Optimization Letters (2023) 17:1317–1335
https://doi.org/10.1007/s11590-022-01949-8

ORIGINAL PAPER

# Polyhedral results and stronger Lagrangean bounds for stable spanning trees

**Phillippe Samer**[1] · **Dag Haugland**[1]

Received: 31 May 2022 / Accepted: 20 October 2022 / Published online: 11 November 2022


## Abstract

Given a graph $G = (V, E)$ and a set $C$ of unordered pairs of edges regarded as being in conflict, a stable spanning tree in $G$ is a set of edges $T$ inducing a spanning tree in $G$, such that for each $\{e_i, e_j\} \in C$, at most one of the edges $e_i$ and $e_j$ is in $T$. The existing work on Lagrangean algorithms to the NP-hard problem of finding minimum weight stable spanning trees is limited to relaxations with the integrality property. We exploit a new relaxation of this problem: fixed cardinality stable sets in the underlying conflict graph $H = (E, C)$. We find interesting properties of the corresponding polytope, and determine stronger dual bounds in a Lagrangean decomposition framework, optimizing over the spanning tree polytope of $G$ and the fixed cardinality stable set polytope of $H$ in the subproblems. This is equivalent to dualizing exponentially many subtour elimination constraints, while limiting the number of multipliers in the dual problem to $|E|$. It is also a proof of concept for combining Lagrangean relaxation with the power of integer programming solvers over strongly NP-hard subproblems. We present encouraging computational results using a dual method that comprises the Volume Algorithm, initialized with multipliers determined by Lagrangean dual-ascent. In particular, the bound is within 5.5% of the optimum in 146 out of 200 benchmark instances; it actually matches the optimum in 75 cases. All of the implementation is made available in a free, open-source repository.

**Keywords** Lagrangean decomposition · Dual ascent · Conflict-free spanning trees · Stable sets · Polyhedra · Open-source software

Dedicated to the memory of Gerhard Woeginger, a lasting inspiration to the first author, and also one of the pioneers in the study of stable spanning trees.

A preliminary version of this work, including the results in Sect. 3, appears in the *open access* proceedings of INOC 2022—the 10th International Network Optimization Conference [29].

✉ Phillippe Samer
samer@uib.no

Extended author information available on the last page of the article

## 1 Introduction

Given an undirected graph $G = (V, E)$, with edge weights $w : E \to \mathbb{Q}$, and a family $C$ of unordered pairs of edges that are regarded as being in conflict, a stable (or conflict-free) spanning tree in $G$ is a set of edges $T$ inducing a spanning tree in $G$, such that for each $\{e_i, e_j\} \in C$, at most one of the edges $e_i$ and $e_j$ is in $T$. The minimum spanning tree under conflict constraints (MSTCC) problem is to determine a stable spanning tree of least weight, or decide that none exists. It was introduced by [12, 13], who also prove its NP-hardness.

Different combinatorial and algorithmic results about stable spanning trees explore the associated conflict graph $H = (E, C)$, which has a vertex corresponding to each edge in the original graph $G$, and where we represent each conflict constraint by an edge connecting the corresponding vertices in $H$. Note that each stable spanning tree in $G$ is a subset of $E$ which corresponds both to a spanning tree in $G$ and to a stable set (or independent set, or co-clique: a subset of pairwise non-adjacent vertices) in $H$. Therefore, one can equivalently search for stable sets in $H$ of cardinality exactly $|V| - 1$ which do not induce cycles in the original graph $G$.

We have recently initiated the combinatorial study of stable sets of cardinality exactly $k$ in a graph [28], where $k$ is a positive integer given as part of the input. There are appealing research directions around algorithms, combinatorics and optimization for problems defined over fixed cardinality stable sets. Also from an applications perspective, conflict constraints arise naturally in operations research and management science. Stable spanning trees, in particular, model real-world settings such as communication networks with different link technologies (which might be mutually exclusive in some cases), and utilities distribution networks. In fact, the latter is a standard application of the quadratic minimum spanning tree problem [1], which generalizes the MSTCC one.

Exact algorithms to find stable spanning trees have been investigated for a decade now, building on branch-and-cut [10, 30], or Lagrangean relaxation [11, 33] strategies. Consider the natural integer programming (IP) formulation for the MSTCC problem:

$$\min \sum_{e \in E} w_e x_e \tag{1}$$

$$\text{s.t.} \sum_{e \in E(S)} x_e \leq |S| - 1, \qquad \text{for each } S \subsetneq V, S \neq \emptyset, \tag{2}$$

$$\sum_{e \in E} x_e = |V| - 1, \tag{3}$$

$$x_{e_i} + x_{e_j} \leq 1, \qquad \text{for each } \{e_i, e_j\} \in C, \tag{4}$$

$$x_e \in \{0, 1\}, \qquad \text{for each } e \in E. \tag{5}$$

While a considerable effort in the development of branch-and-cut algorithms led to more sophisticated formulations and contributed to a better understanding of our capacity to solve MSTCC instances by judicious use of valid inequalities, the existing Lagrangean algorithms are limited to the most elementary approach. Namely, a relaxation scheme dualizing conflict constraints (4), which thus has the integrality property, as proved in the seminal work of Edmonds [16]. We review other aspects of the corresponding references in Sect. 3.1.

The present paper takes the standpoint that the development of a full-fledged Lagrangean strategy to find stable spanning trees is an unsolved problem. While we recognize different merits of previous work, we found it productive to investigate stronger Lagrangean bounds in this context: exploring more creative relaxation schemes, designing improved dual methods, all the while harnessing the polyhedral point of view and progress in IP computation.

The main idea of this paper is to offer an alternative starting point for this problem, building on fixed cardinality stable sets as an alluring handle to work on stable spanning trees. After presenting some elementary properties of the corresponding polytope in Sect. 2, we use cardinality constrained stable sets again in Sect. 3 to design a stronger relaxation scheme, based on Lagrangean *decomposition* (LD). We explain how classical results from the literature guarantee the superiority of such a reformulation: both with respect to the quality of dual bounds, when compared to the straightforward relaxation, and with regard to the number of multipliers, when compared to an alternative framework to determine the same bounds (relax-and-cut dualizing violated subtour elimination constraints (2) dynamically).

We see the opportunity for renewed interest in LD in light of the progress in mixed-integer linear programming (MILP) computation. Given the impressive speedup of MILP solvers over the past two decades, Dimitris Bertsimas and Jack Dunn are among a group of distinguished researchers who make a case for (exact) optimization over integers as the natural, correct model for several tasks within machine learning and towards interpretable artificial intelligence. This is the theme of their recent book [4]; see also [5, 6]. We draw inspiration from this philosophy (challenging assumptions previously deemed computationally intractable) to propose less hesitation towards designing Lagrangean algorithms that exploit subproblems for which, albeit strongly NP-hard, specialized solvers attain good performance. Indeed, we present a proof of concept in the particular case of the MSTCC problem. We leverage a state-of-the-art branch-and-cut algorithm for fixed cardinality stable sets to an effective method to compute strong dual bounds for optimal stable spanning trees by means of LD.

In summary, our contributions are the following.

1. On the polyhedral combinatorics side, we present intersection properties and a bound on the dimension of the fixed cardinality stable set polytope, a relaxation of the stable spanning tree one.
2. We propose a sound analysis of different Lagrangean bounds published in the literature of the MSTCC problem, design a stronger reformulation based on LD, and justify its advantages both in theory and in a numerical evaluation. We make

a case for designing new algorithms combining LD and MILP solvers exploring strongly NP-hard subproblems

3. We present a free, open-source software package implementing the complete algorithm. It welcomes extensions and eventual collaborations, besides offering a series of useful, general-purpose algorithmic components, *e.g.* separation procedures, an LD based dual-ascent framework, an application of the Volume Algorithm framework implemented in COIN-OR.

