Advances in Intelligent Traffic and Transportation Systems
M. Shafik (Ed.)

doi:10.3233/ATDE230015

Compact Models for the Precedence-Constrained Minimum-Cost Arborescence Problem

Mauro DELL'AMICO , Jafar JAMAL and Roberto MONTEMANNI [1]
Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia, Italy

Abstract. The Precedence-Constrained Minimum-Cost Arborescence problem, has been recently proposed. The purpose of the precedence constraints, that are enforced between pairs of vertices, is to prevent certain directed paths to appear in the tree that violate a precedence relationship. In this work we introduce a new mixed integer linear programming model that uses a smaller number of variables and constraints to model the precedence relationships compared to those previously appeared in the literature. Furthermore, two models with a polynomial number of variables and constraints are introduced. It is based on a network-flow formulation to model the connectivity of the arborescence. Extensive computational experiments have been run to validate the new models.

Keywords. Combinatorial optimizations, arborescence, mixed integer linear programming, precedence constraints.

1. Introduction

The *Minimum-Cost Arborescence problem* (MCA) is a classical problem in the area of graph theory. Given a root vertex r, the objective of the problem is to find a directed minimum-cost spanning tree rooted at r. Yoeng-Jin Chu and Tseng-Hong Liu [1], and Jack Edmonds [2], independently proposed the first polynomial time algorithm for solving the problem. An efficient implementation of the algorithm was later on proposed by Gabow and Tarjan [3]. The Minimum-Cost Arborescence problem can be formally described as follows. A directed graph $G = (V,A)$ is given where $V = \{1,\dots,n\}$ 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_{ij} associated with every arc $(i, j) \in A$. The objective of the problem 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.

Several variations of the MCA were introduced in the literature, such as the *Resource-Constrained Minimum-Weight Arborescence problem* [4], where finite resources are associated with the vertices of the input graph. The objective of the problem is to find a minimum-cost arborescence, where for each vertex the sum of its outgoing arcs

[1]Corresponding Author, Roberto MONTEMANNI, Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia, Italy; E-mail: roberto.montemanni@unimore.it.

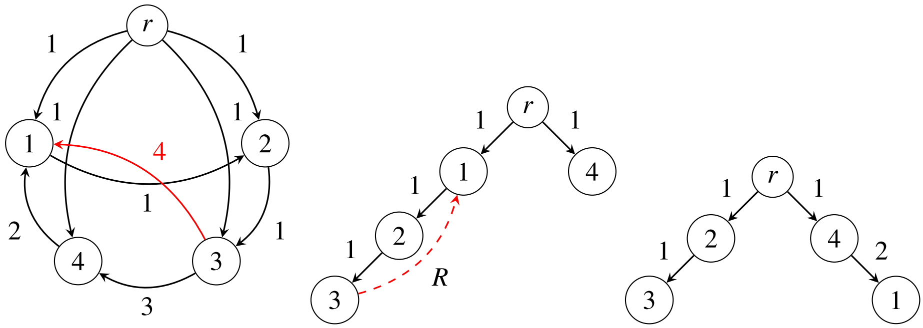

Figure 1. Comparing a MCA and a PCMCA solution. The graph on left is the instance graph with its respective arc costs, with the precedence relationship $(3,1) \in R$ highlighted in red. The graph in the middle shows the optimal MCA, whereas the graph on the right shows the optimal PCMCA. The MCA solution is not a feasible PCMCA solution since vertex 1 precedes vertex 3 on the same directed path and $(3,1) \in R$.

cost is at most equal to the resource associated with that vertex. The *Minimum Spanning Tree problem with Conflict Pairs* [5] is a variation on the *Minimum Spanning Tree Problem*. Given an undirected graph and a set S of *conflicting pairs* of edges, 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. The *Capacitated Minimum-Spanning Tree problem* [6] in which each vertex other than the root is associated with a non-negative integer demand q_j, and an integer Q is given. The problem asks to find a minimum-cost spanning tree rooted at r, such that for any subtree off of the root, the sum of the weights of the vertices in that subtree is at most Q. The *Constrained Arborescence Augmentation problem* [7] that can be described as follows. Given a weighted directed graph $G = (V,A)$, and an arborescence $T = (V,A_r)$ rooted at $r \in V$, the problem asks to find a subset of arcs A' from $A - A_r$ such that there still exists a minimum-cost arborescence in the graph $G' = (V, A_r \cup A' - a)$ for each $a \in A_r$. A relevant problem is the *p-Arborescence Star problem* [8], which can be described as follows. Given a weighted directed graph $G = (V,A)$, a root vertex $r \in V$, and an integer p, the problem asks to find a minimum-cost reverse arborescence rooted at r, that 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 *Maximum Colorful Arborescence problem* [9] can be described as follows. Given a weighted directed acyclic graph, and each vertex having a specified color from a set of colors, the problem asks to find an arborescence of maximum weight, such that no color appears more than once.

A variation of the MCA, named the *Precedence-Constrainted Minimum-Cost Arborescence problem* (PCMCA) was recently proposed in [10], where a set of precedence constraints is included between pairs of vertices as follows. Given a set R of ordered pairs of vertices, then for each precedence $(s,t) \in R$ any path in the arborescence that includes both 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.

A model for the PCMCA has been proposed in [10] based on a well-know formulation for the MCA, and enforces precedence relationships by propagating a value on every path in the arborescence. An alternative model has been discussed in [11].

Figure 1 presents an example that shows the difference between the classic MCA and the PCMCA. The graph with its respective arc costs is shown in the figure on the left, with the precedence relationship $(3,1)$ highlighted in red. The figure in the middle shows a feasible MCA solution with a cost of 4. 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 PCMCA solution with a cost of 5 is shown in the figure on the right.

This work proposes three new mixed integer linear programming (MILP) models for the PCMCA. The first model is based on the model proposed in [10], but uses a smaller number of variables and constraints. The other new model use a polynomial number of constraints to model the connectivity of the solution, instead of the exponential number of constraints used in the previous models. These are the first compact models ever proposed for the problem.

The rest of this paper is organized as follows. Section 2 presents several MILP models for the PCMCA. Section 3 discusses computational results, while conclusions are summarized in Section 4.

2. The Models

In this section we introduce MILP models for the PCMCA. We first start by discussing the precedence-enforcing constraints, and then four MILP models for the PCMCA are introduced, three of which are new.

2.1. Precedence-Enforcing Constraints

The precedence-enforcing constraints adopted throughout this paper, that were first introduced in [10], are based on the following idea. If we consider an arborescence T that includes a simple directed path that starts with t and ends in s, such that $(s,t) \in R$ and $s,t \in V$, then the solution clearly violates a precedence relationship. In order to satisfy the precedence relationship, we need to enforce that no directed path can exist which covers both s and t and visits t before visiting s. This can be achieved by propagating a value down all the paths of the solution starting from t.

