A Feasibility Pump and a Local Branching Heuristics for the Weight-Constrained Minimum Spanning Tree Problem

Cristina Requejo[1(✉)] and Eulália Santos[2]

[1] University of Aveiro, 3810-193 Aveiro, Portugal
crequejo@ua.pt

[2] ISLA-Higher Institute of Santarém and Leiria, 2414-017 Leiria, Portugal

Abstract. The Weight-constrained Minimum Spanning Tree problem (WMST) is a combinatorial optimization problem aiming to find a spanning tree of minimum cost with total edge weight not exceeding a given specified limit. This problem has important applications in the telecommunications network design and communication networks.

In order to obtain optimal or near optimal solutions to the WMST problem we use heuristic methods based on formulations for finding feasible solutions. The feasibility pump heuristic starts with the LP solution, iteratively fixes the values of some variables and solves the corresponding LP problem until a feasible solution is achieved. In the local branching heuristic a feasible solution is improved by using a local search scheme in which the solution space is reduced to the neighborhood of a feasible solution that is explored for a better feasible solution. Extensive computational results show that these heuristics are quite effective in finding feasible solutions and present small gap values. Each heuristic can be used independently, however the best results were obtained when they are used together and the feasible solution obtained by the feasibility pump heuristic is improved by the local branching heuristic.

Keywords: Weighted MST · Minimum spanning tree · Feasibility Pump · Local Branching · Heuristics

1 Introduction

Given a graph with edge costs and edge weights, the aim of the Weight-constrained Minimum Spanning Tree (WMST) problem is to find a spanning tree with minimum cost, such that its total weight does not exceed a given specified integer positive limit W. The WMST problem is a NP-hard [2,21] combinatorial optimization problem with important applications in the telecommunications network design and communication networks.

Several algorithms have already been proposed to the problem, either with exact or approximation approaches for determining a feasible solution. Aggarwal et al. [2] and Shogan [18] propose exact algorithms that use a Lagrangian relaxation to approximate a solution combined with a Branch and Bound strategy.

DOI: 10.1007/978-3-319-62395-5_46

Ravi and Goemans [15], Xue [20], Hong et al. [11] and Hassin and Levin [9] propose approximation schemes. A compilation of results and existing algorithms to solve the problem can be found in Henn [10].

Requejo et al. [16] describe several Integer Linear Programming (ILP) formulations, Agra et al. [4] describe new valid inequalities for the WMST problem, and Requejo and Santos [17] discuss algorithms based on Lagrangian relaxations and propose a Lagrangian relaxation combined with valid inequalities.

Another approach to the problem is to include the weight of the tree as a second objective instead of a hard constraint. The resulting problem is the bicriteria/biobjective spanning tree problem. Many references can be found for this approach, see [19] among many others.

In this paper we present heuristics for the WMST problem that use mixed integer models of the problem and allow interaction with a mixed integer programming (MIP) solver. We describe two heuristics and, additionally these heuristics are used together in one heuristic procedure. In the first one, the heuristic should provide a "good" feasible solution. In the second one, the heuristic should allow the improvement of an available feasible solution. For the first heuristic, a Feasibility Pump scheme is used. Such scheme was proposed by Fischetti, Glover and Lodi [6] with the goal of finding feasible solutions (if any exists) for generic Mixed Integer Linear Programming (MILP) problems and improved by Fischetti, Bertacco and Lodi [5] and by Achterberg and Berthold [1]. A mixed integer formulation for the WMST problem together with a MIP solver are used to obtain (fractional) linear relaxation solutions. These fractional solutions are rounded such that a feasible solution (if any exists) is found. For the second heuristic we consider a Local Branching method proposed by Fischetti and Lodi [7] to solve MIP problems. This enumerative scheme constructs a sequence of feasible solutions with improving (decreasing) value of costs which is considered a very effective improving method for large scale problems. Again a mixed integer formulation for the WMST problem together with a MIP solver are used with the objective to explore reduced feasible regions.

The structure of this paper is as follows. In Sect. 2 we describe the WMST problem and a general formulation that will be used in the heuristic schemes. In Sect. 3 we present a Feasibility Pump heuristic applied to the WMST problem. In Sect. 4 we present a Local Branching heuristic applied to the WMST problem. In Sect. 5, we present computational results for the Feasibility Pump and Local Branching schemes applied to the WMST problem. In the last section, Sect. 6, we conclude the paper.

2 The Weighted-Constrained Minimum Spanning Tree Problem

To define the WMST problem we consider an undirected complete graph $G = (V, E)$, with node set $V = \{0, 1, \ldots, n-1\}$ and edge set $E = \{\{i, j\},\ i, j \in V, i \neq j\}$. Associated with each edge $e = \{i, j\} \in E$ consider positive integer costs c_e and weights w_e. The WMST problem is to find a spanning tree $T = (V, E_T)$

in G, $E_T \subset E$, of minimum cost $C(T) = \sum_{e \in E_T} c_e$ and with total weight $W(T) = \sum_{e \in E_T} w_e$ not exceeding a given limit W.

