is a set of chains, and by preliminary computation we observed that chains of four or more elements don't occur frequently in the subgraph of G induced by the variables with non-zero values. So, in order to use a faster and simpler separation algorithm, we only consider the first three vertices of each chain and search for a violated TCC by enumerating each possible independent set.

3.3. Switching to a MIP Solver

Dyer and Wolsey (1990) proposed a MIP formulation for scheduling problems using one binary variable y_j^t for each job-time to indicate that a job j completes at time t. A time period t is defined as spawning from instant $t-1$ to instant t. This formulation is referred to as the time-indexed formulation (TIF). For the multi machines case, the TIF follows:

$$\text{Minimize} \quad \sum_{j \in J} \sum_{t=p_j}^{T} f_j(t)\, y_j^t \tag{12a}$$

$$\text{Subject to} \quad \sum_{t=p_j}^{T} y_j^t = 1 \qquad (j \in J) \tag{12b}$$

$$\sum_{j \in J} \sum_{s=\max\{p_j,t\}}^{\min\{t+p_j-1,T\}} y_j^s \leq m \qquad (t = 1, \ldots, T) \tag{12c}$$

$$y_j^t \in \{0,1\} \qquad (j \in J;\, t = p_j, \ldots, T). \tag{12d}$$

The objective function (12a) is defined as the sum of the completion cost for all jobs. Constraints (12b) ensure that each job completes once. Constraints (12c) limit the number of jobs on execution to at most m at any time period t. Constraints (12d) enforce integrality.

As performed in the BCP-PMWT, when the solved root node is still fractional, as an alternative to performing branching, we will also feed the residual model to a MIP solver. The model is called residual since some variables may have been fixed to zero by the variable fixation procedure. But instead of feeding the residual ATIF, we project it to the TIF and feed this formulation. In the next section, alternative TIF's to be used in this procedure are describe.

When projecting the ATIF, the bounds provided by the RHECC's, TCC's and OEC's are lost, since such cuts are not translated to the TIF. Also, for a TIF variable y_j^t to be fixed to zero, every variable $x_{ij}^t\,(i \in J_0)$ needs to be fixed to zero, which weakens the fixation. To mitigate this, in Subsection 3.5, we describe a procedure to enhance the TIF linear relaxation bounds, based on the performed ATIF variable fixations.

3.4. Alternative Time-Indexed Formulations

When investigating different formulations for a problem, the usual purpose is to find the one that yields the best bounds. Here, we explore different ways of describing the exactly same polyhedron and discuss how they may influence the MIP solver. We believe two characteristics of a formulation can improve the MIP solver performance.

First is how sparse the constraint matrix is. The sparser, the faster are the bound computations. Second is the impact of fixing one variable over the linear relaxation bound. The more balanced this impact is, the better the MIP solver performs when selecting the branching variables.

When modeling scheduling problems, there are basically two characteristics to be constrained, one is that every job has to be processed, and the other is that no more than m machines are active at any moment. Two ways of modeling the first characteristic are:

1. Use binary variables y_j^t indicating that job j has finished at time t, enforcing for every job j that $\sum_{t=p_j}^{T} y_j^t = 1$.
2. Use binary variables z_j^t indicating that job j has finished until time t, enforcing for every job j that $z_j^{p_j-1} = 0$, $z_j^T = 1$, and $z_j^{t-1} \leq z_j^t$ $(t = p_j, \dots, T)$.

Two ways of modeling the second characteristic are:

1. Formulate the problem as a network flow, where m flow units leave the source, and flow is conserved.
2. Formulate the problem by constraining the number of active machines at any given moment.

By combining these alternatives, four different formulations are yielded. The first two formulations, (13) and (12), use binary variables y_j^t, and differ in how the second characteristic is enforced. The first does it by a network flow (13c, 13d) and the second does it by resource constraints (12c).

$$\text{Minimize} \quad \sum_{j \in J} \sum_{t=p_j}^{T} f_j(t)\, y_j^t \tag{13a}$$

$$\text{Subject to} \quad \sum_{t=p_j}^{T} y_j^t = 1 \quad (j \in J) \tag{13b}$$

$$\sum_{j \in J} y_j^{p_j} = m \tag{13c}$$

$$\sum_{i \in J | t \geq p_i} y_i^t \geq \sum_{j \in J} y_j^{t+p_j} \quad (t = 1, \dots, T) \tag{13d}$$

$$y_j^t \in \{0, 1\} \quad (j \in J;\ t = p_j, \dots, T) \tag{13e}$$

The next two formulations, (14) and (15), use binary variables z_j^t, and again, differ in how the second characteristic is enforced. The first does it by a network flow (14b, 14c) and the second does it by resource constraints (15b).

The main difference of these two formulations is how fixing one variable influences the other variables. For example, by setting $z_j^{t_1} = 1$, every z_j^t with $t > t_1$ are also set to 1, and, by setting $z_j^{t_1} = 0$, every z_j^t with $t < t_1$ are also set to 0. This leads to a more effective branching, since fixing some variable often changes the relaxed solution substantially for both children nodes.

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{j\in J}\sum_{t=p_j}^{T} f_j(t)\,\left(z_j^t - z_j^{t-1}\right) && && \text{(14a)}\\
\text{Subject to} \quad & \sum_{j\in J}\left(z_j^{p_j} - z_j^{p_j-1}\right) = m && && \text{(14b)}\\
& \sum_{j\in J|t\ge p_j}\left(z_j^t - z_j^{t-1}\right) \ge \sum_{j\in J}\left(z_j^{t+p_j} - z_j^{t+p_j-1}\right) && (t=1,\dots,T) && \text{(14c)}\\
& z_j^{t-1} \le z_j^t && (j\in J;\ t=p_j,\dots,T) && \text{(14d)}\\
& z_j^{p_j-1} = 0 && (j\in J) && \text{(14e)}\\
& z_j^t \in \{0,1\} && (j\in J;\ t=p_j,\dots,T-1) && \text{(14f)}\\
& z_j^T = 1 && (j\in J) && \text{(14g)}
\end{aligned}
$$

$$
\begin{aligned}
\text{Minimize} \quad & \sum_{j\in J}\sum_{t=p_j}^{T} f_j(t)\,\left(z_j^t - z_j^{t-1}\right) && && \text{(15a)}\\
\text{Subject to} \quad & \sum_{j\in J}\left(z_j^{\min\{t+p_j-1,T\}} - z_j^{t-1}\right) \le m && (t=1,\dots,T) && \text{(15b)}\\
& z_j^{t-1} \le z_j^t && (j\in J;\ t=p_j,\dots,T) && \text{(15c)}\\
& z_j^{p_j-1} = 0 && (j\in J) && \text{(15d)}\\
& z_j^t \in \{0,1\} && (j\in J;\ t=p_j,\dots,T-1) && \text{(15e)}\\
& z_j^T = 1 && (j\in J) && \text{(15f)}
\end{aligned}
$$

To the best of our knowledge, this is the first time formulation (15) is presented. In the next section, we detail how computational tests will be performed to compare the different formulations, in terms of performance.

3.5. TIF cuts by projecting the ATIF Polytope

As proved by Pessoa et al. (2010), the TIF is dominated by the ATIF, meaning that the ATIF linear relaxation will always yield a better or equal lower bound when compared to the TIF linear relaxation over the same instance. We then devise a procedure to generate cuts from the projection of the ATIF to the TIF. These cuts take further advantage of the variable fixing performed by BCP-PMWT over the ATIF variables.