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. Let u_j^{st} be a variable associated with every vertex $j \in V$ and $(s,t) \in R$, such that $u_s^{st} = 0$ and $u_t^{st} = 1$ for all $(s,t) \in R$. When the value of u_t^{st} is propagated down every path starting from t, and vertex s is reachable from t, then we are propagating a value of 1 to vertex s and $u_s^{st} \geq 1$ which is unsatisfiable (since $u_s^{st} = 0$), and therefore the solution is infeasible. The set of inequalities for enforcing the precedence relationships are as follows.

$$u_j^{st} - u_i^{st} - x_{ij} \geq -1 \qquad \forall (s,t) \in R, (i,j) \in A \tag{1}$$

$$u_s^{st} = 0 \qquad \forall (s,t) \in R \tag{2}$$

$$u_t^{st} = 1 \qquad \forall (s,t) \in R \tag{3}$$

$$u_j^{st} \geq 0 \qquad \forall (s,t) \in R, j \in V \tag{4}$$

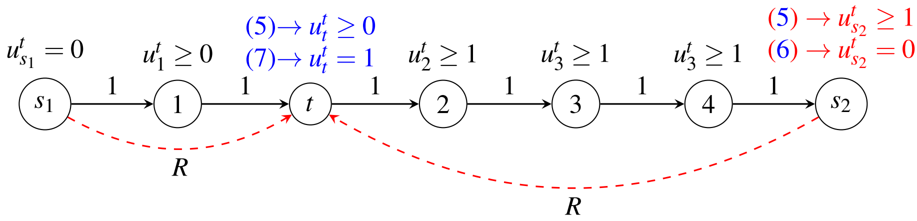

Figure 2. Demonstrating value propagation over feasible and infeasible paths. Each black arc is weighted with its corresponding x_{ij} value. Red dashed arrow shows a precedence relationship $(s,t) \in R$. The u^t_j value is written above each vertex, as for constraint (5). The figure shows a feasible path from s_1 to t, since constraint (5) impose that $u^t_t \geq 0$, and constraint (7) impose that $u^t_t = 1$. The figure also show a violating path from t to s_2, since constraint (5) impose that $u^t_{s_2} \geq 1$, and constraint (6) impose that $u^t_{s_2} = 0$, which means the inequalities are infeasible.

Constraints (1) propagate the value of u^{st}_i down to u^{st}_j if $x_{ij} = 1$ for all $(s,t) \in R$, and $(i,j) \in A$. Constraints (2) and (3) set the values of u^{st}_s and u^{st}_t to 0 and 1 respectively, for all $(s,t) \in R$. Finally, constraints (4) define the domain of the variables. For further details and an example on how the value propagation occurs can be found in [10].

In this paper we observe that, following the same idea of the previous set of constraints, variables u^{st}_j can be instead defined for u^t_j where t is part of a precedence relationship (i.e. $\exists(s,t) \in R$). By doing so, the number of variables and constraints is reduced, and thus solving this model is theoretically faster (the experiments in Section 3 will corroborate this hypothesis). According to these settings, constraints (1)-(4) can be redefined as follows.

$$u^t_j - u^t_i - x_{ij} \geq -1 \qquad \forall t \in V : \exists(s,t) \in R, (i,j) \in A \tag{5}$$

$$u^t_s = 0 \qquad \forall(s,t) \in R \tag{6}$$

$$u^t_t = 1 \qquad \forall t \in V : \exists(s,t) \in R \tag{7}$$

$$u^t_j \geq 0 \qquad \forall t \in V : \exists(s,t) \in R, j \in V \tag{8}$$

Constraints (5) propagate the value of u^t_i down to u^t_j if $x_{ij} = 1$. Constraints (6) and (7) set the values of u^t_s and u^t_t to 0 and 1 respectively, for all $(s,t) \in R$, and $t \in V : \exists(s,t) \in R$. Finally, constraints (8) define the domain of the variables.

Figure 2 shows an example of how the value propagation occurs for the new constraints. The figure contains a feasible path from s_1 to t, and a violating path from t to s_2. The figure demonstrates how a violating path can be detected after redefining the set of variables and constraints.

2.2. Extensive Models

In this section we describe two models that extend a well-known model for the MCA. Each model uses one of the two sets of precedence-enforcing constraints described in Section 2.1. In all the models that follow, let $S \subseteq V \setminus \{r\}$ be a set of vertices. The two models use an exponential set of constraints to enforce the connectivity of the solution (the solution is an arborescence), and a polynomial set of constraints (precedence-enforcing constraints) is used to enforce the precedence relationships in the solution.

2.2.1. U^{st} Model

The following model was originally introduced in [10].

$$\text{minimize} \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{9}$$

$$\text{subject to} \sum_{\substack{(i,j)\in A:\\ i\notin S, j\in S}} x_{ij} \geq 1 \qquad \forall S \subseteq V\backslash\{r\} \tag{10}$$

$$u_j^{st} - u_i^{st} - x_{ij} \geq -1 \qquad \forall (s,t) \in R, (i,j) \in A \tag{11}$$

$$u_s^{st} = 0 \qquad \forall (s,t) \in R \tag{12}$$

$$u_t^{st} = 1 \qquad \forall (s,t) \in R \tag{13}$$

$$u_j^{st} \geq 0 \qquad \forall (s,t) \in R, j \in V \tag{14}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i.j) \in A \tag{15}$$

Constraints (10) are the connectivity constraints that enforce every set of vertices $S \subseteq V\backslash\{r\}$ must be reachable from the root r. The size of the set of constraints (10) is exponential with a size of $O(2^{|A|})$. Constraints (11) propagate the value of u_i^{st} down to u_j^{st} if $x_{ij} = 1$. Constraints (12) and (13) set the values of u_s^{st} and u_t^{st} to 0 and 1 respectively, for all $(s,t) \in R$. Finally, constraints (14) and (15) define the domains of the variables.

2.2.2. U^t Model

The following model is introduced for the first time in this work.

$$\text{minimize} \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{16}$$

$$\text{subject to} \sum_{\substack{(i,j)\in A:\\ i\notin S, j\in S}} x_{ij} \geq 1 \qquad \forall S \subseteq V\backslash\{r\} \tag{17}$$

$$u_j^{t} - u_i^{t} - x_{ij} \geq -1 \qquad \forall t \in V : \exists (s,t) \in R, (i,j) \in A \tag{18}$$

$$u_s^{t} = 0 \qquad \forall (s,t) \in R \tag{19}$$

$$u_t^{t} = 1 \qquad \forall t \in V : \exists (s,t) \in R \tag{20}$$

$$u_j^{t} \geq 0 \qquad \forall t \in V : \exists (s,t) \in R, j \in V \tag{21}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i.j) \in A \tag{22}$$

Constraints (17) are the connectivity constraints which enforce that for any set of vertices $S \subseteq V\backslash\{r\}$, there must be a path which connects r to S. Constraints (18) propagate the value of u_i^t down to u_j^t if $x_{ij} = 1$. Constraints (19) and (20) fix the values of u_s^t and u_t^t to 0 and 1 respectively, for all $(s,t) \in R$, and $t \in V : (\exists (s,t) \in R$. Finally, constraints (21) and (22) define the domains of the variables.

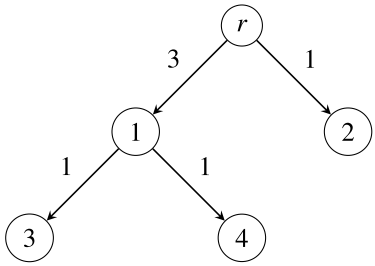

Figure 3. A feasible MCA solution using the Flow-Connectivity constraints. Each arc $(i,j) \in A$ that is part of the solution has a weight equal to the amount of flow passing through that arc. Arcs entering a leaf vertex have a flow value of 1, while arcs entering a non-leaf vertex have a flow value of m, where m is the number of vertices reachable from vertex i.

2.3. *Compact Models*

In this section we propose two models for the PCMCA that are based on a polynomial set of constraints (Flow-Connectivity constraints) which enforce the connectivity of the solution. Each model uses one of the two sets of precedence-enforcing constraints described in Section 2.1.

The Flow-Connectivity constraints that enforce the connectivity of the solution are based on the following idea. If arc $(i,j) \in A$ is part of the solution, then the amount of flow passing through the arc must be equal to the number of vertices reachable from vertex i. This implies that the flow passing through arcs entering a leaf vertex is equal to 1, and the flow passing through arcs entering non-leaf vertices is greater than 1. As there are no arcs entering the root r, and every vertex must be reachable from the root, then the sum of the flow leaving the root must be equal to $|V|-1$, where $|V|$ is the number of vertices in the graph. These set of inequalities will insure that every vertex is reachable from r, and that no feasible solution contains a cycle. An example of the amount of flow passing through every arc in a feasible solution is shown in Figure 3. See [12] for a formal description of the idea.