To obtain formulations for the WMST problem one can easily adapt a Minimum Spanning Tree (MST) formulation. For the MST several formulations are well known (see Magnanti and Wolsey [12]) and in [16] natural and extended formulations for the WMST problem are discussed.

It is well known (see Magnanti and Wolsey [12]) that oriented formulations (based on the underlying directed graph) leads, in general, to tighter formulations (formulations whose lower bounds provided by the linear relaxations are closer to the optimum values). Thus, henceforward consider the corresponding directed graph, with root node 0, where each edge $e = \{0, j\} \in E$ is replaced with arc $(0, j)$ and each edge $e = \{i, j\} \in E, i \neq 0$, is replaced with two arcs, arc (i, j) and arc (j, i). Thus we obtain the arc set $A = \{(i, j),\ i \in V, j \in V \setminus \{0\}, i \neq j\}$. Each arc $(i, j) \in A$ inherits the cost and weight of the corresponding ancestor edge $\{i, j\}$.

Consider the binary variables x_{ij} (for all $(i, j) \in A$) indicating whether arc (i, j) is in the MST solution. Two classical formulations for the MST on the space of the binary variables x_{ij} can be considered [12]. To prevent the existence of circuits in the feasible solutions, and thus ensuring the connectivity of the feasible solutions, one formulation uses the cut-set inequalities and the other formulation uses the circuit elimination inequalities. The linear relaxation of both models provide the same bound value [12]. However the number of inequalities in both sets of constraints increase exponentially with the size of the model. In order to ensure connectivity/prevent circuits, instead of using one of those families with an exponential number of inequalities, one can use compact extended formulations. The well-known multicommodity flow formulation using additional flow variables can be considered. In this formulation the connectivity of the solution is ensured through the flow conservation constraints together with the connecting constraints, see [12]. These three formulations for the MST are easily adapted for the WMST problem through the inclusion of a weight constraint.

A generic formulation for the WMST problem is as follows.

$$\text{(WMST):} \qquad \min \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \quad x \in (MST) \tag{2}$$

$$\sum_{(i,j) \in A} w_{ij} x_{ij} \leq W \tag{3}$$

Where $x = (x_{ij}) \in \mathbb{R}^{|A|}$ is the solution vector and (MST) represents a set of inequalities describing the convex hull of the integer solutions of the MST. As referred, several sets of inequalities can be used. We use the multicommodity flow formulation. Consider the additional set of flow variables $f_{ij}^k \geq 0$, for all $(i, j) \in A$ and $k \in V \backslash \{0, i\}$, indicating weather arc $(i, j) \in A$ is used in the path from the root node to node k. The following flow conservation constraints

$$\sum_{i \in V \setminus \{k\}} f_{ij}^k - \sum_{i \in V \setminus \{0\}} f_{ji}^k = \begin{cases} -1 & j = 0 \\ \;\;0 & j \neq 0, k \\ \;\;1 & j = k \end{cases}, \qquad j \in V,\ k \in V \setminus \{0\},$$

and the connecting constraints $f_{ij}^k \leq x_{ij}$, for all $(i,j) \in A, k \in V \setminus \{0, i\}$ are used to ensure the connectivity of the solution and to represent the set (2) of constraints, together with the set of constraints $\sum_{i \in V} x_{ij} = 1$, for all $j \in V$, guarantying each non-root node receives an arc. Additionally, include the variables integrality constraints $x_{ij} \in \{0,1\}$, for all $(i,j) \in A$ and the variables non-negativity constraints $f_{ij}^k \geq 0$, for all $(i,j) \in A$ and $k \in V \setminus \{0, i\}$. Constraint (3) is the weight constraint.

3 The Feasibility Pump Heuristic for the WMST Problem

In this heuristic procedure the objective is to obtain a "good" feasible solution to the WMST problem. This objective is achieved through a relax-and-fix approach. We use the Feasibility Pump scheme, proposed by Fischetti, Glover and Lodi [6].

The first step of this procedure, the relax step, is to obtain the optimal solution of a linear relaxation formulation for the WMST problem, the LP solution. Let $\bar{x}$ be such solution. If $\bar{x}$ is an integer solution, then it is the optimal solution of the problem.

The second step is the rounding and fixing variables value step. Let the rounding $\tilde{x}$ of vector x be obtained by setting $\tilde{x}_{ij} = round(x_{ij})$, the usual scalar rounding, for all $(i,j) \in A$. The obtained vector $\tilde{x}$ is integer and, in general, $\tilde{x}$ is not a feasible solution.

