Noname manuscript No.
(will be inserted by the editor)

Minimum spanning tree with conflicting edge pairs: a Branch and Cut approach

Francesco Carrabs · Raffaele Cerulli · Rosa Pentangelo · Andrea Raiconi

Received: date / Accepted: date

Abstract In this paper, we show a branch and cut approach to solve the Minimum Spanning Tree problem with conflicting edge pairs. This is a NP-hard variant of the classical Minimum Spanning Tree problem, in which there are mutually exclusive edges. We introduce a new set of valid inequalities for the problem, based on the properties of its feasible solutions, and we develop a branch and cut algorithm based on them. Computational tests are performed both on benchmark instances coming from the literature and on some newly proposed ones. Results show that our approach outperforms a previous branch and cut algorithm proposed for the same problem.

Keywords minimum spanning tree · conflicting edges · branch and cut

1 Introduction

The minimum spanning tree problem with conflicting edge pairs (MSTC) is a very recent variant of the classical minimum spanning tree (MST) problem. Given a connected, undirected and edge-weighted graph, as well as a set of edges pairs in conflict with each other, a feasible MSTC solution is a spanning tree without conflicts whose total weight is minimal, i.e., a minimum spanning tree containing at most an edge for each pair in the conflicts set.

Variants of the same type (that is, with the addition of conflicts) have already been studied for other classic problems, such as the knapsack problem [7], the maximum flow problem [8], the bin packing problem [9] and the minimum cost perfect matching [5].

The specific variant concerning the minimum spanning tree problem was studied for the first time by Darmann et al. [2] in 2009. The authors showed that the problem, in general, is not solvable in polynomial time. In particular,

Department of Mathematics, University of Salerno
Via Giovanni Paolo II 132, 84084 Fisciano (SA), Italy
E-mail: {fcarrabs,raffaele,rpentangelo,araiconi}@unisa.it

there exist two cases in which the problem becomes polynomially solvable: when all pairs of edges in conflict are disjoint ([2], [3]) or when the transitive property holds for the set of such pairs [11]. In [11], the authors presented several meta-heuristic approaches to solve the MSTC problem, while the first authors to face the problem through an exact approach have been Samer and Urrutia in [10]. They presented a mathematical model for the MSTC problem, as well as two sets of valid inequalities. In order to introduce these new sets of valid inequalities, the authors gave an equivalent definition of the problem by defining the concept of *conflict graph*, that we will resume in Section 2.

In this paper, we propose a branch and cut approach for the MSTC, and test its effectiveness and performance on a set of instances originally proposed in [11]. We compared these results with those obtained by the exact algorithm presented in [10], that was tested on the same dataset. The comparison showed that our algorithm outperformed the previous one in all cases except one, and was able to find one additional optimal solution. Furthermore, we also test our approach on a new, wider set of instances that we generated.

The paper is organized as follows. A formal description of the problem, together with the needed definitions and notations, are presented in Section 2. In Section 3, a mathematical formulation for the MSTC is provided. Moreover our novel valid inequalities, together with the ones used in [11], are presented. The proposed branch and cut algorithm is described in Section 4, while computational results are presented in Section 5. Finally, Section 6 contains our conclusions.

2 Notations and problem definition

Let $G = (V, E)$ be an undirected, edge weighted graph, where V is the set of n vertices and E is the set of m edges. We denote by w_e the weight associated to the edge $e \in E$. Furthermore, let P be a set of edge pairs of E, called *conflict set*, defined as follows:

$$P = \{\{e_i, e_j\} : e_i, e_j \in E, e_i \textit{ is in conflict with } e_j\}.$$

For each $e_i \in E$, we indicate with $\chi(e_i)$ the set of edges that are in conflict with it.

The MSTC problem consists of finding the minimum spanning tree $T = (V_T, E_T)$ of G such that its edges are conflict free, i.e.

$$\forall e_i, e_j \in E_T, \{e_i, e_j\} \notin P.$$

We now resume the concept of conflict graph $G' = (E, P)$, originally presented in [10]. G' contains a node for each edge E of G, and two nodes e_i, e_j are connected in G' if and only if $\{e_i, e_j\} \in P$.

Figure 1 shows an example graph G and the related conflict graph G'. We can note that, for instance, $\{e_1, e_3, e_5, e_6, e_8\}$ is a feasible MST solution being a spanning tree of G, but it is not feasible for the MSTC since G' contains both the edges e_1 and e_5 (but $\{e_1, e_5\} \in P$). On the other hand, $\{e_2, e_4, e_5, e_6, e_9\}$ is a conflict free spanning tree and therefore it is a feasible MSTC solution.