In all the models that follow, let T be an arborescence rooted at r. Let y_{ij} be a variable associated with every arc $(i,j) \in A$, such that $y_{ij} = 1$ if arc $(i,j) \in T$, and 0 otherwise. Let x_{ij} be a variable associated with every arc $(i,j) \in A$, where x_{ij} is equal to the amount of flow passing through arc $(i,j) \in A$.

2.3.1. *Compact-U^{st} Model*

The new model introduced in this section is based on a network-flow formulation for the MCA problem [12], and adapts the first precedence-enforcing constraints described in Section 2.1.

$$\text{minimize} \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{23}$$

$$\text{subject to} \sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = \begin{cases} -1 & i \neq r \\ |V|-1 & i = r \end{cases} \qquad \forall i \in V \tag{24}$$

$$u_j^{st} - u_i^{st} - y_{ij} \geq -1 \qquad \forall (s,t) \in R, (i,j) \in A \tag{25}$$

$$u_s^{st} = 0 \qquad \forall (s,t) \in R \tag{26}$$

$$u_t^{st} = 1 \qquad \forall (s,t) \in R \tag{27}$$

$$y_{ij} \geq \frac{x_{ij}}{|V|-1} \qquad \forall (i,j) \in A \tag{28}$$

$$u_j^{st} \geq 0 \qquad \forall (s,t) \in R, j \in V \tag{29}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i.j) \in A \tag{30}$$

$$x_{ij} \in \mathbb{Z}^+ \qquad \forall (i.j) \in A \tag{31}$$

Constraints (24) are the flow-connectivity constraints which enforce every vertex to be reachable from the root r, no arcs enter the root r, and that any feasible solution is acyclic. Constraints (25) propagate the value of u_i^{st} down to u_j^{st} if $y_{ij} = 1$. Constraints (26) and (27) set the values of u_s^{st} and u_t^{st} to 0 and 1 respectively, for all $(s,t) \in R$. Constraints (28) impose that the value of y_{ij} must be between 0 and 1, if the value of x_{ij} is nonzero. The value of x_{ij} is divided by $|V|-1$ since (24) $x_{ij} \leq |V|-1$, which would strict the value of y_{ij} to be between 0 and 1. Finally, constraints (29)-(31) define the domain of the variables.

2.3.2. *Compact-U^t Model*

The new model introduced in this section is based on the same network-flow formulation for the MCA problem [12], but adapts the second precedence-enforcing constraints introduced in Section 2.1.

$$\text{minimize} \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{32}$$

$$\text{subject to} \sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = \begin{cases} -1 & i \neq r \\ |V|-1 & i = r \end{cases} \qquad \forall i \in V \tag{33}$$

$$u_j^t - u_i^t - y_{ij} \geq -1 \qquad \forall t \in V : \exists (s,t) \in R, (i,j) \in A \tag{34}$$

$$u_s^t = 0 \qquad \forall (s,t) \in R \tag{35}$$

$$u_t^t = 1 \qquad \forall t \in V : \exists (s,t) \in R \tag{36}$$

$$y_{ij} \geq \frac{x_{ij}}{|V|-1} \qquad \forall (i,j) \in A \tag{37}$$

$$u_j^t \geq 0 \qquad \forall t \in V : \exists (s,t) \in R, j \in V \tag{38}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i.j) \in A \tag{39}$$

$$x_{ij} \in \mathbb{Z}^+ \qquad \forall (i.j) \in A \tag{40}$$

Constraints (33) are the flow-connectivity constraints that enforce the following. For any vertex i, there must be a unique path which connects r to i. Any feasible solution must be acyclic, and that there are no arcs entering the root. Constraints (34) propagate the value of u_i^t down to u_j^t if $y_{ij} = 1$. Constraints (35) and (36) fix the values of u_s^t and u_t^t

to 0 and 1 respectively, for all $(s,t) \in R$ and $t \in V : \exists(s,t) \in R$. Constraints (37) restrict the value of y_{ij} to be between 0 and 1, if the value of x_{ij} is greater than zero. To impose that, the value of x_{ij} is divided by $|V|-1$ since (33) $x_{ij} \leq |V|-1$. Finally, constraints (38)-(40) define the domain of the variables.

3. Experimental Results

The computational experiments for evaluating and comparing the models discussed in Section 2 are introduced in this section. Experiments are based on the benchmark instances of TSPLIB [13], SOPLIB [14] and COMPILERS [15] originally proposed for the Sequential Ordering Problem (SOP) [16]. The benchmark instances are the same instances previously adopted in [10,11].

All the experiments are performed on an Intel i7 processor running at 1.8 GHz with 8 GB of RAM. CPLEX 12.8[2] is used for solving the MILP models. 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 3 hours is set for the computation time for each computational method/instance.

In all the models constraints (10)-(13), and (17)-(20) are added dynamically to the model when they are violated. The same set of constraints are not added dynamically in the two compact models, as preliminary experiments clearly showed an increase in the solution time. A violated constraint (10) or (17), can be detected by computing a minimum-cut in the graph where the weight of an arc is equal to the value of its corresponding x_{ij} variable. On the other hand, a violated constraint (11) or (18), can be detected by finding a violated $s-t$ path using a DFS algorithm. For more details on how violated constraints are detected and dynamically added to the model, please see [10].

The following tables are split as follows. Tables 1-3 report the computational results for the linear relaxation of the five models, and Tables 4-6 report the computational results for the MILP of the four models. In each table we report the following columns where applicable. Column *Name*, *Size*, and z^*, report the name, size, and the cost of the optimal solution of the instance. Column $\rho(P)$ reports the density of the arcs in the precedence graph computed as $\frac{2\cdot|R|}{|V|(|V|-1)}$. For each computational method, we report the following columns. Column *Cuts* reports the number of constraints that are dynamically added to the model. Column *Nodes* reports the number of nodes in the search tree of the B&C algorithm. Column *Gap* reports the optimality gap of the linear relaxation computed as $\frac{UB-Cost}{UB}$, where UB is the value of the objective function of the MILP, or the optimal solution (z^*) for instances that are solved optimally by the model. Column *IP Gap* reports the optimality gap of the MILP and is computed as $\frac{UB-LB}{UB}$, and is only reported for the model that does not optimally solve all instances within the time limit. Finally, column *Time [s]* reports the solution time in seconds.

[2] IBM ILOG CPLEX Optimization Studio: https://www.ibm.com/products/ilog-cplex-optimization-studio

3.1. Computational Results for the Linear Relaxation of the Models

Tables 1-3 show the results of the linear relaxation models of U^{st}, U^{t}, *Compact-*U^{st}, *Compact-*U^{t}, on the three benchmark sets.

Solving the model *Compact-*U^{st} results in an average optimality gap of 1.1%, when considering the instances that are solved optimally by all four models. The model U^{t} results in an average optiamlity gap of 1.4% (a 30% increase), the model U^{st} results in an average optiamlity gap of 2.1% (a 47% increase), and the model *Compact-*U^{t} results in an average optiamlity gap of 3% (a 63% increase). The results might indicate that the model *Compact-*U^{st} is more efficient at solving the instances, however this is not generally the case as will be shown later.