## 2 Polyhedral results

As a first step towards knowledge about the polytope of stable spanning trees in a graph, we study elementary properties of the larger polytope $\mathfrak{C}(H,k)$ of fixed cardinality stable sets in the conflict graph $H=(E,C)$. The polyhedral results in this section serve their own purpose, and are not necessary for the reformulation and results presented in the remaining of the paper.

We begin with the necessary notation and terminology. For conciseness, we abbreviate "stable set of cardinality $k$" as *kstab* in this work. Let $[n]\stackrel{\text{def}}{=}\{1,\dots,n\}$, and let conv S denote the convex hull of a set $S$. Recall that the *incidence* (or *characteristic*) *vector* of a set $S\subset E=\{e_1,\dots,e_m\}$ is defined as $\chi^S\in\{0,1\}^{|E|}$ such that $\chi^S_i=1$ if and only if $e_i\in S$. The family of all incidence vectors of kstabs in $H$ is denoted $\mathcal{F}_{\text{kstab}}(H,k)$. Hence $\mathfrak{C}(H,k)\stackrel{\text{def}}{=}conv\ \mathcal{F}_{\text{kstab}}(H,k)$.

Also let $\mathcal{F}^{\uparrow}_{\text{kstab}}(H,k)\subset\{0,1\}^{|E|}$ denote the family of incidence vectors of stable sets of cardinality greater than or equal to $k$ in $H$, and let $\mathfrak{C}^{\uparrow}(H,k)\stackrel{\text{def}}{=}conv\ \mathcal{F}^{\uparrow}_{\text{kstab}}(H,k)$ denote their convex hull. Define $\mathcal{F}^{\downarrow}_{\text{kstab}}(H,k)$ and $\mathfrak{C}^{\downarrow}(H,k)$ analogously for stable sets of cardinality at most $k$. We omit the parameters $H$ and $k$ in such notation where it does not cause any confusion. Likewise, we occasionally omit the indices in summations over all coordinates of a point to make a passage more readable, *e.g.* $\sum\mathbf{x}$ when it clearly means $\sum_{i\in[n]}x_i$. Finally, let $ext\ \mathcal{P}$ denote the set of extreme points of a given polyhedron $\mathcal{P}$.

In the following, we present intersection properties connecting $\mathfrak{C}$, $\mathfrak{C}^{\uparrow}$, and $\mathfrak{C}^{\downarrow}$.

**Theorem 1** *Let $H$ be an arbitrary graph on $n$ vertices, and $k$ be a positive integer.*

*i.* $\mathfrak{C}(H,k)=\mathfrak{C}^{\uparrow}(H,k)\cap\mathfrak{C}^{\downarrow}(H,k)$.

*ii.* $\mathfrak{C}(H,k)=\mathfrak{C}^{\uparrow}(H,k)\cap F=\mathfrak{C}^{\downarrow}(H,k)\cap F$, *where* $F\stackrel{\text{def}}{=}\left\{x\in\mathbb{Q}^n\ :\ \sum_{u\in[n]}x_u=k\right\}$.

***Proof*** (*i*) $\mathfrak{C}\subseteq\mathfrak{C}^{\uparrow}\cap\mathfrak{C}^{\downarrow}$ follows from the fact that the convex hull of the intersection of two sets is contained in the intersection of the respective convex hulls.

For the other inclusion, let $\mathbf{x}^*\in\mathfrak{C}^{\uparrow}\cap\mathfrak{C}^{\downarrow}$ be arbitrary. Without loss of generality, we write $\mathbf{x}^*$ as a convex combination of $p$ vertices of $\mathfrak{C}^{\uparrow}$:

$$\mathbf{x}^*=\sum_{i\in[p]}\lambda_i\mathbf{y}^i,\ \text{with}\ \lambda_i\geq 0\ \text{for each}\ i,\ \sum_{i\in[p]}\lambda_i=1,\ \text{and}\ \left\{\mathbf{y}^i\right\}_{i\in[p]}\subseteq ext\mathfrak{C}^{\uparrow}.$$

Note that $\mathbf{y}^i \in \mathfrak{C}^\uparrow \implies \sum_{u\in[n]} y^i_u \geq k$ for each $i$. Now, if $\sum_{u\in[n]} y^i_u > k$ for some $i \in [p]$, we derive from $\lambda_i \geq 0$ and $\sum \lambda_i = 1$ that $\sum_{u\in[n]} x^*_u > k$, and $\mathbf{x} \notin \mathfrak{C}^\downarrow$. Hence $\sum_{u\in[n]} y^i_u = k$ for each $i \in [p]$, and $\left\{\mathbf{y}^i\right\}_{i\in[p]} \subseteq \mathfrak{C}$. By convexity of $\mathfrak{C}$, we conclude that $\mathbf{x}^* \in \mathfrak{C}$.

(*ii*) It is immediate that $\mathfrak{C} \subseteq \mathfrak{C}^\uparrow \cap F$: if $\mathbf{x}^* \in \mathfrak{C}$, we may write $\mathbf{x}^*$ as the convex combination of incidence vectors of kstabs, which is also a convex combination of vertices of $\mathfrak{C}^\uparrow$ within $F$.

For the other inclusion, observe that $\mathfrak{C}^\uparrow \cap F$ is the face of $\mathfrak{C}^\uparrow$ induced by valid inequality $\sum \mathbf{x} \geq k$. Let $\mathbf{x}^*$ denote a point in that face. Viewing the face as a polytope, $\mathbf{x}^*$ may be written as a convex combination of vertices of the face, which in turn are vertices of $\mathfrak{C}^\uparrow$ satisfying $\sum \mathbf{x} = k$. We thus write $\mathbf{x}^*$ as a convex combination of incidence vectors of kstabs, and $\mathbf{x}^* \in \mathfrak{C}$.

The proof is analogous for the second equality, observing that $F$ is the face determined by inequality $\sum \mathbf{x} \leq k$, valid for $\mathfrak{C}^\downarrow$. □

Note that it is not necessary that a vertex of the intersection of two polytopes is a vertex of any of the polytopes. For a counterexample, consider two squares $A$, $B$ in $\mathbb{Q}^2$ such that $A \cap B$ is another square; vertices of the intersection need not be vertices of $A$ or $B$. The result in Theorem 3 below shows a rather favourable situation when it comes to our cardinality constrained stable set polytopes. In order to prove it, we use the following fact, which is an elementary exercise in polyhedral theory, *e.g.* Exercise 3-8 in the 2017 lecture notes *Linear programming and polyhedral combinatorics*, by Michel Goemans (https://math.mit.edu/~goemans/18453S17/polyhedral.pdf). We remind the reader of the equivalence of extreme points, vertices, and basic feasible solutions of a polyhedron.

**Lemma 2** *Let* $\mathcal{P} = \{\mathbf{x} \in \mathbb{Q}^n : \mathbf{Ax} \leq \mathbf{b}, \mathbf{Cx} \leq \mathbf{d}\}$, *and* $\mathcal{Q} = \{\mathbf{x} \in \mathbb{Q}^n : \mathbf{Ax} \leq \mathbf{b}, \mathbf{Cx} = \mathbf{d}\}$. *It follows that ext* $\mathcal{Q} \subseteq$ *ext* $\mathcal{P}$.

