Precedence-Constrained Arborescences

Xiaochen Chou    Mauro Dell’Amico    Jafar Jamal
Roberto Montemanni*

University of Modena and Reggio Emilia, Italy

Abstract

The minimum-cost arborescence problem is a well-studied problem in the area of graph theory, with known polynomial-time algorithms for solving it. Previous literature introduced new variations on the original problem with different objective function and/or constraints. Recently, the Precedence-Constrained Minimum-Cost Arborescence problem was proposed, in which precedence constraints are enforced on pairs of vertices. These constraints prevent the formation of directed paths that violate precedence relationships along the tree. We show that this problem is NP-hard, and we introduce a new scalable mixed integer linear programming model for it. With respect to the previous models, the newly proposed model performs substantially better. This work also introduces a new variation on the minimum-cost arborescence problem with precedence constraints. We show that this new variation is also NP-hard, and we propose several mixed integer linear programming models for formulating the problem.

1 Introduction

The Minimum-Cost Arborescence problem (MCA) is a well-known problem that consists in finding a directed minimum-cost spanning tree rooted at some vertex \( r \) called the root in a directed graph. The first polynomial time algorithm for solving the problem was proposed independently by Yoeng-Jin Chu and Tseng-Hong Liu [8], and Jack Edmonds [11]. The problem can be formally described as follows. A directed graph \( G = (V, A) \) is given where \( V = \{1, \cdots , n\} \) is the set of vertices, \( r \in V \) is the root of the arborescence, and \( A \subseteq V \times V \) is the set of arcs with a cost \( c_a \) associated with every arc \( a \in A \). The goal is to find a minimum-cost directed spanning tree in \( G \) rooted at \( r \), i.e. a set \( T \subseteq A \) of \( n - 1 \) arcs, such that there is a unique directed path from \( r \) to any other vertex \( j \in V \setminus \{r\} \) in the subgraph induced by \( T \). A different polynomial time algorithm for solving the MCA that operates directly on the cost matrix was discussed by Bock [4].

*Corresponding author: roberto.montemanni@unimore.it
Since the MCA was first proposed, different variations have been introduced such as the **Resource-Constrained Minimum-Weight Arborescence problem** [14], where finite resources are associated with each vertex in the input graph. The objective of the problem is to find an arborescence with minimum total cost where the sum of the costs of outgoing arcs from each vertex is at most equal to the resource of that vertex. The problem is categorized as NP-hard as it generalizes the Knapsack problem [15]. The **Capacitated Minimum Spanning Tree problem** [19] is another variation, where non-negative integer node demands \( q_j \) is associated with each node \( j \in V \setminus \{ r \} \), and an integer \( Q \) is given. The objective is to find a minimum spanning tree rooted at \( r \) such that the sum of the weights of the vertices in any subtree off the root is at most \( Q \). The problem is shown to be NP-hard as the particular case with zero cost arcs is a bin packing problem [19]. The **p-Arborescence Star problem** [30] is a relevant problem that is described as follows. Given a weighted directed graph \( G = (V, A) \), a root vertex \( r \in V \), and an integer \( p \), the objective of the problem is to find a minimum-cost reverse arborescence rooted at \( r \), such that the arborescence spans the set of vertices \( H \subseteq V \setminus \{ r \} \) of size \( p \), and each vertex \( v \in V \setminus \{ H \cup r \} \) must be assigned to one of the vertices in \( H \). The problem is NP-hard [29] in the general case by a reduction from the p-median problem [20]. Frieze and Tkocz [16] study the problem of finding a minimum-cost arborescence such that the cost of the arborescence is at most \( c_0 \). The problem is studied on randomly weighted digraphs where each arc in the graph has a weight \( w \) and a cost \( c \), each being an independent uniform random variable \( U^* \) where \( 0 < s \leq 1 \), and \( U \) is uniform \([0, 1]\). The problem is NP-hard [16] through a reduction from the knapsack problem. Another problem is the **Maximum Colorful Arborescence problem** [14] which can be described as the following. Given a weighted directed acyclic graph with each vertex having a specified color from a set of colors \( C \), the objective is to find an arborescence of maximum weight, in which no color appears more than once. The problem is known to be NP-hard [5] even when all arcs have a weight of 1. The **Constrained Arborescence Augmentation problem** [25] is a different variation that can be described as follows. Given a weighted directed graph \( G = (V, A) \), and an arborescence \( T = (V, A_r) \) in \( G \) rooted at vertex \( r \in V \), the objective of the problem is to find an arc subset \( A' \) from \( A - A_r \) such that there still exists an arborescence in the new graph \( G' = (V, A_r \cup A' - a) \) for each arc \( a \in A_r \), where the sum of the weights of the arcs in \( A' \) is minimized. The problem is an extension on the augmentation problem [13], and is shown to be NP-hard [25]. The **Minimum k Arborescence with Bandwidth Constraints** [6] is another variation, where every arc \( a \in A \) has an integer bandwidth \( b(a) \) that indicates the number of times such an arc can be used. The objective of the problem is to find \( k \) arborescences of minimum-cost rooted at the \( k \) given root vertices, covering every arc \( a \in A \) at most \( b(a) \) times. It has been shown that the problem can be solved in polynomial time [6]. The **Degree-Constrained Minimal Spanning Tree problem with unreliable links and node outage costs** [23] is modeled as a directed graph with the root vertex being the central node of the network, and all other vertices being terminal nodes. The problem consists in finding links in a network to connect a set of terminal nodes to a central
node, while minimizing both link costs and node outage costs. Node outage cost is the economic cost incurred by the network user whenever that node is disabled due to failure of a link. The problem is shown to be NP-hard by reducing the problem to an equivalent Traveling Salesman problem [18]. The Minimum Changeover Cost Arborescence [17] is another variation, where each arc is labeled with a color out of a set of \( k \) available colors. A changeover cost is defined on every vertex \( v \) in the arborescence other than the root. The cost over a vertex \( v \) is paid for each outgoing arc from \( v \) and depends on the color of its outgoing arcs, relative to the color of its incoming arc. The costs are given through a \( k \times k \) matrix \( C \), where each entry \( C_{ab} \) specifies the cost to be paid at vertex \( v \) when its incoming arc is colored \( a \) and one of its outgoing arcs is colored \( b \). A changeover cost at vertex \( v \) is calculated as the sum of costs paid for every outgoing arc at that vertex. The objective of the problem is to find an arborescence \( T \) with minimum total changeover cost for every vertex \( j \in V \) other than the root. The problem is shown to be NP-hard and very hard to approximate [17]. Finding a pair of arc-disjoint in-arborescence and out-arborescence is another problem, with the objective of finding a pair of arc-disjoint \( r \)-out-arborescence rooted at \( r_1 \) and \( r \)-in-arborescence rooted at \( r_2 \) where \( r_1, r_2 \in V \). An \( r \)-out-arborescence has all its arcs directed away from the root, and an \( r \)-in-arborescence has all its arcs directed towards the root. The problem was studied by Bérczi et al. [3] where a linear-time algorithm for solving the problem in directed acyclic graphs is proposed. The problem is shown to be NP-Complete in general graphs even if \( r_1 = r_2 \) [2]. Yingshu et al. [26] studied the problem of constructing a strongly connected broadcast arborescence with bounded transmission delay, where they devise a polynomial time algorithm for constructing a broadcast network with minimum energy consumption that respects the transmission delays of the broadcast tree simultaneously. The Minimum Spanning Tree Problem with Conflict Pairs is a variation of the minimum spanning tree problem where given an undirected graph and a set \( S \) that contains conflicting pairs of edges called a conflict pair, the objective of the problem is to find a minimum-cost spanning tree that contains at most one edge from each conflict pair in \( S \) [7]. The problem is shown to be NP-hard [9]. The Least-Dependency Constrained Spanning Tree problem [34] is another variation that can be defined as follows. Given a connected graph \( G = (V, E) \) and a directed graph \( D = (E, A) \) whose vertices are the edges of \( G \), the directed graph \( D \) is a dependency graph for \( E \), and \( e_1 \in E \) is a dependency of \( e_2 \in E \) if \( (e_1, e_2) \in A \). The objective of the problem is to decide whether there is a spanning tree \( T \) of \( G \) such that each edge in \( T \) has either an empty dependency or at least one of its dependencies is also in \( T \). The All-Dependency Constrained Spanning Tree problem [34] is a similar problem that consists in deciding whether there is a spanning tree \( T \) of \( G \) such that each of its edges either has no dependency or all of its dependencies are in \( T \). The two problems are shown to be NP-Complete [34].

The Precedence-Constrained Minimum-Cost Arborescence problem (PCMCA) was first introduced by Dell’Amico et al. [10], where a set of precedence constraints is included as follows. Given a set \( R \) of ordered pairs of ver-
tices, then for each precedence \((s, t) \in R\) any path of the arborescence covering both vertices \(s\) and \(t\) must visit \(s\) before visiting \(t\). The objective of the problem is to find an arborescence of minimum total cost that satisfies the precedence constraints. By definition of the PCMCA, we always assume that if \((s, t) \in R\) then \((t, s) \notin A\). The PCMCA has applications in infrastructure design such as designing a commodity distribution network. As an example, assume we have a commodity distribution network, where the distribution starts from a main vertex (root of the arborescence), and the distribution travels in a single direction away from the root to every other vertex in the graph. Such a structure follows the definition of an arborescence. Now assume that transit duties that are higher than the travel costs have to be paid by vertex \(s\) in the graph, if the commodity passes through vertex \(t\) on its way to vertex \(s\). To avoid for such duties to be paid by vertex \(s\), we can impose a precedence relationship between the vertex pair \(s\) and \(t\), i.e. \((s, t) \in R\). This will guarantee that no directed path from \(t\) to \(s\) will appear in the distribution network, and vertex \(s\) can avoid paying the transit duties (see [10] for more details).

A new variation on the MCA named the Precedence-Constrained Minimum-Cost Arborescence problem with Waiting Times (PCMCA-WT) is introduced in this work. The problem is an extension on the PCMCA characterized by an additional constraint. Given a spanning arborescence rooted at vertex \(r\), with arc costs indicating the time required to traverse an arc, assume there is a flow which starts at the root vertex \(r\) and traverses each path of the arborescence. For each precedence \((s, t) \in R\), we must guarantee that the time at which the flow enters \(s\) is smaller than or equal to the time at which the flow enters \(t\). As an example, assume that \((b, a), (c, a), (d, a) \in R\), and the flow enters vertex \(b\) at time step 5, vertex \(c\) at time step 10, and vertex \(d\) at time step 15. Therefore, the flow must enter vertex \(a\) at a time step greater than or equal to 15, and if the cost of the path from \(r\) to \(a\) is equal to 10, then this will result in a waiting time of 5 at vertex \(a\). The objective of the problem is to find an arborescence \(T\) that has a minimum total cost, plus total waiting times, where the flow never enters \(t\) earlier than entering \(s\) for all \((s, t) \in R\).

The contributions of this paper can be summarized as:

1. Introducing a scalable and efficient integer linear programming model for the PCMCA.
2. Introducing the PCMCA-WT as a new variation of the MCA.
3. Proving that both the PCMCA and the PCMCA-WT are NP-hard.

The rest of the paper is organized as follows. Section 2 presents a proof of complexity and a new mixed integer linear programming model (MILP) for the PCMCA. Section 3 presents a proof of complexity and several mixed integer linear programming models for the PCMCA-WT. Section 4 discusses computational results, while some conclusions are summarized in Section 5.
2 The Precedence-Constrained Minimum-Cost Arborescence Problem

The Precedence-Constrained Minimum-Cost Arborescence problem can be formally described as follows. Let $G = (V, A)$ be a directed graph, $r \in V$, and $P = (V, R)$ be a precedence graph. Let $c_{ij}$ be a cost associated with every arc $(i, j) \in A$. An arc $(s, t) \in R$ is a precedence relationship between the two vertices $s, t \in V$. The objective of the problem is to find a minimum-cost arborescence $T$ rooted at vertex $r$ such that, for each $(s, t) \in R$, $t$ must not belong to the unique path in $T$ that connects $r$ to $s$. For simplicity, we always assume that for the root $r \in V$, $(s, r) \notin A$ for all $s \in V \setminus \{r\}$, as by definition none of these arcs would be part of an arborescence rooted at $r$, and $(s, r) \notin R$ for all $s \in V \setminus \{r\}$ as the problem would be infeasible otherwise.