In terms of the number of cuts that are dynamically added to the model's linear relaxation. The solver adds 81 cuts on average when solving the model U^{t}, and adds 182 cuts on average (a 55% increase) when solving the model U^{st}. The smaller number of cuts that are added to the model, combined with the size of the model, indicates that the linear relaxation of the model U^{t} is easier to solve which can be verified by inspecting the solution times. The solver has an average solution time of 13 seconds when solving the model U^{t}, an average solution time of 15 seconds (a 13% increase) when solving the model *Compact-*U^{t}, an average solution time of 18 seconds (a 28% increase) when solving the model *Compact-*U^{st}, and an average solution time of 55 seconds (a 76% increase) when solving the model U^{st}. The smaller average solution time of the solver using the model U^{t} shows that the model's linear relaxation is relatively easier to solve.

The results of the solution time and optimality gap, indicate that the two models U^{t} and *Compact-*U^{t}, are the best two models out of the four models introduced. The following observations can be derived from the results. The model U^{t} finds a smaller optimality gap for 40 instances, their densities are in the range $[0.046, 0.995]$, with 83% of those instances are in the range $[0.046, 0.840]$. On the other hand, the model *Compact-*U^{t} finds a smaller optimality gap for 19 instances, their densities are in the range $[0.075, 0.980]$, with 74% of those instances are in the range $[0.847, 0.980]$. Moreover, the model U^{t} solves the linear relaxation model faster for 82 instances, their densities are in the range $[0.008, 0.997]$, with 87% of those instances are in the range $[0.008, 0.633]$. On the other hand, the model *Compact-*U^{t} solves the linear relaxation model faster for 31 instances, their densities are in the range $[0.008, 0.986]$, with 71% of those instances are in the range $[0.740, 0.986]$. From these observations, it can be concluded that the model *Compact-*U^{t} generally performs better on instances with very dense precedence graph, when solving the linear relaxation model. See Tables 1-3 for the complete computational results.

3.2. Computational Results for the MILP

Tables 4-6 show the results of the MILP models for U^{st}, U^{t}, *Compact-*U^{st}, *Compact-*U^{t}, on the three benchmark sets.

In terms of the number of cuts that are dynamically added to the model, the solver dynamically adds 211 cuts on average when solving the model U^{t}, and 4755 cuts on average when solving the model U^{st} (a 95% increase). This further shows that the solver is more efficient at solving the problem using the model U^{t} as the size of the MILP is relatively smaller.

Table 1. Computational results for the linear relaxation of the models on TSPLIB instances.

Instance				Extensive						Compact			
				U^{st} [10]			U^t			U^{st}		U^t	
Name	Size	$\rho(p)$	z^*	Cuts	Time [s]	Gap	Cuts	Time [s]	Gap	Time [s]	Gap	Time [s]	Gap
br17.10	18	0.314	25	25	0.032	0.000	17	0.047	0.000	0.270	0.140	0.160	0.412
br17.12	18	0.359	25	25	0.047	0.000	19	0.060	0.000	0.230	0.061	0.080	0.454
ESC07	9	0.611	1531	12	0.031	0.000	14	0.047	0.000	0.062	0.000	0.062	0.000
ESC11	13	0.359	1752	5	0.031	0.000	3	0.032	0.000	0.078	0.000	0.110	0.027
ESC12	14	0.396	1138	3	0.016	0.000	3	0.016	0.000	0.094	0.000	0.141	0.000
ESC25	27	0.177	1041	14	0.062	0.000	22	0.078	0.000	0.078	0.000	0.156	0.000
ESC47	49	0.108	703	102	0.484	0.003	105	0.253	0.000	0.812	0.000	1.062	0.000
ESC63	65	0.173	56	14	0.329	0.000	270	2.484	0.000	2.328	0.000	0.609	0.000
ESC78	80	0.139	502	12	0.094	0.000	3	0.050	0.000	1.594	0.000	3.406	0.000
ft53.1	54	0.082	3917	123	1.172	0.004	72	0.760	0.003	0.630	0.024	0.700	0.023
ft53.2	54	0.094	3978	3674	0.281	0.076	49	0.250	0.004	0.580	0.021	0.630	0.021
ft53.3	54	0.225	4242	77	1.890	0.056	51	0.980	0.015	0.560	0.017	0.670	0.021
ft53.4	54	0.604	4882	11	0.156	0.027	8	0.203	0.000	1.032	0.000	1.015	0.000
ft70.1	71	0.036	32846	144	2.891	0.000	136	2.813	0.000	5.375	0.000	4.391	0.000
ft70.2	71	0.075	32930	158	2.985	0.000	129	2.750	0.000	8.562	0.000	9.093	0.000
ft70.3	71	0.142	33431	45	0.750	0.024	131	3.550	0.003	1.500	0.005	1.580	0.004
ft70.4	71	0.589	35179	217	13.015	0.006	21	0.110	0.026	1.060	0.000	13.906	0.000
rbg048a	50	0.444	204	3	0.047	0.000	4	0.047	0.000	0.360	0.000	0.406	0.000
rbg050c	52	0.459	191	35	0.313	0.000	16	0.141	0.000	1.344	0.000	1.609	0.000
rbg109	111	0.909	256	47	11.578	0.000	6	0.109	0.000	0.797	0.000	1.125	0.000
rbg150a	152	0.927	373	6	2.485	0.000	7	0.297	0.000	4.641	0.000	6.734	0.000
rbg174a	176	0.929	365	56	29.610	0.003	32	1.047	0.000	2.750	0.019	3.970	0.019
rbg253a	255	0.948	375	2	13.985	0.000	9	1.094	0.000	8.094	0.000	9.500	0.000
rbg323a	325	0.928	754	16	1.547	0.000	5	1.391	0.000	29.750	0.000	23.890	0.000
rbg341a	343	0.937	610	395	23.344	0.033	30	15.530	0.011	17.390	0.010	12.890	0.010
rbg358a	360	0.886	595	4	0.312	0.000	26	21.343	0.000	40.515	0.000	40.750	0.000
rbg378a	380	0.894	559	464	16.079	0.039	29	31.250	0.000	15.750	0.007	13.860	0.009
kro124p.1	101	0.046	32597	39	0.734	0.060	222	0.520	0.001	11.760	0.026	12.080	0.026
kro124p.2	101	0.053	32851	23	0.578	0.069	228	3.030	0.006	10.300	0.021	12.110	0.023
kro124p.3	101	0.092	33779	225	8.672	0.027	198	6.980	0.023	437.890	0.133	11.170	0.219
kro124p.4	101	0.496	37124	277	41.828	0.014	143	5.926	0.011	6.450	0.028	5.660	0.026
p43.1	44	0.101	2720	112	0.594	0.127	384	5.125	0.000	1.060	0.048	1.060	0.048
p43.2	44	0.126	2720	237	1.016	0.084	471	2.062	0.000	1.140	0.051	1.280	0.036
p43.3	44	0.191	2720	111	0.547	0.144	270	1.531	0.000	1.640	0.033	0.970	0.577
p43.4	44	0.164	2820	132	1.218	0.087	324	5.060	0.000	0.720	0.009	0.750	0.007
prob.100	100	0.048	650	282	11.766	0.013	249	12.141	0.011	5.160	0.011	7.280	0.011
prob.42	42	0.116	143	29	0.125	0.000	32	0.125	0.000	0.407	0.000	0.328	0.000
ry48p.1	49	0.091	13095	118	0.828	0.009	81	1.192	0.009	0.720	0.012	1.000	0.017
ry48p.2	49	0.103	13103	128	1.031	0.006	90	1.110	0.005	1.260	0.011	6.560	0.024
ry48p.3	49	0.193	13886	183	2.109	0.037	121	1.095	0.034	1.230	0.045	5.000	0.051
ry48p.4	49	0.588	15340	65	2.531	0.072	93	0.981	0.034	0.810	0.046	3.300	0.054
Average				187	4.808	0.025	101	3.259	0.005	15.287	0.019	5.392	0.052

Considering the instances that are solved optimally by all four models, the efficiency of the model U^t can be shown by comparing the solution times of the solver when solving each model. The solver has an average solution time of 23 seconds when solving the model U^t, an average solution time of 131 seconds (a 82% increase) when solving the model *Compact-*U^t, an average solution time of 173 seconds (a 86% increase) when solving the model *Compact-*U^{st}, and an average solution time of 221 seconds (a 90% increase) when solving the model U^{st}. In terms of the number of nodes generated in the search tree by the solver, and considering the instances that are solved optimally by all models. The solver generates, 211 nodes on average when solving the model U^t, 1008 nodes on average (a 79% increase) when solving the model *Compact-*U^{st}, 1468 nodes on average (a 86% increase) when solving the model *Compact-*U^t, and 3387 nodes on

Table 2. Computational results for the linear relaxation of the models on SOPLIB instances.