***Proof*** If $\mathbf{x}^* \in$ *ext* $\mathcal{Q}$, then $\mathbf{x}^*$ is a basic feasible solution of $\mathcal{Q}$. Let $I$ denote the subset of indices of constraints in $\mathbf{Ax} \leq \mathbf{b}$ that are active at $\mathbf{x}^*$, which is thus the unique solution of the subsystem

$$\begin{cases} \mathbf{a}_i\mathbf{x} &= b_i, \quad \text{for } i \in I, \\ \mathbf{Cx} &= \mathbf{d}. \end{cases} \tag{6}$$

This subsystem also corresponds to a selection of inequalities in the definition of $\mathcal{P}$ to be satisfied with equality. The same $n$ linearly independent constraint vectors in (6) determine that $\mathbf{x}^*$ is a basic solution of $\mathcal{P}$. Since $\mathbf{x}^* \in \mathcal{P}$ as well, it follows that $\mathbf{x}^* \in$ *ext* $\mathcal{P}$. □

**Theorem 3** *ext*$\mathfrak{C}(H,k) =$ *ext*$\mathfrak{C}^\uparrow(H,k) \cap$ *ext*$\mathfrak{C}^\downarrow(H,k)$ *for arbitrary H and k.*

***Proof*** Let $\mathbf{x}^*$ denote a vertex of both $\mathfrak{C}^\uparrow$ and $\mathfrak{C}^\downarrow$. Then $\mathbf{x}^*$ is the incidence vector of a kstab in $H$, and $\mathbf{x}^* \in$ *ext*$\mathfrak{C}$. For the other inclusion, we use Lemma 2 twice: once with $\mathcal{P}$ denoting a description of $\mathfrak{C}^\uparrow$ (whence $\mathcal{Q}$ is identified with $\mathfrak{C}$, by item

(*ii*) in Theorem 1) to show that $ext\mathfrak{C} \subseteq ext\mathfrak{C}^{\uparrow}$, and again with $\mathcal{P} = \mathfrak{C}^{\downarrow}$ to show that $ext\mathfrak{C} \subseteq ext\mathfrak{C}^{\downarrow}$. □

**Corollary 4** *Let $H$ be a graph on $n$ vertices, and $k$ be a positive integer. Also let $\mathcal{P} = \left\{ \mathbf{x} \in \mathbb{Q}^n : \mathbf{Ax} \leq \mathbf{b}, \sum_{u\in[n]} x_u \geq k \right\}$ be a formulation for stable sets of cardinality at least $k$ in that graph, that is, $\mathcal{P} \cap \{0,1\}^n = \mathcal{F}^{\uparrow}_{\text{kstab}}(H,k)$. If $\mathcal{P}$ is actually integral ($\mathcal{P} = \mathfrak{C}^{\uparrow}$), then so is the formulation $\mathcal{P}' = \left\{ \mathbf{x} \in \mathbb{Q}^n : \mathbf{Ax} \leq \mathbf{b}, \sum_{u\in[n]} x_u = k \right\} = \mathfrak{C}(H,k)$. The analogous result holds for $\mathfrak{C}^{\downarrow}(H,k)$.*

These results might be explored in future work that benefit from optimizing over kstabs with a reformulation based on stable sets of *bounded* cardinality. They may also be useful when dealing with classes of graphs for which an explicit characterization of the corresponding polytopes $\mathfrak{C}^{\uparrow}$ or $\mathfrak{C}^{\downarrow}$ is known.

Finally, we give a lower bound on the dimension of the polytope $\mathfrak{C}(H,k)$ as a function of the stability number $\alpha(H)$, that is, the size of the largest stable set in $H$.

**Theorem 5** *Let $k$ be a positive integer, and $H$ be an arbitrary graph on $n$ vertices such that $\alpha(H) \geq k+1$. Then $\alpha(H) - 1 \leq \dim \mathfrak{C}(H,k) \leq n-1$.*

***Proof*** The upper bound is trivial, given the presence of the cardinality constraint in the equality system of any linear inequality description of $\mathfrak{C}(H,k)$. For the lower bound, we prove by induction on $\alpha(H)$ that we can find $\alpha(H)$ linearly independent (l.i.) incidence vectors of kstabs in $H$. The result then follows immediately.

Suppose first that $\alpha(H) = k+1$, and let $\chi \in \mathfrak{C}$ be the incidence vector of a stable set of cardinality $k+1$ in $H$. Let $I \subset [n]$, $|I| = k+1$, denote the coordinates corresponding to vertices in that stable set, that is, $\chi_i = 1$ for each $i \in I$. Denoting the $i$-th unit vector in $\mathbb{R}^n$ by $\mathbf{e}^i$, we have that $\left\{ \chi - \mathbf{e}^i \right\}_{i\in I}$ are $k+1$ l.i. points in $\mathfrak{C}(H,k)$.

Assume inductively that we can determine $p$ l.i. incidence vectors of kstabs in a graph if its stability number is equal to $p$. Now, given $H$ such that $\alpha(H) = p+1$, and $\chi$ the incidence vector of a maximum stable set in $H$, we may proceed as above to again determine $p+1$ l.i. incidence vectors of *pstabs* (cardinality $p$ stable sets) in $H$. Let $\phi, \psi$ be two such vectors.

As the subgraph induced by $\phi$ has no edges, we have $\alpha(H[\phi]) = p$. The inductive hypothesis thus yields a collection $\left\{ \chi^1, \ldots, \chi^p \right\} \subset \{0,1\}^p$ of l.i. incidence vectors of kstabs in the induced subgraph. Let $\left\{ \overline{\chi}^1, \ldots, \overline{\chi}^p \right\}$ be the lifting of this collection to space $\mathbb{R}^n$ with zeros in the coordinates corresponding to missing vertices.

Since $\phi$ and $\psi$ are l.i., we claim that it is possible to discard $p-k$ vertices from the stable set induced by $\psi$ in such a way that the incidence vector $\overline{\psi}$ of the resulting kstab is l.i. of $\left\{ \overline{\chi}^1, \ldots, \overline{\chi}^p \right\}$. Indeed, $\phi$ and $\psi$ induce *different* pstabs, so that there exists a vertex in the subgraph induced by $\psi$ that is not in the subgraph induced by $\phi$. Let $u \in [n]$ be such that $\psi_u = 1$, $\phi_u = 0$, and choose $\overline{\psi}$ (kstab inducing) with $\overline{\psi}_u = 1$. In turn, note that $\overline{\chi}^j_u = 0$ for each $j \in [p]$, by construction: from $\phi_u = 0$ it

follows that $u$ is one of the coordinates padded with zero when mapping $\chi^j$ to $\overline{\chi}^j$. This means that $\overline{\psi} \notin span\left\{\overline{\chi}^1, \ldots, \overline{\chi}^p\right\}$, and hence we determine $p+1$ l.i. incidence vectors of kstabs in $H$, completing the proof. □

We remark that the down-monotone polytope $\mathfrak{C}^{\downarrow}(H,k)$ is full-dimensional for arbitrary $H$ and $k$, as it contains the $|V(H)|+1$ affinely independent points corresponding to the unit vectors and zero. The problem of determining $\dim \mathfrak{C}$ may therefore be cast in terms of $\mathfrak{C}^{\uparrow}$ in future research.

## 3 Lagrangean relaxation and decomposition

In this section, we present the main contributions of the paper. We give special attention to justifying carefully the drawbacks of previous reformulations based on Lagrangean duality, and how a decomposition approach optimizing over the fixed cardinality stable set polytope leads to an *effective* algorithm to compute strong dual bounds for optimal stable trees.

In this section, *effectiveness* is taken from the analytical point of view: we argue that the decomposition is superior *in theory* both with respect to bound quality and tractability of the dual problem. In the next section, we discuss the practical evaluation of our (free, open-source) software implementing the resulting algorithm, and argue that it indeed contributes as an effective tool to determine tight dual bounds on a representative subset of benchmark instances of the problem.