Figure 1 presents an example that shows the difference between the classic MCA and the PCMCA. The example instance graph with its respective arc costs is shown in the figure on the left, with the precedence relationship $(3, 1) \in R$ highlighted in red. The figure in the middle shows a feasible MCA solution with a cost of 3. The MCA solution is infeasible for the PCMCA since $(3, 1) \in R$, and vertex 1 belongs to the directed path connecting $r$ to vertex 3. To make the solution feasible for the PCMCA, vertex 1 must succeed vertex 3 on the same directed path, or the two vertices must reside on two disjoint paths. A feasible
solution with a cost of 4 is shown in the figure on the right.

2.1 Computational Complexity

Some of the Minimum-Cost Arborescence variations mentioned in Section 1 belong to the \( \text{NP} \)-hard complexity class. In this section we show that the Precedence-Constrained Minimum-Cost Arborescence Problem is also \( \text{NP} \)-hard. The proof is inspired by the one introduced in [24] for the Path Avoiding Forbidden Pairs problem.

**Theorem 1.** The PCMCA is \( \text{NP} \)-hard.

**Proof.** By reduction from 3-SAT: Let \( X = \{x_1, x_2, \ldots, x_t\} \) be a set of variables. Let \( \Phi = C_1 \land C_2 \land \cdots \land C_m \) be a boolean expression in 3-conjunctive normal form, such that each clause \( C_i \), \( i = 1, \ldots, m \), is denoted by \( (v_{i1} \lor v_{i2} \lor v_{i3}) \), where each literal \( v_{ik}, 1 \leq k \leq 3 \), is associated to one variable in \( X \) or its negation. We will construct a graph \( G \) and a set of precedence constraints \( R \) such that there exists a feasible solution of the PCMCA problem in \( G \) if and only if \( \Phi \) is satisfiable.

Let \( G = (V, A) \) where \( V = \{r\} \cup \{s\} \cup \{s'\} \cup \{t\} \cup C \), with \( C, A \) and \( R \)
defined as follows.

\[
C = \{ v_{ik} : 1 \leq i \leq m, 1 \leq k \leq 3 \}
\]

\[
A = \{ (r,s), (r,s') \} \cup \{ (s,v_{ij}), 1 \leq j \leq 3 \} \cup \{ (v_{mj},t), 1 \leq j \leq 3 \}
\]

\[
R = \{ (t,s') \} \cup \{ (v_{hk}, v_{ij}) : h > i, v_{hk} \text{ and } v_{ij} \text{ refer to the same variable, but exactly one of the two literals is negated} \}
\]

Note that \( C \) contains \( 3m \) vertices, one for each literal of each clause \( C_i \), with all arcs having an equal positive cost. The three sets \( C \), \( A \), and \( R \) induce the graph shown in Figure 2. The set of precedence constraints, besides \((t,s')\), is between two vertices that refer to the same literal, but exactly one of the two literals is negated. If a feasible solution \( T \) of the PCMCA problem can be found in \( G \), this implies that:

1. no path from \( s' \) to \( t \) exists in \( T \)
2. in any (rooted) path there is no pair of vertices corresponding to a variable and its negation
3. there is a unique path \( P \) from \( r \) to \( t \) which passes through \( s \) and through a vertex of each clause

The formula can be satisfied by assigning true values to all the literals corresponding to the vertices in \( P \cap C \), and assigning false values to all the variables not associated with these literals. This satisfies all the clauses.

Conversely, if the formula is satisfied then each clause has at least one literal with true value, and no variable is assigned to both true and false (in different clauses). We construct a PCMCA feasible solution as follows. We start by building a path \( P \) from \( r \) to \( t \) which includes \( s \) and exactly one vertex from each clause, corresponding to a literal with true value. We complete the arborescence by adding \((r,s')\) and \((s',v)\) for each \( v \notin P \).

\[\square\]

### 2.2 A Set-Based Model

The MILP previously proposed in [10] suffers from scalability issues because of the cubic number of variables (relative to the number of vertices and the precedence relationships) used to model the precedence relationships between vertex pairs. A new integer linear program for the PCMCA is introduced in this section.

We extend the classic connectivity constraints for the MCA [22] in such a way to take precedences into account. When considering a set \( S \subseteq V \setminus \{r\} \) we add a constraint for each \( j \in S \), and we force that at least one active arc must enter \( S \) coming from the set of vertices allowed to precede \( j \) on the path connecting \( j \) to \( r \).

Let \( x_{ij} \) be a variable associated with every arc \((i,j) \in A\) such that \( x_{ij} = 1 \) if \((i,j) \in T\) and 0 otherwise, where \( T \) is the resulting optimal arborescence.
Let $V_j = \{i \in V : (j, i) \notin R\}$ be the set of vertices that can precede $j$ on a directed path from the root without introducing precedence violations or, in turn, a violating path, which is a directed path that violates some precedence relationship in $R$.

\[
\begin{align*}
\text{minimize} & \quad \sum_{(i,j) \in A} c_{ij} x_{ij} \quad (1) \\
\text{subject to} & \quad \sum_{(i,j) \in A} x_{ij} = 1 \quad \forall j \in V \setminus \{r\} \quad (2) \\
& \quad \sum_{(i,k) \in A: \quad i \in V_j \setminus S, k \in S} x_{ik} \geq 1 \forall j \in V \setminus \{r\}, \forall S \subseteq V_j \setminus \{r\} : j \in S \quad (3) \\
& \quad x_{ij} \in \{0, 1\} \quad \forall (i,j) \in A \quad (4)
\end{align*}
\]

Constraints (2) implies the first property of an arborescence namely that every vertex $v \in V \setminus \{r\}$ must have a single parent. Constraints (3) model the connectivity constraints, that is every vertex $j \in V \setminus \{r\}$ must be reachable from the root. Note that $V_j \setminus S$ contains at least $r$, while $S$ contains at least $j$. The set of constraints (3) reduces to the classical connectivity constraints for the MCA which are $\sum_{(i,k) \in A: \quad i \notin S, k \in S} x_{ik} \geq 1 \forall S \subseteq V \setminus \{r\}$ when the set $R$ of precedence relationships is empty. This is because when $R$ is an empty set, $V_j = V$ for all $j \in V \setminus \{r\}$. Constraints (3) also impose the precedence relationships. Inequality (3) implies that the resulting arborescence will not include vertex $t$ in the directed path connecting $r$ to $s$ when $(s, t) \in R$. Note that this is the same inequality named weak $\sigma$-inequality considered by Ascheuer, Jünger & Reinelt [1] for the Sequential Ordering Problem. Finally, constraints (4) define the domain of the variables. The MILP model proposed has $O(|A|)$ variables, and $O(|A|)$ constraints, plus an exponential number of connectivity constraint (3). Although the number of constraints of the Set-Based Model is exponential, it is more efficient at solving the problem than the model introduced in [10]. This is because in practice the number of constraints that are dynamically added to the model is small. Moreover, the model uses a smaller number of variables. An experimental validation for these considerations will be provided by the experiments in Section 4.

One approach to solve a linear relaxation model (LR-model) that has an exponential number of constraints is to start by solving the LR-model without including a large set of constraints (such as (3)), then iteratively adding a constraint once it is violated, then solving the new LR-model again. A procedure for finding a violated constraint is called a separation procedure. The optimal solution of the LR-model is found as soon as there are no violated constraints. When solving a MILP, the optimality gap needs to be closed to find the optimal solution even if no violated inequality is found.

In the literature, the large set of constraints that are necessary to model the problem, but are added dynamically to the model only when they are violated,
are known as *lazy constraints*. Note that using this approach, the separation procedure must also be used to check the feasibility of integer solutions found by the linear relaxation or primal heuristics.

**Algorithm 1** Separation Procedure for Inequalities (3)

1: \( G = (V, A) \) is a directed graph, \( \bar{x} \) is a fractional LP-solution
2: procedure \textsc{FindViolatedInequality}(G, \bar{x})
3: for \( j \in V \setminus \{r\} \) do
4: Construct a directed graph \( D_j = (V_j, A') \) such that:
5: \( V_j = \{ i \in V : (j, i) \notin R \} \)
6: \( A' = \{ (i, k) : i, k \in V_j \} \)
7: \( c_{ik} = \bar{x}_{ik} \quad \forall (i, k) \in A' \)
8: Calculate a minimum \((r, j)\)-cut \( C \) in \( D_j \)
9: if the cost of \( C < 1 \) then
10: return the violated inequality \( \sum_{(i,k) \in C} \bar{x}_{ik} \geq 1 \)
11: end if
12: end for
13: end procedure

Algorithm 1 describes the separation procedure for inequalities (3). Let \( \bar{x} \) be a solution of the linear relaxation or a candidate primal solution. An inequality (3) that is violated by the solution \( \bar{x} \) can be detected by computing a minimum \((r, j)\)-cut \( C \) in a directed graph \( D_j = (V_j, A') \), where \( A' \) is equal to the set of arcs \( A \) minus the arcs incident to the immediate successors of \( j \) in the precedence graph. The cost \( c_{ik} \) of an arc \( (i, k) \in A' \) is equal to \( \bar{x}_{ik} \). The value of the minimum \((r, j)\)-cut \( C \) in \( D_j \) can tell us the following about the given fractional solution:

1. If the cost of a minimum cut is equal to 0, then vertex \( j \) is not reachable from \( r \) in \( D_j \). In this case, the solution does not contain a path from \( r \) to \( j \), or contains a single or multiple paths from \( r \) to \( j \), all of which pass through a successor of \( j \).

2. If the cost of a minimum cut is in the range \((0, 1)\), then vertex \( j \) is reachable from \( r \) in \( D_j \). In this case, the solution contains multiple paths from \( r \) to \( j \), and at least one of them passes through a successor of \( j \).

3. If the cost of a minimum cut is equal to 1, then vertex \( j \) is reachable from \( r \) through a single or multiple paths in \( D_j \), although possibly some of them pass through a successor of \( j \).

In the first two cases, the minimum cut \( C \) defines an inequality (3) violated by \( \bar{x} \), however in the last case a violated inequality (3) does not exist even if the fractional solution \( \bar{x} \) contains a violating path. Therefore, although inequalities (3) are valid inequalities for that PCMCA, there are fractional LP-solutions that contain violating paths, but satisfy inequalities (3).
Figure 3: An example of a fractional solution of the Set-Based model that contains a violating path, and does not violate an inequality (3). Every arc cost is associated with the value of its respective variable $x_{ij}$. For this solution we have the violated precedence $(s, t) \in R$. The figure on the left shows the solution, and the figure on the right shows the graph $D_j$ which has a minimum $(r, s)$-cut of value 1 indicated by the red dashed line.

Figure 3 shows an example on how the separation procedure works. The figure also shows a fractional solution of the Set-Based model that contains a violating path, but does not violate any inequality (3). Another example, showing how the Path-Based model introduced in [10] fails to detect a violating path in a fractional solution, is presented in [A].

A drawback of the Set-Based model is the high computational complexity of the separation procedure of inequalities (3), which has a complexity of $O(n^4)$, assuming it uses an $O(n^3)$ algorithm for computing a minimum $(s, t)$-cut in $D_j$ [21].