We define a fractional solution of the TIF as y^* and the set of remaining arcs A_t from the ATIF, for each time period t, as below.

$$A_t = \left\{(i,j)^t \in A \mid x_{ij}^t \text{ is not fixed}\right\} \tag{16}$$

From the network flow time-indexed formulation (13), the following inequality is valid in the y variable space,

$$\sum_{i \in J} y_i^t \geq \sum_{j \in J} y_j^{t+p_j} \qquad (t = 1, \ldots, T). \tag{13d}$$

This inequality can be interpreted as the association of some variable $y_j^{t+p_j}$ equal to one to some variable y_i^t also equal to one, meaning that some job j is the successor of job some job i. Since, by the ATIF (and the variable fixing), not all such successions are allowed, the following inequalities must be valid for the TIF,

$$\sum_{i:\exists j \in S | (i,j)^t \in A_t} y_i^t \geq \sum_{j \in S} y_j^{t+p_j} \qquad (S \subset J;\, t = 1, \ldots, T). \tag{17}$$

For a given time period t, exact separation of (17) can be done by finding the minimum cut on a digraph built as follows. We lay one source node (s) and one sink node (t), n nodes $\{1, 2, ..., n\}$ and other n nodes $\{1', 2', ..., n'\}$. For each variable ${y^*}_i^t \neq 0$, we draw an arc connecting the source node to the node i with capacity $c_{si} = {y^*}_i^t$. For each variable ${y^*}_j^{t+p_j} \neq 0$, we draw an arc connecting the node j' to the sink with capacity $c_{j't} = {y^*}_j^{t+p_j}$. Finally, for each non-fixed variable x_{ij}^t we draw an arc from i to j' with capacity $c_{ij'} = \infty$. After calculating the minimum cut, the set S will be all nodes i' in the sink side of the cut. Figure 2 shows how the graph is constructed.

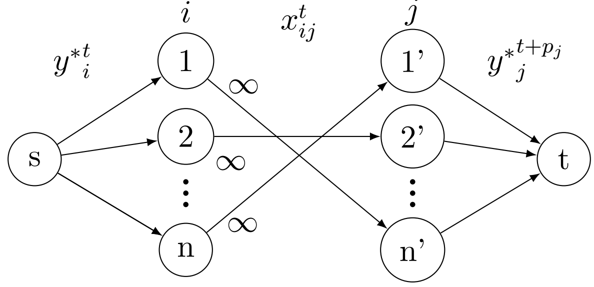

Figure 2 Support graph for separation of cuts derived from ATIF

This procedure can be viewed as a way to check if a solution in the y variables space is feasible in the x variables space. For shortness, we refer to the cuts proposed here as projected cuts.

4. Experimental Results

In this section, we detail how computational tests were performed and comment its results. First, we evaluate the effectiveness of the Triangle Clique and Overload Elimination Cuts on improving the ATIF bounds. Next, regarding the four time-indexed formulations presented, we perform two experiments. The first feeds the 40 and 50 jobs instances residual model to a MIP solver, using all four time-indexed formulations, in order to evaluate which performs better. The second evaluates the effect of variable fixation and projected cuts on the best TIF. Last, we analyze the results of the full BCP-PWMT-OTI algorithm.

The $P||\Sigma w_j T_j$ instances were generated by transforming the $1||\Sigma w_j T_j$ instances of Potts and Wassenhove (1985), found on the OR-Library. As described by Pessoa et al. (2010), for $n \in \{40, 50, 100\}$, and $m \in \{2, 4\}$, the first instance for each group (n, m) was picked (instance numbers ending with 1 or 6) and had its due dates d_j divided by m.

For the primal bounds, we used the solutions computed by the heuristic procedure of Kramer and Subramanian (2015). For better evaluating our improvements to the BCP algorithm, we highlight where those values are better than the ones of Rodrigues et al. (2008), used in the BCP-PMWT. Tests are run only on instances not proven optimal by the ATIF linear relaxation, i.e., the first linear relaxation solution is less than the primal bound.

The LP/MIP solver used was the IBM ILOG CPLEX 12.5. All tests ran on a Intel Core i7-3770 PC with a 3.4Ghz clock (using one thread), 12GB of RAM, and the Linux operating system.

4.1. Solving the Root Node

Table A1 presents the results for the root relaxation after cut separation. The first three columns identify the instance (n, m and Instance number), the next column (UB) gives the upper bound provided by Kramer and Subramanian (2015), the next column gives the ATIF linear relaxation solution (First LB), then, two sets of columns show the root lower bound, its corresponding integrality gap and the running time, for the BCP-PMWT and the BCP-PMWT-OTI. The integrality gap is calculated as $\left(\frac{\text{UB}-\text{Root LB}}{\text{UB}}\right)$. By root lower

Table 1 Root relaxation and cut separation results

		BCP-PMWT		BCP-PMWT-OTI	
n	m	Avg. Gap	Avg. Time	Avg. Gap	Avg. Time
40	2	0.525%	78.0	0.235%	51.9
40	4	0.456%	23.4	0.448%	18.8
50	2	0.379%	256.8	0.276%	193.8
50	4	0.571%	67.8	0.583%	29.9
100	2	0.878%	6297.0	0.114%	3398.8
100	4	0.494%	984.0	0.322%	481.6

bound, we refer to the objective value of the LP after all rounds of cut separation. A cut separation round consist of separating the RHECCs, TCCs and OECs, at once, for a fractional solution. At most 50 cuts from each family are inserted in each round. When column Gap is empty, an optimal solution has been found. The best LB value of each row is highlighted in bold.

Table 1 summarizes the results of both algorithms in solving the root node. It can be seen that the increase in lower bound compensates the time expense of separating OEC cuts. Especially for instances with $n = 100$ and $m = 2$, for example, instance (100-2m-81) which was not solved by BCP-PMWT, was solved at the root using the new cuts.

4.2. Evaluating the best alternative Time-Indexed Formulation

We ran tests for each of the four time-indexed formulations presented in Section 3.4, in order to compare them. To save time, we constrained the tests to 40 and 50 jobs instances. But we believe that any conclusions drawn can be assumed true for bigger instances. For all formulations, we used the residual model after the root node of the BCP-PMWT-OTI.

Table B1 shows the test results for the 29 instances of 40 and 50 jobs not solved at the root node. In it, we refer to formulations (13), (12), (14) and (15) as Fy, Ry, Fz and Rz respectively. The first three columns (n, m and Instance number) identify the instance, and the two sets of columns gives the time for solving the linear relaxation plus all cut generations (LP Time), and the time the MIP solver took to solve the TIF formulation strengthened by projected cuts (MIP Time).

Table 2 Comparison of Alternative Time-Indexed Formulations – Summary

	Average LP Time (s)				Average MIP Time (s)				# Solved			
n	Fy	Ry	Fz	Rz	Fy	Ry	Fz	Rz	Fy	Ry	Fz	Rz
40	0.72	0.84	7.17	0.97	63.17	351.97	122.92	58.28	12	10	12	12
50	1.77	1.98	47.08	2.43	53.46	150.26	70.56	16.47	13	11	14	16

Table 3 Effect of Variable Fixation in the Rz Time-Index Formulation – Summary

	Average LP Time (s)		Average MIP Time (s)		# Solved	
n	Fix.	w/ Fix.	Fix.	w/ Fix.	Fix.	w/ Fix.
40	0.74	22.54	11.11	561.59	12	10
50	2.04	105.84	11.63	496.61	17	9

Table 2 summarizes Table B1 by giving the average values for the two instance sets, as well as the number of instances solved in up to 3,600 seconds (# Solved). The averages consider only the 10 instances solved by all four formulation. It can be observed that the best choice is formulation Rz, even though solving the linear relaxation sometimes may be faster for a different TIF, the branching performed by the MIP solver is more effective for Rz.