Instance				Extensive						Compact			
				U^{st} [10]			U^t			U^{st}		U^t	
Name	Size	$\rho(p)$	z^*	Cuts	Time [s]	Gap	Cuts	Time [s]	Gap	Time [s]	Gap	Time [s]	Gap
R.200.100.1	200	0.020	29	1	0.219	0.000	13	1.843	0.000	4.719	0.000	4.578	0.000
R.200.100.15	200	0.847	454	1274	3235.391	0.057	168	13.365	0.048	17.690	0.019	4.050	0.020
R.200.100.30	200	0.957	529	27	12.922	0.112	43	3.051	0.029	1.840	0.017	1.630	0.022
R.200.100.60	200	0.991	6018	0	3.593	0.000	0	0.157	0.000	1.141	0.000	1.016	0.000
R.200.1000.1	200	0.020	887	0	0.203	0.000	2	0.625	0.000	9.562	0.000	9.875	0.000
R.200.1000.15	200	0.876	5891	170	203.234	0.043	30	3.750	0.049	3.280	0.010	2.640	0.010
R.200.1000.30	200	0.958	7653	45	56.000	0.000	7	0.953	0.000	1.520	0.001	1.830	0.002
R.200.1000.60	200	0.989	6666	0	3.797	0.000	0	0.157	0.000	1.469	0.000	1.579	0.000
R.300.100.1	300	0.013	13	0	0.500	0.000	22	5.313	0.000	10.515	0.000	3.360	0.000
R.300.100.15	300	0.905	575	149	3.985	0.103	228	46.330	0.036	9.330	0.026	9.990	0.026
R.300.100.30	300	0.970	756	57	1.672	0.000	8	1.313	0.000	3.630	0.007	4.470	0.005
R.300.100.60	300	0.994	708	57	1.531	0.000	11	1.718	0.000	19.672	0.000	18.656	0.000
R.300.1000.1	300	0.013	715	69	10.546	0.000	69	10.343	0.000	58.562	0.000	62.375	0.000
R.300.1000.15	300	0.905	6660	65	0.812	0.060	75	15.082	0.009	6.020	0.006	7.410	0.006
R.300.1000.30	300	0.965	8693	11	1.531	0.000	1	1.016	0.000	7.328	0.000	10.718	0.000
R.300.1000.60	300	0.994	7678	4	23.234	0.000	0	0.469	0.000	10.672	0.000	11.109	0.000
R.400.100.1	400	0.010	6	1	0.391	0.000	7	1.142	0.000	18.093	0.000	19.515	0.000
R.400.100.15	400	0.927	699	22	0.328	0.108	62	26.167	0.033	24.470	0.011	10.450	0.014
R.400.100.30	400	0.978	712	58	10.156	0.000	2	8.078	0.000	21.110	0.000	24.875	0.000
R.400.100.60	400	0.996	557	2	0.219	0.000	1	0.181	0.000	10.265	0.000	11.031	0.000
R.400.1000.1	400	0.010	780	13	6.734	0.000	17	11.672	0.000	12.953	0.000	13.140	0.000
R.400.1000.15	400	0.930	7382	27	0.625	0.085	42	31.662	0.023	8.380	0.019	7.260	0.019
R.400.1000.30	400	0.977	9368	541	34.531	0.011	36	13.551	0.025	8.190	0.021	11.750	0.023
R.400.1000.60	400	0.995	7167	44	2.016	0.000	3	1.453	0.000	33.078	0.000	36.500	0.026
R.500.100.1	500	0.008	3	579	217.172	0.000	172	35.726	0.000	37.921	0.000	34.469	0.000
R.500.100.15	500	0.945	860	20	1.016	0.085	51	35.192	0.041	31.550	0.017	17.050	0.016
R.500.100.30	500	0.980	710	333	14.453	0.031	16	23.592	0.006	68.609	0.000	61.734	0.000
R.500.100.60	500	0.996	566	0	0.687	0.000	0	0.625	0.000	43.265	0.000	43.234	0.000
R.500.1000.1	500	0.008	297	0	0.609	0.000	0	0.611	0.000	17.469	0.000	17.250	0.000
R.500.1000.15	500	0.940	8063	648	82.015	0.000	37	48.410	0.006	14.520	0.001	88.218	0.000
R.500.1000.30	500	0.981	9409	28	11.141	0.000	2	7.985	0.000	30.516	0.000	32.954	0.000
R.500.1000.60	500	0.996	6163	0	0.671	0.000	0	0.634	0.000	36.984	0.000	40.718	0.000
R.600.100.1	600	0.007	1	858	659.156	0.000	1262	840.446	0.000	81.797	0.000	81.532	0.000
R.600.100.15	600	0.950	568	387	31.516	0.000	10	12.750	0.000	58.094	0.000	63.406	0.000
R.600.100.30	600	0.985	776	263	13.484	0.017	0	15.578	0.000	38.810	0.007	43.580	0.007
R.600.100.60	600	0.997	538	0	0.359	0.000	0	0.265	0.000	24.453	0.000	24.672	0.000
R.600.1000.1	600	0.007	322	0	0.844	0.000	0	0.735	0.000	32.016	0.000	33.969	0.000
R.600.1000.15	600	0.945	9763	216	17.984	0.022	20	55.640	0.000	42.063	0.000	66.703	0.000
R.600.1000.30	600	0.984	9497	14	7.219	0.000	3	3.714	0.000	48.735	0.000	97.906	0.000
R.600.1000.60	600	0.997	6915	1	0.406	0.000	1	0.125	0.000	33.422	0.000	34.578	0.000
R.700.100.1	700	0.006	2	0	1.250	0.000	0	1.152	0.000	117.281	0.000	35.234	0.000
R.700.100.15	700	0.957	675	106	41.000	0.000	7	12.766	0.000	176.047	0.000	86.688	0.000
R.700.100.30	700	0.987	590	0	3.984	0.000	0	2.813	0.000	61.203	0.000	74.266	0.000
R.700.100.60	700	0.997	383	0	0.500	0.000	0	0.435	0.000	46.437	0.000	45.469	0.000
R.700.1000.1	700	0.006	611	3	1.625	0.000	7	5.156	0.000	57.156	0.000	61.797	0.000
R.700.1000.15	700	0.956	2792	3	1.500	0.000	1	1.156	0.000	28.609	0.000	35.828	0.000
R.700.1000.30	700	0.986	2658	0	0.360	0.000	0	0.259	0.000	20.078	0.000	23.500	0.000
R.700.1000.60	700	0.997	1913	0	0.515	0.000	0	0.315	0.000	55.750	0.000	60.719	0.000
Average				127	98.409	0.015	51	27.197	0.006	31.381	0.003	31.152	0.004

average (a 94% increase) when solving the model U^{st}.