3 The Precedence-Constrained Minimum-Cost Arborescence Problem with Waiting Times

In the Precedence-Constrained Minimum-Cost Arborescence Problem with Waiting Times (PCMCA-WT) a flow starts from the root $r$ at time 0, and traverses each path of the arborescence. The cost $c_{ij}$ of an arc $(i, j) \in A$ represents the time required to traverse that arc. Let $d_j$ be the time at which the flow enters vertex $j \in V$. For any $(s, t) \in R$, $d_t \geq d_s$, which means that the flow must enter vertex $t$ at the same time step or after entering vertex $s$. Let $w_j$ be the waiting time before the flow enters vertex $j$ required to respect the aforementioned constraint. The objective is to find an arborescence $T$ that has a minimum total cost plus total waiting time, where the flow never enters
Figure 4: Comparing an instance solved as a PCMCA, and solved as a PCMCA-WT. The graph on the left shows the instance with its respective arc costs, and the precedence relationship \((2, 3) \in R\) highlighted in red. The graph in the middle shows the optimal PCMCA solution of cost 3, and the graph on the right shows the optimal PCMCA-WT solution of cost 4. The PCMCA solution is not a feasible PCMCA-WT solution since the flow enters vertex 3 before entering vertex 2 and \((2, 3) \in R\).

t earlier than entering \(s\) for all \((s, t) \in R\). For simplicity, we always assume that for the root \(r \in V\), \((i, r) \notin A\) for all \(i \in V \setminus \{r\}\), as by definition none of these arcs would be part of an arborescence rooted at \(r\), and \((s, r) \notin R\) for all \(s \in V \setminus \{r\}\), as the problem would be infeasible otherwise.

Figure 4 presents an example that shows the difference between the PCMCA and the PCMCA-WT. Next to each vertex we have its corresponding \(d_t\) value. In this example we have the precedence relationship \((2, 3)\) highlighted in red. The two solutions depicted are valid solutions for the PCMCA, since they both satisfy the precedence constraints, that is \(t\) never precede \(s\) on the same directed path for all \((s, t) \in R\). The solution in the middle shows the optimal PCMCA solution with a total cost of 3 (sum of all the arcs). We can see that the solution in the middle is not a feasible PCMCA-WT solution since \((2, 3) \in R\) but \(d_3 < d_2\). The solution on the right shows an optimal PCMCA-WT solution with a cost of 4 (sum of all the arcs plus waiting time at each vertex). The solution results in a waiting time of 1 at vertex 3, since the time from \(r\) to 2 is 2, and the time from \(r\) to 3 is 1.

### 3.1 Computational Complexity

In this section we show that the PCMCA-WT is NP-hard.

The *Rectilinear Steiner Arborescence* (RSA) *Problem* [32] is an NP-hard problem formally defined as follows. Let \(P = \{p_1, p_2, \ldots, p_n\}\) be a set of points in the first quadrant of the Cartesian plane, where \(p_i = (x_i, y_i)\) with \(x_i, y_i \geq 0\), and \(p_1 = (0, 0)\). A complete grid can be created, where the points in \(P\) are on the intersections of vertical and horizontal lines. A set \(S\) of Steiner vertices can be added, corresponding to the \(O(|P|^2)\) intersection points not overlapping with the
Figure 5: The figure shows an RSA instance with 5 points and 10 Steiner vertices, while the dashed lines represent the arcs of the instance.

points in $P$. The arcs of the problem are the right-directed horizontal segments and the up-directed vertical segments between two adjacent points of the grid $P \cup S$. The cost associated with each arc $(p_i, p_j)$ is defined as $|x_i - x_j| + |y_i - y_j|$. Figure 5 shows an example of an RSA instance with 5 points, and the relative Steiner vertices.

Given a positive value $k$, the decision version of the RSA problem consists in deciding whether there is an arborescence with total length not greater than $k$ such that the arborescence is rooted at $p_1$ and it contains a unique path from $p_1$ to $p_i$ for all $i \in \{1, 2, \ldots, n\}$. Note that the length of each path from $p_1$ to $p_i$ is $x_i + y_i$ by construction.

**Theorem 2.** The PCMCA-WT is NP-hard.

**Proof.** By a reduction from the decision version of the RSA problem: we construct a graph $G = (V, A)$ and a set $R$ of precedence constraints such that there exist a PCMCA-WT solution of cost at most $k$ if and only if a RSA of cost at most $k$ exists. Given an instance of the RSA problem with a set of points $P$ and a set of Steiner points $S$, consider the PCMCA-WT instance defined as follows:

- $V = P \cup S$
- $A' = \{(i, j) : j$ is immediately on the top of $i$ in the grid, or $j$ is immediately on the right of $i$ in the grid\}$
- $A = A' \cup \{(P_{\text{FAR}}, s_i), s_i \in S\}$, with $P_{\text{FAR}} \in \arg\max_{p_i \in P} \{x_i + y_i\}$
- $R = \{(p, P_{\text{FAR}}) : p \in P \setminus \{P_{\text{FAR}}\}\}$
- $c_{ij} = (x_j - x_i) + (y_j - y_i)$ for $(i, j) \in A'$
- $c_{P_{\text{FAR}}, s_i} = 0$ for $s_i \in S$

If the instance of RSA has a solution of cost $k$, then a solution of cost $k$ for the instance of PCMCA-WT can be obtained. Starting from the solution of the
Figure 6: The PCMCA-WT instance associated with the RSA instance depicted in Figure 5. A RSA solution of minimum cost is given by the blue arcs. The red arcs have cost 0 and, together with the blue ones, form an optimal PCMCA-WT solution.

RSA problem, it is possible to complete the solution of the associated PCMCA-WT problem by adding 0-cost arcs (red arcs) to connect the node \(P_{FAR}\) to the Steiner nodes not used in the RSA solution. The solution of an RSA instance and a solution of the associated PCMCA-WT problem are depicted in Figure 6.

Conversely, assume that there is a feasible solution of PCMCA-WT with cost at most \(k\). Without loss of generality suppose that such a solution is optimal. Note that a path starting at \(P_{FAR}\) and passing through a vertex in \(P\) cannot exist due to the precedence constraints. Besides, every leaf of the arborescence that is in \(S\) must have \(P_{FAR}\) as parent; otherwise, making \(P_{FAR}\) its parent would reduce the cost. Therefore, removing all the leaves of the PCMCA-WT arborescence connected through \(P_{FAR}\) results in a tree that uses only arcs in \(A'\) and whose leaves are all in \(P\). It follows that the resulting tree is a feasible solution for the RSA.

\[\square\]

3.2 MILP Models

This section introduces three different MILP models for formulating the Precedence-Constrained Minimum-Cost Arborescence Problem with Waiting Times. For all the models, let \(d_j\) be a variable associated with every vertex \(j \in V\) to represent the time at which the flow enters vertex \(j\), with \(d_r = 0\). The value of \(d_j\) is bounded from below by summing the time from \(r\) to the parent \(i\) of \(j\) and the cost of the arc \((i, j) \in A\) that is part of the arborescence. To ensure
that the resulting arborescence satisfies the precedence constraints, we enforce that the time from \( r \) to \( t \) is greater than or equal to the time from \( r \) to \( s \) for all \((s, t) \in R\). A variable \( x_{ij} \) is associated with every arc \((i, j) \in A\) such that \( x_{ij} = 1 \) if \((i, j) \in T\) and 0 otherwise, where \( T \) is the resulting optimal arborescence.

In all the models proposed for the PCMCA-WT, the value of \( M \), which is an upper bound on the value of an optimal solution, is equal to the solution cost of solving the instance as a Sequential Ordering Problem (SOP) \cite{28} using a nearest neighbor algorithm \cite{33}. This is a valid upper bound on the solution for the PCMCA-WT, since a valid solution for the SOP consists of a simple directed path that includes all the vertices of the graph such that \( t \) never precede \( s \) for all \((s, t) \in R\), which implies that \( d_t \geq d_s \) for all \((s, t) \in R\), and the waiting time on each vertex is equal to zero.

### 3.2.1 A Multi-Commodity Flow Model

The model introduced in this section extends the one introduced in \cite{10} for the PCMCA, and formulates the sub-problem of finding an arborescence rooted at \( r \) that does not violate precedence relationships in \( R \) as a multi-commodity flow problem. The model uses a polynomial set of constraints instead of inequalities \cite{3} to ensure that every vertex in the graph is reachable from the root, and that for any \((s, t) \in R\) there is no path from \( r \) to \( s \) that passes through \( t \) in the resulting arborescence. This can be ensured by having a flow value of 1 that enters every vertex \( k \) in the graph, and that for any vertex \( k \) the flow to that vertex does not pass through a successor of \( k \). Let \( y_{ij}^k \) be a variable associated with every vertex \( k \in V\text{\(\setminus\{r\}\) and every arc \((i, j) \in A\), such that \( y_{ij}^k = 1 \) if arc \((i, j) \in A\) is part of the path from the root \( r \) to vertex \( k \), and 0 otherwise. Let \( w_i \) be the waiting time at vertex \( i \in V\).

\[
\text{(MCF) minimize } \sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{i \in V} w_i \tag{5}
\]

subject to

\[
\sum_{(i,j) \in A} x_{ij} = 1 \quad \forall j \in V\text{\(\setminus\{r\}\) (6)}
\]

\[
\sum_{(i,j) \in A: (k,j) \notin R} y_{ij}^k - \sum_{(j,i) \in A: (k,i) \notin R} y_{ji}^k = \begin{cases} 
1 & \text{if } i = r \\
-1 & \text{if } i = k \\
0 & \text{otherwise} 
\end{cases} \quad \forall i \in V : (k,i) \notin R, k \in V \setminus \{r\} \tag{7}
\]

\[
d_r = 0 \tag{8}
\]

\[
d_j \geq d_i - M + (M + c_{ij}) x_{ij} \quad \forall (i, j) \in A \tag{9}
\]

\[
w_j \geq d_j - d_i - (M - c_{ij}) x_{ij} \quad \forall (i, j) \in A \tag{10}
\]

\[
d_t \geq d_s \quad \forall (s, t) \in R \tag{11}
\]

\[
y_{ij}^k \leq x_{ij} \quad \forall k \in V \setminus \{r\}, (i, j) \in A \tag{12}
\]

\[
x_{ij} \in \{0, 1\} \quad \forall (i, j) \in A \tag{13}
\]

\[
y_{ij}^k \in \{0, 1\} \quad \forall k \in V \setminus \{r\}, (i, j) \in A \tag{14}
\]
Constraints (6) impose the first property of an arborescence namely that every vertex \( v \in V \setminus \{r\} \) must have a single parent. Constraints (7) are the multi-commodity flow constraints: every vertex \( k \in V \) must be reachable from the root, and any path from \( r \) to \( k \) must not pass through the successors of \( k \) in the precedence graph \( P \) (otherwise this would violate a precedence relation). Constraint (8) sets the distance from the root \( r \) to itself to be equal to 0. Constraints (9) impose that when arc \((i, j)\) is selected to be part of the arborescence, then the time at which the flow enters vertex \( j \) is greater than or equal to the time at which the flow enters vertex \( i \) plus \( c_{ij} \). Constraints (10) enforce that the waiting time at each vertex \( j \) is greater than or equal to the difference between the time at which the flow enters vertex \( j \) and the time at which the flow enters vertex \( i \) plus \( c_{ij} \), where \( i \) is the parent of \( j \) in the arborescence. Constraints (11) enforce that the time at which the flow enters vertex \( t \) must be greater than or equal to the time at which the flow enters vertex \( s \), for all \((s, t) \in R\). Finally, constraints (12)-(15) define the domain of the variables. The MILP model proposed has \( O(|V||A|) \) variables, and \( O(|V||A|) \) constraints.

The major drawback of this model is the large number of variables used which might result in memory issues when solving large-sized instances, similar to what happens in the model proposed in [10] for the PCMCA.