The third step is to obtain the closest feasible solution to vector $\tilde{x}$. For that define a distance function $\Delta(x, \tilde{x})$ between a generic vector x and the given integer solution $\tilde{x}$. To define the distance function use the following L_1-norm function $\Delta(x, \tilde{x}) := \sum_{(i,j) \in A} |x_{ij} - \tilde{x}_{ij}|$. Notice that this function is equivalent to the linear function defined by $\sum_{(i,j) \in S}(1 - x_{ij}) + \sum_{(i,j) \in \overline{S}} x_{ij}$, with set $S = \{(i,j) \in A : \tilde{x}_{ij} = 1\}$ and its complement set $\overline{S} = \{(i,j) \in A : \tilde{x}_{ij} = 0\}$. Thus consider the distance function

$$\Delta(x, \tilde{x}) := \sum_{(i,j) \in S} (1 - x_{ij}) + \sum_{(i,j) \in \overline{S}} x_{ij}. \tag{4}$$

Hence, given an integer $\tilde{x}$, the closest vector x satisfying constraints (2) and (3) can be determined by minimizing the value of the distance function $\Delta(x, \tilde{x})$ as follows

$$\text{(D-WMST):} \qquad \begin{array}{ll} \min & \Delta(x, \tilde{x}) \\ \text{s.t.} & (2), (3). \end{array} \tag{5}$$

Let $\hat{x}$ be the optimal solution of the LP relaxation of problem (5), the D-WMST problem, and the value $\Delta(\hat{x}, \tilde{x})$ is its optimal value. The vector $\hat{x}$ is the closest solution to the integer $\tilde{x}$ and two cases may occur. If $\Delta(\hat{x}, \tilde{x}) = 0$, then $\hat{x} = \tilde{x}$ is an integer feasible solution for the WMST problem. If $\Delta(\hat{x}, \tilde{x}) > 0$, then a new integer $\tilde{\tilde{x}}$ is going to be obtained by rounding $\hat{x}$. By rounding $\hat{x}$ two cases may occur. Either $\tilde{\tilde{x}} \neq \tilde{x}$, a different vector was obtained, or $\tilde{\tilde{x}} = \tilde{x}$. When $\tilde{\tilde{x}} \neq \tilde{x}$ occurs the iterative process continues by solving the D-WMST problem again and a new solution closest to the new integer vector $\tilde{\tilde{x}}$ is going to be obtained. When $\tilde{\tilde{x}} = \tilde{x}$, the same solution was obtained and a cycle occurs. To avoid cycling the following perturbation mechanism [6] is applied to an integer solution $\tilde{x}$. For a given parameter $\delta > 0$, modify $\tilde{x}$ such that $|x_{ij} - \tilde{x}_{ij}| + max\{\rho_{ij}, 0\} > \delta$, for all $(i, j) \in A$ and ρ_{ij} randomly selected in $[-0.3, 0.7]$. Notice that this perturbation mechanism gives the possibility to modify the variables such that $|x_{ij} - \tilde{x}_{ij}| = 0$.

Iteratively another step is performed in order to further reduce the distance $\Delta(\hat{x}, \tilde{x})$ between the vector $\tilde{x}$, the integer rounded vector, and the closer solution $\hat{x}$ of the LP problem (5). Therefore the pair $(\hat{x}, \tilde{x})$ is iteratively updated by performing this relax-and-fix procedure until a feasible solution to the WMST problem is found.

In practice it may be much time consuming to achieve such feasible solution. Therefore a *time limit*, denoted maxtime, and a *maximum number of iterations*, denoted maxiter, are imposed. Hence we obtain a heuristic procedure, the feasibility pump heuristic which is briefly described in Algorithm 1.

Algorithm 1. Feasibility Pump heuristic for the WMST problem (FP)

```
Require: problem data (graph $G = (V, E)$; costs $c_{ij}$; weights $w_{ij}$; weight limit $W$);
    parameters (maxtime; maxiter).
 1: solve the LP relaxation of a formulation to the WMST problem
 2: let $\hat{x}$ be its optimal solution
 3: if $\hat{x}$ is integer then
 4:     stop, $\hat{x}$ is the optimal solution of the WMST problem
 5: end if
 6: $\tilde{x}^0 \leftarrow round(\hat{x})$
 7: $t \leftarrow 0$
 8: while ((time < maxtime) and ($t$ < maxiter)) do
 9:     obtain $\hat{x}$, the optimal solution of the LP relaxation of D-WMST problem with
    $\Delta(x, \tilde{x}^t)$ as objective function
10:     let $\Delta(\hat{x}, \tilde{x}^t)$ be its optimal value
11:     if $\Delta(\hat{x}, \tilde{x}^t) = 0$ then
12:         stop, $\hat{x}$ is an integer feasible solution for the WMST problem
13:     else
14:         $t \leftarrow t + 1$
15:         $\tilde{x}^t \leftarrow round(\hat{x})$
16:         if $\tilde{x}^t = \tilde{x}^{t-1}$ then
17:             apply perturbation mechanisms
18:         end if
19:     end if
20: end while
    return feasible solution $\hat{x}$ and integer solution $\tilde{x}^t$
```

In the first line of Algorithm 1 a solver is used to obtain the LP solution $\hat{x}$. A solver is again used in line 9 to obtain a solution to the D-WMST problem.