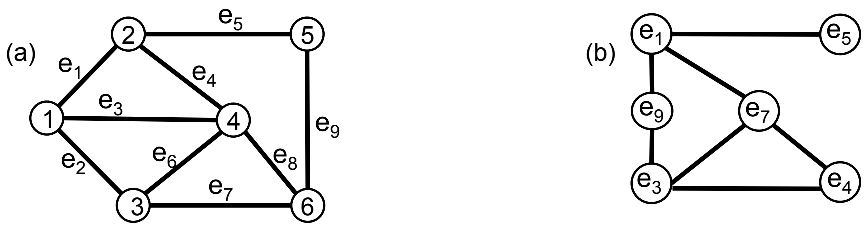

Fig. 1: (a) An example graph G with $|V| = 6$, $|E| = 10$ and conflicts set $P = \{\{e_1, e_5\}, \{e_1, e_7\}, \{e_1, e_9\}, \{e_3, e_4\}, \{e_3, e_7\}, \{e_3, e_9\}, \{e_4, e_7\}\}$ ($|P| = 7$). (b) The related conflict graph $G'(E, P)$, where each node corresponds to an edge of G and each edge corresponds to a pair in P.

3 Basic Mathematical Model

In this section we present a mathematical model for the MSTC problem, based on a traditional Subtour Elimination formulation for the MST with the additional constraints to avoid the conflicts. This model was also considered in [10]. The formulation only uses a type of decision variables x_e associated with the edges of G, with the following meaning:

$$x_e = \begin{cases} 1 & \text{if } e \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

The mathematical programming formulation of the MSTC is the following:

$$(\mathbf{ILP})\ \min \sum_{e \in E} w_e x_e \tag{1}$$

$$s.t. \sum_{e \in E} x_e = n - 1 \tag{2}$$

$$\sum_{e \in E(S)} x_e \leq |S| - 1, \qquad \forall S \subseteq V, S \neq \emptyset \tag{3}$$

$$x_{e_i} + x_{e_j} \leq 1, \qquad \forall (e_i, e_j) \in P \tag{4}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{5}$$

The objective function (1) minimizes the weight of the spanning tree. Constraint (2) imposes the selection of $n - 1$ edges (recall that $|V| = n$) while Constraints (3) are the classical subtour elimination constraints. Finally, Constraints (4) assure that two edge in conflict cannot be simultaneously selected in the solution while constraints (5) are variable definitions.

3.1 Valid inequalities

In this section we present three classes of valid inequalities for the MSTC that we used to design a Branch and Cut approach for this problem. The first class, named *degree-cut* inequalities, assure that there are not isolated vertices in the solution; we use them to enforce the Subtour Elimination model. The second one, the *conflict-cycle* inequalities combine the request of avoiding both cycles and conflicts and represent our main contribution. Finally, the third class of inequalities are the well known *odd-cycle* inequalities that are derived from the conflict graph structure. In the following section we describe in details these valid inequalities.

3.1.1 The degree-cut inequalities

Since the solution of the MSTC is a spanning tree then for each node we have at least one incident edge selected. For this reason, we add to our model the following valid inequalities:

$$\sum_{e \in \delta(v)} x_e \geq 1, \qquad \forall v \in V. \tag{6}$$

The constraints (6) state explicitly that the degree of any node into the solution must be greater than or equal to 1. These inequalities improve the relaxed solution value of ILP model. Indeed, by removing the constraints (3) from ILP model, the optimal solution is obtained by selecting the cheapest $n - 1$ edges of the graph. This could lead to the presence of isolated nodes (i.e. with degree equal to zero) in the solution. The inequalities (6) prevent the construction of these type of solutions.

Since the number of inequalities (6) is equal to n, no separation procedures are applied but they are directly introduced into the ILP model as a priori constraints. Obviously, these constraints are not necessary to represent the solutions space but, in our experiments, they speed up the convergence of our branch an cut.

3.1.2 Conflict-cycle inequalities

The conflict-cycle inequalities are a stronger version of the subtour elimination constraints obtained by exploiting the conflicts among the edges. More in details:

Let ζ be a set of edges that generate a cycle in G and let us suppose that two of these edges are in conflict with another edge e_c that does not belong to ζ. Then, in any feasible solutions of MSTC, the number of edges of ζ plus the edge e_c must be lower than or equal to $|\zeta| - 1$. The following theorem prove that these inequalities are valid for the MSTC.

Theorem 1 *Let ζ be a cycle of G and let e_c be an edge outside this cycle that is in conflict with two edges e_k and e_j of ζ. Then*

$$\sum_{e_i \in \zeta} x_{e_i} + x_{e_c} \leq |\zeta| - 1, \qquad \forall\, \zeta \subseteq E : \exists \{e_k, e_j\} \subseteq \zeta \cap \chi(e_c),\ e_c \notin \zeta \tag{7}$$

are valid inequalities for the MSTC problem.

Proof By contradiction, let us suppose that in a feasible solution of MSTC we have:

$$\sum_{e_{i'} \in \zeta'} x_{e_{i'}} + x_{e_g} > |\zeta'| - 1.$$

where $\zeta' \subseteq E$ is a cycle of G, $e_{j'}, e_{k'} \in \zeta'$, $e_g \in E \setminus \zeta'$, and $e_{j'}, e_{k'} \in \chi(e_g)$. We have to consider the following two cases:

- If $x_{e_g} = 0$ then $\sum_{e_{i'} \in \zeta'} x_{e_{i'}} > |\zeta'| - 1$. However, this last condition violates Constraints (3). A contradiction.
- if $x_{e_g} = 1$ then

$$\sum_{e_{i'} \in \zeta'} x_{e_{i'}} + 1 > |\zeta'| - 1 \quad \Rightarrow \quad \sum_{e_{i'} \in \zeta'} x_{e_{i'}} > |\zeta'| - 2.$$

Due to this last condition at least one of variables $x_{e_{j'}}$ and $x_{e_{k'}}$ must be equal to 1 thereby violating the Constraints (4).