### 3.2.2 A Distance-Accumulation Model

The model introduced in this section extends the model introduced in Section 2.2 for the PCMCA. As mentioned earlier, the time from the root \( r \) to vertex \( j \) in the arborescence is bounded from below by summing the time from \( r \) to the parent \( i \) of \( j \) and the cost of the arc \((i, j) \in A\), with \( d_r = 0 \). To ensure that the resulting arborescence satisfies the precedence constraints, we enforce that the time from \( r \) to \( t \) is greater than or equal to the time from \( r \) to \( s \) for all \((s, t) \in R\). We recall that \( w_i \) is the waiting time at vertex \( i \in V \).

\[
\begin{align*}
(dA) \text{ minimize} & \quad \sum_{(i, j) \in A} c_{ij} x_{ij} + \sum_{i \in V} w_i \\
\text{subject to} & \quad \sum_{(i, j) \in A} x_{ij} = 1 \quad \forall j \in V \setminus \{r\} \quad (17) \\
& \quad \sum_{(i, k) \in A: i \in V \setminus S, k \in S} x_{ik} \geq 1 \quad \forall j \in V \setminus \{r\}, \forall S \subseteq V_j \setminus \{r\} : j \in S \quad (18) \\
& \quad d_r = 0 \quad (19) \\
& \quad d_j \geq d_i - M + (M + c_{ij}) x_{ij} \quad \forall (i, j) \in A \quad (20) \\
& \quad w_j \geq d_j - d_i - M + (M - c_{ij}) x_{ij} \quad \forall (i, j) \in A \quad (21) \\
& \quad d_t \geq d_s \quad \forall (s, t) \in R \quad (22) \\
& \quad x_{ij} \in \{0, 1\} \quad \forall (i, j) \in A \quad (23)
\end{align*}
\]
\(d_i, w_i \geq 0 \quad \forall i \in V\) (24)

Constraints (17) impose the first property of an arborescence, namely that every vertex \(v \in V \setminus \{r\}\) must have a single parent. Constraints (18) model the connectivity constraint, that is every vertex \(v \in V \setminus \{r\}\) must be reachable from the root, and they also impose the precedence constraints where the resulting arborescence should not include vertex \(t\) in the directed path connecting \(r\) to \(s\) when \((s, t) \in R\). This will lead to an arborescence such that the flow never enters \(t\) before entering \(s\), if \(s\) precedes \(t\) on the same directed path. Constraint (19) sets the distance from the root \(r\) to itself to be equal to 0. Constraints (20) impose that when arc \((i, j)\) is selected to be part of the arborescence, then the time at which the flow enters vertex \(j\) is greater than or equal to the time at which the flow enters vertex \(i\) plus \(c_{ij}\). Constraints (21) enforce that the waiting time at vertex \(j\) is greater than or equal to the difference between the time at which the flow enters vertex \(j\) and the time at which the flow enters vertex \(i\) plus \(c_{ij}\). Constraints (22) enforce that the time at which the flow enters vertex \(t\) is greater than or equal to the time at which the flow enters vertex \(s\) for all \((s, t) \in R\). Finally, constraints (23) and (24) define the domain of the variables. The MILP model proposed, without constraints (18), has \(O(|A|)\) variables, and \(O(|A|)\) constraints. Constraints (18) are dynamically added to the model using the same separation procedure described in Section 2.2.

### 3.2.3 An Adjusted Arc-Cost Model

The model introduced in this section is originated by removing inequalities (21) from the model introduced in Section 3.2.2 and representing the value of \(w_j\) by the nonlinear term

\[w_j = \sum_{i \in (i, j) \in A} (d_j - d_i - c_{ij})x_{ij}\] (25)

A different linear model is then derived as follows.

**Proposition 1.** The waiting time at vertex \(j \in V\) can be expressed by the nonlinear equality (25).

**Proof.** Inequalities (21) can be rewritten as \(w_j \geq d_j - d_i - c_{ij} - M(1 - x_{ij})\) \(\forall (i, j) \in A\). If \(x_{ij} = 0\) then \(w_j\) has to be greater than or equal to a negative value, however the value of \(w_j\) should be greater than or equal to zero by definition. Accordingly, the inequality would be active and affect the solution only when \(x_{ij} = 1\). Therefore, we can represent the waiting time at vertex \(j\) using equality (25). \(\square\)

Based on Proposition 1 we can replace the second term in the objective function (16) as follows:

\[\sum_{j \in V} w_j = \sum_{j \in V} \sum_{i : (i, j) \in A} (d_j - d_i - c_{ij})x_{ij} = \sum_{(i, j) \in A} (d_j - d_i - c_{ij})x_{ij}\]
This means that inequalities (21) are no longer necessary as the objective function no longer depends on \( w \), which results in the following nonlinear model.

\[
\begin{align*}
\text{minimize} & \quad \sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{(i,j) \in A} (d_j - d_i - c_{ij}) x_{ij} \\
\text{subject to} & \quad \sum_{(i,j) \in A} x_{ij} = 1 \quad \forall j \in V \setminus \{r\} \\
& \quad \sum_{(i,k) \in A : i \in V_j \setminus S, k \in S} x_{ik} \geq 1 \quad \forall j \in V \setminus \{r\}, \forall S \subseteq V_j \setminus \{r\} : j \in S \\
& \quad d_r = 0 \\
& \quad d_j \geq d_i - M + (M + c_{ij}) x_{ij} \quad \forall (i,j) \in A \\
& \quad d_i \geq d_s \quad \forall (s,t) \in R \\
& \quad x_{ij} \in \{0,1\} \quad \forall (i,j) \in A \\
& \quad d_i \geq 0 \quad \forall i \in V \\
\end{align*}
\]

**Proposition 2.** Using a new set of \(|A|\) variables \( z \) and \(2|A|\) new constraints, the objective function (26) can be linearized as follows:

\[
\begin{align*}
\text{minimize} & \quad \sum_{j \in V \setminus \{r\}} d_j - \sum_{(i,j) \in A} z_{ij} \\
\end{align*}
\]

**Proof.** The objective function (26) can be rewritten as follows:

\[
\begin{align*}
\sum_{(i,j) \in A} c_{ij} x_{ij} + \sum_{(i,j) \in A} (d_j - d_i - c_{ij}) x_{ij} = \\
\sum_{(i,j) \in A} d_j x_{ij} - \sum_{(i,j) \in A} d_i x_{ij} = \sum_{j \in V \setminus \{r\}} d_j - \sum_{(i,j) \in A} d_i x_{ij} \\
\end{align*}
\]

We use the fact that \( \sum_{(i,j) \in A} d_j x_{ij} = \sum_{j \in V \setminus \{r\}} d_j \) as each \( j \in V \setminus \{r\} \) has exactly one \( x_{ij} \) assigned to 1 in an arborescence, as imposed by (27). Since the term \( d_i x_{ij} \) is summed over each arc \((i,j) \in A\), then we need at least \(2|A|\) constraints to linearize the product. We can substitute each term \( d_i x_{ij} \) by a new continuous variable \( z_{ij} \) and the following two inequalities:

\[
\begin{align*}
z_{ij} & \leq M x_{ij} \quad \forall (i,j) \in A \\
z_{ij} & \leq d_i \quad \forall (i,j) \in A \\
\end{align*}
\]

Inequalities (35) ensure that if \( x_{ij} = 0 \) then \( z_{ij} = 0 \). On the other hand, if \( x_{ij} = 1 \), then inequalities (35) ensure that \( z_{ij} \) is less than the upper bound on the optimal solution which is further tightened by inequalities (36). This results in a total of \(2|A|\) new constraints and (26) can now be expressed as \( \sum_{j \in V \setminus \{r\}} d_j - \sum_{(i,j) \in A} z_{ij} \) by elaborating on (34). \( \square \)
Based on Proposition 2, we can derive the following MILP model that contains $O(|A|)$ variables, and $O(|A|)$ constraints, plus an exponential number of constraints (28).

$$\text{(AAC)} \quad \text{minimize} \quad \sum_{j \in V \setminus \{r\}} d_j - \sum_{(i,j) \in A} z_{ij} \quad \forall j \in V \setminus \{r\}$$

subject to

$$\sum_{(i,j) \in A} x_{ij} = 1 \quad \forall j \in V \setminus \{r\} \quad (38)$$

$$\sum_{i \in V \setminus \{r\} : (i,j) \in A} x_{ik} \geq 1 \quad \forall j \in V \setminus \{r\}, \forall S \subseteq V \setminus \{r\} : j \in S \quad (39)$$

$$d_r = 0 \quad (40)$$

$$d_j \geq d_i - M + (M + c_{ij})x_{ij} \quad \forall (i,j) \in A \quad (41)$$

$$d_i \geq d_s \quad \forall (s,t) \in R \quad (42)$$

$$z_{ij} \leq d_i \quad \forall (i,j) \in A \quad (43)$$

$$z_{ij} \leq Mx_{ij} \quad \forall (i,j) \in A \quad (44)$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A \quad (45)$$

$$z_{ij} \geq 0 \quad \forall (i,j) \in A \quad (46)$$

$$d_i \geq d_s \quad \forall i \in V \quad (47)$$

**Proposition 3.** The following inequalities are valid for the (AAC) model:

$$\sum_{i \in V : (i,j) \in A} z_{ij} \leq d_j - \sum_{(i,j) \in A} c_{ij}x_{ij} \quad \forall j \in V \setminus \{r\} \quad (48)$$

**Proof.** Since for each vertex $j \in V \setminus \{r\}$ there is only one active arc $(i,j) \in A$ entering $j$ (from inequalities (38)), from inequalities (41) we can derive the following new quadratic inequalities:

$$d_j \geq \sum_{i \in V : (i,j) \in A} d_ix_{ij} + \sum_{(i,j) \in A} c_{ij}x_{ij} \quad \forall j \in V \setminus \{r\} \quad (49)$$

From inequalities (43) and (44) we have $z_{ij} \leq d_ix_{ij}$ (see Proposition 2), then inequality (48) can be derived from inequality (49) as follows.

$$d_j \geq \sum_{i \in V : (i,j) \in A} d_ix_{ij} + \sum_{(i,j) \in A} c_{ij}x_{ij} \implies \sum_{i \in V : (i,j) \in A} d_ix_{ij} \leq d_j - \sum_{(i,j) \in A} c_{ij}x_{ij}$$

$$\implies \sum_{i \in V : (i,j) \in A} z_{ij} \leq d_j - \sum_{(i,j) \in A} c_{ij}x_{ij}$$

$$\implies d_j \geq \sum_{i \in V : (i,j) \in A} z_{ij} + \sum_{(i,j) \in A} c_{ij}x_{ij}$$
It should be noted that inequalities (48) are not an integral part of the AAC model, but are added to have a stronger linear relaxation. If the inequalities are not included in the model, then the value of the $z_{ij}$s can be substantially larger than the value of the $d_j$s in order to minimize the value of the objective function. This could result in feasible solutions of the linear relaxation with a negative objective function. This would make the MILP much harder to solve. Therefore, inequalities (48) are considered for all the experiments reported in this paper.

4 Experimental Results

The computational experiments for evaluating the proposed models are based on the benchmark instances of TSPLIB [31], SOPLIB [27] and COMPILERS [33] originally proposed for the SOP [12]. The benchmark instances are the same instances previously adopted in [10].

All the experiments are performed on an Intel i7 processor running at 1.8 GHz with 8 GB of RAM. CPLEX 12.8 is used for solving the MILPs. CPLEX is run with its default parameters, and single threaded standard Branch-and-Cut (B&C) algorithm is applied for solving the MILP models, with BestBound node selection, and MIP emphasis set to MIPEmphasisOptimality. A time limit of 1 hour is set for the computation time for each computational (new) method/instance. No time limit was instead considered for the computational time of the Path-Based Model (see [10]).

In all the tables that follow, Name and Size columns report the name and size of the instance, Density of $P$ reports the density of arcs in the precedence graph computed as $\frac{2|R|}{|V|(|V|-1)}$, $z^*$ reports the value of the optimal solution for that instance. For each model we report the following columns. Cuts column reports the number of model-dependent cuts (inequalities) that are dynamically added to the model, Nodes column reports the number of nodes in the search decision-tree, Time (s) reports the solution time in seconds. The same set of columns is reported for both the results of the model’s linear relaxation (grouped under LR), and for the mixed integer linear programming model (grouped under IP).

4.1 Computational Results for the PCMCA

A MILP model for the PCMCA was previously proposed in [10], where precedence constraints are imposed by propagating a value along every path with end-points $s$ and $t$ for $(s, t) \in R$ in order to detect a precedence violation. This results in a cubic number of variables (a variable for each precedence relationship and vertex), and a quadratic number of constraints for the value propagation. The model is known to suffer from scalability and performance issues [10]. Tables 1-3 report the overall results of the model proposed in Section 2 that will
be named *Set-Based Model*, and compare its results with the results obtained by the model previously proposed in [10], that is here named *Path-Based Model*. In Tables 1-3, the *Gap* column indicates the percentage relative difference between the optimal solution ($z^*$) of the PCMCA instance and the objective function value of the model’s linear relaxation ($\text{Cost}_{LR}$), computed as $100 \cdot \frac{z^* - \text{Cost}_{LR}}{z^*}$.