The results of the solution time and the number of nodes generated by the solver, also indicate that the two models U^t and *Compact*-U^t, are the best two models out of the four models introduced, as they balance computation time and memory usage. The following observations can be derived from the results. The model U^t solves the MILP model faster for 106 instances, their densities are in the range $[0.008, 0.996]$, spread uniformly across this range. On the other hand, the model *Compact*-U^t solves the MILP

Table 3. Computational results for the linear relaxation of the models on COMPILERS instances.

Instance				Extensive						Compact			
				U^{st} [10]			U^t			U^{st}		U^t	
Name	Size	$\rho(p)$	z^*	Cuts	Time [s]	Gap	Cuts	Time [s]	Gap	Time [s]	Gap	Time [s]	Gap
gsm.153.124	126	0.970	185	180	0.578	0.000	26	0.297	0.000	1.610	0.000	1.125	0.029
gsm.444.350	353	0.990	1542	2	0.078	0.000	12	0.375	0.000	0.594	0.000	0.454	0.000
gsm.462.77	79	0.840	292	48	3.422	0.000	31	0.296	0.000	2.109	0.000	2.234	0.003
jpeg.1483.25	27	0.484	71	50	0.234	0.000	48	0.082	0.000	1.156	0.000	0.190	0.017
jpeg.3184.107	109	0.887	411	96	14.640	0.006	56	0.660	0.004	1.391	0.000	0.500	0.007
jpeg.3195.85	87	0.740	13	2548	278.844	0.385	661	12.312	0.385	3.390	0.088	3.610	0.000
jpeg.3198.93	95	0.752	140	2093	252.734	0.029	647	13.280	0.021	17.390	0.000	28.203	0.000
jpeg.3203.135	137	0.897	507	104	47.578	0.004	88	1.751	0.004	1.090	0.016	0.980	0.017
jpeg.3740.15	17	0.257	33	72	1.782	0.030	24	0.067	0.030	0.220	0.045	0.240	0.046
jpeg.4154.36	38	0.633	74	52	0.641	0.050	42	0.160	0.051	0.310	0.041	0.220	0.041
jpeg.4753.54	56	0.769	146	154	2.766	0.007	67	0.342	0.010	4.391	0.000	2.641	0.000
susan.248.197	199	0.939	588	75	76.329	0.003	54	2.451	0.002	0.750	0.008	0.610	0.008
susan.260.158	160	0.916	472	20	12.156	0.017	75	2.080	0.006	1.340	0.010	0.780	0.010
susan.343.182	184	0.936	468	201	194.188	0.010	123	6.842	0.009	1.520	0.007	1.470	0.006
typeset.10192.123	125	0.744	241	14	4.859	0.103	103	2.890	0.016	2.740	0.037	1.910	0.041
typeset.10835.26	28	0.349	60	8	0.063	0.000	9	0.047	0.000	0.079	0.000	0.078	0.000
typeset.12395.43	45	0.518	125	37	0.531	0.005	28	0.125	0.000	1.250	0.000	0.280	0.013
typeset.15087.23	25	0.557	89	84	0.297	0.011	25	0.030	0.011	0.170	0.011	0.190	0.011
typeset.15577.36	38	0.555	93	7	0.031	0.000	6	0.062	0.000	0.516	0.000	0.468	0.000
typeset.16000.68	70	0.658	67	787	21.891	0.000	425	17.725	0.000	4.530	0.093	13.560	0.093
typeset.1723.25	27	0.245	54	88	0.203	0.056	52	0.160	0.056	0.340	0.092	0.440	0.989
typeset.19972.246	248	0.993	979	14	0.110	0.000	7	0.069	0.000	0.234	0.000	0.250	0.000
typeset.4391.240	242	0.981	837	131	378.172	0.001	95	3.782	0.000	5.156	0.000	6.359	0.000
typeset.4597.45	47	0.493	133	16	0.437	0.000	16	0.079	0.000	0.234	0.000	0.297	0.000
typeset.4724.433	435	0.995	1819	374	4.000	0.000	68	1.823	0.000	2.030	0.003	1.980	0.003
typeset.5797.33	35	0.748	93	85	0.234	0.000	37	0.125	0.000	0.407	0.000	0.297	0.000
typeset.5881.246	248	0.986	979	49	191.813	0.003	55	1.885	0.003	0.530	0.003	0.810	0.003
Average				274	55.134	0.027	107	2.585	0.023	2.055	0.017	2.599	0.049

model faster for 8 instances, their densities are in the range $[0.008, 0.752]$, with 75% of those instances are in the range $[0.008, 0.173]$. Moreover, The model U^t generates less nodes for 41 instances, their densities are in the range $[0.046, 0.997]$, where 40% of those instances have a density in the range $[0.046, 0.359]$, and 56% of those instances have a density in the range $[0.847, 0.997]$. On the other hand, the model *Compact*-U^t generates less nodes for 16 instances, their densities are in the range $[0.048, 0.980]$, with 62% of those instances are in the range $[0.484, 0.769]$. From these observations, it can be concluded that the model *Compact*-U^t generally performs better on instances with medium density precedence graph, when solving the MILP model. It should be noted that another advantage of the model *Compact*-U^t, is that it is easier to implement, as there is no need to handle dynamic constraints in some contexts. See Tables 4-6 for the complete computational results.

4. Conclusions

This work introduced a formulation of the precedence-enforcing constraints that uses a smaller number of variables and constraints compared to previous work in the literature. Moreover, a formulation for the PCMCA is introduced that uses a polynomial set of constraints to model both the connectivity of the solution and the precedence relationships.

The computational results show that reducing the number of variables and constraints that are used to model the precedence relationships achieves a 77% decrease on

Table 4. Computational results for MILP models on TSPLIB instances.