In Figure 2 is shown an example of how the inequalities (7) work. Figure 2(a) is the initial graph. In particular, we consider a cycle $\zeta = \{e_4, e_5, e_9, e_8\}$ noting that e_5 and e_9 belong to $\chi(e_1)$ (see Fig. 1). Notice that in this situation the application of the classical subtour elimination constraints could generate the solution depicted in Figure 2(c) that, instead, is cut off by inequalities (7). In particular, since inequalities (7) allow to select at most three our of five edges in Figure 2(b), as a result we can have the solution shown in Figure 2(d), when e_1 is selected, or a solution containing three edges of ζ, when e_1 is not selected (Figure 2(e)).

3.1.3 Odd-Cycle inequalities

Another set of valid inequalities for the MSTC are the well-known odd-cycle inequalities. These inequalities are based on the conflict graph G' described in Section 2. Each vertex of G' is associated to an edge of G and two nodes are connected if the respective edges of G are in conflict. This means that the selection of two connected vertices in G' is equivalent to select two edges in conflict in G. For this reason, given a cycle ζ' of G', having an odd number k of edges, it is easy to see that it is possible to select at most $\frac{k-1}{2}$ vertices of the cycle (that is, edges of G) without violating the conflict constraints. Formally,

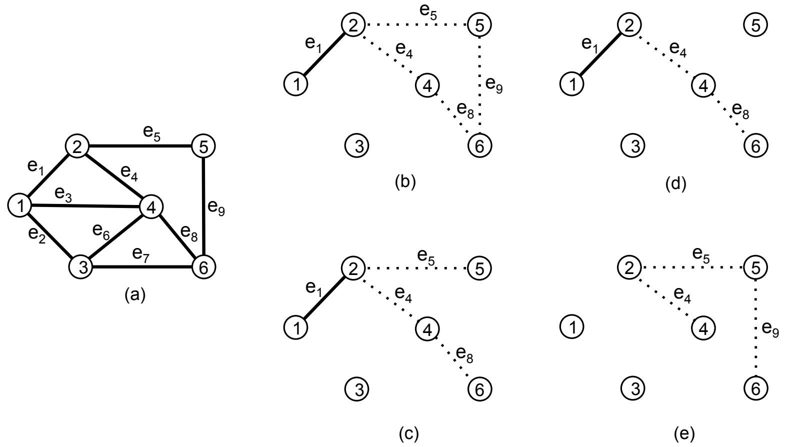

Fig. 2: (a) The input graph G. (b) A cycle $\zeta = \{e_4, e_5, e_9, e_8\}$ (dotted lines) and edge e_1 in conflict with e_5 and e_9. (c) A selection of four edges that respect constraints (2)-(5) and violates constraints (7). (d,e) Two selections of three edges respecting constraints (2)-(5) and (7).

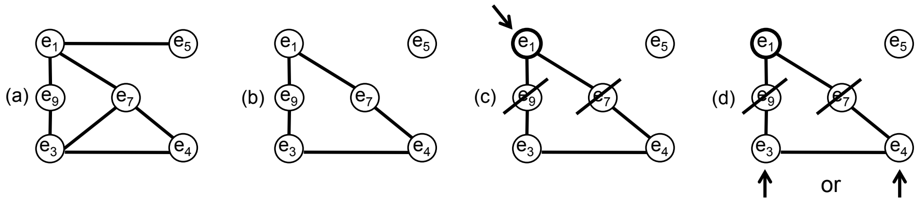

Fig. 3: (a) The conflict graph G' for the G graph in Figure 1. (b) An odd-cycle of length 5 in G'. (c) If we choose e_1, it is not possible to choose e_5 and e_9. (d) At this point, only one between e_3 and e_4 can be part of an MSTC solution.

$$\sum_{e \in OD} x_e \leq \frac{|OD| - 1}{2}, \qquad \forall OD \subseteq E \quad odd-cycle \quad in \quad G' \tag{8}$$

In Figure 3 we show that, given an odd cycle of length 5 in the conflict graph of the example in Figure 1, the maximum number of edges that can be chosen is $\frac{5-1}{2} = 2$.

A branch and cut approach based on the ILP model and using, among the others, the odd-cycle inequalities was presented in [10]. In the computational test section we will carry out a comparison between our branch and cut approach and theirs.

4 Branch and Cut approach

In this section, we outline the main ingredients of our branch-and-cut algorithm for the optimal MSTC solution as well as the separation procedures for the valid inequalities described in previous section. To obtain upper bounds that help pruning the search tree, we use the genetic algorithm proposed in [1]. However, since it is known that even finding a feasible MSTC solution is NP-hard, there are several instances where these upper bounds are not available because the genetic algorithm did not found them within a fixed time limit.

4.1 Initial relaxation

The initial relaxation of ILP, named $\mathscr{R}(ILP)$, is composed by constraints (2),(4),(6) and the inequalities $0 \leq x_e \leq 1$.