In Algorithm 1 all the solutions, except the first LP solution that uses the objective function of the WMST problem, are obtained by using as objective function the distance function (4). This objective function does not take into account the objective function of the WMST problem. As a consequence the solution obtained at the end of the procedure may have a value far from the best objective value of the WMST problem. To overcome this disadvantage, Achterberg and Berthold [1] propose the use of a different objective function that is a convex linear combination of the objective function (1) of the WMST problem and the distance function (4) of the D-WMST problem. The proposed objective function is

$$\Delta_\alpha(x,\tilde{x}) := (1-\alpha)\Delta(x,\tilde{x}) + \alpha\frac{\sqrt{|A|}}{||c||}\sum_{(i,j)\in A} c_{ij}x_{ij}, \tag{6}$$

with $\alpha \in [0,1]$ and where $||c||$ is the Euclidean norm of the cost vector c and $|A|$ is the cardinality of set A. Thus $\sqrt{|A|}$ is the Euclidean norm of the objective function vector in (4). In the objective function (6), the influence of the objective function (1) of the WMST problem is controlled by the α parameter. For values of α near to 1 the influence of the objective function is high.

In order to obtain an Improved Feasibility Pump heuristic for the WMST problem (IFP) one has to obtain, in line 9 of the Algorithm 1, the optimal solution $\hat{x}$ of (5) with (6) as the objective function.

Both the FP and the IFP return an integer solution, which may not satisfy constraints (2) and (3), and a solution satisfying constraints (2) and (3), but not necessarily integer. Two trajectories of solutions, hopefully convergent, are constructed. One is formed by a sequence of solutions satisfying constraints (2) and (3), solutions $\hat{x}$ that may not be integer. The other sequence is formed by integer solutions $\tilde{x}^t$ that may not satisfy constraints (2) and (3).

4 The Local Branching for the WMST Problem

The improvement of a previously obtained feasible solution to the WMST problem is the objective of the second heuristic we describe. The improvement is done through a local search scheme that uses a local branching method based on the method proposed by Fischetti and Lodi [7].

Consider, as a reference solution, a previously obtained feasible solution $\tilde{x}$ to the WMST problem. This integer solution corresponds to a spanning tree $T_{\tilde{x}}$ with cost $C(\tilde{x}) = C(T_{\tilde{x}})$ and weight $W(\tilde{x}) = W(T_{\tilde{x}})$. Define two sets, set $S = \{(i,j) \in A : \tilde{x}_{ij} = 1\}$ and its complement set $\overline{S} = \{(i,j) \in A : \tilde{x}_{ij} = 0\}$. For a given positive integer parameter k', define the *neighborhood* of $\tilde{x}$ as the set of feasible solutions of the WMST problem satisfying the additional *local branching constraint*:

$$\Delta(x,\tilde{x}) = \sum_{(i,j)\in S}(1-x_{ij}) + \sum_{(i,j)\in\overline{S}} x_{ij} \leq k'. \tag{7}$$

The linear constraint (7) limits to k' the total number of binary variables flipping their value with respect to the solution $\tilde{x}$, either from 1 to 0 or from 0 to 1, respectively. Notice that the local branching constraint uses the same distance function $\Delta(x, \tilde{x})$ that is used in the FP heuristic as the objective function (4).

For every feasible solution to the WMST problem the cardinality of the set S is constant and equal to the number of edges of the corresponding feasible tree $T_{\tilde{x}}$. Further the number of variables exchanging from 1 to 0 must be equal to the number of variables exchanging from 0 to 1. Thus the *local branching constraint* may assume the asymmetric form:

$$\Lambda(x, \tilde{x}) = \sum_{(i,j)\in S} (1 - x_{ij}) \leq k \tag{8}$$

with $k = \frac{k'}{2}$. Constraint (8) can be used as a branching criterion within an enumerative scheme and the solution space associated with the current branching node can be partitioned by means of the disjunction

(i) $\Lambda(x, \tilde{x}) \leq k$ (left branch) or (ii) $\Lambda(x, \tilde{x}) \geq k + 1$ (right branch).

With each one of those constraints the solution space is reduced. Define the *neighborhood* $\mathcal{N}(\tilde{x}, k)$ of $\tilde{x}$ as the set of feasible solutions of the WMST problem satisfying the additional local branching constraint $\Lambda(x, \tilde{x}) \leq k$, and the *neighborhood* $\mathcal{N}_{+}(\tilde{x}, k)$ of $\tilde{x}$ as the set of feasible solutions of the WMST problem satisfying the additional local branching constraint $\Lambda(x, \tilde{x}) \geq k + 1$. The choice of the size of the neighborhoods given by the parameter k is a problem which may depend on the size and structure of the instances used. On one hand, the k must be large enough so that the neighborhood $\mathcal{N}(\tilde{x}, k)$ contains better valued solutions than $\tilde{x}$. On the other hand, the k should be small enough to ensure that the neighborhood $\mathcal{N}(\tilde{x}, k)$ is quickly explored. Note that neighborhood $\mathcal{N}(\tilde{x}, k)$ of $\tilde{x}$ has solutions similar to $\tilde{x}$ and neighborhood $\mathcal{N}_{+}(\tilde{x}, k)$ contains solutions that differ from $\tilde{x}$ in more than $2 \times (k + 1)$ variables. The method explores both neighborhoods, but the neighborhood $\mathcal{N}_{+}(\tilde{x}, k)$ is only explored when a feasible solution better valued than $\tilde{x}$ is not found in the neighborhood $\mathcal{N}(\tilde{x}, k)$. Algorithm 2 displays a brief description of the local branching scheme applied to the WMST problem.