Instance				Extensive						Compact					
				U^{st} [10]			U^t			U^{st}			U^t		
Name	Size	$\rho(p)$	z^*	Nodes	Cuts	Time [s]	Nodes	Cuts	Time	Nodes	Time [s]	IP Gap	Nodes	Time [s]	IP Gap
br17.10	18	0.314	25	3	26	0.060	0	17	0.047	1024	0.922	-	483	0.938	-
br17.12	18	0.359	25	3	26	0.063	15	20	0.094	44	1.125	-	516	1.563	-
ESC07	9	0.611	1531	0	12	0.031	0	14	0.047	0	0.062	-	0	0.047	-
ESC11	13	0.359	1752	0	5	0.031	0	3	0.032	0	0.078	-	6	0.078	-
ESC12	14	0.396	1138	0	3	0.016	0	3	0.016	0	0.094	-	0	0.079	-
ESC25	27	0.177	1041	0	14	0.062	0	22	0.078	0	0.078	-	0	0.093	-
ESC47	49	0.108	703	5	106	0.469	0	105	0.253	0	0.812	-	0	0.890	-
ESC63	65	0.173	56	0	14	0.329	0	270	2.484	0	2.328	-	0	0.985	-
ESC78	80	0.139	502	0	12	0.094	0	3	0.050	0	1.594	-	0	1.703	-
ft53.1	54	0.082	3917	7	129	1.172	7	82	0.812	966	10.282	-	1477	10.547	-
ft53.2	54	0.094	3978	104	302	0.688	16	55	0.297	481	17.094	-	1580	19.594	-
ft53.3	54	0.225	4242	122	416	2.547	33	82	1.156	616	10.469	-	361	10.093	-
ft53.4	54	0.604	4882	9	46	0.250	0	8	0.203	0	1.032	-	0	1.015	-
ft70.1	71	0.036	32846	1	144	2.828	0	136	2.813	0	5.375	-	0	4.391	-
ft70.2	71	0.075	32930	2	160	3.016	2	138	2.781	0	8.562	-	0	9.093	-
ft70.3	71	0.142	33431	954	3061	63.171	280	288	6.531	547	113.250	-	462	59.719	-
ft70.4	71	0.589	35179	53	457	13.438	37	217	1.515	6	7.860	-	0	13.906	-
rbg048a	50	0.444	204	0	3	0.047	0	4	0.047	0	0.360	-	0	0.406	-
rbg050c	52	0.459	191	0	35	0.313	0	16	0.141	0	1.344	-	0	1.609	-
rbg109	111	0.909	256	0	47	11.578	0	6	0.109	0	0.797	-	0	1.125	-
rbg150a	152	0.927	373	0	6	2.485	0	7	0.297	0	4.641	-	0	6.734	-
rbg174a	176	0.929	365	2	57	29.609	0	32	1.047	438	20.312	-	16	25.391	-
rbg253a	255	0.948	375	0	2	13.985	0	9	1.094	0	8.094	-	0	9.500	-
rbg323a	325	0.928	754	0	16	1.547	0	5	1.391	0	29.750	-	0	23.890	-
rbg341a	343	0.937	610	376	11958	278.859	60	40	23.547	54	373.516	-	385	499.281	-
rbg358a	360	0.886	595	0	4	0.312	0	26	21.343	0	40.515	-	0	40.750	-
rbg378a	380	0.894	559	543	4390	178.515	0	29	31.250	523	615.093	-	510	181.703	-
kro124p.1	101	0.046	32597	47	312	1.844	6	234	0.703	500	44.594	-	504	43.313	-
kro124p.2	101	0.053	32851	1433	801	11.203	1052	416	9.859	1459	810.547	-	1263	909.469	-
kro124p.3	101	0.092	33779	258648	4253	6599.140	187246	1552	4625.841	821	-	0.105	2966	-	0.094
kro124p.4	101	0.496	37124	198	981	59.359	206	226	8.157	521	212.687	-	994	207.360	-
p43.1	44	0.101	2720	238	1202	4.203	0	384	5.125	487	4.297	-	487	4.250	-
p43.2	44	0.126	2720	119	589	1.781	0	471	2.062	491	9.859	-	496	15.281	-
p43.3	44	0.191	2720	283	1113	2.829	0	270	1.531	508	151.453	-	610	112.109	-
p43.4	44	0.164	2820	198	926	3.516	99	435	5.516	1004	26.844	-	2016	25.437	-
prob.100	100	0.048	650	1428	2555	36.594	3633	2401	119.406	2265	389.265	-	2826	60.641	-
prob.42	42	0.116	143	0	29	0.125	0	32	0.125	0	0.407	-	0	0.328	-
ry48p.1	49	0.091	13095	879	380	1.656	880	183	2.594	1338	85.531	-	1523	40.375	-
ry48p.2	49	0.103	13103	220	450	1.593	37	130	1.296	578	31.500	-	617	60.641	-
ry48p.3	49	0.193	13886	123233	2793	638.344	68658	1006	567.140	58651	3421.736	-	33288	2111.234	-
ry48p.4	49	0.588	15340	8610	1034	24.156	1830	286	4.578	1269	357.203	-	718	79.484	-
Average				9700	948	194.923	6441	236	133.010	1819	170.534		1320	114.876	

average in terms of solution time, and a 56% decrease on average in the number of nodes generated in the search tree generated while solving the model.

The computational results also show that the *Extensive* models are faster on average compared to their *Compact* form, even though they contain an exponential set of constraints. The computational results have also shown that the newly proposed model U^t is the most effective model compared to the models introduced in this work at solving PCMCA instances in terms of both computation time and memory usage. However, the model *Compact*-U^t is generally more effective on specific subset of the instances.

References

[1] Chu YJ, Liu TH. On the shortest arborescence of a directed graph. Scientia Sinica. 1965;14:1396-400.

[2] Edmonds J. Optimum branchings. Journal of research of the national bureau of standards. 1967;B 71(4):233-40.

[3] Gabow HN, Galil Z, Spencer T, Tarjan RE. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. Combinatorica. 1986;6(2):109-22.

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

Table 5. Computational results for MILP models on SOPLIB instances.

Instance				Extensive						Compact			
				U^{st} [10]			U^{t}			U^{st}		U^{t}	
Name	Size	$\rho(p)$	z^*	Nodes	Cuts	Time [s]	Nodes	Cuts	Time	Nodes	Time [s]	Nodes	Time [s]
R.200.100.1	200	0.020	29	0	1	0.219	0	13	1.843	0	4.719	0	4.578
R.200.100.15	200	0.847	454	382	3314	4034.859	269	729	29.531	1118	677.000	2608	338.312
R.200.100.30	200	0.957	529	59	142	54.828	28	68	3.656	28	19.922	28	17.562
R.200.100.60	200	0.991	6018	0	0	3.593	0	0	0.157	0	1.000	0	1.016
R.200.1000.1	200	0.020	887	0	0	0.203	0	2	0.625	0	9.562	0	9.875
R.200.1000.15	200	0.876	5891	132	731	329.313	74	328	9.360	1156	52.266	997	44.719
R.200.1000.30	200	0.958	7653	2	46	57.141	0	7	0.953	43	6.547	29	8.469
R.200.1000.60	200	0.989	6666	0	0	3.797	0	0	0.157	0	1.469	0	1.579
R.300.100.1	300	0.013	13	0	0	0.500	0	22	5.313	0	10.515	0	3.360
R.300.100.15	300	0.905	575	87859	119299	2220.656	476	1681	114.484	506	199.688	552	208.141
R.300.100.30	300	0.970	756	0	57	1.672	0	8	1.313	481	13.953	61	20.375
R.300.100.60	300	0.994	708	2	57	2.469	0	11	1.718	0	19.672	0	18.656
R.300.1000.1	300	0.013	715	0	69	10.546	0	69	10.343	0	58.562	0	62.375
R.300.1000.15	300	0.905	6660	3304	4165	91.938	66	95	19.203	57	83.328	194	124.828
R.300.1000.30	300	0.965	8693	0	11	1.531	0	1	1.016	0	7.328	0	10.718
R.300.1000.60	300	0.994	7678	0	4	23.234	0	0	0.469	0	10.672	0	11.109
R.400.100.1	400	0.010	6	0	1	0.391	0	7	1.142	0	18.093	0	19.515
R.400.100.15	400	0.927	699	52858	105054	2021.813	499	1979	161.828	1582	1895.359	564	1598.750
R.400.100.30	400	0.978	712	0	58	10.156	0	2	8.078	0	21.110	0	24.875
R.400.100.60	400	0.996	557	0	2	0.219	0	1	0.181	0	10.265	0	11.031
R.400.1000.1	400	0.010	780	0	13	6.734	0	17	11.672	0	12.953	0	13.140
R.400.1000.15	400	0.930	7382	56018	170012	8935.188	328	153	75.516	1199	1265.079	608	1296.484
R.400.1000.30	400	0.977	9368	4797	5545	209.593	58	130	20.547	1979	174.156	2308	417.172
R.400.1000.60	400	0.995	7167	0	44	2.016	0	3	1.453	0	33.078	0	36.500
R.500.100.1	500	0.008	3	0	579	217.172	0	172	35.726	0	37.921	0	34.469
R.500.100.15	500	0.945	860	9879	8120	443.125	186	279	104.687	516	375.407	520	468.468
R.500.100.30	500	0.980	710	11490	19359	696.922	9	16	25.969	0	68.609	0	61.734
R.500.100.60	500	0.996	566	0	0	0.687	0	0	0.625	0	43.265	0	43.234
R.500.1000.1	500	0.008	297	0	0	0.609	0	0	0.611	0	17.469	0	17.250
R.500.1000.15	500	0.940	8063	57	819	100.640	7	43	54.422	289	109.625	0	88.218
R.500.1000.30	500	0.981	9409	0	28	11.141	0	2	7.985	0	30.516	0	32.954
R.500.1000.60	500	0.996	6163	0	0	0.671	0	0	0.634	0	36.984	0	40.718
R.600.100.1	600	0.007	1	0	858	659.156	0	1262	840.446	0	81.797	0	81.532
R.600.100.15	600	0.950	568	1	387	34.985	0	10	12.750	0	58.094	0	63.406
R.600.100.30	600	0.985	776	659	8656	298.109	0	13	15.578	1059	152.672	116	180.375
R.600.100.60	600	0.997	538	0	0	0.359	0	0	0.265	0	24.453	0	24.672
R.600.1000.1	600	0.007	322	0	0	0.844	0	0	0.735	0	32.016	0	33.969
R.600.1000.15	600	0.945	9763	31	2092	159.515	4	20	56.547	0	42.063	0	66.703
R.600.1000.30	600	0.984	9497	0	14	7.219	0	3	3.714	0	77.047	0	97.906
R.600.1000.60	600	0.997	6915	0	1	0.406	0	1	0.125	0	33.422	105	34.578
R.700.100.1	700	0.006	2	0	0	1.250	0	0	1.152	0	117.281	0	116.328
R.700.100.15	700	0.957	675	0	106	41.000	0	7	12.766	0	176.047	0	86.688
R.700.100.30	700	0.987	590	0	0	3.984	0	0	2.813	0	61.203	0	74.266
R.700.100.60	700	0.997	383	0	0	0.500	0	0	0.435	0	46.437	0	45.469
R.700.1000.1	700	0.006	611	0	3	1.625	0	7	5.156	0	57.156	0	61.797
R.700.1000.15	700	0.956	2792	0	3	1.500	0	1	5.156	0	28.609	0	35.828
R.700.1000.30	700	0.986	2658	0	0	0.360	0	0	0.259	0	20.078	0	23.500
R.700.1000.60	700	0.997	1913	0	0	0.515	0	0	0.315	0	55.750	0	60.719
Average				4740	9368	431.352	42	149	34.780	209	133.130	181	128.707