4.2 Separation procedures

The odd-cycle inequalities are separated by using the exact algorithm proposed in [4] while the subtour elimination constraints are separated by using the exact algorithm presented in [6].

In the following, we describe our procedure to separate conflict-cycle inequalities (7). Given a solution $\bar{x}$ of $\mathscr{R}(ILP)$, we build a new graph $\tilde{G} = (V, \tilde{E})$ where $\tilde{E} = \{e = (i,j) \in E : \bar{x}_e > 0\}$. To each edge $\tilde{e} \in \tilde{E}$ the weight $w_{\tilde{e}} = 1 - \bar{x}_{\tilde{e}}$ is assigned. The conflict-cycle inequalities (7) are heuristically separated by using the graph $\tilde{G}$ with the following procedure. Given any couple of nodes $\tilde{v}_1, \tilde{v}_2 \in V$ such that $(\tilde{v}_1, \tilde{v}_2) \in \tilde{E}$, we look for the shortest path between them in $\tilde{G}$ which does not include the edge $(\tilde{v}_1, \tilde{v}_2)$. If such a path exists, we append $(\tilde{v}_1, \tilde{v}_2)$ to it, obtaining a cycle $\tilde{\zeta} \subseteq \tilde{E}$. To individuate a violated inequality, we look for an edge $\tilde{e}_3 \in \chi(\tilde{e}_1) \cap \chi(\tilde{e}_2) \setminus \tilde{\zeta}$ where $\tilde{e}_1, \tilde{e}_2 \in \tilde{\zeta}$ and such that $\sum_{\tilde{e} \in \tilde{\zeta}} \bar{x}_{\tilde{e}} + \bar{x}_{\tilde{e}_3} > |\tilde{\zeta}| - 1$; hence we look for all possible edges of this type and all the violated inequalities are introduced in the model. Note that if $\sum_{\tilde{e} \in \tilde{\zeta}} \bar{x}_{\tilde{e}} + 1 \leq |\tilde{\zeta}| - 1$, it is impossible to find violated inequalities of the type (7), hence we don't look for them in this case. Furthermore, we decided to use an additional tolerance parameter $\epsilon_c \geq 0$, meaning that we only consider violated inequalities if $\sum_{\tilde{e} \in \tilde{\zeta}} \bar{x}_{\tilde{e}} + \bar{x}_{\tilde{e}_3} > |\tilde{\zeta}| - 1 + \epsilon_c$. The computational complexity of this algorithm is $O(m^2 logn)$, in fact the individuation of a shortest path requires $|\tilde{E}| log |V|$ and it is invoked for each edge in $\tilde{E}$. We use an m x m binary matrix to state in $O(1)$ that two edges are in conflict.
Note that the separation procedure for the subtour elimination constraints cannot be used for inequalities (7) because it is not sufficient to individuate the set of vertices S that generate a cycle. We need to know what are the edges of the cycle to separate the conflict-cycle inequalities.

4.3 Cutting plane phase

At each iteration of the cutting-plane algorithm:
- if the variables in the LP solution are all integer, the subtour elimination constraints (3) are heuristically separated through a DFS procedure;
- otherwise, the following separation procedures are used:
  1. Exact separation procedure [6] for the subtour elimination constraints (3).
  2. Heuristic algorithm for separating the conflict-cycle inequalities (7) with $\epsilon_c = 0.1$.
  3. Exact algorithm for separating the odd-cycle inequalities (8) only at the root node.

If all separation procedures fail to find violated inequalities or a tailing-off criterium is met, we branch on variables using the default parameters of CPLEX. The tailing-off is applied when the improvement in the upper bound is less than 10^{-5} in five consecutive iterations.
To keep the size of the LP as small as possible, in each node of the search tree we never add more than 50 valid inequalities. The value of this parameter was chosen after a preliminary tuning phase.

5 Computational results

In this section we present the computational results of the tests we made in order to evaluate the performance of our branch and cut algorithm (from now on called BC). The algorithm was coded in C++ on an OSX platform (iMac, mid 2011), running on an Intel(R) Core(TM) i7-2600 CPU 3.40GHz (family 6, model 42, stepping 7) with 8 GB of RAM, equipped with the IBM ILOG CPLEX 12.6.1 solver (single thread mode).

We compared the results of BC with the branch and cut algorithm (from now on called SU) proposed in [10]. Following [10], for all the experiments we considered a time limit equal to 5000 seconds. Furthermore, we considered a memory limit set to 3 GB. In this previous work, the authors propose a preprocessing procedure to simplify the instances before solving them. They divided the instances in two subsets, namely type 1 and type 2. Instances belonging to type 2 resulted to be very easy to solve after the preprocessing phase. Indeed, the authors do not present results about the SU performances on these instances, since they state that after this preprocessing (taking up to 18 seconds) all instances of this group were solved in negligible time. On these same instances, the genetic algorithm that we used to initialize our method always found (in up to 26 seconds) solutions that were very quickly certified to be optimal by BC. For these reasons, we compare our results only on the harder type 1 instances. We want to remark that we did not apply any preprocessing before solving them with BC. As will be shown, despite this, we obtained better results in all cases except one. This result, in our opinion, emphasizes the effectiveness of our algorithm.