In the first line of Algorithm 2 a solver is used to obtain the first feasible integer solution $\tilde{x}^1$ which is taken as a reference solution. A solver is again used in lines 7 and 13 to obtain the solutions in the reduced solution space. The sequence of solutions $\tilde{x}^t$ generated by the LB, corresponds to a decreasing sequence of costs.

The Local Branching is a MIP technique planned to be an exact method, but that acts as a heuristic method when a time limit is set and reached before the optimal solution is found [8]. In case the time limit is exceeded, the obtained solution $\tilde{x}$ is not the optimal solution and the exploration of the neighborhood is not complete. In that case the size of the neighborhood that is to be explored has to be modified in order to either reduce or enlarge the region where the

Algorithm 2. Local branching scheme for the WMST problem (LB)

```
Require: problem data (graph $G = (V, E)$; costs $c_{ij}$; weights $w_{ij}$; weight limit $W$);
    parameters ($k$; maxtime; maxiter).
 1: obtain a feasible solution $\tilde{x}$, the reference solution
 2: $t \leftarrow 1$
 3: $\tilde{x}^t \leftarrow \tilde{x}$
 4: while ((time < maxtime) and ($t$ < maxiter)) do
 5:     define $k$
 6:     introduce the local branching constraint $\Lambda(x, \tilde{x}^t) \leq k$ in the WMST problem
 7:     solve the problem and let $\tilde{x}$ be its optimal solution within the neighborhood
    $\mathscr{N}(\tilde{x}^t, k)$
 8:     if $C(\tilde{x}) < C(\tilde{x}^t)$ then
 9:         $t \leftarrow t + 1$
10:         $\tilde{x}^t \leftarrow \tilde{x}$
11:     else
12:         introduce the local branching constraint $\Lambda(x, \tilde{x}^t) \geq k + 1$ in the WMST
    problem
13:         solve the problem and let $\tilde{x}$ be its optimal solution within the neighborhood
    $\mathscr{N}_+(\tilde{x}^t, k)$
14:         if $C(\tilde{x}) < C(\tilde{x}^t)$ then
15:             $t \leftarrow t + 1$
16:             $\tilde{x}^t \leftarrow \tilde{x}$
17:         end if
18:     end if
19: end while
    return integer solution $\tilde{x}^t$
```

solution is sought. The following mechanisms, see [13], modify the size of the neighborhood.

Intensification mechanism. The intensification mechanism aims to reduce the size of the neighborhood in an attempt to speed-up its exploration. The right hand side of the constraint $\Lambda(x, \tilde{x}) \leq k$ is reduced to $\lfloor \frac{k}{2} \rfloor$.

Diversification mechanism. The diversification mechanism aims to enlarge the size of the neighborhood. However, and consequently the exploration time is also increased. First apply a "weak" diversification mechanism, in which the right hand side of the constraint $\Lambda(x, \tilde{x}) \leq k$ is increased by $\lceil \frac{k}{2} \rceil$, i.e., it is introduced the constraint $\Lambda(x, \tilde{x}) \leq k + \lceil \frac{k}{2} \rceil$. In case an improved solution is not found, apply a "strong" diversification mechanism, in which the right hand side of the constraint $\Lambda(x, \tilde{x}) \leq k$ is increased with $2 \times \lceil \frac{k}{2} \rceil$, i.e., it is introduced the constraint $\Lambda(x, \tilde{x}) \leq k + 2\lceil \frac{k}{2} \rceil$.

When a time limit is exceeded and the obtained solution $\tilde{x}$ is not the optimal solution the following cases may occur. (i) If the solution $\tilde{x}$ has an improved value, then the reference solution is updated, but the value of parameter k is not modified. (ii) If the solution $\tilde{x}$ does not have an improved value, $C(\tilde{x}) > C(\tilde{x}^t)$, then apply the intensification mechanism in order to reduce the neighborhood.

If again an improved solution is not found, apply a "weak" diversification mechanism. (iii) If the solution $\tilde{x}$ is infeasible, then apply the "strong" diversification mechanism in order to enlarge the neighborhood.

5 Computational Experience

In this section we report computational tests of the FP and of the LB heuristics applied to the WMST problem. We compare the performance of the heuristics against an upper bound obtained by a branch-and-cut algorithm based on the strengthened weighted Miller-Tucker-Zemlin formulation (Agra et al. [3] and Requejo et al. [16]).