### 3.1 Drawbacks of existing Lagrangean approaches for MSTCC

The work of [33] contributes in many research directions about stable spanning trees, including particular cases which are polynomially solvable, feasibility tests, several heuristics, and two exact algorithms based on Lagrangean relaxation. The first formulation is straightforward, dualizing all conflict constraints (4); they denote the corresponding dual bound $L^*$. The second approach relaxes a subset of inequalities (4): using an approximation to the maximum edge clique partitioning problem [14], this scheme dualizes a subset of conflict constraints such that the remaining conflict graph is a collection of disjoint cliques; the resulting dual bound is denoted $\ell^*$. The authors argue that the latter reformulation is stronger than the former, and present extensive computational results justifying their claims.

Unfortunately, the Lagrangean dual bounds $L^*$ and $\ell^*$ in [33] are in fact identical, as we show next. The first relaxation clearly has the integrality property, as the remaining constraints correspond to a description of the spanning tree polytope or, equivalently, to bases of the graphic matroid of $G$ [16]. The second relaxation scheme is designed so that the conflict constraints which remain in the subproblem of relaxation $\ell^*$ induce a collection of disjoint cliques in $H$. The subproblem thus corresponds to the intersection of two matroids: the graphic matroid of $G$ and the partition matroid of subsets of $E$ that intersect the enumerated cliques in $H$ at most once. It follows that the second relaxation also has the integrality property

[24, Theorem III.3.5.9], and consequently, $L^*$ and $\ell^*$ both equal the optimal objective function value in the continuous relaxation of (1–5) [24, Corollary II.3.6.6]. In this perspective, the computational results in Tables 2–4 of [33] diverge from what Lagrangean duality theory prescribes.

Recently, [11] presented thorough computational experiments of a new Lagrangean algorithm for the MSTCC problem. They use the same relaxation scheme dualizing all conflict constraints, and focus on a combination of dual ascent and the subgradient method to compute the Lagrangean bound, namely, $L^*$ in [33], equal to the LP-relaxation of (1–5). In Table 1 of [11], the performance of the new algorithm is compared to the results published in [33]. That is, the issue we analyse above regarding the computational results of [33] is repeated as a baseline of the new numerical evaluation.

Another drawback of the new algorithm is that dual ascent steps are intertwined with subgradient optimization. While *not* incorrect, this choice undermines the advantages of a strategy to solve the dual problem in fewer iterations. A passage from a classical work of Guignard and Rosenwein [22] is conclusive: *"An ascent procedure may also serve to initialize multipliers in a subgradient procedure. This scheme is particularly useful at the root node of an enumeration tree. However, an ascent method cannot guarantee improved bounds over bounds obtained by solving the Lagrangean dual with a subgradient procedure."*

Moreover, the ascent steps rely on a greedy heuristic, and not on *maximal ascent directions*, *i.e.* optimal step size in a direction of bound increase; see Definition 7. In the algorithm of [11], if a conflicting pair of edges exists in a Lagrangean solution, the multiplier adjustment is derived from the observation that the dual bound shall improve by at least the increased cost of replacing one of the edges by its cheapest successor (in a list of edges ordered by current costs). The authors remedy the resulting low adjustment values by alternating subgradient optimization iterations and the ascent procedure.

We stress again that references [11] and [33] have many virtues and present concrete contributions to the MSTCC literature. Our only remark is that the first Lagrangean strategy designed to improve upon the LP-relaxation bound is matter-of-factly yet to be introduced. In the next sections, we offer an interesting approach to tackle this challenge.

### 3.2 Lagrangean decomposition

Renaming the variables in (4) as $\mathbf{y}$, and introducing linking constraints $x_e = y_e$ for each $e \in E$, we have the same formulation. Now, dualizing the linking constraints with Lagrangean multipliers $\lambda \in \mathbb{Q}^{|E|}$, we arrive at the Lagrangean decomposition (LD) formulation:

$$z(\lambda) \overset{\text{def}}{=} \min_{\mathbf{x} \in \mathcal{F}_{\text{sp.tree}}(G)} (\mathbf{w} - \lambda)^{\intercal}\mathbf{x} + \min_{\mathbf{y} \in \mathcal{F}_{\text{kstab}}(H, |V|-1)} \lambda^{\intercal}\mathbf{y} \tag{7}$$

where $\mathcal{F}_{\text{sp.tree}}(G)$ is given by

$$\sum_{e \in E(S)} x_e \leq |S| - 1, \qquad \text{for each } S \subsetneq V, S \neq \emptyset, \tag{8}$$

$$\sum_{e \in E} x_e = |V| - 1, \tag{9}$$

$$x_e \in \{0, 1\}, \qquad \text{for each } e \in E, \tag{10}$$

and $\mathcal{F}_{\text{kstab}}(H, |V| - 1)$ is as in Sect. 2, given by

$$\sum_{e \in E} y_e = |V| - 1, \tag{11}$$

$$y_{e_i} + y_{e_j} \leq 1, \text{ for each } \left\{e_i, e_j\right\} \in C, \tag{12}$$

$$y_e \in \{0, 1\}, \text{ for each } e \in E. \tag{13}$$

The Lagrangean dual problem is to determine the tightest such bound:

$$\zeta \stackrel{\text{def}}{=} \max_{\lambda \in \mathbb{Q}^{|E|}} \{z(\lambda)\}. \tag{14}$$

The first systematic study of LD as a general purpose reformulation technique was presented by [21]. They indicate earlier applications of variable splitting/layering, especially [27] and [31]. See also the outstanding presentation in [20, Section 7].

One of the main virtues of the decomposition principle over traditional Lagrangean relaxation schemes is that the bound from the LD dual is equal to the optimum of the primal objective function over the intersection of the convex hulls of both constraint sets [21, Corollary 3.4]. The decomposition bound is thus equal to the strongest of the two Lagrangean relaxation schemes corresponding to dualizing either of the constraint sets.

In our application to the MSTCC problem, we recognize the integrality of the spanning tree formulation described by (8–9) over $\mathbf{x} \in \mathbb{Q}^{|E|}$, following a classical result of Edmonds [16]. Hence the decomposition bound matches that of the stronger scheme where constraints (11–12) enforcing fixed cardinality stable sets are kept in the subproblem (which is thus *convexified*), and all subtour elimination constraints (8) are dualized. This means that we can compute stronger Lagrangean bounds, while limiting the number of multipliers in the dual problem to $|E|$, instead of dealing with exponentially many multipliers *e.g.* in a relax-and-cut approach.

We defend the advantages of breaking the original problem into two parts, exploiting their rich combinatorial and polyhedral structures, so as to derive stronger dual bounds. The price of this strategy is to solve a strongly NP-hard subproblem, which naturally leads to the design of more sophisticated dual algorithms, requiring the fewest iterations possible.

### 3.3 Dual algorithm

We combine two techniques to solve the problem of approximating $\zeta$ in the dual problem (14). The first is customized dual ascent, an *ad-hoc*, analytical method that integrates naturally with LD [21]. It guarantees monotone bound improvement, and could be employed as a stand-alone dual algorithm – though likely converging to a sub-optimal bound $z(\lambda^*) < \zeta$ due to incomplete information of ascent directions. We circumvent this by continuing the search (from the dual ascent solution $\lambda^*$) with an iterative, subgradient-based method: the Volume Algorithm (VA) of [3].