Table 1 reports the results of the comparison between BC and SU.

Instance				SU		BC		
id	n	m	p	LB	UB	LB	UB	Time
1	50	200	199	708		708		0.2
2	50	200	398	770		770		0.5
3	50	200	597	917		917		1.8
4	50	200	995	1324		1324		7.4
5	100	300	448	4041		4041		4.6
6	100	300	897	5658		5658		178.5
7	100	300	1344	6621.2	-	**6635.4**	-	5010.0
8	100	500	1247	4275		4275		11.5
9	100	500	2495	5951.4	6006	**5997**		1239.4
10	100	500	3741	6510.8	9440	**6707.8**	**8049**	5010.1
11	100	500	6237	7568.7	-	**7729.3**	-	5010.0
12	100	500	12474	9816.9	-	**10560.2**	-	5010.0
13	200	600	1797	13072.9	14707	**13171.2**	**14086**	5010.0
14	200	600	3594	17532.7	-	**17595.0**	-	5010.0
15	200	600	5391	Infeasible		Infeasible		16.4
16	200	800	3196	20744.2	21852	**20941.5**	**21553**	5010.1
17	200	800	6392	26361.3	-	**26526.7**	-	5010.1
18	200	800	9588	29443.6	-	**30634.2**	-	5010.0
19	200	800	15980	33345.1	-	**36900.2**	-	5010.0
20	300	800	3196	Infeasible		Infeasible		2911.1
21	300	1000	4995	51451.3	-	51398.4	-	5010.0
22	300	1000	9990	60907.8	-	**61878.9**	-	5010.0
23	300	1000	14985	Infeasible		Infeasible		1820.0

Table 1: Comparison between the solution values of SU and BC algorithms.

The first fourth columns of the table show the information concerning the instance: a numerical identifier (id), the number of nodes (n), of edges (m) and of conflict pairs (p). The next two columns report the lower (LB) and upper (UB) bounds found by SU. When the lower and the upper bounds coincide, i.e. an optimal solution is found, the optimal value is reported between the LB and UB columns. Finally, the last three columns report the lower bound, the upper bound and the computational time ($Time$), in seconds, of BC. The bounds are shown in bold whenever the solution found by BC is better than the solution found by SU. In [10] the authors did not report the computational time spent by SU on these instances, and therefore we cannot carry out a precise comparison between the two branch and cut from this point of view.

The first 6 instances and instance n°8 are solved optimally by both algorithms. The instance n°9, instead, is solved to optimality by BC in 1239.4 seconds, while it was not solved by SU within 5000 seconds. Therefore, our algorithm provides a new optimal solution for this set of instances. Both the algorithms certify the infeasibility of instances n°15, 20 and 23. For the remaining 12 instances, BC produces better lower bounds in all cases except one (instance n°21).

With respect to the subset of instances that are not solved by BC and SU, both algorithms found upper bounds in the same 3 cases (instances n° 10, 13 and 16), and those found by BC are always better. It is worth noting the percentage gap value between the upper and the lower bounds in these cases.

This value is computed as $100 \times \frac{UB-LB}{UB}$. On the instance n°10, the percentage gap is equal to 31.03% for SU and 16.66% for BC. On the instance n°13, it is equal to 11.11% for SU and 6.49% for BC. Finally, for the instance n°16, it is equal to 5.07% for SU and 2.84% for BC. That is, the percentage gap of BC for these instances is about half of the percentage gap of SU.

Regarding the performance, all the instances optimally solved by BC required less than 12 seconds, except for the instance n°9 for which, as previously mentioned, about 1240 seconds were spent. To certify the infeasibility, BC required around 16 seconds on the instance n°15, 2911 seconds on the instance n°20 and 1820 seconds on the instance n°23.

In order to further investigate the effectiveness and performance of BC, we generated a new set of benchmark instances[1]. The number of nodes n in this new set ranges from 25 to 100, with incremental steps of size 25. The number of edges m is assigned according to the following density values: 0.2, 0.3, 0.4. That is, a graph with density d has $m = dn(n-1)/2$ edges. This means that our instances are much denser than the previous ones, in which the highest density value is about 0.16.

Given m edges, we can generate at most $\binom{m}{2} = m(m-1)/2$ conflict pairs. The number of conflict pairs associated to each instance is equal to 1%, 4% and 7% of $m(m-1)/2$. We generated 5 different instances for each combination of parameters n, m and p. Thus, in total we generated 180 new instances. The combinations of these parameters allow us to determine which of them affects most the BC performances. It is also worth noting that the new instances were generated ensuring their feasibility, therefore there are no unfeasible instances as in the previous set.

We show the results on this new dataset in Tables 2 and 3. The meaning of id, n, m and p are the same as for Table 1. Under the s columns we report the value of a seed parameter that was used to generate different instances with the same parameter values. The last three columns report, as in Table 1, the results of our approach (lower bound (LB) and upper bound (UB), or a value in between when an optimum is found) and the computational times in seconds.