An overview of the results for the *Path-Based Model* shows that its linear relaxation optimally solves 47% of the instances with a 2.1% average optimality gap. On the other hand, the linear relaxation of the *Set-Based Model* optimally solves 68% of the instances (a 44% improvement compared to *Path-Based Model*) with an average optimality gap of 1.7% (a 23% improvement compared to the *Path-Based Model*). The solution times for the integer *Path-Based Model* range between milliseconds and 2.5 hours (the maximum computing time allowed was longer in [10]), with an average of 276 seconds, a median of 3 seconds, and standard deviation of 1116 seconds. The solution times for the integer *Set-Based Model* range between milliseconds and 15 minutes, with an average of 27 seconds, a median of 0.8 seconds, and standard deviation of 129 seconds (this is on average a 90% improvement compared to the *Path-Based Model*). In the integer *Set-Based Model*, the number of cuts generated by exploring the whole branch-decision-tree increases by 80% on average, compared to the root of the branch-decision-tree itself, and the solver explores 77 nodes on average. On the other hand, for the integer *Path-Based Model* the solver explores 5588 nodes on average (a 98% increase).

By inspecting Tables 1-3, we can observe that the *Path-Based Model* from [10] optimally solves a subset of the instances faster than the *Set-Based Model*. We can see that those instances (underlined in the tables) are relatively large in size and have either a very sparse or very dense precedence graph. More specifically, in terms of size the *Path-Based Model* is faster at solving 54% of the instances that have a size larger than 500. In terms of precedence graph density, the *Path-Based Model* is faster at solving 62% of the instance with density smaller than 0.005 and is faster at solving 57% of the instances with density larger than 0.990. Considering the two factors simultaneously, the *Path-Based Model* is faster at solving 57% of the instances with size larger than 500 and precedence graph density that is smaller than 0.008 or larger than 0.940. A low density precedence graph implies a small number of variables and constraints used to model the precedence relationships in the *Path-Based Model*, and since finding a violated precedence inequality is much faster in that model, it is sometimes more efficient at solving those instances. In other instances, the increase in solution time is justified by the time it takes to find a violated inequality in the *Set-Based Model*. In general, if we look at Figure 7, which shows the distribution of solution times for each model, we see that the *Set-Based Model* is much faster at solving the instances, even when we consider or exclude outliers. The large solution time in the *Set-Based Model* for the two instances prob.100 and R.700.100.1, compared to the *Path-Based Model*, can be explained by the number of cuts generated while solving the LR, which also increases the overall solution time. We can verify that by observing the solution time of the LR for the first instance.

In conclusion, the *Set-Based Model* is a significant improvement over previ-
Figure 7: A box plot showing the distribution of solution times (in seconds) of all the 116 instances for the Path-Based Model from [10] and the Set-Based Model. The box plot on the right excludes outliers.

ous methods. Indeed, it provides optimal solutions in substantially less time, and memory usage for the majority of the instances considered. The same cannot be said about the Set-Based Model when linear relaxations only are considered, as the Path-Based Model can be solved much faster in most cases because of the fewer number of constraints, although it sometimes generates a looser estimate on the value of the optimal integer solution. In terms of memory usage, the Set-Based Model consumes approximately an average of 95MB, with a standard deviation of 132 and median of 41 when solving the instances considered. On the other hand, the Path-Based Model consumes approximately an average of 363MB, with a standard deviation of 486 and median of 116. For considerably large sized instances such as R.700.100.30 and R.700.100.60, the Path-Based Model consumes 1841MB and 1097MB for each of those two instances, whereas the Set-Based Model consumes 263MB and 286MB for the same instances. The two instances considered are solved at the root node of the branch-decision tree by both models.

4.2 Computational Results for the PCMCA-WT

For the computational experiments of the PCMCA-WT, we omit the detailed results for SOPLIB benchmark instances. However, we can draw the following conclusions on these instances. The Multi-Commodity Flow (MCF) model is unable to solve large sized instances because of memory issues (building the model consumes around 5GB of memory on average) or time out while solving the model’s linear relaxation. Since the linear relaxation of MCF model was unable to solve any single instance from SOPLIB benchmark set, we concluded that it is highly unsuitable for solving such instances. The characteristics of SOPLIB instances are summarized in Table 2. The Distance-Accumulation (DA) model and the Adjusted Arc-Cost (AAC) model are able to optimally solve SOPLIB instances with low density precedence graphs within the time limit, with an average of 800 seconds, and achieve an average optimality gap of 63.8% for the LR models and 63.3% for the IP models before timing out for the remaining instances. These figures are much higher compared to the other two benchmark
sets as will be shown later. The computational experiments have shown that large-sized instances with a highly dense precedence graph are outside the reach of the models proposed due to the intrinsic complexity of the problem.
| Instance     | Name      | Size  | Density of P^* | LR IP | LR IP | LR IP | LR IP | LR IP | LR IP |
|--------------|-----------|-------|----------------|-------|-------|-------|-------|-------|-------|
| br17.10      | 18        | 0.314 | 25             | 0.032 | 0.000 | 3     | 0.060 | 21    | 0.015 |
| br17.12      | 18        | 0.359 | 25             | 0.047 | 0.000 | 3     | 0.063 | 22    | 0.016 |
| ESC07        | 9         | 0.611 | 1531           | 0.031 | 0.000 | 0     | 0.031 | 13    | 0.014 |
| ESC11        | 13        | 0.350 | 1752           | 0.031 | 0.000 | 0     | 0.031 | 1     | 0.031 |
| ESC12        | 14        | 0.396 | 1138           | 0.016 | 0.000 | 0     | 0.016 | 1     | 0.016 |
| ESC25        | 27        | 0.177 | 1041           | 0.062 | 0.000 | 0     | 0.062 | 31    | 0.063 |
| ESC47        | 49        | 0.108 | 703            | 0.484 | 0.284 | 5     | 0.469 | 142   | 0.547 |
| ESC63        | 65        | 0.173 | 56             | 0.329 | 0.000 | 0     | 0.329 | 42    | 0.218 |
| ESC78        | 80        | 0.130 | 502            | 0.094 | 0.000 | 0     | 0.094 | 1     | 0.047 |
| H5.1         | 54        | 0.082 | 3917           | 1.172 | 0.408 | 7     | 1.172 | 78    | 0.328 |
| H5.2         | 54        | 0.094 | 3978           | 0.281 | 7.642 | 104   | 0.688 | 57    | 0.188 |
| H5.3         | 54        | 0.225 | 4242           | 1.890 | 5.877 | 122   | 2.547 | 96    | 0.453 |
| H5.4         | 54        | 0.604 | 4882           | 0.156 | 2.663 | 9     | 0.250 | 13    | 0.047 |
| H7.1         | 71        | 0.036 | 32846          | 2.891 | 0.035 | 2     | 2.828 | 158   | 2.750 |
| H7.2         | 71        | 0.075 | 32980          | 2.985 | 0.035 | 2     | 3.016 | 163   | 2.719 |
| H7.3         | 71        | 0.142 | 33431          | 0.750 | 2.423 | 954   | 63.171| 66    | 0.265 |
| krb04Sa      | 50        | 0.444 | 264            | 0.047 | 0.000 | 0     | 0.047 | 5     | 0.031 |
| krb05sc      | 52        | 0.450 | 191            | 0.313 | 0.000 | 0     | 0.313 | 11    | 0.047 |
| krb100       | 111       | 0.909 | 256            | 11.578| 0.000 | 0     | 11.578| 14    | 0.094 |
| krb150a      | 152       | 0.927 | 373            | 2.485 | 0.000 | 0     | 2.485 | 14    | 0.187 |
| krb174a      | 174       | 0.929 | 362            | 20.931| 0.074 | 1     | 22    | 23    | 0.297 |
| krb253a      | 255       | 0.948 | 753            | 13.985| 0.000 | 0     | 13.985| 22    | 1.125 |
| krb323a      | 325       | 0.928 | 754            | 1.547 | 0.000 | 0     | 1.547 | 26    | 1.047 |
| krb344a      | 344       | 0.937 | 610            | 23.344| 3.279 | 376   | 278.859| 89    | 3.031 |
| krb358a      | 360       | 0.886 | 595            | 0.312 | 0.000 | 0     | 0.312 | 67    | 5.812 |
| krb378a      | 380       | 0.894 | 559            | 10.079| 3.936 | 543   | 178.315| 21    | 4.472 |
| kro124p.1    | 101       | 0.046 | 32597          | 0.734 | 5.997 | 47    | 1.844 | 95    | 1.782 |
| kro124p.2    | 101       | 0.053 | 32851          | 0.578 | 6.929 | 1433  | 11.203| 169   | 1.828 |
| kro124p.3    | 101       | 0.092 | 33779          | 8.672 | 2.680 | 25068 | 6599.140| 69    | 3.486 |
| kro124p.4    | 101       | 0.496 | 37124          | 41.828| 1.375 | 198   | 59.359| 128   | 1.672 |
| p33.1        | 44        | 0.211 | 270200         | 1.016 | 8.364 | 119   | 1.781 | 33    | 1.189 |
| p34.2        | 44        | 0.126 | 27200          | 0.547 | 14.407| 283   | 2.829 | 77    | 7.537 |
| p45.3        | 44        | 0.191 | 27200          | 0.121 | 8.688 | 198   | 3.516 | 11    | 8.333 |
| prob.100     | 100       | 0.048 | 650            | 11.766| 1.308 | 1428  | 46.594| 1840  | 622.437|
| prob.42      | 42        | 0.116 | 143            | 0.125 | 0.000 | 0     | 0.125 | 2     | 0.032 |
| ry48p.1      | 49        | 0.691 | 13065          | 0.828 | 8.886 | 879   | 1.658 | 31    | 1.894 |
| ry48p.2      | 49        | 0.103 | 13103          | 1.031 | 5.551 | 220   | 1.593 | 58    | 0.235 |
| ry48p.3      | 49        | 0.193 | 13886          | 2.157 | 1.657 | 12233 | 638.514| 34    | 6.852 |
| ry48p.4      | 49        | 0.588 | 15340          | 2.531 | 7.210 | 8610  | 24.156| 65    | 1.572 |
| Average      |           |       |                | 4.808 | 2.484 | 9700  | 194.923| 94    | 15.878|

Table 1: Overall computational results for comparing the Path-Based Model from [10] and the Set-Based Model for the PCMCA for TSPLIB instances.
Table 2: Overall computational results for comparing the Path-Based Model from [10] and Set-Based Model for the PCMCA for SOPLIB instances

| Instance | Path-Based Model | Set-Based Model |
|----------|------------------|-----------------|
|          | LR               | IP              | LR               | IP              |
| Name     | Size             | Density of P    | Time (s) Gap Nodes | Time (s) Cuts | Time (s) Gap Nodes | Cuts | Cuts | Time (s) |
| R.200.100.1 | 200              | 0.020           | 11,979            | 0.000          | 321               | 0.000 | 321   | 77     |
| R.200.100.15 | 200             | 0.957           | 0.219             | 0.000          | 4.35              | 0.000 | 4.35  | 77     |
| R.200.100.30 | 200             | 0.991           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.200.100.60 | 200             | 0.989           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.200.150.1 | 200             | 0.970           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.200.150.15 | 200            | 0.958           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.200.150.30 | 200            | 0.958           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.200.150.60 | 200            | 0.958           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.300.100.1 | 300             | 0.970           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.300.100.15 | 300            | 0.958           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.300.100.30 | 300            | 0.958           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| R.300.100.60 | 300            | 0.958           | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |
| Average   | 98.409           | 11.141          | 0.194             | 0.000          | 4.21              | 0.000 | 4.21  | 77     |