[5] Carrabs F, Cerulli R, Pentangelo R, Raiconi A. Minimum spanning tree with conflicting edge pairs: a branch-and-cut approach. Annals of Operations Research. 2021;298(1):65-78.

[6] Gouveia L, Lopes MJ. The capacitated minimum spanning tree problem: On improved multistar constraints. European Journal of Operational Research. 2005;160(1):47-62.

[7] Li J, Liu X, Lichen J. The constrained arborescence augmentation problem in digraphs. In: 2017 3rd IEEE International Conference on Computer and Communications (ICCC). IEEE; 2017. p. 1204-9.

[8] Pereira AH, Mateus GR, Urrutia S. Branch-and-cut algorithms for the-arborescence star problem. International Transactions in Operational Research. 2022;29(4):2374-400.

[9] Fertin G, Fradin J, Jean G. Algorithmic aspects of the maximum colorful arborescence problem. In: International Conference on Theory and Applications of Models of Computation. Springer; 2017. p.

Table 6. Computational results for MILP models on COMPILERS instances.

Instance				Extensive						Compact			
				U^{st} [10]			U^{t}			U^{st}		U^{t}	
Name	Size	$\rho(p)$	z^*	Nodes	Cuts	Time [s]	Nodes	Cuts	Time	Nodes	Time [s]	Nodes	Time [s]
gsm.153.124	126	0.970	185	0	180	0.578	0	26	0.297	0	1.610	0	1.125
gsm.444.350	353	0.990	1542	0	2	0.078	0	12	0.375	0	0.594	0	0.454
gsm.462.77	79	0.840	292	17	79	4.047	0	31	0.296	0	2.109	0	2.234
jpeg.1483.25	27	0.484	71	43	197	0.266	16	55	0.125	0	1.156	8	1.062
jpeg.3184.107	109	0.887	411	24	117	16.844	37	83	0.922	0	1.391	2616	5.765
jpeg.3195.85	87	0.740	13	4041	47994	1366.985	155	4003	120.359	47	87.672	1	55.156
jpeg.3198.93	95	0.752	140	2204	6649	529.781	48	1921	34.329	0	17.390	0	28.203
jpeg.3203.135	137	0.897	507	31	196	56.703	35	132	2.141	495	19.046	485	13.141
jpeg.3740.15	17	0.257	33	231	185	0.234	391	78	0.188	57	3.922	40	4.079
jpeg.4154.36	38	0.633	74	1462	364	2.500	692	156	0.812	321	8.344	251	3.984
jpeg.4753.54	56	0.769	146	11	192	2.984	6	74	0.375	0	4.391	0	2.641
susan.248.197	199	0.939	588	22	178	106.672	9	79	2.922	1984	27.968	1984	25.031
susan.260.158	160	0.916	472	570	461	123.594	261	111	4.578	1534	180.062	6909	55.953
susan.343.182	184	0.936	468	776	896	474.391	318	399	12.812	516	44.016	3567	72.703
typeset.10192.123	125	0.744	241	5565	1134	297.859	4537	644	40.766	4674	2267.172	2879	1437.594
typeset.10835.26	28	0.349	60	0	8	0.063	0	9	0.047	0	0.079	0	0.078
typeset.12395.43	45	0.518	125	10	64	0.437	0	28	0.125	0	1.250	29	1.594
typeset.15087.23	25	0.557	89	32	148	0.297	67	51	0.109	23	0.985	16	2.250
typeset.15577.36	38	0.555	93	0	7	0.031	0	6	0.062	0	0.516	0	0.468
typeset.16000.68	70	0.658	67	0	787	21.891	0	425	17.725	16981	3872.728	6317	2532.500
typeset.1723.25	27	0.245	54	7660	781	4.094	9000	283	7.094	4237	95.156	82554	65.891
typeset.19972.246	248	0.993	979	0	14	0.110	0	7	0.069	0	0.234	0	0.250
typeset.4391.240	242	0.981	837	46	1291	6.250	0	95	3.782	0	5.156	0	6.359
typeset.4597.45	47	0.493	133	0	16	0.437	0	16	0.079	0	0.234	0	0.297
typeset.4724.433	435	0.995	1819	0	374	4.000	0	68	1.823	10	17.109	22	23.704
typeset.5797.33	35	0.748	93	0	85	0.234	0	37	0.125	0	0.407	0	0.297
typeset.5881.246	248	0.986	979	191	184	356.218	394	137	7.031	1289	34.547	1258	38.062
Average				849	2318	125.095	591	332	9.606	1191	247.972	4035	162.255

216-30.

[10] Dell'Amico M, Jamal J, Montemanni R. A mixed integer linear program for a precedence-constrained minimum-cost arborescence problem. In Proc The 8^{th} International Conference on Industrial Engineering and Applications (Europe). 2021:216-21.

[11] Chou X, Dell'Amico M, Jamal J, Montemanni R. Precedence-Constrained Arborescences. arXiv preprint arXiv:220802327. 2022.

[12] Abdelmaguid TF. An Efficient Mixed Integer Linear Programming Model for the Minimum Spanning Tree Problem. Mathematics. 2018;6(10).

[13] Reinelt G. TSPLIB–A travelling salesman problem library. ORSA journal on computing. 1991;3(4):376-84.

[14] Montemanni R, Smith DH, Gambardella LM. A heuristic manipulation technique for the sequential ordering problem. Computers & Operations Research. 2008;35(12):3931-44.

[15] Shobaki G, Jamal J. An exact algorithm for the sequential ordeing problem and its application to switching energy minimization in compilers. Computational Optimizations and Applications. 2015;61(2):343-72.

[16] Escudero LF. An inexact algorithm for the sequential ordering problem. European Journal of Operational Research. 1988;37(2):236-49.