All the tests were performed using a computer with an Intel(R) Core(TM)2 Duo CPU (T7100) 2.00 GHz processor and 4 GB of RAM, and were conducted using the Xpress-Optimizer 23.01.03 solver with the default options.

We present computational results for instances to the WMST problem defined on complete graphs with a number of nodes varying between 10 and 1000, in a total of 215 instances.

5.1 Instances Generation Description

To generate an instance of the WMST problem, the cost c_e and the weight w_e of each edge e have to be defined. Afterwards, a (feasible) value to the weight limit W must also be defined. We built three sets of instances, constituting three different ways of generating costs and weights.

In a first set of instances, costs c_e and weights w_e are generated similarly to a set of instances described in Pisinger [14] and named therein as Spanner instances. A value for W is selected between 1000 and 3500 proportional to the number n of nodes of the instance. The costs and the weights are multiples of a small set (the spanner set in [14]) of costs and weights following one particular distribution, we use the Uncorrelated distribution (which is in the Pisinger's [14] proposed list of distributions), and the following two parameters $s = 2$ and $m = 10$. At the end, the weights of some edges are manipulated in such a way that the optimal solution has a desired predefined structure. After testing a few structures we obtained some challenging instances when the optimal structure of the WMST problem instance solution has large diameter values, almost $n-1$, but not equal to $n-1$, in such way that the tree is almost a path. Thus we name this instance's set as "Almost Path" (AP).

For the second set of instances, named Random (R) instances, the costs c_e and the weights w_e are uniformly generated in the interval $[1, 1000]$.

For the third set of instances, named Euclidean (E) instances, costs c_e and weights w_e are obtained using Euclidean distances. After randomly generating the coordinates of n points/nodes in a 100×100 grid, the cost c_e of each edge $e = \{i, j\}$ is the integer part of the Euclidean distance between points/nodes i and j. We proceed independently and similarly to obtain the weights.

To define a (feasible) value to the weight limit W for each instance of these two sets of instances (sets R and E), we start by obtaining the weight of the minimum spanning tree $W(T_c)$ and the weight of the minimum weight spanning tree $W(T_w)$ and we select W to be one of the values $W_i = \frac{W(T_c) + W(T_w)}{2^i}$, for $i \in \{1, \ldots, 10\}$.

A total of 215 instances were generated, 95 of the set AP and 60 of each set R and E. For each set AP and each instance size between 10 and 150 we have 10 instances and for each instance size between 200 and 1000 we have 5 instances. For each set R and E and each instance size we have 5 instances.

5.2 Computational Tests for the FP and IFP Heuristics

Computational tests performed in all groups of instances allow us to conclude that the heuristics FP and IFP, described in Sect. 3, obtain similar quality solutions. However the heuristic FP uses less computational time in all groups of instances, see Fig. 1. Also the number of iterations used by the IFP heuristic is approximately three times higher than the number of iterations used by the FP heuristic. On average the number of iterations of the FP is 4 and the number of iterations of the IFP is 14. Therefore, the remaining computational results will be presented only for the FP heuristic.

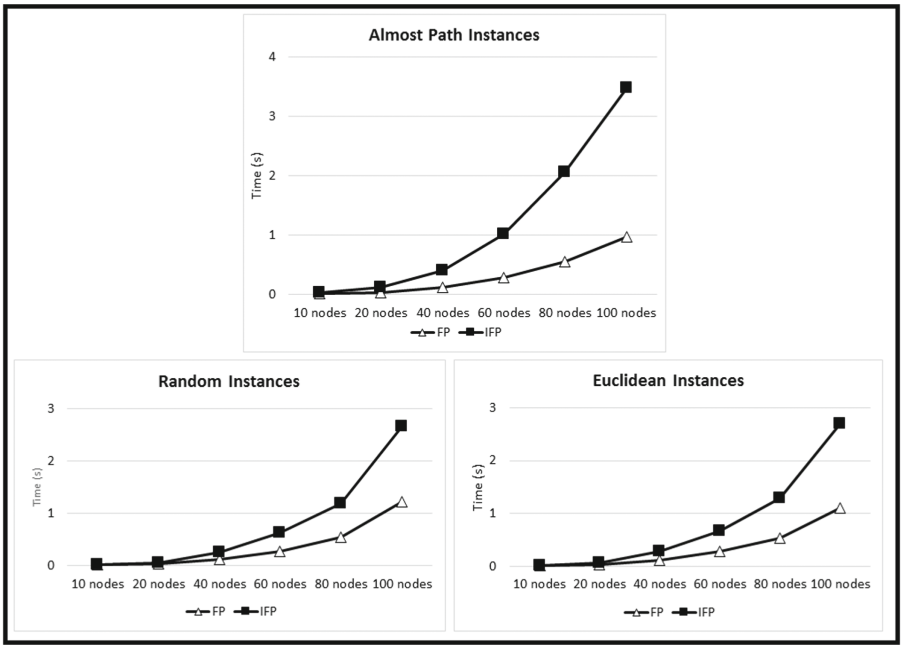

Fig. 1. Comparing the mean execution times (in seconds) for FP and IFP heuristics.