Table B2 shows, for the same instances of the previous test, the effect of fixing variables in the TIF, based on the state achieved in the ATIF. It can be seen that both the LP Time and MIP time decreases dramatically. Table 3 summarizes B2. Again, the averages consider only the 19 instances solved in up to 3,600 seconds by all four formulation.

Table B3 shows the projected cuts improvement on the TIF linear relaxation bounds. There is a huge loss of lower bound when projecting the ATIF into the TIF, even with variable fixation because the RHECCs, TCCs and OECs are not translated to the time-indexed variables. To alleviate this loss, the projected cuts plays an important role. For example, on instance (100-4m-86), it closes 27.36% of the integrality gap. Without such improvement, the optimal solution would possibly not be achieved. Table 4 summarizes table B3.

It can be seen that both the generation of cuts and the fixing of TIF variables are very effective, being the latter the most significant. For some instances, the generation of cuts

Table 4 Effect of Projected Cuts in the Rz Time-Indexed Formulation – Summary

		ATIF	TIF		
n	m	Root Gap	1st LP Gap	Root Gap	Gap Improv.
100	2	0.114%	0.294%	0.249%	16.76%
100	4	0.322%	0.660%	0.646%	11.20%

worsen the Branch and Bound performance due to the LP increase in complexity. We tried to balance the generation of cuts and this increase of complexity by implementing a rollback procedure, in which the cut generation stops and the cuts inserted in the last iteration are removed if the time for solving the strengthened relaxation more than doubles. Also, we try to avoid tailing off by stopping the cut generation if the objective increases less than 10^{-3} on 5 sequential iterations.

4.3. The Full Improved Algorithm

Tables C1 to C6 present the results of solving the 106 instances by the BCP-PMWT and the BCP-PMWT-OTI. Each table corresponds to a set (m,n) of instances, where the first column identify the instance. The results for the two algorithms are separated into two sets of columns, where column UB gives the upper bound used by both the BCP and the MIP solver to explore the search tree, column Root LB and Root Time gives the lower bound for the ATIF strengthened with the algorithm's family of cuts and the time to solve the first LP and all subsequent cut insertions by CG, columns BCP Time and ATIF/TIF MIP Time gives the time it took to achieve the integral optimal solution, by BCP from root and by feeding the ATIF/TIF to the MIP solver. Column Best gives which part of the algorithm achieved the optimal solution faster and column Overall Time give the overall time (Root Time, Root+BCP Time or Root + ATIF/TIF MIP Time). We highlight (by bold printed numbers) in column Inst the instances solved for the first time. In column UB for the BCP-PMWT-OTI, we highlight where the improved bounds were used, and, in columns Root LB, we highlight the best lower bound for the instance.

For the TIF, we used model Rz (15), which proved to be the best choice. When feeding the MIP solver, the ATIF carries the cuts separated in the root node, and the TIF do not. Each set (m,n) have 25 instances, the instances proven optimal by the first ATIF LP

Table 5 Full Results - Summary

		BCP-PMWT		BCP-PMWT-OTI	
n	m	# Solved	Avg. Time	# Solved	Avg. Time
40		50	357.9	50	48.1
50		50	5734.9	50	241.9
100	2	18	22523.8	21	7058.5
100	4	16	37667.7	22	5672.0

Table 6 BCP-PMWT-OTI Best Procedure

		BCP-PMWT				BCP-PMWT-OTI			
n	m	Root	BCP	ATIF MIP	Unsolved	Root	BCP	TIF MIP	Unsolved
40		38	2	10	0	38	1	11	0
50		33	4	13	0	33	3	14	0
100	2	13	2	3	7	16	1	4	4
100	4	7	5	4	9	7	1	14	3

relaxation are omitted from the table, but are accounted in the number of Solved instances. Tables 5 summarizes the six tables. All 40 and 50 jobs instances were already solved, but the better lower bounds and the TIF improved running times. Three instances of the 100-2m set and five instances of the 100-4m set were solved for the first time. Also, there was an improvement of running time.

Table 6 compares the BCP approach to MIP solver approach, for both the BCP-PMWT and the BCP-PMWT-OTI. It can be noted that in both algorithms, the best approach was the MIP solver for most of the instances. Even though we did not run the experiment of feeding the residual ATIF model to a MIP solver again, as the Root LB of some instances did not improve with the new cuts, we can say that the residual TIF model performs better.

Figure 3 displays the integrality gap achieved on different settings, for each instance, a set of five columns represents the lower bound achieved by the ATIF, the ATIF after the

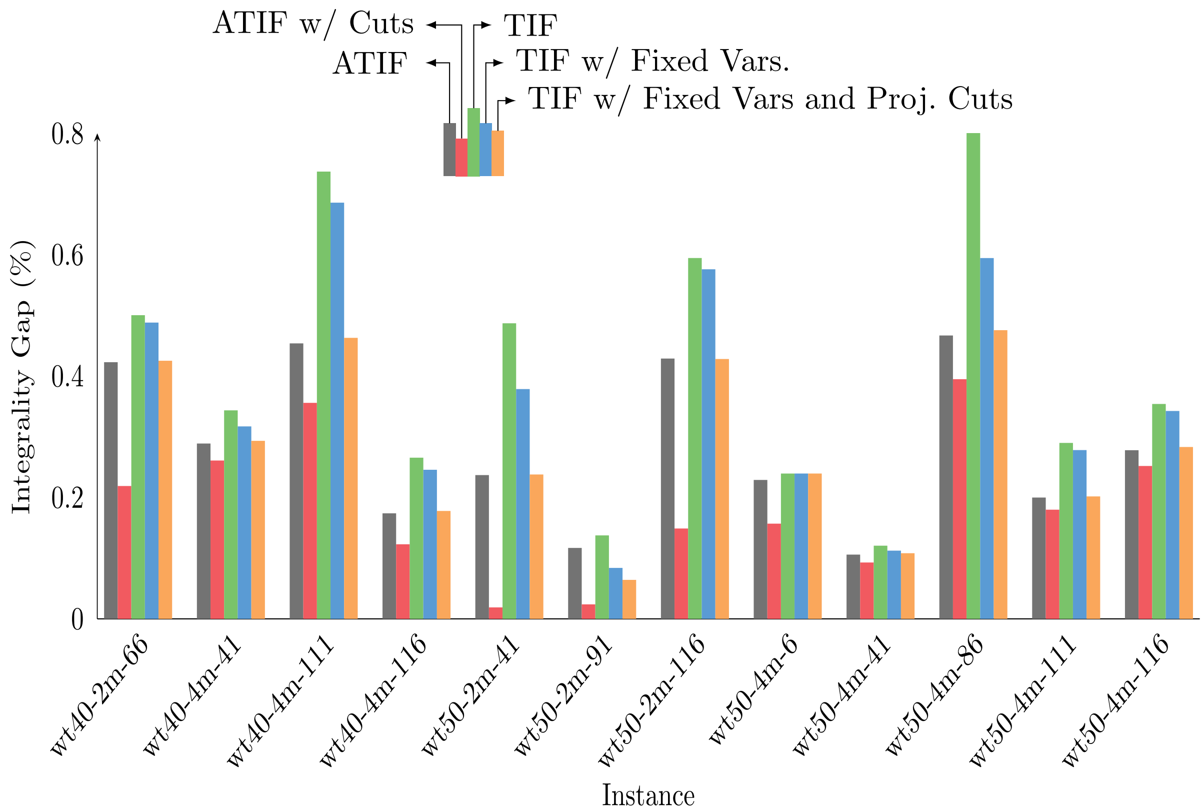

Figure 3 **Integrality Gap along different settings**

separation of all cuts, the TIF, the residual TIF and the residual TIF after separating all projected cuts. A few interesting observations can be made from the figure, some are:

- how the cuts improve the ATIF LB,
- how far the TIF LB is from the ATIF LB,
- how considering fixed variables improves the TIF,
- how the projected cuts improve the TIF, taking its LB very close to the one of the ATIF, possibly surpassing it, as occurred with instances wt50-2m-91 and wt50-2m-116.