We can see that all instances with $n = 25$ (Table 2(a)) are solved to optimality within 0.5 seconds, with 38 out of 45 of them requiring under 0.1 seconds. We can, however, start noticing a trend with respect to how parameters affect the complexity of the instances. Indeed, the 4 instances that required the most time to be solved all correspond to cases in which the number of conflict pairs is the highest, with respect to the other instances with the same number of edges. These cases correspond to instances n°36 ($m = 60, p = 124$), n°50 ($m = 90, p = 281$), n°65 and 66 ($m = 120, p = 500$), that are solved in 0.3, 0.5, 0.5 and 0.4 seconds, respectively.

The trend is confirmed for instances of all sizes. For $n = 50$ (Table 2(b)) we can observe that all instances with p equal or less than the 4% of $\binom{m}{2}$ are

[1] Instance files are available at
http://www.dipmat2.unisa.it/people/carrabs/www/DataSet/MSTC.zip

again solved optimally, with computational times growing up to 6.3 seconds for $m = 245$, 7.5 seconds for $m = 367$ and 13.8 seconds for $m = 490$. When p grows to the 7% of the maximum number of conflicts, the related instances result to be considerably more difficult to solve, since we reach a certified optimal solution only for 2 out of 15 of them, namely instances n°80 and 83, solved in 1938.69 and 25.7 seconds, respectively. In the other 13 cases, the gap between the returned lower and upper bounds are between 2% and 8% for $m = 245$, between 3% and 8% for $m = 367$ and between 3% and 7% for $m = 490$.

When $n = 75$ (Table 3(a)) we are able to find optimal solutions for all the 15 instances with p equal to the 1% of $\binom{m}{2}$. Computational times in these cases grow up to 23.9 seconds once (instance n°132) and are lower than 5.5 seconds for the remaining 14 instances. None of the remaining 30 instances is solved to optimality. When $p = 4\%$ of $\binom{m}{2}$, we were always able to find both an upper and a lower bound, with gaps between 1% and 6% for $m = 555$, between 1% and 5% for $m = 832$ and between 2% and 6% for $m = 1110$. It can be noticed that in 3 out of 5 cases for $m = 832$ as well as in all 5 cases with $m = 1110$ the computational times are lower than the time limit, as in these cases it was the memory limit to be reached first. The instances with the p equal to the 7% of $\binom{m}{2}$ are again the hardest, since we were able to identify a lower bound only for one of them (instance n°156). Even in this case, the gap between upper and lower bound is considerably high, being equal to 26%. In 13 out of 15 cases we reached the memory limit.

Finally, we consider the results for instances with $n = 100$, reported in Table 3(b). Again, all instances with $p = 1\%$ of $\binom{m}{2}$ could be solved to optimality, within 71.8 seconds for $m = 990$, 249.6 seconds for $m = 1485$ and 214.4 seconds for $m = 1980$. None of the instances with $p = 4\%$ of $\binom{m}{2}$ was solved to optimality, and we were able to identify a lower bound for each of them except one (instance n°164). The gaps between lower and upper bounds are between 12% and 20% for $m = 990$, between 11% and 17% for $m = 1485$ and between 10% and 13% for $m = 1980$. The time limit was always reached for the 5 instances with the smallest number of edges, while the memory limit was always reached in the remaining 10 cases. Finally, when $p = 7\%$ of $\binom{m}{2}$ we were never able to find an upper bound. The memory limit was reached for 6 of these instances, while the time limit was reached in the other 9 cases.

To conclude we can note that, predictably, the factor that most affects the BC performances is the ratio between the number of edges and the number of conflict pairs. Indeed, as p grows with respect to m, it becomes more difficult to find feasible solutions. Between instances with the same number of nodes, increasing the number of edges while keeping constant this ratio have in many cases either marginal or unnoticeable effect on the performances. While increasing the number of nodes leads to harder instances, even the largest ones (with up to 100 nodes and 1980 edges) with the fewest number of conflict pairs could be solved to optimality within about 4 minutes.

(a)

Instance					BC		
id	n	m	p	s	LB	UB	Time
24	25	60	18	1	347		0.0
25	25	60	18	7	389		0.0
26	25	60	18	13	353		0.0
27	25	60	18	19	346		0.0
28	25	60	18	25	336		0.0
29	25	60	71	31	381		0.0
30	25	60	71	37	390		0.1
31	25	60	71	43	372		0.0
32	25	60	71	49	357		0.0
33	25	60	71	55	406		0.0
34	25	60	124	61	385		0.0
35	25	60	124	67	432		0.0
36	25	60	124	73	458		0.3
37	25	60	124	79	400		0.0
38	25	60	124	85	420		0.0
39	25	90	41	91	311		0.0
40	25	90	41	97	306		0.0
41	25	90	41	103	299		0.0
42	25	90	41	109	297		0.0
43	25	90	41	115	318		0.0
44	25	90	161	121	305		0.0
45	25	90	161	127	339		0.0
46	25	90	161	133	344		0.0
47	25	90	161	139	329		0.0
48	25	90	161	145	326		0.0
49	25	90	281	151	349		0.0
50	25	90	281	157	385		0.5
51	25	90	281	163	335		0.0
52	25	90	281	169	348		0.1
53	25	90	281	175	357		0.0
54	25	120	72	181	282		0.0
55	25	120	72	187	294		0.0
56	25	120	72	193	284		0.0
57	25	120	72	199	281		0.0
58	25	120	72	205	292		0.0
59	25	120	286	211	321		0.0
60	25	120	286	217	317		0.0
61	25	120	286	223	284		0.0
62	25	120	286	229	311		0.0
63	25	120	286	235	290		0.0
64	25	120	500	241	329		0.1
65	25	120	500	247	339		0.5
66	25	120	500	253	368		0.4
67	25	120	500	259	311		0.0
68	25	120	500	265	321		0.0