Proposed as an extension of subgradient optimization to attain better numerical results, VA was later characterized by [2] as an intermediate method between classical subgradient and more robust bundle methods, using combinations of past and present subgradient vectors available at each iteration. In contrast to most bundle-type methods, which require the solution of a potentially expensive quadratic program, the computation of a new dual point in VA uses a correction factor determined by a simple recurrence relation. The revision of [2] introduces a classification of *green/yellow/red* steps, like *serious/null* ones in bundle methods, and demonstrates the theoretical convergence of such revised VA. The combined simplicity and comparatively good computational experience reported in applications of VA make it an attractive alternative; see [8] for a systematic evaluation.

***Remark 6*** Like many other subgradient-like methods, the Volume Algorithm also determines primal sequences of (fractional) points approximating the dual optimal solution. We do not explore this aspect in the present work. See our suggestions for further research in the discussion following our numerical results in Sect. 4.3.

Since VA is precisely defined, and we use it as a black-box solver, the remainder of this section is devoted to its initialization by the dual ascent procedure. In what follows, let $\mathbf{e}_i \in \mathbb{R}^m$ denote the standard unit vector in the $i$-th direction, and $\mathcal{P}_{\text{sp.tree}}(G) \overset{\text{def}}{=} conv\ \mathcal{F}_{\text{sp.tree}}(G)$ denote the spanning tree polytope of graph $G$. Note that $\mathcal{P}_{\text{sp.tree}}$ and $\mathfrak{G}$ are bounded (polytopes contained in the 0,1 hypercube), and do not contain extreme rays.

The Lagrangean dual function $z : \mathbb{Q}^{|E|} \to \mathbb{Q}$ is an implicit function of $\lambda$. It is determined by the lower envelope of $\left\{ (\mathbf{w} - \lambda)^\intercal \mathbf{x}^r + \lambda^\intercal \mathbf{y}^s \ :\ \mathbf{x}^r \in ext\ \mathcal{P}_{\text{sp.tree}}(G), \mathbf{y}^s \in ext\mathfrak{G}(H, |V| - 1) \right\}$. Hence, it is piecewise linear concave, and differentiable almost everywhere, with breakpoints at all $\lambda'$ where the optimal solution to $z(\lambda')$ is not unique.

Such breakpoints are the key ingredient in the dual ascent paradigm to solve a Lagrangean dual problem. In particular, the following kind of point deserves special attention to guide progress in this framework.

**Definition 7** A **maximal ascent direction** of the Lagrangean dual function $z : \mathbb{Q}^m \to \mathbb{Q}$ at $\lambda^r$ is a vector $\mathbf{u} \in \mathbb{Q}^m$ satisfying two conditions: (i) $\mathbf{u}$ determines a direction of increase from $z(\lambda^r)$, *i.e.* $z(\lambda^r + \mathbf{u}) > z(\lambda^r)$; (ii) $\lambda^r + \mathbf{u}$ is a breakpoint of $z$, that is, if $(\mathbf{x}^r, \mathbf{y}^r)$ is an optimal solution to $z(\lambda^r)$, then $(\mathbf{x}^r, \mathbf{y}^r)$ also optimizes $z(\lambda^r + \mathbf{u})$, but it is not the unique solution.

A maximal ascent direction determines an optimal multiplier adjustment in a given direction of increase of the Lagrangean dual function. It need not correspond to a *steepest* ascent direction from $z(\lambda^r)$, in general.

The technique of optimizing the Lagrangean dual function by means of ascent directions uses the formulation structure to determine monotone bound improving sequences of multipliers. It was pioneered by [7] and [17] in the context of the facility location problem. An actual algorithm of this kind thus relies on analysing the specific problem and the information available from subproblem solutions. Although there is no pragmatic, problem-independent algorithm, we found it instructive to summarize and systematically review the following instructions in the derivation of our results.

***Remark 8*** [*Guiding principle of LD based dual ascent*] We may derive a maximal ascent direction by analysing the implications of updating a single multiplier $\lambda_e$, corresponding to a violation $x_e \neq y_e$. The update must improve the Lagrangean dual bound and induce an alternative optimal solution.

To avoid overloading the notation in the next two results, we omit the transposition symbol in vector products like $(\mathbf{w} - \lambda^r)^\intercal \mathbf{x}^r$.

**Theorem 9** *Let $e \in E$ and let $(\mathbf{x}^r, \mathbf{y}^r)$ be an optimal solution to subproblem $z(\lambda^r)$, such that $x^r_e = 0 < 1 = y^r_e$. Define the non-negative quantities*

$$\Delta^r_{-e} \stackrel{\text{def}}{=} \min \left\{ \lambda^r \mathbf{y} \,:\, \mathbf{y} \in \mathcal{F}_{\text{kstab}}(H, |V| - 1), y_e = 0 \right\} - \lambda^r \mathbf{y}^r, \tag{15}$$

$$\partial^r_{+e} \stackrel{\text{def}}{=} \min \left\{ (\mathbf{w} - \lambda^r)\mathbf{x} \,:\, \mathbf{x} \in \mathcal{F}_{\text{sp.tree}}(G), x_e = 1 \right\} - (\mathbf{w} - \lambda^r)\mathbf{x}^r. \tag{16}$$

*If* $\min \left\{ \Delta^r_{-e}, \partial^r_{+e} \right\} \neq 0$, *then* $\min \left\{ \Delta^r_{-e}, \partial^r_{+e} \right\} \cdot \mathbf{e}_e$ *is a maximal ascent direction of $z$ at $\lambda^r$.*

***Proof*** See [29, Theorem 4.2]. □

We remark that determining a minimum spanning tree with edge $e = \{i, j\}$ fixed *a priori* in (16) can be accomplished efficiently by *contracting* that edge in $G$. If the contraction operator is defined so as to allow parallel edges between the new vertex $ij$ and $k \in N(i) \cap N(j)$, where $N(u) \subset V$ denotes the neighbourhood of vertex $u$, we must ensure that not more than one edge between two vertices is chosen (*e.g.* in Kruskal's algorithm; this is not an issue in Prim's method). Now, if the contraction operator forbids parallel edges, we make an unambiguous choice in the original

graph $G$ by recognizing the proper edge ($\{i,k\}$ or $\{j,k\}$) yielding the correct spanning tree.

The next result is analogous, now identifying maximal ascent directions from Lagrangean solutions where $x^r_e = 1$ but $y^r_e = 0$.

**Theorem 10** *Let $e \in E$ and let $(\mathbf{x}^r, \mathbf{y}^r)$ be an optimal solution to subproblem $z(\lambda^r)$, such that $x^r_e = 1 > 0 = y^r_e$. Define the non-negative quantities*

$$\Delta^r_{+e} \overset{\text{def}}{=} \min\left\{\lambda^r \mathbf{y} \,:\, \mathbf{y} \in \mathcal{F}_{\text{kstab}}(H, |V|-1), y_e = 1\right\} - \lambda^r \mathbf{y}^r, \tag{17}$$

$$\partial^r_{-e} \overset{\text{def}}{=} \min\left\{(\mathbf{w}-\lambda^r)\mathbf{x} \,:\, \mathbf{x} \in \mathcal{F}_{\text{sp.tree}}(G), x_e = 0\right\} - (\mathbf{w}-\lambda^r)\mathbf{x}^r. \tag{18}$$

*If $\min\left\{\Delta^r_{+e}, \partial^r_{-e}\right\} \neq 0$, then $\min\left\{\Delta^r_{+e}, \partial^r_{-e}\right\} \cdot \left(-\mathbf{e}_e\right)$ is a maximal ascent direction of $z$ at $\lambda^r$.*

***Proof*** See [29, Theorem 4.3]. □

## 4 Experimental evaluation

The main goal of our computational endeavour is to assess the strength of the LD bound $\zeta = \max_{\lambda \in \mathbb{Q}^{|E|}} \{z(\lambda)\}$ in (14) over benchmark instances of the MSTCC problem. This is fundamental to verify the practicality of that reformulation, as well as to understand its limitations.