The fact that the TIF with fixed variables and projected cuts can achieve a LB better than the one achieved by the ATIF shows that some information from the ATIF cuts can be carried to the TIF by means of the fixed variables.

5. Conclusions

This work proposed a set of improvements to the BCP algorithm of (Pessoa et al. 2010), referred to as BCP-PMWT throughout the text. The resulting algorithm, referred to as BCP-PMWT-OTI, included a new family of cuts, the Overload Elimination Cuts, with an efficient algorithm for separation. Also, we projected the Arc-Time-Indexed formulation into the Time-Indexed formulation for solving the residual model with a MIP solver, which we consider to be the main contribution of this work. For the instances that could be solved both by feeding the ATIF and the TIF residual model to the MIP solver, the average MIP solution time decreases 92.7% by using the latter, even though its bounds are worse.

For the $P||\sum w_jT_j$ instances that could be solved by both algorithms, the proposed improvements resulted in an average solution time decrease of 84.1%, solving 9 instances for the first time, leaving 7 instances still unsolved.

For future works, a few suggestions are:

- Explore if performing variable fixation on an extended formulation, and projecting it onto a more tractable one could be applied to other combinatorial optimization problems.
- Explore new branching schemes for the BCP algorithm.
- Strengthen the Time-Indexed Formulation with different family of cuts.
- Translate the RHECCs, Triangle Clique Cuts and OECs to the Time-Indexed Formulation.

Acknowledgments

Daniel Oliveira received financial support from CAPES and Artur Pessoa from both CNPq and FAPERJ.

References

Abdul-Razaq T, Potts CN, Van Wassenhove LN (1990) A survey of algorithms for the single machine total weighted tardiness scheduling problem. *Discrete Applied Mathematics* 26(2):235–253.

Abdul-Razaq TS, Potts CN (1988) Dynamic programming state-space relaxation for single-machine scheduling. *The Journal of the Operational Research Society* 39(2):141–152.

Dyer ME, Wolsey LA (1990) Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Applied Mathematics* 26(2):255–270.

Ibaraki T (1987) Successive sublimation methods for dynamic programming computation. *Annals of Operations Research* 11(1):398–439i.

Ibaraki T, Nakamura Y (1994) A dynamic programming method for single machine scheduling. *European Journal of Operational Research* 76(1):72 – 82.

Kramer A, Subramanian A (2015) A unified heuristic and an annotated bibliography for a large class of earliness-tardiness scheduling problems. *CoRR* abs/1509.02384, URL `http://arxiv.org/abs/1509.02384`.

Marcus Poggi de Aragão EU (2003) Integer Program Reformulation for Robust Branch-and-Cut-and-Price Algorithms. *In Proceedings of the Conference Mathematical Program in Rio: A Conference in Honour of Nelson Maculan* 56–61 p.

Pessoa A, Uchoa E, de Aragão M, Rodrigues R (2010) Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation* 2(3-4):259–290.

Pessoa A, Uchoa E, De Aragão MP (2009) A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. *Networks* .

Potts CN, Wassenhove LNV (1985) A branch and bound algorithm for the total weighted tardiness problem. *Operations Research* .

Rodrigues R, Pessoa A, Uchoa E, Poggi de Aragão M (2008) Heuristic algorithm for the parallel machine total weighted tardiness scheduling problem .

Tanaka S, Araki M (2006) A branch-and bound algorithm based on lagrangian relaxation for single machine scheduling. *Proceedings of International Symposium on Scheduling*, volume 18–20, 148–153.

Tanaka S, Araki M (2013) An exact algorithm for the single-machine total weighted tardiness problem with sequence-dependent setup times. *Computers & Operations Research* 40(1):344–352.

Tanaka S, Fujikuma S (2012) A dynamic-programming-based exact algorithm for general single-machine scheduling with machine idle time. *Journal of Scheduling* 15(3):347–361.

Tanaka S, Fujikuma S, Araki M (2009) An exact algorithm for single-machine scheduling without machine idle time. *Journal of Scheduling* 12(6):575–593.

Uchoa E, Fukasawa R, Lysgaard J, Pessoa A, de Aragão MP, Andrade D (2006) Robust branch-cut-and-price for the Capacitated Minimum Spanning Tree problem over a large extended formulation. *Mathematical Programming* 112(2):443–472.

Wentges P (1997) Weighted dantzig–wolfe decomposition for linear mixed-integer programming. *International Transactions in Operational Research* 4(2):151–162.

Appendix

A. Root Node Full Results

Table A1: Root relaxation and cut separation results

					BCP-PMWT-OTI			BCP-PMWT		
n	m	Inst	UB	First LB	Root LB	Gap	Time	Root LB	Gap	Time
40	2	1	606	584	606		35.80	606		63.50
40	2	6	3886	3875	3886		6.30	3886		11.80
40	2	11	9617	9592	9617		5.40	9617		6.40
40	2	16	38356	38279	38356		25.10	38356		41.80

Table A1: continued

					BCP-PMWT-OTI			BCP-PMWT		
n	m	Inst	UB	First LB	Root LB	Gap	Time	Root LB	Gap	Time
40	2	31	3812	3758	3812		24.10	3812		29.20
40	2	36	10713	10662	10713		33.30	10713		50.70
40	2	56	1279	1272	1279		5.10	1279		8.30
40	2	61	11488	11311	**11459**	0.252%	312.90	11394	0.818%	684.80
40	2	66	35279	35130	**35202**	0.218%	224.90	35197	0.232%	200.60
40	2	71	47952	47935	47952		5.40	47952		0.90
40	2	81	571	452	571		105.40	571		67.60
40	2	86	6048	5996	6048		20.40	6048		31.90
40	2	96	66116	66111	66116		3.90	66116		1.20
40	2	111	17936	17898	17936		6.50	17936		3.50
40	2	116	25870	25786	25870		11.70	25870		44.40
40	2	121	64516	64507	64516		3.60	64516		1.00
40	4	1	439	438	439		3.60	439		1.70
40	4	6	2374	2372	2374		3.20	2374		2.30
40	4	11	5737	5735	5737		2.70	5737		0.50
40	4	16	21493	21484	**21493**		26.80	21490	0.014%	19.90
40	4	31	2525	2496	**2507**	0.713%	20.40	2500	0.990%	39.30
40	4	36	6420	6355	**6368**	0.810%	30.50	6364	0.872%	26.50
40	4	41	17685	17634	**17639**	0.260%	12.00	17637	0.271%	24.30
40	4	56	826	798	816	1.211%	17.70	**817**	1.090%	45.50
40	4	61	7357	7316	**7331**	0.353%	76.80	7322	0.476%	22.80
40	4	66	20251	20247	20251		2.50	20251		1.70
40	4	81	564	550	560	0.709%	12.40	560	0.709%	37.90
40	4	86	4725	4719	4725		5.00	4725		6.80
40	4	91	15569	15557	15562	0.045%	16.60	15562	0.045%	27.00
40	4	111	11263	11212	**11222**	0.364%	29.00	11219	0.391%	58.60
40	4	116	15566	15539	**15565**	0.006%	18.60	15545	0.135%	19.70
40	4	121	35751	35739	**35749**	0.006%	22.60	35741	0.028%	39.10
50	2	1	1268	1232	1268		107.40	1268		296.60
50	2	6	14272	14261	14272		28.60	14272		63.10
50	2	11	23028	23000	23028		12.80	23028		21.30
50	2	16	46072	46011	46072		17.80	46072		25.10
50	2	21	111069	111067	111069		5.40	111069		0.70
50	2	31	5378	5289	**5355**	0.428%	110.80	5349	0.539%	1060.20
50	2	36	18956	18895	18956		15.90	18956		26.80
50	2	41	38058	37968	**38051**	0.018%	641.00	38050	0.021%	341.60
50	2	46	82105	82085	82105		32.20	82105		31.50
50	2	56	761	730	761		79.40	761		232.70
50	2	61	13682	13589	**13682**		1250.80	13619	0.460%	614.00
50	2	66	40907	40904	40907		5.40	40907		0.60
50	2	81	542	538	542		8.30	542		5.30
50	2	86	12557	12277	**12460**	0.772%	415.60	12427	1.035%	946.70
50	2	91	47349	47294	**47342**	0.015%	406.50	47330	0.040%	492.30
50	2	96	92822	92803	92822		8.50	92822		6.30
50	2	111	15564	15544	15564		7.70	15564		3.50
50	2	116	19608	19524	**19579**	0.148%	334.80	19573	0.178%	453.30
50	4	1	785	777	785		26.90	785		24.80
50	4	6	8317	8298	8304	0.156%	31.60	8304	0.156%	110.80
50	4	11	12879	12871	**12876**	0.023%	40.80	12875	0.031%	71.20
50	4	16	25376	25375	25376		7.80	25376		8.00
50	4	36	10796	10794	10796		4.70	10796		5.00
50	4	41	21806	21783	**21786**	0.092%	65.40	21785	0.096%	44.90
50	4	46	44455	44452	44453	0.004%	8.40	**44455**		15.10