21
Table 3: Overall computational results for comparing the Path-Based Model from [10] and Set-Based Model for the PCMCA for COMPILERS instances

| Instance   | Name      | Size  | Density of P | $z^*$ | LR Time (s) | IP Time (s) | Gap Nodes | Cuts Time (s) | Gap Cuts | Nodes IP Time (s) | Cuts IP Time (s) | Gap Cuts |
|------------|-----------|-------|---------------|------|-------------|-------------|-----------|--------------|----------|----------------|----------------|----------|
| gsm.153.124| 126       | 0.970 | 185           | 0.578| 0.000       | 0.578       | 0         | 0.125        | 1.081    | 3              | 53             | 0.140    |
| gsm.444.350| 353       | 0.990 | 1542          | 0.078| 0.000       | 0.078       | 0         | 0.094        | 0.000    | 0              | 0              | 0.094    |
| jpeg.1483.25| 27        | 0.842 | 292           | 3.422| 0.000       | 4.047       | 17        | 0.031        | 0.000    | 14             | 0.031          | 0.031    |
| jpeg.3184.107| 109     | 0.887 | 411           | 14.640| 0.487       | 16.844      | 24        | 0.093        | 0.000    | 32             | 0.093          | 0.093    |
| jpeg.3195.85| 87        | 0.740 | 13             | 278.844| 38.462   | 30.882      | 4041      | 1366.985    | 45.012   | 5674           | 16979          | 897.312  |
| jpeg.3198.93| 95        | 0.752 | 140            | 252.734| 2.857     | 255.591     | 2204      | 529.781     | 29.041   | 401            | 1686           | 9.704    |
| jpeg.3203.135| 137      | 0.897 | 507            | 47.578| 0.394       | 50.972      | 31        | 56.703      | 18.097   | 7              | 41             | 0.125    |
| jpeg.3740.15| 17        | 0.257 | 33             | 1.782 | 3.030       | 0.234       | 231       | 0.031        | 0.000    | 0              | 17             | 0.031    |
| jpeg.4154.36| 38        | 0.633 | 74             | 0.641 | 5.405       | 6.046       | 1462      | 2.500       | 43.063   | 0              | 43             | 0.063    |
| jpeg.4753.54| 56        | 0.769 | 146            | 2.766 | 0.685       | 2.841       | 11        | 2.984       | 38.062   | 0              | 6              | 0.109    |
| susan.2418.197| 199     | 0.939 | 588            | 76.329| 0.340       | 76.670      | 22        | 106.672     | 21.032   | 0              | 21             | 0.125    |
| susan.260.158| 480      | 0.916 | 472            | 12.156| 1.695       | 12.321      | 570       | 123.594     | 33.014   | 0              | 33             | 0.141    |
| susan.343.182| 184      | 0.936 | 468            | 194.188| 1.068   | 195.256     | 776       | 474.391     | 47.023   | 19             | 89             | 0.359    |
| typeset.10192.123| 125     | 0.744 | 241            | 4.859 | 10.373     | 5.665       | 5565      | 297.859     | 93.050   | 0              | 93             | 0.500    |
| typeset.10835.26| 28       | 0.349 | 60             | 0.063 | 0.000       | 0.063       | 0         | 0.031        | 0.000    | 0              | 14             | 0.031    |
| typeset.12395.43| 45       | 0.518 | 125            | 0.531 | 0.800       | 1.047       | 10        | 0.437       | 27.078   | 0              | 27             | 0.078    |
| typeset.15087.23| 25       | 0.557 | 89             | 0.297 | 1.124       | 0.327       | 32        | 0.297       | 24.047   | 0              | 24             | 0.047    |
| typeset.15577.36| 38       | 0.555 | 93             | 0.031 | 0.000       | 0.031       | 0         | 0.031       | 4.015    | 0              | 4              | 0.015    |
| typeset.16000.68| 70       | 0.658 | 67             | 21.891| 0.000       | 21.891      | 0         | 21.891      | 643.281  | 8.955         | 144            | 1316.72  |
| typeset.1723.25| 27       | 0.245 | 54             | 0.203 | 5.556       | 4.094       | 7660      | 5.556       | 19.031   | 21             | 99             | 0.110    |
| typeset.19972.246| 248     | 0.993 | 979            | 0.110 | 0.000       | 0.110       | 0         | 0.062       | 0.000    | 0              | 0              | 0.062    |
| typeset.4391.240| 242     | 0.981 | 837            | 378.172| 0.119   | 380.172     | 46        | 6.250       | 18.094   | 0              | 18             | 0.094    |
| typeset.4597.45| 47       | 0.493 | 133            | 0.437 | 0.000       | 0.437       | 0         | 0.437       | 7.031    | 0              | 7              | 0.031    |
| typeset.4724.433| 335      | 0.995 | 1819           | 4.000 | 0.000       | 4.000       | 0         | 4.000       | 8.172    | 0              | 8              | 0.172    |
| typeset.5797.33| 35       | 0.748 | 93             | 0.234 | 0.000       | 0.234       | 0         | 0.234       | 9.032    | 0              | 9              | 0.032    |
| typeset.5881.246| 248     | 0.986 | 979            | 191.813| 0.306   | 191.813     | 191       | 356.218     | 52.343   | 0              | 52             | 0.343    |
| Average    | 55.134   | 2.693 | 849            | 125.095| 4.226   | 2.276       | 233       | 769         | 33.965   | 25             |                |          |
4.2.1 Computational Results for LR Models

Tables 4-5 show the overall results for the linear relaxation of the MILP models proposed for the PCMCA-WT. In all the tables the Cost column reports the value of the objective function. The Gap column indicates the percentage relative difference between the cost of the best known integer solution of the instance (Cost\textsubscript{Best}), and the objective function cost of the model’s linear relaxation (Cost\textsubscript{LR}), computed as $100 \cdot \frac{\text{Cost}_{\text{Best}} - \text{Cost}_{\text{LR}}}{\text{Cost}_{\text{Best}}}$. The Cuts column indicates the number of inequalities that are dynamically added to the model, that is inequalities (18) and (39) for each model. The solution information are not reported for instances where the solver times out or runs out of memory before finding the optimal solution.

The linear relaxation of the MCF model has an average optimality gap of 20.22%, and the solver times out before finding the optimal solution for the model’s linear relaxation for instances that are larger than 240. Comparing the results for the DA and AAC models, the first model’s linear relaxation has an average optimality gap of 23.96%, whereas the second model has an average optimality gap of 23.99% across all the instances. Comparing the number of generated cuts, the AAC model generates 6% less cuts compared to the DA model. We can notice that the DA model finds higher estimates for the optimal integer solution compared to the other two models, however the AAC model finds better estimates on the symmetrical COMPILERS instances which have symmetric costs. Instances where the AAC model and MCF model found tighter estimates are underlined in the tables.

A major problem that we can notice in the MCF model is that the solution times are much larger when compared to the other two models. For example, the MCF model finds the optimal solution of ESC78 instance within 9 minutes compared to 4 and 6 seconds of computing time by the other two models. The same increased solution time can be noticed in other instances, sometimes reaching almost an hour to solve the linear relaxation compared to few seconds. For the instances that are optimally solved by all three LR models, the solution time is on average 889 seconds for the MCF model, 19 seconds for the DA model, and 38 seconds for the AAC model.

In general, it is hard to decide which linear relaxation would perform better on some instances, however the DA model seems to be the most suitable, as its linear relaxation is much easier to solve compared to the other two, and its result exhibits a lower average optimality gap compared to the other two models.
| Instance | Name | Size | Density of P | Cost Time (s) | Gap | Cost Cuts Time (s) | Gap | Cost Cuts Time (s) | Gap |
|----------|------|------|--------------|---------------|-----|-------------------|-----|-------------------|-----|
| br17.12  | 18   | 0.359| 25.12 1.032 42.917 | 25.17 15 | 0.265 42.795 | 25.15 18 | 0.263 42.841 |
| ESC07    | 9    | 0.611| 1878.50 0.204 0.971 | 1900.75 3 0.110 0.800 | 1782.07 7 0.031 6.502 |
| ESC11    | 13   | 0.359| 2137.05 0.297 2.162 | 2067.00 10 0.187 4.922 | 2040.30 8 0.312 6.150 |
| ESC12    | 14   | 0.296| 1318.00 0.109 0.000 | 1318.00 0 0.063 0.000 | 1138.00 1 0.078 0.000 |
| ESC25    | 27   | 0.177| 1043.05 3.297 9.927 | 1082.41 37 0.672 6.528 | 1064.20 40 0.890 8.100 |
| ESC47    | 49   | 0.108| 703.14 36.969 5.872 | 703.12 257 9.250 5.874 | 703.14 80 3.625 5.871 |
| ESC63    | 65   | 0.173| 560.00 266.610 0.000 | 560.00 6 1.594 0.000 | 560.00 67 20.937 0.000 |
| ESC78    | 80   | 0.139| 502.16 523.810 58.014 721.93 8 4.453 39.638 | 718.00 6 5.969 39.667 |
| f51.4    | 54   | 0.082| 3953.05 188.391 3.325 | 3962.45 34 5.904 3.995 | 3993.84 52 8.547 6.773 |
| f51.5    | 54   | 0.094| 3997.50 180.250 6.688 | 3998.74 40 5.531 6.659 | 3993.84 52 8.547 6.773 |
| f53.4    | 54   | 0.225| 4286.70 191.200 21.442 | 4286.35 69 7.640 19.583 | 4249.72 97 13.562 22.124 |
| f53.5    | 54   | 0.604| 5023.27 52.062 21.940 | 5149.40 18 4.875 20.028 | 5010.26 21 5.250 22.189 |
| f70.1    | 71   | 0.036| 3280.40 1021.590 1.492 | 3298.40 148 16.100 0.554 | 3285.51 130 39.453 1.341 |
| f70.2    | 71   | 0.075| 3299.06 1523.521 4.141 | 3301.60 160 22.235 4.161 | 3293.71 171 48.172 4.384 |
| f70.3    | 71   | 0.142| 3341.93 2048.220 21.740 | 3361.84 402 47.500 21.272 | 3367.54 264 63.344 21.201 |
| kro12.4p-1| 101  | 0.046| 3259.08 3482.940 7.476 | 3295.90 106 47.765 7.384 | 3260.69 123 89.266 7.457 |
| kro12.4p-2| 101  | 0.053| 3276.06 3482.630 13.687 | 3305.63 135 48.688 12.916 | 3292.44 171 134.109 13.262 |
| kro12.4p-3| 101  | 0.092| 3715.31 3490.750 37.549 | 3951.74 270 76.203 37.112 | 3928.66 303 212.000 37.446 |
| kro12.4p-4| 101  | 0.496| 3786.23 2552.250 32.255 | 3802.91 132 35.250 31.096 | 3733.59 174 88.859 32.532 |
| p43.1    | 44   | 0.101| 2825.00 864.844 36.801 | 2825.00 49 2.140 36.801 | 2797.37 53 3.787 37.419 |
| p43.2    | 44   | 0.126| 2759.38 1036.547 35.453 | 2759.38 98 2.672 33.918 | 2722.91 140 9.422 36.306 |
| p43.3    | 44   | 0.191| 2759.53 573.968 48.666 | 2845.00 113 3.469 47.070 | 2722.79 197 10.407 49.343 |
| p43.4    | 44   | 0.164| 2925.07 1193.780 40.305 | 2930.08 115 2.984 40.202 | 2822.27 93 4.968 42.403 |
| prob.100 | 100  | 0.048| 643.00 3448.390 36.210 | 661.13 1225 596.594 33.717 | 657.65 1000 999.453 34.757 |
| ry48.p.1 | 49   | 0.091| 1313.04 99.141 4.285 | 1313.93 62 4.953 4.286 | 1313.56 54 8.391 4.421 |
| ry48.p.2 | 49   | 0.103| 13195.09 95.937 9.986 | 13243.77 48 4.703 9.654 | 13206.48 34 5.203 9.909 |
| ry48.p.3 | 49   | 0.193| 13092.14 136.850 14.700 | 13097.71 207 12.469 14.371 | 13055.41 191 19.016 14.704 |
| ry48.p.4 | 49   | 0.588| 16168.48 167.781 17.713 | 16316.13 60 5.344 16.962 | 16186.84 93 9.406 17.629 |
| Average  |      |      | 835.671 19.921 | 108 | 61.691 24.784 | 99 | 166.982 25.626 |
Table 5: LR Models computational results for PCMCA-WT for COMPILERS instances