A second intention of the project is to offer a careful implementation of the complete algorithm as a free, open-source software package. The code was crafted with attention to time and space efficiency, fairly tested for correctness, and is available in the LD-davol repository on GitHub (https://github.com/phillippesamer/stable-trees-ld-davol). It welcomes collaboration towards extensions and facilitates the direct comparison with eventual algorithms designed for the MSTCC problem in the future, besides offering useful, general-purpose algorithmic components. In the remainder of this section, we refer to our implementation of the algorithm by its repository name, LD-davol.

### 4.1 Implementation details

LD-davol is written in C++, with the support of two libraries integrating the COIN-OR project [23], as we describe next. We also include the preprocessing algorithm introduced by [30], a collection of probing tests that removes variables and identifies implied conflicts in the original input instance.

Recall that the two building blocks of the dual algorithm presented in Sect. 3.3 are a dual ascent initialization, followed by the Volume Algorithm. For the latter, we use the implementation in COIN-OR Vol (see https://github.com/coin-or/Vol, and

the overview document "An implementation of the Volume Algorithm" by F. Barahona and L. Ladanyi in the same repository).

There are two Lagrangean subproblems to solve in each iteration of both the dual ascent and the volume procedures. We solve the minimum spanning tree subproblem in the original graph $G = (V, E)$ using the efficient implementation of Kruskal's algorithm in COIN-OR LEMON 1.3.1 [15], while we solve the fixed cardinality stable set subproblem in the conflict graph $H = (E, C)$ with a branch-and-cut algorithm, implemented using the Gurobi 9.5.1 solver.

We reinforce formulation (11–13) with two further classes of valid inequalities from the classic stable set polytope, exactly as first presented by [30] for the MSTCC problem. Namely, odd-cycle inequalities

$$\sum_{u \in U} y_u \leq \frac{|U| - 1}{2}, \text{ for each } U \subset E \text{ inducing an odd-cycle in } H, \tag{19}$$

are added dynamically using the separation algorithm of [19, Remark 1], while maximal clique inequalities

$$\sum_{u \in Q} y_u \leq 1, \text{ for each } Q \subset E \text{ inducing a maximal clique in } H, \tag{20}$$

are enumerated *a priori* using the algorithm of [32], since this can be done efficiently over the MSTCC benchmark instances. The interested reader is referred to [30], as well as the eminently readable tutorial by [26].

### 4.2 Experimental setup and benchmark instances

Our computational evaluation was performed on a desktop machine with an Intel® Core™ i5-8400 processor, with 6 CPU cores at 2.80 GHz, and 16 GB of RAM, runnning GNU/Linux kernel 5.4.0 under the Ubuntu 18.04.1 distribution. All the code is compiled with g++ 7.5.0, and we consider a numerical precision of $10^{-10}$. We limit the execution time to 3600 seconds, allowing the dual ascent procedure to run for at most 1800 seconds, and the volume algorithm to run for the remaining time.

After preliminary experiments with the different algorithm parameters, we considered that the following combination exhibits better performance. Dual ascent follows the first maximal ascent direction available in each iteration (instead of identifying the steepest ascent). The volume algorithm implementation from COIN-OR is used with default parameters, except for screen log settings and warm-starting with the multipliers found by dual ascent. Gurobi 9.5.1 is used with default settings, except for screen log settings and switches to indicate the presence of the callback for user cuts. Odd-cycle inequalities are generated only at the root node of the enumeration tree, with the following strategy for balancing bound quality and cut pool size. When separating a relaxation solution, only the most violated cut and those close to being orthogonal to it are added; we accept hyperplanes having inner product of 0.01 or less with the most violated one.

There are two sets of benchmark instances for evaluating MSTCC algorithms. The original one was proposed by [33], and more recently [10] introduced a new collection. The total number of instances can be misleading, as only a small fraction correspond to interesting (*i.e.* computationally challenging) problems. Moreover, it is not possible to discriminate the hard ones by the input size, especially in the latter collection. More specifically, the available problem instances fall into three categories.

i. *Type 1* instances in [33]: 23 instances, most of which are difficult; 12 still have an open optimality gap in the experiments discussed in the literature.
ii. *Type 2* instances in [33]: 27 instances, all of which are trivial; the preprocessing algorithm of [30] solves (or reduces to a classic MST problem without conflicts) all of them in negligible time.
iii. Instances introduced by [10]: 180 instances, 107 of which (spanning each group of the collection, ordered by $|E|$) are easily solved within few seconds. The remaining 73 instances are interesting. The collection was only considered in that original work and continuing research from the same group [9, 11].

In summary, only instances in (*i*) and less than half of the large collection in (*iii*) serve the purpose of benchmarking MSTCC algorithms, in our opinion. Our discussion contemplates both benchmarks in full, but we choose to include full numerical results for the instances in (*i*) in the next section, while longer tables corresponding to (*iii*) are present in Appendix (online supplement).

### 4.3 Numerical results

We present the information on bound quality and computing time for three classes of dual bounds: the combinatorial bound corresponding to the kstab relaxation (also the first subproblem solved in LD-davol), the LP relaxation bound, and the LD bound, *i.e.* the approximation of $\zeta$ by LD-davol. For a fair, unbiased comparison, note that the linear program whose bound we refer by LP is also reinforced with odd-cycle and clique inequalities in (19–20).

Table 1 covers *type 1* instances in the original benchmark of [33] (apart from three that could be identified efficiently as infeasible in previous works). In this set, a problem defined on a graph $(V, E)$ and conflict set $C$ has identifier `z|V|-|E|-|C|`. Tables 2, 3, 4 and 5 in Appendix (online supplement) contain the corresponding results over instances proposed by [10]. The second column in each table contains the instance optimal value, or the best dual bound reported in the literature (we mark instances with unknown optimal solution with an asterisk$^*$).

Given the time limit that we allocate to the dual algorithms, we only report LD-davol results for instances where the kstab bound is computed within 1800 seconds. If that is not the case, we report the available dual bound for the fractional kstab relaxation and the corresponding entry appears with a mark ($z^\dagger$). Moreover, we use boldface ($\mathbf{z}^\dagger$) in case this bound is actually stronger than those previously appearing

in the literature. We remark that $\zeta$, or any Lagrangean bound, is greater than or equal to the LP bound. Nevertheless, in the seven cases where the approximation attained by LD-davol is an inferior bound, a negative number appears in the *% above LP* column. Finally, if the Lagrangean bound is better than the previously best known bound (applies only to instances with unknown optima), a negative value in bold appears in the *% from OPT* column.

We read from Table 1 that the Lagrangean bound can be up to 27.21% above the LP relaxation one. We consider it even more remarkable that LD-davol computes $\zeta$ exactly and this actually matches the optimum in 2 instances in this collection, and in 73 instances out of 180 in the remaining tables. Otherwise, the bound is within 9% of the optimum. This figure actually corresponds to one of two outliers in this table, where LD-davol does not improve on the initial kstab bound; disregarding instance `z100-500-3741`, the bound is within 5.5% of the optimum across all experiments.

Concerning the instances introduced by [10], the bound is within

(i) 2.1% of the optimum in instances with 25 vertices ($60 \leq |E| \leq 120$, $18 \leq |C| \leq 500$);
(ii) 4.4% of the optimum in instances with 50 vertices ($245 \leq |E| \leq 490$, $299 \leq |C| \leq 8387$);
(iii) 2.6% of the optimum in instances with 75 vertices ($555 \leq |E| \leq 1110$, $1538 \leq |C| \leq 43085$);
(iv) 0.1% of the optimum in instances with 100 vertices ($990 \leq |E| \leq 1980$, $4896 \leq |C| \leq 137145$).