Table A1: continued

					BCP-PMWT-OTI			BCP-PMWT		
n	m	Inst	UB	First LB	Root LB	Gap	Time	Root LB	Gap	Time
50	4	56	570	538	540	5.263%	13.80	**541**	5.088%	45.80
50	4	61	7898	7850	7857	0.519%	30.90	7857	0.519%	81.80
50	4	71	42645	42625	42627	0.042%	25.50	**42628**	0.040%	138.40
50	4	81	495	478	495		24.80	495		64.90
50	4	86	8369	8330	8336	0.394%	65.70	8336	0.394%	163.30
50	4	91	26551	26546	**26551**		10.70	26548	0.011%	111.60
50	4	96	50326	50312	**50324**	0.004%	58.20	50317	0.018%	91.00
50	4	111	10069	10049	10051	0.179%	32.10	10051	0.179%	63.20
50	4	116	11552	11520	11523	0.251%	24.80	11523	0.251%	54.40
50	4	121	23792	23769	23775	0.071%	35.50	23775	0.071%	58.50
100	2	1	3339	3314	**3339**		1006.50	3322	0.509%	814.60
100	2	6	30665	30644	30665		379.10	30665		328.10
100	2	11	93894	93894	93894		33.80	93894		17.20
100	2	16	209100	209062	209100		98.30	209100		5086.70
100	2	21	457836	457814	457836		425.40	457836		24.30
100	2	31	12729	12725	12729		135.80	12729		210.50
100	2	36	56671	56576	**56620**	0.090%	1212.10	56590	0.143%	804.70
100	2	41	237964	237773	**237881**	0.035%	3462.30	237841	0.052%	19685.20
100	2	46	422831	422804	**422831**		303.20	422830	0.000%	1525.30
100	2	56	5047	4983	5047		271.90	5047		796.90
100	2	61	45573	45424	**45487**	0.189%	1896.70	45481	0.202%	6818.20
100	2	66	126513	126408	**126493**	0.016%	6123.10	126477	0.028%	6969.70
100	2	71	327305	327300	327305		55.60	327305		3.10
100	2	81	908	792	**908**		451.30	830	8.590%	968.90
100	2	86	36581	36218	**36402**	0.489%	2192.60	36322	0.708%	3234.00
100	2	91	129929	129619	**129798**	0.101%	8362.50	129752	0.136%	3617.10
100	2	96	254194	254141	254194		274.60	254194		400.20
100	2	111	84220	84005	**84139**	0.096%	5217.10	84097	0.146%	6578.00
100	2	116	191186	191085	**191185**	0.001%	4645.10	191173	0.007%	11118.60
100	2	121	242018	241954	**241999**	0.008%	31429.90	241997	0.009%	56938.90
100	4	1	2001	1990	2001		145.70	2001		668.10
100	4	11	50232	50199	50203	0.058%	291.40	50203	0.058%	951.30
100	4	16	110219	110114	**110120**	0.090%	326.40	110118	0.092%	950.60
100	4	21	237392	237389	**237391**	0.000%	378.40	237390	0.001%	1047.80
100	4	31	7130	7080	7082	0.673%	301.60	7082	0.673%	1156.70
100	4	36	30791	30773	30782	0.029%	347.40	**30783**	0.026%	1654.20
100	4	41	126185	126130	**126147**	0.030%	1186.30	126144	0.032%	923.40
100	4	46	219536	219526	219529	0.003%	592.30	219529	0.003%	910.00
100	4	56	3076	3021	3030	1.495%	590.90	3030	1.495%	1102.50
100	4	61	24856	24806	**24825**	0.125%	726.40	24821	0.141%	1094.90
100	4	66	67970	67948	**67955**	0.022%	445.90	67954	0.024%	819.70
100	4	71	170691	170674	170675	0.009%	418.20	170675	0.009%	398.80
100	4	81	819	754	**797**	2.686%	425.70	772	5.739%	1242.50
100	4	86	21286	21209	**21220**	0.310%	519.40	21218	0.319%	798.60
100	4	91	70608	70582	70587	0.030%	605.10	**70589**	0.027%	1640.70
100	4	96	133587	133572	133575	0.009%	321.40	133575	0.009%	443.80
100	4	111	46719	46616	**46633**	0.184%	382.60	46630	0.191%	1094.30
100	4	116	101551	101514	**101521**	0.030%	512.80	101520	0.031%	871.70
100	4	121	127619	127593	**127598**	0.016%	633.20	127597	0.017%	926.70