5.3 Computational Tests for the LB Heuristic

Computational experiments performed with the LB heuristic, described in Sect. 4, showed that when the value k of the local branching constraints (8) increases, the mean execution times also increases. To obtain the computational results of the LB heuristic we used the value $k = 5$ and several strategies were tested. First strategy (LB1): a time limit of 10000 s is imposed. Second strategy (LB2): in addition to the time limit, the solver is stopped when the first integer solution is obtained, and intensification and diversification mechanisms are used. Third strategy (LB3): in addition to the second strategy conditions, the reference solution is the first integer solution found by the solver and a time limit is imposed for the exploration of the neighborhoods. Fourth strategy (LB4): differs from the third strategy because the reference solution is obtained with the FP heuristic. To evaluate these four strategies some preliminary computational tests were performed to instances with up to $n = 100$ nodes in a total of 120 instances. Table 1 displays mean computational times and mean number of iterations for the four LB strategies. The LB1 strategy obtained good valued solutions, but it uses high computational times and 18.3% of the instances (13 AP instances and 9 E instances) exceed a computational time limit imposed.

Table 1. Mean execution time (in seconds) and mean number of iterations of the four LB strategies.

	n	LB1		LB2		LB3		LB4	
		Time	Iter	Time	Iter	Time	Iter	Time	Iter
Almost Path	10	0.42	3	0.58	5	0.40	3	0.40	3
	20	3.75	3	4.66	6	3.49	3	2.37	3
	40	64.06	4	29.27	8	24.32	5	20.00	4
	60	2273.08	4	17.85	7	34.10	4	24.00	3
	80	3443.43	3	41.93	7	61.95	3	51.05	3
	100	9999.25	3	67.04	7	123.66	4	98.63	3
Random	10	0.41	3	0.55	4	0.61	3	0.37	3
	20	2.68	3	4.73	6	2.82	3	2.35	3
	40	17.26	4	26.27	7	13.67	3	10.06	3
	60	18.89	4	20.32	8	20.04	4	13.47	3
	80	64.02	5	48.81	9	49.26	4	30.50	3
	100	1942.81	11	90.72	10	238.08	8	75.49	3
Euclidean	10	0.89	3	2.28	4	2.69	4	2.69	4
	20	6.52	4	14.92	9	13.41	7	6.85	5
	40	88.60	7	63.84	11	75.01	9	29.52	4
	60	3802.02	11	62.20	11	241.64	12	28.98	4
	80	8084.06	12	476.79	12	1927.29	14	44.58	4
	100	9102.89	18	319.51	14	3471.08	19	132.12	5

Table 2. Mean gap values (in percentage) of the HP procedure, the FP and the LB heuristics.

	n	AP			R			E		
		HP	FP	LB	HP	FP	LB	HP	FP	LB
Mean gap	10	0.00	7.14	0.00	0.00	3.84	0.50	0.00	5.27	0.00
	20	0.00	0.57	0.00	0.00	5.28	0.00	0.00	2.25	0.10
	40	0.00	0.58	0.00	0.00	2.74	0.00	0.00	1.96	0.12
	60	0.00	0.02	0.00	0.00	1.31	0.00	0.00	0.75	0.08
	80	0.00	0.06	0.09	0.00	1.50	0.00	0.00	0.58	0.08
	100	0.00	0.67	0.28	0.00	0.55	0.01	0.00	0.56	0.06
	150	0.00	0.77	0.00	0.00	0.79	0.39	0.00	0.28	0.02
	200	0.00	1.18	0.00	0.00	0.53	0.02	0.00	0.40	0.03
	300	0.00	1.20	0.00	0.00	0.24	0.16	0.00	0.22	0.02
	400	0.00	0.89	0.00	0.00	0.14	0.14	13.58	0.00	0.03
	500	0.00	1.06	0.00	0.00	0.22	0.14	$*$	0.00	0.00
	1000	0.00	1.41	0.00	0.04	0.07	0.07	$*$	0.00	0.00
σ_{gap}	10	0.00	8.08	0.00	0.00	3.43	1.12	0.00	3.49	0.00
	20	0.00	0.61	0.00	0.00	5.01	0.00	0.00	2.68	0.23
	40	0.00	0.61	0.00	0.00	1.68	0.00	0.00	0.39	0.00
	60	0.00	0.03	0.00	0.00	1.07	0.00	0.00	0.71	0.13
	80	0.00	0.05	0.11	0.00	1.32	0.00	0.00	0.43	0.15
	100	0.00	1.40	0.63	0.00	0.26	0.02	0.00	0.66	0.07
	150	0.00	1.27	0.00	0.00	0.68	0.86	0.00	0.19	0.02
	200	0.00	1.42	0.00	0.00	0.38	0.03	0.00	0.20	0.03
	300	0.00	0.88	0.00	0.00	0.62	0.21	0.00	0.16	0.02
	400	0.00	0.73	0.00	0.00	0.10	0.10	4.50	0.00	0.06
	500	0.00	1.32	0.00	0.00	0.14	0.19	$*$	0.00	0.00
	1000	0.00	0.43	0.00	0.08	0.08	0.08	$*$	0.00	0.00

The symbol $*$ means we could not obtain an upper bound within the time limit imposed.