The initial kstab bound is the only one computed in 8 out of 20 instances in Table 1 (45 out of 180 instances in the remaining tables). Nevertheless, in 5 of these cases (respectively, in 39 of those 45) it is stronger than the previously known best bound. Note that, even though the machines and implementations cannot be compared directly, the 1800 second time limit set for this initial combinatorial relaxation is much lower than the standard (5000s) used in the literature of the MSTCC problem.

The main negative remark is as expected: the LD bound might be too expensive to compute. Even though it can be efficiently determined in a large number of instances (*e.g.* at most sixty seconds for 96 cases across all tables), the execution of LD-davol is terminated due to the time limit in 4 instances appearing in Table 1 (29 appearing in the other tables). An intuitive rule of thumb is that LD-davol yields stronger bounds in reasonable time as long as the combinatorial relaxation bound (the initial kstab problem) can be computed in reasonable time.

We avoid direct comparison of implementations/solvers altogether. As declared in the beginning of this section, our goal is to assess the strength and practicality of our ideas: exploring fixed cardinality stable sets and the reformulation by LD. It should be clear from our numerical results that the method yields high-quality dual bounds in the allotted computing time. It is probably not suited for embedding in a branch-and-bound scheme without successful work on heuristic aspects,

**Table 1** Results attained over hard instances in the original benchmark

| Instance | | KSTAB | | LP | | LD-davol | | | |
|---|---|---|---|---|---|---|---|---|---|
| ID | OPT | Bound | Time (s) | Bound | Time (s) | Bound | Time (s) | % above LP | % from OPT |
| z50-200-199 | 708 | 612 | 0.0 | 706 | 0.0 | 705 | 1.2 | −0.14 | 0.4 |
| z50-200-398 | 770 | 652 | 0.0 | 770 | 0.1 | 770 | 1.4 | 0 | 0 |
| z50-200-597 | 917 | 726 | 0.0 | 876 | 0.1 | 900 | 12.7 | 2.74 | 1.9 |
| z50-200-995 | 1324 | 1164 | 0.3 | 1037 | 0.0 | 1251 | 315.9 | 20.64 | 5.5 |
| z100-300-448 | 4041 | 3440 | 0.0 | 4038 | 0.6 | 4037 | 5.0 | −0.02 | 0.1 |
| z100-300-897 | 5658 | 4785 | 0.0 | 5070 | 0.4 | 5371 | 1402.2 | 5.94 | 5.1 |
| z100-300-1344 | 6635.4* | 6970 | 563.1 | 5479 | 0.2 | 6970 | 3602.9 | 27.21 | **−5.0** |
| z100-500-1247 | 4275 | 3454 | 0.0 | 4275 | 0.7 | 4275 | 10.0 | 0.02 | 0 |
| z100-500-2495 | 5997 | 5022 | 0.1 | 5363 | 0.4 | 5693 | 2225.9 | 6.15 | 5.1 |
| z100-500-3741 | 6707.8* | 6101 | 2.5 | 5830 | 0.3 | 6101 | 3609.2 | 4.65 | 9.0 |
| z100-500-6237 | 7729.3* | **8506**† | 1800.0 | 6789 | 0.3 | – | – | – | – |
| z100-500-12474 | 10560.2* | 10506† | 1800.0 | 9008 | 1.3 | – | – | – | – |
| z200-600-1797 | 13171.2* | 12213 | 0.1 | 12580 | 5.5 | 12993 | 3603.7 | 3.28 | 1.4 |
| z200-600-3594 | 17595.0* | **17785**† | 1800.0 | 14763 | 2.5 | – | – | – | – |
| z200-800-3196 | 20941.5* | 18477 | 0.0 | 20002 | 5.0 | 20437 | 3609.3 | 2.17 | 2.4 |
| z200-800-6392 | 26526.7* | **27124**† | 1800.0 | 22923 | 3.3 | – | – | – | – |
| z200-800-9588 | 30634.2* | **31132**† | 1800.0 | 27616 | 2.5 | – | – | – | – |
| z200-800-15980 | 36900.2* | 34648† | 1800.0 | 32050 | 1.6 | – | – | – | - |
| z300-1000-4995 | 51398.4* | **51621**† | 1800.0 | 45599 | 10.5 | – | – | – | – |
| z300-1000-9990 | 61878.9* | 61732† | 1800.0 | 54593 | 16.4 | – | – | – | – |

namely: learning effective LD-davol parameters – especially setting a time limit in each node, implementing *repair heuristics* to search for primal solutions from the sequence of fractional points produced by the Volume Algorithm, as well as designing local search methods to explore neighbourhoods of the kstab and spanning tree solutions found during the Lagrangean subproblems. (Note that [11] describes successful results from such a Lagrangean heuristic derived from the integral relaxation scheme discussed in Sect. 3.1.) Alternatively, one could experiment with calling LD-davol selectively in a branch-and-cut framework to strengthen dual bounds, *e.g.* when an incumbent solution is found, or when the optimality gap is not decreasing effectively.

Additional ideas that we leave for future work include improving the kstab subproblem solver, fine-tuning the Volume Algorithm to perform faster, and experimenting with different dual methods *e.g.* the sophisticated framework for subgradient optimization made available by [18], or, more ambitiously, the approximate solution using nonsmooth optimization techniques with *inexact* function/subgradient evaluation [25].

## 5 Concluding remarks

Stable spanning trees are not only interesting structures in combinatorial optimization, but pose a computationally challenging problem. We explore a new relaxation (fixed cardinality stable sets) to present polyhedral results and to derive stronger Lagrangean bounds. The latter builds on a careful analysis of different relaxation schemes, both old and new. Our Lagrangean decomposition (LD) bounds are also evaluated in practice, using a dual method comprising an original dual-ascent initialization followed by the Volume Algorithm. Finally, we also made great efforts to offer a high-quality, useful, open-source software in a free repository.

The LD bound actually matches the optimum in 75 out of 200 benchmark instances. We verify that, in at least 146 of these instances (where the kstab subproblem can be solved fast enough), the LD bound is within 5.5% of the optimum or the best known bound. In 44 of the remaining instances, the initial combinatorial bound from kstabs at least improves the previously known best bounds.

We reinforce the position put forth at the end of the introduction. In light of the progress in MILP computation, it seems worthwhile to further investigate the strategy of LD based on harder subproblems, possibly replacing the common sense boundary of weakly NP-hard choices by the weaker requirement that our choice be *computationally tractable*.

**Supplementary Information** The online version contains supplementary material available at (https://doi.org/10.1007/s11590-022-01949-8).

**Acknowledgements** This research is partly supported by the Research Council of Norway through the research project 249994 CLASSIS.

**Author contributions** Conception and design of study: PS; Investigation and methodology: PS and DH; Software implementation: PS; Writing - original draft: PS; Writing - review and editing: PS and DH.

**Funding** Open access funding provided by University of Bergen (incl Haukeland University Hospital).

**Data availability** The complete source code discussed in this work, as well as benchmark datasets, is publicly available in the LD-davol repository on GitHub (https://github.com/phillippesamer/stable-trees-ld-davol).