| Instance   | MCF   | DA      | AAC      |
|------------|-------|---------|----------|
| Name       | Size  | Density of P | Cost | Time (s) | Gap | Cost | Cuts | Time (s) | Gap | Cost | Cuts | Time (s) | Gap |
| gsm.153.124| 126   | 0.97     | 221.14 | 135.500  | 29.348 | 222.23 | 15 | 0.610 | 29.000 | 223.41 | 15 | 3.312 | 29.623 |
| gsm.444.350| 353   | 0.99     | -     | -         | -     | -     | 1914.83 | 6 | 4.531 | 33.351 | 2042.75 | 4 | 5.156 | 28.898 |
| gsm.462.77 | 79    | 0.84     | 377.54 | 231.625  | 22.636 | 384.41 | 27 | 6.016 | 21.227 | 380.96 | 29 | 5.375 | 21.934 |
| jpeg.1483.25| 27    | 0.484    | 84.00  | 1.844    | 3.446  | 78.97 | 17 | 0.406 | 9.230 | 76.89 | 16 | 0.719 | 11.621 |
| jpeg.3184.107| 109  | 0.887    | 419.22 | 254.187  | 38.710 | 441.65 | 60 | 2.875 | 35.431 | 451.07 | 76 | 16.047 | 34.054 |
| jpeg.3195.85| 87    | 0.74     | 13.04  | 3595.590 | 47.837 | 9.00  | 126 | 7.857 | 64.000 | 13.00 | 195 | 9.130 | 48.000 |
| jpeg.3198.93| 95    | 0.752    | 140.26 | 3594.700 | 31.244 | 151.87 | 214 | 9.296 | 25.554 | 152.79 | 152 | 11.730 | 25.103 |
| jpeg.3203.135| 15    | 0.257    | 33.00  | 0.313    | 0.000  | 33.00 | 5    | 0.093 | 0.000  | 33.00 | 3    | 0.125 | 0.000  |
| jpeg.4154.36| 56    | 0.633    | 86.88  | 7.483    | 3.469  | 85.06 | 26 | 2.125 | 5.489 | 84.01 | 21 | 0.765 | 6.656  |
| jpeg.4753.54| 56    | 0.769    | 150.20 | 76.875   | 8.413  | 153.08 | 30 | 3.500 | 6.659 | 150.19 | 40 | 2.250 | 8.421  |
| susan.248.197| 199  | 0.939    | 613.41 | 3519.028 | 48.192 | 658.84 | 108 | 9.672 | 44.355 | 682.70 | 138 | 34.656 | 42.340 |
| susan.260.158| 160  | 0.916    | 494.65 | 1681.391 | 43.533 | 519.01 | 116 | 7.796 | 40.752 | 534.43 | 244 | 55.359 | 38.992 |
| susan.343.182| 184  | 0.936    | 469.79 | 1488.110 | 45.500 | 539.47 | 72 | 6.156 | 37.416 | 554.04 | 94 | 20.234 | 35.726 |
| typeset.10192.123| 125 | 0.744     | 246.52 | 3579.440 | 40.599 | 264.30 | 131 | 22.078 | 36.313 | 260.60 | 90 | 23.328 | 37.205 |
| typeset.10835.26| 28   | 0.349    | 93.55  | 2.187    | 16.470 | 81.83 | 7    | 0.328 | 26.938 | 92.34 | 9    | 0.625 | 17.554 |
| typeset.12395.43| 45   | 0.518    | 139.02 | 57.437   | 4.784  | 137.85 | 110 | 3.094 | 5.582 | 137.27 | 107 | 4.484 | 5.979  |
| typeset.15087.23| 25   | 0.557    | 92.27  | 2.046    | 4.878  | 93.00 | 13 | 0.157 | 4.124 | 93.00 | 30 | 0.516 | 4.124  |
| typeset.15577.36| 38   | 0.555    | 120.01 | 4.141    | 3.995  | 120.69 | 21 | 0.531 | 3.448 | 120.01 | 14 | 1.015 | 3.992  |
| typeset.16000.68| 70   | 0.658    | 70.00  | 2051.330 | 12.500 | 70.07 | 121 | 4.062 | 12.413 | 69.48 | 486 | 32.745 | 13.150 |
| typeset.19972.246| 248  | 0.993    | 56.00  | 5.566    | 6.607  | 55.33 | 93 | 1.062 | 7.783 | 55.50 | 90 | 2.047 | 7.500  |
| typeset.4391.240| 242  | 0.981    | -     | -        | -     | -     | 1229.52 | 4 | 1.891 | 36.261 | 1234.43 | 11 | 4.328 | 36.007 |
| typeset.4597.45| 47    | 0.493    | 144.01 | 23.484   | 7.094  | 143.13 | 89 | 3.282 | 7.658 | 141.18 | 169 | 9.281 | 8.916  |
| typeset.4724.433| 435  | 0.995    | -     | -        | -     | -     | 2351.03 | 29 | 12.016 | 31.517 | 2351.76 | 33 | 21.531 | 31.495 |
| typeset.5797.33| 35    | 0.748    | 105.93 | 2.843    | 6.258  | 106.00 | 9  | 0.625 | 6.195 | 104.21 | 21 | 0.891 | 7.779  |
| typeset.5881.246| 248  | 0.986    | -     | -        | -     | -     | 1204.29 | 27 | 5.234 | 29.159 | 1229.51 | 28 | 9.422 | 27.676 |
| Average    | 978.753 | 20.713  | 58     | 4.495    | 22.718 | 85 | 11.348 | 21.518 |
4.2.2 Computational Results for IP Models

Tables 6-7 show the overall results of the MILP models proposed for the PCMCA-WT. In Tables 6-7, the Cost column reports the lower bound (LB) and upper bound (UB) on the value of the objective function obtained from solving the respective model. The IP Gap column measures the percentage relative difference between the upper and lower bound obtained from solving the respective model, calculated as $100 \cdot \frac{UB - LB}{UB}$. Cuts column indicates the number of inequalities that are dynamically added to the model, that is inequalities (15) and (39) for each model. The solution time is not reported in the tables for the instances that are not optimally solved within the time limit. Moreover, the solution information is not reported for instances where it was not possible to solve the model’s linear relaxation (see Tables 4-5).

The MCF model achieves an optimality gap of 29% on average across all the instances it is able to solve, compared to 14% for the DA model, and 15% for the AAC model. We can notice that the solver is finding it difficult to solve the linear relaxation of the MCF model by observing the small number of nodes generated in the time limit for some of the instances, compared to the other two models. However, the MCF model is able to provide the best known solution (underlined in the tables) for 1 out of 68 instances, and the best lower bound for 1 instance.

Comparing the results of the DA and AAC models, we can see that the DA model has an average optimality gap of 19% compared to 21% across all the instances. Furthermore, the DA model solves three extra instances for the TSPLIB instances, while the AAC model solves one extra instance for the COMPILERS instances. For the instances (marked bold in the tables) that are optimally solved by both models, the DA model is 119% faster at solving those instances on average, computed as $100 \cdot \frac{T_{DA} - T_{AAC}}{T_{DA}}$, where $T_{DA}$ and $T_{AAC}$ is the average solution time for solving those instances using the DA model and AAC model. Comparing the number of branch-decision-tree nodes generated by the two models, the DA model generates 36% more nodes on average across all the instances, and 10% more nodes on average for the instances that are optimally solved by the two models. The DA model found the best lower bound for 55 out of 68 instances, and better upper bounds for 51 out of 68 instances. On the other hand, the AAC model found tighter lower bounds for 12 out of 68 instances, and better upper bounds for 16 of those instances (underlined in the tables). In general, the DA model performs better than the AAC model, except on symmetrical instances and/or instances with extreme high densities larger than 0.9.

We can conclude that the AAC model is more suitable for symmetrical instances with extreme densities, while the DA model is more suitable for general instances. The MCF finds better bounds compared to the other two models for some instances, however it is not suitable for instances with size larger than 200.
Table 6: IP Models computational results for PCMCA-WT for TSPLIB instances

| Instance | MCF          | DA           | AAC          |
|----------|--------------|--------------|--------------|
| br17.10  | 0.31        | [35, 44]      | 20.455       |
| br17.12  | 0.35        | [35, 44]      | 20.455       |
| ESC07    | 0.11        | 1906.00      | 17.09        |
| ESC11    | 0.35        | 2174.00      | 20.455       |
| ESC12    | 0.24        | 1138.00      | 4.578        |
| ESC25    | 0.06        | 1158.00      | 24.625       |
| ESC47    | 0.12        | 747.00       | 0.000        |
| ESC63    | 0.17        | [65, 61]      | 4.578        |
| ESC78    | 0.15        | [922, 1346]  | 3.510        |
| f53.1    | 0.08        | [3984, 4089] | 2.586        |
| f53.2    | 0.09        | [4054, 4500] | 12.986       |
| f53.3    | 0.25        | [4472, 4671] | 26.388       |
| f53.4    | 0.06        | [5409, 6871] | 21.278       |
| f70.1    | 0.06        | [3267, 3728] | 11.168       |
| f70.2    | 0.075       | [3308, 4326] | 23.562       |
| f70.3    | 0.14        | 5993.00      | 24.625       |
| f70.4    | 0.26        | 1392.00      | 4.578        |
| fbg050c  | 0.25        | [255, 267]   | 2.996        |
| fbg199   | 0.12        | [320, 499]   | 5.249        |
| fbg199a  | 0.25        | [403, 972]   | 5.249        |
| fbg17n   | 0.09        | [400, 1121]  | 34.538       |
| fbg25a   | 0.08        | -            | -            |
| fbg23a   | 0.24        | -            | -            |
| fbg41a   | 0.33        | -            | -            |
| fbg35a   | 0.86        | -            | -            |
| fbg37a   | 0.84        | -            | -            |
| fbg050c1 | 0.046       | [3259, 4565] | 28.595       |
| fbg199   | 0.09        | [320, 699]   | 5.249        |
| prob.100 | 0.048       | [643, 361]   | 8.219        |
| prob.42  | 0.12        | [163, 171]   | 4.678        |
| ry48p.1  | 0.49        | [3276, 5256] | 40.718       |
| ry48p.2  | 0.05        | [3712, 3927] | 89.770       |
| ry48p.3  | 0.01        | [2825, 4470] | 36.801       |
| ry48p.4  | 0.15        | [2828, 6995] | 59.339       |
| prob.100 | 0.01        | [643, 361]   | 8.219        |
| prob.42  | 0.12        | [163, 171]   | 4.678        |
| ry48p.1  | 0.49        | [3276, 5256] | 40.718       |
| ry48p.2  | 0.05        | [3712, 3927] | 89.770       |
| ry48p.3  | 0.01        | [2825, 4470] | 36.801       |
| ry48p.4  | 0.15        | [2828, 6995] | 59.339       |