(b)

Instance					BC		
id	n	m	p	s	LB	UB	Time
69	50	245	299	271	619		0.0
70	50	245	299	277	604		0.0
71	50	245	299	283	634		0.0
72	50	245	299	289	616		0.1
73	50	245	299	295	595		0.0
74	50	245	1196	301	678		1.4
75	50	245	1196	307	681		3.2
76	50	245	1196	313	709		6.3
77	50	245	1196	319	639		1.5
78	50	245	1196	325	681		3.8
79	50	245	2093	331	791.2	833	5010.1
80	50	245	2093	337	835		1938.7
81	50	245	2093	343	773.231	840	5010.1
82	50	245	2093	349	820.029	836	5010.1
83	50	245	2093	355	769		25.7
84	50	367	672	361	570		0.1
85	50	367	672	367	561		1.4
86	50	367	672	373	573		0.0
87	50	367	672	379	560		0.0
88	50	367	672	385	549		0.5
89	50	367	2687	391	612		7.5
90	50	367	2687	397	615		6.6
91	50	367	2687	403	587		3.0
92	50	367	2687	409	634		7.3
93	50	367	2687	415	643		3.2
94	50	367	4702	421	701.265	726	5010.1
95	50	367	4702	427	719.451	770	5010.0
96	50	367	4702	433	723.89	786	5010.0
97	50	367	4702	439	669.848	711	5010.0
98	50	367	4702	445	737.318	764	5010.0
99	50	490	1199	451	548		0.1
100	50	490	1199	457	530		0.5
101	50	490	1199	463	549		0.0
102	50	490	1199	469	540		0.2
103	50	490	1199	475	540		0.0
104	50	490	4793	481	594		7.8
105	50	490	4793	487	579		13.8
106	50	490	4793	493	589		3.0
107	50	490	4793	499	577		7.5
108	50	490	4793	505	592		6.0
109	50	490	8387	511	631.436	678	5010.0
110	50	490	8387	517	626.721	651	5010.0
111	50	490	8387	523	658.385	689	5010.0
112	50	490	8387	529	662.224	682	5010.1
113	50	490	8387	535	641.31	674	5010.0

Table 2: Computational results of BC on new instances: (a) $n = 25$, (b) $n = 50$

6 Conclusions

In this work, we described a novel branch and cut approach to solve the MSTC problem. In particular, our main contribution is related to the proposal of a new set of valid inequalities, based on combined properties belonging to any feasible solution. Furthermore, we tested the approach we designed on the benchmark instances and compared it with a previous one. Our tests showed our approach to perform better on all instances except one, despite not using a preprocessing algorithm presented in the previous work in order to simplify the instances. Moreover, we created a new set of feasible instances, in order to further test our approach and allow other researchers to have access to a wider set of benchmark instances for the problem.

References

1. Carrabs, F., Cerulli, R., Pentangelo, R.: A genetic algorithm for minimum spanning trees problem with conflicting edge pairs. submitted to Annals of Operations Research (2016)
2. Darmann, A., Pferschy, U., Schauer, J.: Determining a minimum spanning tree with disjunctive constraints. Lecture Notes in Computer Science (including subseries Lecture

(a)

Instance					BC		
id	n	m	p	s	LB	UB	Time
114	75	555	1538	541	868		0.7
115	75	555	1538	547	871		3.0
116	75	555	1538	553	838		0.3
117	75	555	1538	559	855		4.4
118	75	555	1538	565	857		4.1
119	75	555	6150	571	1023.72	1047	5010.0
120	75	555	6150	577	1008.82	1069	5010.2
121	75	555	6150	583	987.318	1040	5010.1
122	75	555	6150	589	985.642	998	5010.1
123	75	555	6150	595	962.557	994	5010.1
124	75	555	10762	601	1054.25	-	4647.3
125	75	555	10762	607	1069.51	-	3483.6
126	75	555	10762	613	1040.97	-	5010.0
127	75	555	10762	619	1006.3	-	5010.0
128	75	555	10762	625	1046.43	-	3208.1
129	75	832	3457	631	798		4.8
130	75	832	3457	637	821		1.1
131	75	832	3457	643	816		4.0
132	75	832	3457	649	820		23.9
133	75	832	3457	655	815		1.4
134	75	832	13828	661	873.837	903	3463.3
135	75	832	13828	667	901.812	953	4443.7
136	75	832	13828	673	873.675	892	5010.0
137	75	832	13828	679	885.573	915	4570.8
138	75	832	13828	685	886.873	896	5010.0
139	75	832	24199	691	949.556	-	1305.3
140	75	832	24199	697	907.804	-	1161.2
141	75	832	24199	703	910	-	953.0
142	75	832	24199	709	943.984	-	1224.2
143	75	832	24199	715	956.312	-	962.3
144	75	1110	6155	721	787		2.1
145	75	1110	6155	727	785		5.4
146	75	1110	6155	733	783		0.0
147	75	1110	6155	739	784		3.3
148	75	1110	6155	745	797		5.7
149	75	1110	24620	751	846.698	867	4067.5
150	75	1110	24620	757	829.232	851	4573.5
151	75	1110	24620	763	841.546	892	2189.8
152	75	1110	24620	769	841.621	864	2866.0
153	75	1110	24620	775	835.045	882	1783.3
154	75	1110	43085	781	868.725	-	1049.9
155	75	1110	43085	787	853.45	-	1350.7
156	75	1110	43085	793	884.679	1194	1522.3
157	75	1110	43085	799	853	-	1466.3
158	75	1110	43085	805	853.987	-	1461.4