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

## References


1. Assad, A., Xu, W.: The quadratic minimum spanning tree problem. Naval Res. Logist. (NRL) **39**(3), 399–417 (1992). https://doi.org/10.1002/1520-6750(199204)39:33.0.CO;2-0
2. Bahiense, L., Maculan, N., Sagastizábal, C.: The volume algorithm revisited: relation with bundle methods. Math. Program. **94**(1), 41–69 (2002). https://doi.org/10.1007/s10107-002-0357-3
3. Barahona, F., Anbil, R.: The volume algorithm: producing primal solutions with a subgradient method. Math. Program. **87**(3), 385–399 (2000). https://doi.org/10.1007/s101070050002
4. Bertsimas, D., Dunn, J.: Machine learning under a modern optimization lens. Dynamic Ideas LLC, Charlestown (2019)
5. Bertsimas, D., King, A., Mazumder, R.: Best subset selection via a modern optimization lens. Ann. Stat. **44**(2), 813–852 (2016). https://doi.org/10.1214/15-AOS1388
6. Bertsimas, D., Pauphilet, J., Parys, B.V.: Sparse regression: scalable algorithms and empirical performance. Stat. Sci. **35**(4), 555–578 (2020). https://doi.org/10.1214/19-STS701
7. Bilde, O., Krarup, J.: Sharp lower bounds and efficient algorithms for the simple plant location problem. In: Hammer, P., Johnson, E., Korte, B., Nemhauser, G. (eds.) Studies in integer programming, annals of discrete mathematics, vol. 1, pp. 79–97. Elsevier (1977). https://doi.org/10.1016/S0167-5060(08)70728-3
8. Briant, O., Lemaréchal, C., Meurdesoif, P., Michel, S., Perrot, N., Vanderbeck, F.: Comparison of bundle and classical column generation. Math. Program. **113**(2), 299–344 (2008). https://doi.org/10.1007/s10107-006-0079-z
9. Carrabs, F., Cerrone, C., Pentangelo, R.: A multiethnic genetic approach for the minimum conflict weighted spanning tree problem. Networks **74**(2), 134–147 (2019). https://doi.org/10.1002/net.21883
10. Carrabs, F., Cerulli, R., Pentangelo, R., Raiconi, A.: Minimum spanning tree with conflicting edge pairs: a branch-and-cut approach. Ann. Op. Res. **298**(1), 65–78 (2021). https://doi.org/10.1007/s10479-018-2895-y
11. Carrabs, F., Gaudioso, M.: A lagrangian approach for the minimum spanning tree problem with conflicting edge pairs. Networks **78**(1), 32–45 (2021). https://doi.org/10.1002/net.22009
12. Darmann, A., Pferschy, U., Schauer, J.: Determining a minimum spanning tree with disjunctive constraints. In: Rossi, F., Tsoukias, A. (eds.) Algorithmic decision theory, Lecture Notes in Computer Science, vol. 5783, pp. 414–423. Springer Berlin Heidelberg (2009). https://doi.org/10.1007/978-3-642-04428-1_36
13. Darmann, A., Pferschy, U., Schauer, J., Woeginger, G.J.: Paths, trees and matchings under disjunctive constraints. Discr. Appl. Math. **159**(16), 1726–1735 (2011). https://doi.org/10.1016/j.dam.2010.12.016
14. Dessmark, A., Jansson, J., Lingas, A., Lundell, E.M., Persson, M.: On the approximability of maximum and minimum edge clique partition problems. Int. J. Found. Comput. Sci. **18**(02), 217–226 (2007). https://doi.org/10.1142/S0129054107004656
15. Dezső, B., Jüttner, A., Kovács, P.: LEMON - an Open Source C++ Graph Template Library. Electronic notes in theoretical computer science **264**(5), 23–45 (2011). https://doi.org/10.1016/j.entcs.2011.06.003
16. Edmonds, J.: Matroids and the greedy algorithm. Math. Program. **1**(1), 127–136 (1971). https://doi.org/10.1007/BF01584082
17. Erlenkotter, D.: A dual-based procedure for uncapacitated facility location. Op. Res. **26**(6), 992–1009 (1978). http://www.jstor.org/stable/170260
18. Frangioni, A., Gendron, B., Gorgone, E.: On the computational efficiency of subgradient methods: a case study with lagrangian bounds. Math. Program. Comput. **9**(4), 573–604 (2017). https://doi.org/10.1007/s12532-017-0120-7
19. Gerards, A., Schrijver, A.: Matrices with the Edmonds-Johnson property. Combinatorica **6**(4), 365–379 (1986). https://doi.org/10.1007/BF02579262
20. Guignard, M.: Lagrangean relaxation. Top **11**(2), 151–200 (2003). https://doi.org/10.1007/BF02579036
21. Guignard, M., Kim, S.: Lagrangean decomposition: a model yielding stronger lagrangean bounds. Math. Program. **39**, 215–228 (1987). https://doi.org/10.1007/BF02592954

22. Guignard, M., Rosenwein, M.B.: An application-oriented guide for designing lagrangean dual ascent algorithms. Eur. J. Oper. Res. **43**(2), 197–205 (1989). https://doi.org/10.1016/0377-2217(89)90213-0
23. Lougee-Heimer, R.: The common optimization interface for operations research: promoting open-source software in the operations research community. IBM J. Res. Dev. **47**(1), 57–66 (2003). https://doi.org/10.1147/rd.471.0057
24. Nemhauser, G.L., Wolsey, L.A.: Integer and combinatorial optimization, Wiley-Interscience series in discrete mathematics and optimization, vol. 55. John Wiley & Sons, Inc (1999). https://doi.org/10.1002/9781118627372
25. de Oliveira, W., Sagastizábal, C., Lemaréchal, C.: Convex proximal bundle methods in depth: a unified analysis for inexact oracles. Math. Program. **148**(1), 241–277 (2014). https://doi.org/10.1007/s10107-014-0809-6
26. Rebennack, S., Reinelt, G., Pardalos, P.M.: A tutorial on branch and cut algorithms for the maximum stable set problem. Int. Trans. Op. Res. **19**(1–2), 161–199 (2012). https://doi.org/10.1111/j.1475-3995.2011.00805.x
27. Ribeiro, C., Minoux, M.: Solving hard constrained shortest path problems by Lagrangean relaxation and branch-and-bound algorithms. Methods Op. Res. **53**, 303–316 (1986)
28. Samer, P., Haugland, D.: Fixed cardinality stable sets. Discr. Appl. Math. **303**, 137–148 (2021). https://doi.org/10.1016/j.dam.2021.01.019
29. Samer, P., Haugland, D.: Towards stronger Lagrangean bounds for stable spanning trees. In: Büsing, C., Koster, A.M.C.A. (eds.) Proceedings of the 10th international network optimization conference, INOC 2022, Aachen, Germany, June 7-10, 2022, pp. 29–33. OpenProceedings.org (2022). https://doi.org/10.48786/inoc.2022.06
30. Samer, P., Urrutia, S.: A branch and cut algorithm for minimum spanning trees under conflict constraints. Optim. Lett. **9**(1), 41–55 (2015). https://doi.org/10.1007/s11590-014-0750-x
31. Shepardson, F., Marsten, R.E.: A Lagrangean relaxation algorithm for the two duty period scheduling problem. Manage. Sci. **26**(3), 274–281 (1980). https://doi.org/10.1287/mnsc.26.3.274
32. Tomita, E., Tanaka, A., Takahashi, H.: The worst-case time complexity for generating all maximal cliques and computational experiments. Theoret. Comput. Sci. **363**(1), 28–42 (2006). https://doi.org/10.1016/j.tcs.2006.06.015
33. Zhang, R., Kabadi, S.N., Punnen, A.P.: The minimum spanning tree problem with conflict constraints and its variations. Discr. Optim. **8**(2), 191–205 (2011). https://doi.org/10.1016/j.disopt.2010.08.001



## Authors and Affiliations

**Phillippe Samer**[1] · **Dag Haugland**[1]

Dag Haugland
dag@ii.uib.no

[1] Department of Informatics, University of Bergen, P.O. Box 7800, 5020 Bergen, Norway