B. Alternative Time-Index Formulations Performance

Table B1: Comparison of Alternative Time-Indexed Formulations

			LP Time(s)				MIP Time (s)			
n	m	Inst	Fy	Ry	Fz	Rz	Fy	Ry	Fz	Rz
40	2	61	0.78	0.57	7.05	1.15	235.6	1379.7	712.1	238.9
40	2	66	2.72	2.75	43.87	3.01	218.4	1350.1	460.3	316.7
40	4	31	0.47	0.93	10.30	0.82	996.8	≥3600	57.4	32.0
40	4	36	0.38	0.37	4.25	0.40	140.8	676.4	15.7	7.1
40	4	41	0.44	0.51	4.47	0.58	13.3	34.9	4.5	4.7
40	4	56	0.46	1.21	8.33	1.54	28.0	≥3600	340.0	52.9
40	4	61	0.18	0.22	1.32	0.39	12.3	44.3	13.8	5.8
40	4	81	0.05	0.06	0.27	0.08	0.1	0.7	0.2	0.6
40	4	91	0.01	0.04	0.03	0.02	0.1	0.3	0.3	0.2
40	4	111	0.19	0.23	1.78	0.46	5.6	19.3	11.1	3.7
40	4	116	0.14	0.19	0.99	0.22	4.7	9.3	7.9	3.1
40	4	121	0.12	0.34	0.70	0.19	1.0	4.6	3.3	2.0
50	2	31	10.56	9.29	515.22	13.62	≥3600	≥3600	≥3600	≥3600
50	2	41	1.54	2.33	18.56	1.80	3.9	18.0	30.3	12.9
50	2	61	3.44	2.30	94.61	3.39	≥3600	≥3600	296.3	26.4
50	2	86	2.28	1.42	38.53	4.04	2631.5	≥3600	≥3600	2109.8
50	2	91	1.09	1.31	7.24	1.71	6.1	30.8	50.8	23.6
50	2	116	1.20	1.28	20.39	1.70	≥3600	≥3600	≥3600	3066.9
50	4	6	0.86	2.08	51.40	1.42	1001.5	≥3600	9.2	16.9
50	4	11	0.52	0.63	4.61	0.57	4.7	15.4	49.5	11.4
50	4	41	0.56	1.10	7.67	0.67	4.3	29.8	25.9	15.1
50	4	46	0.01	0.05	0.04	0.03	0.1	0.1	0.1	0.1
50	4	56	1.24	3.20	93.93	5.61	≥3600	≥3600	2352.9	317.9
50	4	61	0.98	0.76	16.16	0.79	≥3600	≥3600	260.4	804.5
50	4	71	0.62	0.88	5.30	0.46	3.1	14.1	26.2	10.7
50	4	86	0.21	0.31	1.67	0.34	27.4	46.9	9.6	3.1
50	4	91	0.05	0.18	0.22	0.10	0.2	0.3	0.7	0.2
50	4	96	0.39	0.71	3.37	0.31	8.8	23.2	13.3	17.0
50	4	111	0.15	0.19	0.96	0.44	1.8	3.6	3.6	3.1
50	4	116	0.43	0.47	3.74	0.53	489.8	1415.7	558.1	76.0
50	4	121	0.49	0.94	5.01	0.48	38.2	55.3	8.8	8.2

Table B2: Effect of Variable Fixation in the Rz Time-Index Formulation

			First LP Time (s)		MIP Time (s)	
n	m	Inst	Fix.	w/ Fix.	Fix.	w/ Fix.
40	2	61	1.16	83.16	232.8	≥3600
40	2	66	3.03	68.65	389.4	≥3600
40	4	31	0.81	15.28	11.4	698.4
40	4	36	0.40	12.70	16.2	2562.7
40	4	41	0.57	15.05	6.1	70.0
40	4	56	1.54	10.72	59.8	1528.7
40	4	61	0.39	11.36	6.8	157.2

Table B2: continued

			First LP Time (s)		MIP Time (s)	
n	m	Inst	Fix.	w/ Fix.	Fix.	w/ Fix.
40	4	81	0.08	18.15	0.4	44.6
40	4	91	0.02	9.06	0.2	50.3
40	4	111	0.45	14.55	6.0	296.1
40	4	116	0.22	8.22	2.0	106.4
40	4	121	0.19	3.56	2.1	101.5
50	2	31	13.71	717.17	3010.9	≥3600
50	2	41	1.80	193.44	20.3	≥3600
50	2	86	4.03	257.36	2616.2	≥3600
50	2	91	1.71	172.52	31.0	≥3600
50	2	116	1.70	120.13	3081.3	≥3600
50	4	6	1.43	71.88	12.8	1921.7
50	4	11	0.56	17.88	15.4	394.7
50	4	41	0.67	19.62	18.7	193.5
50	4	46	0.03	3.56	0.1	9.3
50	4	56	5.62	20.79	252.5	≥3600
50	4	61	0.79	70.42	97.1	≥3600
50	4	71	0.46	10.51	17.7	157.9
50	4	86	0.35	37.98	5.3	475.7
50	4	96	0.32	6.87	14.1	162.7
50	4	111	0.45	50.12	8.4	907.0
50	4	116	0.53	13.87	395.6	≥3600
50	4	121	0.48	15.13	12.2	246.9

Table B3: Effect of Projected Cuts in the Rz Time-Indexed Formulation

				ATIF		TIF				
n	m	Inst	Heu UB	Root LB	Root Gap	1st LP LB	1st LP Gap	Root LB	Root Gap	Gap Improv.
100	2	36	56671	56620	0.090%	56557	0.201%	56575	0.169%	15.79%
100	2	41	237964	237881	0.035%	237734	0.097%	237771	0.081%	16.09%
100	2	61	45573	45487	0.189%	45370	0.445%	45421	0.334%	25.12%
100	2	66	126513	126493	0.016%	126374	0.110%	126405	0.085%	22.30%
100	2	86	36581	36402	0.489%	36171	1.121%	36217	0.995%	11.22%
100	2	91	129929	129798	0.101%	129557	0.286%	129618	0.239%	16.40%
100	2	111	84220	84139	0.096%	83970	0.297%	84004	0.256%	13.60%
100	2	116	191186	191185	0.001%	191072	0.060%	191084	0.053%	10.53%
100	2	121	242018	241999	0.008%	241937	0.033%	241953	0.027%	19.75%
100	4	11	50232	50203	0.058%	50194	0.076%	50198	0.068%	10.53%
100	4	16	110219	110120	0.090%	110102	0.106%	110111	0.098%	7.69%
100	4	21	237392	237391	0.000%	237388	0.002%	237388	0.002%	0.00%
100	4	31	7130	7082	0.673%	7080	0.701%	7080	0.701%	0.00%
100	4	36	30791	30782	0.029%	30773	0.058%	30773	0.058%	0.00%
100	4	41	126185	126147	0.030%	126126	0.047%	126130	0.044%	6.78%
100	4	46	219536	219529	0.003%	219525	0.005%	219526	0.005%	9.09%
100	4	56	3076	3030	1.495%	3021	1.788%	3021	1.788%	0.00%
100	4	61	24856	24825	0.125%	24795	0.245%	24806	0.201%	18.03%
100	4	66	67970	67955	0.022%	67938	0.047%	67947	0.034%	28.13%
100	4	71	170691	170675	0.009%	170671	0.012%	170673	0.011%	10.00%
100	4	81	819	797	2.686%	754	7.937%	754	7.937%	0.00%

Table B3: continued

				ATIF		TIF				
n	m	Inst	Heu UB	Root LB	Root Gap	1st LP LB	1st LP Gap	Root LB	Root Gap	Gap Improv.
100	4	86	21286	21220	0.310%	21180	0.498%	21209	0.362%	27.36%
100	4	91	70608	70587	0.030%	70575	0.047%	70582	0.037%	21.21%
100	4	96	133587	133575	0.009%	133561	0.019%	133572	0.011%	42.31%
100	4	111	46719	46633	0.184%	46608	0.238%	46615	0.223%	6.31%
100	4	116	101551	101521	0.030%	101510	0.040%	101513	0.037%	7.32%
100	4	121	127619	127598	0.016%	127590	0.023%	127592	0.021%	6.90%