(b)

Instance					BC		
id	n	m	p	s	LB	UB	Time
159	100	990	4896	811	1119		43.7
160	100	990	4896	817	1137		11.8
161	100	990	4896	823	1113		71.8
162	100	990	4896	829	1110		48.6
163	100	990	4896	835	1090		35.8
164	100	990	19583	841	1249.38	-	5010.0
165	100	990	19583	847	1225.76	1491	5010.0
166	100	990	19583	853	1215	1510	5010.0
167	100	990	19583	859	1264.17	1441	5010.2
168	100	990	19583	865	1257.27	1560	5010.1
169	100	990	34269	871	1262	-	3006.9
170	100	990	34269	877	1290.68	-	3371.9
171	100	990	34269	883	1318.54	-	3684.2
172	100	990	34269	889	1282.38	-	3939.1
173	100	990	34269	895	1304.45	-	3103.7
174	100	1485	11019	901	1079		248.6
175	100	1485	11019	907	1056		113.4
176	100	1485	11019	913	1059		47.9
177	100	1485	11019	919	1046		195.2
178	100	1485	11019	925	1072		249.6
179	100	1485	44075	931	1143.95	1374	3018.3
180	100	1485	44075	937	1143.61	1291	2144.6
181	100	1485	44075	943	1137.62	1344	3075.6
182	100	1485	44075	949	1136.9	1286	3523.3
183	100	1485	44075	955	1134.63	1370	2954.2
184	100	1485	77131	961	1164.44	-	4773.8
185	100	1485	77131	967	1168.2	-	5010.0
186	100	1485	77131	973	1180.02	-	5010.0
187	100	1485	77131	979	1183.53	-	5010.0
188	100	1485	77131	985	1159.25	-	5010.0
189	100	1980	19593	991	1031		214.4
190	100	1980	19593	997	1036		42.8
191	100	1980	19593	1003	1024		21.8
192	100	1980	19593	1009	1025		27.4
193	100	1980	19593	1015	1028		151.0
194	100	1980	78369	1021	1096.83	1234	1938.3
195	100	1980	78369	1027	1065.64	1187	2160.3
196	100	1980	78369	1033	1087.39	1213	3595.6
197	100	1980	78369	1039	1081.26	1221	2411.8
198	100	1980	78369	1045	1084.09	1245	2385.6
199	100	1980	137145	1051	1098.61	-	5010.1
200	100	1980	137145	1057	1126.27	-	5010.1
201	100	1980	137145	1063	1111.27	-	5010.1
202	100	1980	137145	1069	1114.58	-	5010.1
203	100	1980	137145	1075	1114.07	-	5010.2

Table 3: Computational results of BC on new instances: (a) $n = 75$, (b) $n = 100$

Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **5783 LNAI**, 414–423 (2009)

3. Darmann, A., Pferschy, U., Schauer, J., Woeginger, G.: Paths, trees and matchings under disjunctive constraints. Discrete Applied Mathematics **159**(16), 1726–1735 (2011)
4. Gerards, A.M.H., Schrijver, A.: Matrices with the edmonds–johnson property. Combinatorica **6**(4), 365–379 (1986)
5. Oncan, T., Zhang, R., Punnen, A.: The minimum cost perfect matching problem with conflict pair constraints. Computers and Operations Research **40**(4), 920–930 (2013)
6. Padberg, M., Wolsey, L.: Trees and cuts. North-Holland Mathematics Studies **75**(C), 511–517 (1983)
7. Pferschy, U., Schauer, J.: The knapsack problem with conflict graphs. Journal of Graph Algorithms and Applications **13**(2), 233–249 (2009)
8. Pferschy, U., Schauer, J.: The maximum flow problem with disjunctive constraints. Journal of Combinatorial Optimization **26**(1), 109–119 (2013)
9. Sadykov, R., Vanderbeck, F.: Bin packing with conflicts: A generic branch-and-price algorithm. INFORMS Journal on Computing **25**(2), 244–255 (2013)
10. Samer, P., Urrutia, S.: A branch and cut algorithm for minimum spanning trees under conflict constraints. Optimization Letters **9**(1), 41–55 (2014)
11. Zhang, R., Kabadi, S., Punnen, A.: The minimum spanning tree problem with conflict constraints and its variations. Discrete Optimization **8**(2), 191–205 (2011)