Average: 28.164 14679 20.723 252211 5772 693.392
Table 7: IP Models computational results for PCMCA-WT for COMPILERS instances

| Instance   | Name       | Size | Density of P | Cost IP Gap | Nodes | Time (s) | Cost IP Gap | Nodes | Cuts | Time (s) | Cost IP Gap | Nodes | Cuts | Time (s) |
|------------|------------|------|--------------|-------------|-------|----------|-------------|-------|------|----------|-------------|-------|------|----------|
| cet.19972  | jpeg.1483.25 | 27   | 0.484        | 87          | 0.000  | 2338     | 30.526      | 50228 | 1051 | 20.667   | 270786      | 30000 | 30000 | 270786   |
| cet.19972  | jpeg.3198.93 | 95   | 0.752        | 60          | 0.000  | 213      | 19.675      | 27085 | 7545 | 27.680   | 18262       | 30000 | 30000 | 18262    |
| cet.19972  | jpeg.3203.135| 137  | 0.997        | 65          | 0.000  | 2338     | 20.530      | 179785 | 1051 | 20.667   | 270786      | 30000 | 30000 | 270786   |
| cet.19972  | jpeg.3740.15 | 17   | 0.257        | 33          | 0.000  | 2338     | 33          | 0.000  | 5    | 0.000    | 270786      | 30000 | 30000 | 270786   |
| cet.19972  | jpeg.4154.36 | 38   | 0.633        | 50          | 0.000  | 10271    | 39.358      | 62265 | 1519 | 45.602   | 39582       | 30000 | 30000 | 39582    |
| cet.19972  | jpeg.4753.54 | 56   | 0.769        | 817          | 1826   | [161, 165]| 2.112       | 112802 | 1079| 1.644    | 94353       | 30000 | 30000 | 94353    |
| cet.19972  | susan.249.197| 190  | 0.939        | 79           | 0.000  | 2338     | 30.526      | 50228 | 1051 | 20.667   | 270786      | 30000 | 30000 | 270786   |
| cet.19972  | susan.260.158| 160  | 0.916        | 80           | 0.000  | 2338     | 541         | 11490 | 3999| 35.636   | 40000       | 30000 | 30000 | 40000    |
| cet.19972  | susan.343.182| 184  | 0.936        | 63           | 0.000  | 2338     | 586         | 11330 | 5011| 33.371   | 53066       | 30000 | 30000 | 53066    |
| cet.19972  | typeset.10192.123| 125 | 0.744        | 68           | 0.000  | 2338     | 280         | 415   | 3643| 38.596   | 59000       | 30000 | 30000 | 59000    |
| cet.19972  | typeset.10935.26| 28   | 0.349        | 13           | 119310| 599       | 11.607      | 896684 | 7000| 12.369   | 57777       | 30000 | 30000 | 57777    |
| cet.19972  | typeset.12955.43| 45   | 0.518        | 47           | 5556   | [143, 146]| 2.055       | 392093 | 4782| 4.062    | 177106      | 30000 | 30000 | 177106   |
| cet.19972  | typeset.15067.23 | 25  | 0.557        | 97           | 10941  | 0.000    | 60     | 1212   | 281  | 1.085    | 112526      | 30000 | 30000 | 112526   |
| cet.19972  | typeset.15577.36| 38   | 0.555        | 125          | 173848 | 0.000    | 1350     | 168000 | 467  | 51.688   | 106552      | 30000 | 30000 | 106552   |
| cet.19972  | typeset.16000.68| 70   | 0.658        | 81           | 13917  | 0.000    | 77       | 10465  | 25666| 3.750    | 29319       | 30000 | 30000 | 29319    |
| cet.19972  | typeset.1723.25 | 27  | 0.245        | 60           | 682    | 84.75    | 60         | 1212   | 281  | 1.085    | 112526      | 30000 | 30000 | 112526   |
| cet.19972  | typeset.19597.246| 246 | 0.963        | -            | -      | -        | -          | -     | -    | -        | -           | -     | -    | -        |
| cet.19972  | typeset.4391.240| 242 | 0.981        | -            | -      | -        | -          | -     | -    | -        | -           | -     | -    | -        |
| cet.19972  | typeset.5977.33 | 35   | 0.748        | 113          | 300    | 28272    | 53766      | 25358 | 214  | 24.25    | 30000       | 30000 | 30000 | 30000   |
| cet.19972  | typeset.5881.246| 248 | 0.986        | -            | -      | -        | -          | -     | -    | -        | -           | -     | -    | -        |
| Average    |            |     |              |             |       |          | 31.318     | 7848  | 17   | 9.06     | 27086       | 30000 | 30000 | 27086    |
5 Conclusions

This work introduces a more scalable model for the PCMCA, and three models for solving the PCMCA-WT. A proof of complexity shows that the two problems fall inside the NP-hard complexity class. The experimental results show that the benchmark instances for the PCMCA could be solved much more efficiently compared to a previously proposed model. Moreover, the results show that the benchmark instances are much harder to solve under the PCMCA-WT settings. In general, large sized instances with dense precedence graphs are outside the reach of the three PCMCA-WT models proposed, therefore further studies to find better formulations for the problem are needed.

The two problems proposed have applications in designing distribution networks for commodities such as oil and gas, where the network is designed in such a way to avoid passing through a certain location when delivering the commodity to another location. Moreover, the delivery can be scheduled to reach certain locations with higher priority first. The Sequential Ordering Problem is a related problem with a wide range of applications in the domains of scheduling and logistics. The two problems proposed can be seen as relaxations of the Sequential Ordering Problem. Therefore, lower bounds for the SOP based on them (or their linear relaxations) could be derived, together with some new valid inequalities. This in turn could have impact on solving the several real-world problems that can be formulated as a Sequential Ordering Problem.

Acknowledgment

We would like to thank the anonymous reviewers for their constructive comments and suggestions.

A The Path-Based model from [10] fails to detect the violation of precedence constraints in fractional solutions

The MILP model previously proposed for the PCMCA in [10], imposes the precedence relationships by propagating a value along every path with endpoints \( s \) and \( t \) for \((s, t) \in R\) in order to detect a precedence violation. The violation is detected if a non-zero value is propagated from \( t \) down to \( s \). The model sometimes fail to detect a violating path in a fractional solution due to the diminishing propagated values along that path [11]. However, the Set-Based Model proposed in Section 2.2 is able to detect the violation for the same fractional solution. In order to show this, consider the example shown in Figure 8. Referring back to Algorithm 1, we construct the graph \( D_j \) (consisting of all the vertices of the original graph \( G \) excluding the successors of vertex \( s \) in the precedence graph \( P \)), then we compute a minimum \((r, s)\)-cut in \( D_j \), namely cut \( \{(s, 1, 2), \{r\}\} \), which has a value less than 1 in the example. On the other
hand in the model proposed in [10] a value of 0 will be propagated down from $t$ to $s$ failing to detect the violating path. The details about how the value is propagated can be found in [10].

Figure 8: Example of a fractional solution that violates the precedence relationship $(s, t) \in R$, and how the violation cannot be detected by the Path-Based Model. The figure on the left shows the fractional solution with the violating path from $t$ to $s$. The figure in the middle shows the value propagated from $t$ to $s$ next to each vertex, where the violation is not detected using the Path-Based Model because a value of 0 is propagated down to vertex $s$ from $t$. The figure on the right shows how the Set-Based Model is able to detect the violating path, since using Algorithm 1 the graph has a minimum $(r, s)$-cut in $D_j$ with value less than 1.

References

[1] N. Ascheuer, N. Jünger, and G. Reinelt. A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints. Computational Optimization and Applications, 17(1):61–84, 2000.

[2] J. Bang-Jensen. Edge-disjoint in- and out-branchings in tournaments and related path problems. Journal of Combinatorial Theory - Series B, 51(1):1–23, 1991.

[3] K. Bérczi, S. Fujishige, and N. Kamiyama. A linear-time algorithm to find a pair of arc-disjoint spanning in-arborescence and out-arborescence in a directed acyclic graph. Information Processing Letters, 109(23):1227–1231, 2009.

[4] F. Bock. An algorithm to construct a minimum directed spanning tree in a directed network. Developments in Operations Research, pages 29–44, 1971.
[5] S. Böcker and F. Rasche. Towards de novo identification of metabolites by analyzing tandem mass spectra. In Seventh European Conference on Computational Biology and Bioinformatics, ECCB 2008, volume 24, pages 49–55. Springer, 2008.

[6] M.C. Cai, X. Deng, and L. Wang. Minimum k arborescence with bandwidth constraints. Algorithmica, 38(4):529–537, 2004.

[7] F. Carrabs, R. Cerulli, R. Pentangelo, and A. Raiconi. Minimum spanning tree with conflicting edge pairs: a branch-and-cut approach. Annals of Operations Research, 298:65–78, 2021.

[8] Y.J. Chu and T.H. Liu. On the shortest arborescence of a directed graph. Scientia Sinica, 14:1396–1400, 1965.

[9] A. Darmann, U. Pferschy, and J. Schauer. Determining a minimum spanning tree with disjunctive constraints. In Algorithmic Decision Theory, pages 414–423. Springer Berlin Heidelberg, 2009.

[10] M. Dell’Amico, J. Jamal, and R. Montemanni. A mixed integer linear program for a precedence-constrained minimum-cost arborescence problem. In Proc. The 8th International Conference on Industrial Engineering and Applications (Europe), pages 216–221, 2021.

[11] J. Edmonds. Optimum branchings. Journal of research of the national bureau of standards, B 71(4):233–240, 1967.

[12] L.F. Escudero. An inexact algorithm for the sequential ordering problem. European Journal of Operational Research, 37(2):236–249, 1988.

[13] K.P. Eswaran and R.E. Tarjan. Augmentation problems. SIAM Journal on Computing, 5:653–665, 1976.

[14] G. Fertin, J. Fradin, and G. Jean. Algorithmic aspects of the maximum colorful arborescence problem. In Theory and Applications of Models of Computation. TAMC 2017, pages 216–230. Springer, 2017.

[15] M. Fischetti and D. Vigo. A branch-and-cut algorithm for the resource-constrained minimum-weight arborescence problem. Networks: An International Journal, 29(1):55–67, 1997.

[16] A.M. Frieze and T. Tkocz. A randomly weighted minimum arborescence with a random cost constraint. Mathematics of Operations Research, 2021.

[17] G. Galbiati, S. Gualandi, and F. Maffioli. On minimum changeover cost arborescences. Lecture Notes in Computer Science, 6630:112–123, 2011.

[18] M.R. Garey and D.S. Johnson. Computers and intractability: A Guide to the Theory of NP-Completeness. W H Freeman & Co., San Francisco, 1979.
[19] L. Gouveia and M.J. Lopes. The capacitated minimum spanning tree problem: On improved multistar constraints. *European Journal of Operational Research*, 160(1):47–62, 2005.

[20] S.L. Hakimi. Optimum distribution of switching centers in a communication network and some related graph theoretic problems. *Operations Research*, 13(3):462–475, 1965.

[21] J.X. Hao and J.B. Orlin. A faster algorithm for finding the minimum cut in a directed graph. *Journal of Algorithms*, 17(3):424–446, 1994.

[22] V.R Houndji, P. Schaus, M.N. Hounkonnou, and L. Wolsey. The weighted arborescence constraint. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 185–201. Springer, 2017.

[23] R. Kawatra and D. Bricker. Design of a degree-constrained minimal spanning tree with unreliable links and node outage costs. *European Journal of Operational Research*, 156(1):73–82, 2004.

[24] J. Kováč. Complexity of the path avoiding forbidden pairs problem revisited. *Discrete Applied Mathematics*, 161(10):1506–1512, 2013.

[25] J. Li, X. Liu, and J. Lichen. The constrained arborescence augmentation problem in digraphs. In *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pages 1204–1209. IEEE, 2017.

[26] Y. Li, M.T. Thai, F. Wang, and D. Zhu Du. On the construction of a strongly connected broadcast arborescence with bounded transmission delay. *IEEE Transactions on Mobile Computing*, 5(10):1460–1470, 2006.

[27] R. Montemanni, D.H. Smith, and L.M. Gambardella. A heuristic manipulation technique for the sequential ordering problem. *Computers & Operations Research*, 35(12):3931–3944, 2008.

[28] R Montemanni, D.H. Smith, A.E. Rizzoli, and L.M. Gambardella. Sequential ordering problems for crane scheduling in port terminals. *International Journal of Simulation and Process Modelling*, 5(4):348–361, 2009.

[29] V. Morais, B. Gendron, and G.R. Mateus. The p-arborescence star problem: Formulations and exact solution approaches. *Computers & Operations Research*, 102:91–101, 2019.

[30] A.H. Pereira, G.R. Mateus, and S. Urrutia. Branch-and-cut algorithms for the p-arborescence star problem. *International Transactions in Operational Research*, 29(4):2374–2400, 2021.

[31] G. Reinelt. TSPLIB–A travelling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.
[32] W. Shi and C. Su. The rectilinear steiner arborescence problem is NP-complete. *SIAM Journal on Computing*, 35(3):729–740, 2005.

[33] G. Shobaki and J. Jamal. An exact algorithm for the sequential ordering problem and its application to switching energy minimization in compilers. *Computational Optimizations and Applications*, 61(2):343–372, 2015.

[34] L.A. do Carmo Viana and M. Campêlo. Two dependency constrained spanning tree problems. *International Transactions in Operational Research*, 27(2):867–898, 2020.