C. Full Results

Table C1: Detailed Results for $m = 2$ and $n = 40$ instances

	BCP-PMWT								BCP-PMWT-OTI							
		Root		BCP	CPLEX	Overall				Root		BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
1	606	606	80.9			Root	80.9	606	606	606	35.8			Root	35.8	606
6	3886	3886	27.7			Root	27.7	3886	3886	3886	6.3			Root	6.3	3886
11	9617	9617	22.3			Root	22.3	9617	9617	9617	5.4			Root	5.4	9617
16	38356	38356	59.1			Root	59.1	38356	38356	38356	25.1			Root	25.1	38356
31	3812	3812	49.2			Root	49.2	3812	3812	3812	24.1			Root	24.1	3812
36	10713	10713	65.7			Root	65.7	10713	10713	10713	33.3			Root	33.3	10713
56	1279	1279	24.3			Root	24.3	1279	1279	1279	5.1			Root	5.1	1279
61	11488	11394	705.7	2398.5	1994.0	MIP	2699.7	11488	11488	**11459**	312.9	526.3	237.67	MIP	550.6	11488
66	35279	35197	220.4	307.8	1776.0	BCP	528.2	35279	35279	**35202**	224.9	699.8	86.47	MIP	311.4	35279
71	47952	47952	15.6			Root	15.6	47952	47952	47952	5.4			Root	5.4	47952
81	573	571	83.7			Root	83.7	571	**571**	571	105.4			Root	105.4	571
86	6048	6048	46.2			Root	46.2	6048	6048	6048	20.4			Root	20.4	6048
96	66116	66116	22.8			Root	22.8	66116	66116	66116	3.9			Root	3.9	66116
111	17936	17936	28.1			Root	28.1	17936	17936	17936	6.5			Root	6.5	17936
116	25874	25870	66.6			Root	66.6	25870	**25870**	25870	11.7			Root	11.7	25870
121	64516	64516	24.3			Root	24.3	64516	64516	64516	3.6			Root	3.6	64516

Table C2: Detailed Results for $m = 4$ and $n = 40$ instances

	BCP-PMWT								BCP-PMWT-OTI							
		Root		BCP	CPLEX	Overall				Root		BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
1	439	439	11.4			Root	11.4	439	439	439	3.6			Root	3.6	439
6	2374	2374	11.1			Root	11.1	2374	2374	2374	3.2			Root	3.2	2374
11	5737	5737	10.3			Root	10.3	5737	5737	5737	2.7			Root	2.7	5737
16	21493	21490	30.4			Root	30.4	21493	21493	**21493**	26.8			Root	26.8	21493
31	2525	2500	50.1	2932.6	12172.3	BCP	2982.7	2525	2525	**2507**	20.4	35.4	12.32	MIP	32.7	2525
36	6420	6364	37.3	18799.5	2829.7	MIP	2867.0	6420	6420	**6368**	30.5	9287.9	5.96	MIP	36.5	6420
41	17685	17637	36.1	1856.0	128.0	MIP	164.1	17685	17685	**17639**	12.0	378.4	3.23	MIP	15.2	17685
56	826	**817**	54.6	>86400	878.0	MIP	932.6	826	826	816	17.7	>14387.3	49.65	MIP	67.4	826
61	7357	7322	33.8	230.2	143.0	MIP	176.8	7357	7357	**7331**	76.8	194.3	4.70	MIP	81.5	7357

Table C2: Detailed Results for $m=4$ and $n=40$ instances

	BCP-PMWT								BCP-PMWT-OTI							
		Root		BCP	CPLEX	Overall				Root		BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
66	20251	20251	12.9			Root	12.9	20251	20251	20251	2.5			Root	2.5	20251
81	565	560	46.4	9.7	0.9	MIP	47.3	564	**564**	560	12.4	0.4	0.04	MIP	12.4	564
86	4725	4725	15.3			Root	15.3	4725	4725	4725	5.0			Root	5.0	4725
91	15569	15562	39.4	9.6	0.7	MIP	40.1	15569	15569	15562	16.6	0.5	0.07	BCP	17.1	15569
111	11263	11219	72.0	303.9	105.3	MIP	177.3	11263	11263	**11222**	29.0	111.8	5.54	MIP	34.5	11263
116	15566	15545	31.3	101.4	40.4	MIP	71.7	15566	15566	**15565**	18.6	21.1	2.93	MIP	21.5	15566
121	35751	35741	52.0	42.6	4.7	MIP	56.7	35751	35751	**35749**	22.6	2.3	1.28	MIP	23.9	35751

Table C3: Detailed Results for $m=2$ and $n=50$ instances

	BCP-PMWT								BCP-PMWT-OTI							
		Root		BCP	CPLEX	Overall				Root		BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
1	1268	1268	345.8			Root	345.8	1268	1268	1268	107.4			Root	107.4	1268
6	14272	14272	122.8			Root	122.8	14272	14272	14272	28.6			Root	28.6	14272
11	23028	23028	67.1			Root	67.1	23028	23028	23028	12.8			Root	12.8	23028
16	46072	46072	59.4			Root	59.4	46072	46072	46072	17.8			Root	17.8	46072
21	111069	111069	36.1			Root	36.1	111069	111069	111069	5.4			Root	5.4	111069
31	5378	5349	1110.2	11608.9	11997.5	BCP	12719.1	5378	5378	**5355**	110.8	154.7	4698.26	BCP	265.5	5378
36	18956	18956	73.3			Root	73.3	18956	18956	18956	15.9			Root	15.9	18956
41	38058	38050	381.5	16.7	43.5	BCP	398.2	38058	38058	**38051**	641.0	639.0	15.99	MIP	657.0	38058
46	82105	82105	66.0			Root	66.0	82105	82105	82105	32.2			Root	32.2	82105
56	761	761	267.4			Root	267.4	761	761	761	79.4			Root	79.4	761
61	13682	13619	661.4	83122.8	36094.6	MIP	36756.0	13682	13682	**13682**	1250.8			Root	1250.8	13682
66	40907	40907	37.5			Root	37.5	40907	40907	40907	5.4			Root	5.4	40907
81	542	542	40.5			Root	40.5	542	542	542	8.3			Root	8.3	542
86	12557	12427	999.3	13817.7	23070.0	BCP	14817.0	12557	12557	**12460**	415.6	2758.8	1894.57	MIP	2310.2	12557
91	47349	47330	552.6	210.6	204.5	MIP	757.1	47349	47349	**47342**	406.5	348.4	26.55	MIP	433.0	47349
96	92822	92822	60.9			Root	60.9	92822	92822	92822	8.5			Root	8.5	92822
111	15564	15564	66.6			Root	66.6	15564	15564	15564	7.7			Root	7.7	15564
116	19609	19573	508.8	2390.7	8539.5	BCP	2899.5	19608	**19608**	**19579**	334.8	664.3	2917.71	BCP	999.1	19608

Table C4: Detailed Results for $m = 4$ and $n = 50$ instances

	BCP-PMWT								BCP-PMWT-OTI							
	Root			BCP	CPLEX	Overall			Root			BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
1	785	785	56.4			Root	56.4	785	785	785	26.9			Root	26.9	785
6	8317	8304	147.1	>86400	78140.0	MIP	78287.1	8317	8317	8304	31.6	1211.5	21.42	MIP	53.0	8317
11	12879	12875	98.9	5691.6	43.8	MIP	142.7	12879	12879	**12876**	40.8	56.3	9.77	MIP	50.6	12879
16	25376	25376	31.2			Root	31.2	25376	25376	25376	7.8			Root	7.8	25376
36	10796	10796	35.1			Root	35.1	10796	10796	10796	4.7			Root	4.7	10796
41	21806	21785	72.7	232.4	68.1	MIP	140.8	21806	21806	**21786**	65.4	147.5	21.58	MIP	87.0	21806
46	44455	**44455**	40.5			Root	40.5	44455	44455	44453	8.4	0.9	0.05	BCP	9.3	44455
56	570	**541**	67.8	>86400	18126.0	MIP	18193.8	570	570	540	13.8	>14399.6	449.66	MIP	463.5	570
61	7898	7857	113.7	>86400	27997.4	MIP	28111.1	7898	7898	7857	30.9	7402.9	840.28	MIP	871.2	7898
71	42645	**42628**	218.8	336.6	42.3	MIP	261.1	42645	42645	42627	25.5	84.7	17.08	MIP	42.6	42645
81	495	495	118.9			Root	118.9	495	495	495	24.8			Root	24.8	495
86	8369	8336	241.3	575.2	355.6	MIP	596.9	8369	8369	8336	65.7	49.9	4.82	MIP	70.5	8369
91	26552	26548	187.6			Root	187.6	26551	**26551**	**26551**	10.7			Root	10.7	26551
96	50326	50317	183.2	261.1	32.4	MIP	215.6	50326	50326	**50324**	58.2	58.0	10.37	MIP	68.6	50326
111	10069	10051	134.6	52.0	43.8	MIP	178.4	10069	10069	10051	32.1	31.2	4.85	MIP	36.9	10069
116	11552	11523	114.3	45693.5	4087.2	MIP	4201.5	11552	11552	11523	24.8	6373.1	325.91	MIP	350.7	11552
121	23792	23775	85.3	438.5	248.5	MIP	333.8	23792	23792	23775	35.5	105.4	6.72	MIP	42.2	23792

Table C5: Detailed Results for $m = 2$ and $n = 100$ instances

	BCP-PMWT								BCP-PMWT-OTI							
	Root			BCP	CPLEX	Overall			Root			BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
1	3339	3322	1606.0	>14400			>14400	$\leq$3339	3339	**3339**	1006.5			Root	1006.5	3339
6	30665	30665	1001.5			Root	1001.5	30665	30665	30665	379.1			Root	379.1	30665
11	93894	93894	613.0			Root	613.0	93894	93894	93894	33.8			Root	33.8	93894
16	209100	209100	5603.6			Root	5603.6	209100	209100	209100	98.3			Root	98.3	209100
21	457836	457836	1248.7			Root	1248.7	457836	457836	457836	425.4			Root	425.4	457836
31	12729	12729	927.7			Root	927.7	12729	12729	12729	135.8			Root	135.8	12729
36	56671	56590	1435.7	>14400			>14400	$\leq$56671	56671	**56620**	1212.1	>13242.7	22506.30	MIP	23718.4	56671
41	237964	237841	20436.3	>20436	120721.7	MIP	141158.0	237964	237964	**237881**	3462.3	42735.2	32047.52	MIP	35509.8	237964

Table C5: continued

	BCP-PMWT								BCP-PMWT-OTI							
		Root		BCP	CPLEX	Overall				Root		BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
46	422831	422830	2077.9	44.1	51.9	BCP	2122.0	422831	422831	**422831**	303.2			Root	303.2	422831
56	5047	5047	1582.3			Root	1582.3	5047	5047	5047	271.9			Root	271.9	5047
61	45573	45481	7353.8	>14400	>86400		>14400	≤45573	45573	**45487**	1896.7	>12559.0	≥172800		1896.7	≤45573
66	126522	126477	7526.5	23425.6	>86400	BCP	30952.1	126512	**126513**	**126493**	6123.1	5134.1	5413.94	BCP	11257.2	126513
71	327305	327305	630.2			Root	630.2	327305	327305	327305	55.6			Root	55.6	327305
81	908	830	1299.2	>14400			>14400	≤908	908	**908**	451.3			Root	451.3	908
86	36581	36322	3905.2	>14400	>86400		>14400	≤36581	36581	**36402**	2192.6	>13291.9	≥172800		2192.6	≤36581
91	129931	129752	4259.4	>14400	>86400		>14400	≤129931	**129929**	**129798**	8362.5	> 6221.1	≥172800		8362.5	≤129929
96	254194	254194	1030.5			Root	1030.5	254194	254194	254194	274.6			Root	274.6	254194
111	84274	84097	7424.3	>14400	>86400		>14400	≤84250	**84220**	**84139**	5217.1	> 9778.6	≥172800		5217.1	≤84220
116	191198	191173	11740.2	>14400	32567.7	MIP	44307.9	191186	**191186**	**191185**	4645.1	2601.7	1540.90	MIP	6186.0	191186
121	242022	241997	57559.6	>86400	4072.4	MIP	61632.0	242018	**242018**	**241999**	31429.9	> 0.3	1398.56	MIP	32828.5	242018

Table C6: Detailed Results for $m = 4$ and $n = 100$ instances

	BCP-PMWT								BCP-PMWT-OTI							
		Root		BCP	CPLEX	Overall				Root		BCP	CPLEX	Overall		
Inst	UB	LB	Time	Time	Time	Best	Time	Opt	UB	LB	Time	Time	Time	Best	Time	Opt
1	2001	2001	1337.3			Root	1337.3	2001	2001	2001	145.7			Root	145.7	2001
11	50236	50203	1439.4	>14400			>14400	≤50236	**50232**	50203	291.4	>14187.3	3571.20	MIP	3862.6	50232
16	110222	110118	1449.2	>14400	>86400		>14400	≤110222	**110219**	**110120**	326.4	>14094.4	≥172800		326.4	≤110219
21	237392	237390	1493.8	5.5	29.0	BCP	1499.3	237392	237392	**237391**	378.4	10.0	0.55	MIP	378.9	237392
31	7130	7082	1901.5	>14400			>14400	≤7130	7130	7082	301.6	>14392.5	≥3600		301.6	≤7130
36	30791	**30783**	2201.3	>14400			>14400	≤30791	30791	30782	347.4	>14075.4	190.75	MIP	538.1	30791
41	126193	126144	1493.6	11041.3	41682.9	BCP	12534.9	126185	**126185**	**126147**	1186.3	5643.1	435.71	MIP	1622.0	126185
46	219537	219529	1372.0	256.3	800.9	BCP	1628.3	219536	**219536**	219529	592.3	394.2	25.27	MIP	617.6	219536
56	3076	3030	1727.3	>14400			>14400	≤3076	3076	3030	590.9	>14636.7	≥3600		590.9	≤3076
61	24868	24821	1682.5	>14400	>86400		>14400	≤24868	**24856**	**24825**	726.4	>13697.5	751.04	MIP	1477.4	24856
66	67979	67954	1267.7	2885.7	3738.2	BCP	4153.4	67967	**67970**	**67955**	445.9	879.0	232.73	MIP	678.6	67969
71	170699	170675	862.9	>14400	81028.1	MIP	81891.0	170689	**170691**	170675	418.2	>85994.6	1299.60	MIP	1717.8	170690
81	819	772	1594.4	>14400			>14400	≤819	819	**797**	425.7	1017.6	1233.73	BCP	1443.3	819
86	21299	21218	1379.0	>14400	>86400		>14400	≤21299	**21286**	**21220**	519.4	>13913.1	11857.18	MIP	12376.6	21282
91	70612	**70589**	2119.4	1436.3	7318.0	BCP	3555.7	70606	**70608**	70587	605.1	2157.4	210.80	MIP	815.9	70606
96	133591	133575	878.6	3156.3	1930.3	MIP	2808.9	133587	**133587**	133575	321.4	420.9	58.09	MIP	379.5	133587
111	46763	46630	1704.6	>14400	>86400		>14400	≤46747	**46719**	**46633**	382.6	>14090.1	57566.09	MIP	57948.7	46704
116	101563	101520	1374.6	>14400	156440.1	MIP	157814.7	101546	**101551**	**101521**	512.8	65651.5	2612.53	MIP	3125.3	101546
121	127639	127597	1389.6	>14400	108063.9	MIP	109453.5	127618	**127619**	**127598**	633.2	23821.9	2990.62	MIP	3623.8	127619