The LB2 and LB3 strategies use less computational time than strategy LB1, however the LB4 strategy uses less computational time, obtains the best valued solutions and the optimal solution was obtained in 83.3% of the instances (in 100 of the 120 instances). Table 1 shows that the LB4 strategy presents the lowest average execution times and also uses the smallest average number of iterations, for all group of instances. Therefore we report results obtained with this LB4 strategy, denoted as LB in what follows. Notice that this strategy corresponds to the FP heuristic followed by a LB heuristic.

Table 3. Mean execution time (in seconds) for the HP procedure, the FP and the LB heuristics.

	n	AP			R			E		
		HP	FP	LB	HP	FP	LB	HP	FP	LB
Mean time	10	0.11	0.01	0.40	0.19	0.01	0.37	0.11	0.01	2.69
	20	1.32	0.03	2.37	1.13	0.03	2.35	0.89	0.03	6.85
	40	41.67	0.12	20.00	4.13	0.12	10.06	5.60	0.12	29.52
	60	517.22	0.29	24.00	5.96	0.27	13.47	12.36	0.29	28.98
	80	2535.07	0.55	51.05	14.25	0.54	30.50	41.12	0.54	44.58
	100	10596.60	0.97	98.63	20.68	1.21	75.49	116.40	1.10	132.12
	150	10804.61	3.01	176.11	95.07	4.98	175.32	585.16	2.30	181.10
	200	10813.50	7.89	355.65	310.59	12.52	299.32	1905.27	4.94	314.05
	300	10866.94	40.86	688.63	1566.21	39.65	727.49	12368.56	18.87	1205.03
	400	10929.46	90.46	1525.50	2628.47	92.53	1406.42	31103.45	39.50	2310.27
	500	11057.40	268.78	3624.23	9225.34	311.44	2357.79	27611.01	98.89	4530.66
	1000	11988.70	1014.11	9621.59	17636.70	1724.17	10808.70	31071.30	1171.19	10833.06
σ_{time}	10	0.04	0.01	0.14	0.14	0.01	0.17	0.06	0.01	1.78
	20	0.87	0.00	0.69	0.62	0.00	0.27	0.56	0.01	3.26
	40	29.75	0.02	5.84	1.15	0.01	2.60	1.21	0.01	7.64
	60	338.83	0.01	5.32	1.19	0.01	3.96	4.65	0.01	6.31
	80	2095.42	0.02	10.49	2.32	0.04	12.24	13.01	0.05	6.02
	100	645.98	0.06	19.44	6.10	0.35	35.28	34.54	0.26	48.59
	150	1.82	0.15	37.14	37.46	1.18	84.89	181.11	0.11	26.14
	200	4.73	1.41	176.84	222.72	2.55	64.89	737.56	0.41	83.14
	300	34.44	7.36	177.79	866.04	15.76	62.21	3070.60	3.47	234.47
	400	79.30	4.31	173.93	519.69	45.28	92.43	21167.74	1.55	482.86
	500	132.57	76.00	327.01	2740.35	31.91	617.46	7483.64	20.97	381.97
	1000	488.97	88.88	1048.95	2945.74	264.51	421.43	86.23	136.92	13.18

5.4 Results Description and Analysis

We compare the performance of the FP and of the LB heuristics with the performance of the HP procedure.

In [3,16] the best results to obtain the optimal value using the software Xpress 7.3 (Xpress Release 2012 with Xpress-Optimizer 23.01.03 and Xpress-Mosel 3.4.0), were obtained with the Branch and Cut algorithm based on a weighted MTZ (Miller-Tucker-Zemlin) formulation with the inclusion of cuts preventing cycles at the root node. This procedure will be denoted by HP (Hybrid Procedure) and is used to access the quality of the solutions obtained with the FP and LB heuristics.

Having an upper bound on the value of the cost, the upper bound gap is $gap = \frac{UB - OPT}{OPT} \times 100$, where UB is the upper bound obtained through the considered method (HP procedure, FP heuristic, LB heuristic) and OPT is the optimal value obtained with the HP procedure or the best obtained value with this procedure within a time limit of 10000 s.

For each instance set AP, R, and E and each instance size set we display in the upper part of the Table 2 the average upper bound gap (in percentage) and the corresponding standard deviation values, in the lower part of the Table 2. In the upper part of the Table 3 are displayed the average execution times (in seconds) and corresponding standard deviation values, in the lower part of the Table 3.

The LB heuristic improves the solutions obtained by the FP heuristic in 68.37% of the instances (147 from 215 instances). In 75.34% of the instances (162 from 215 instances) the gap is null, i.e. the optimal solution was obtained, when using the LB heuristic. The set of instances with the highest gap is the E instance set, the Euclidean instances.

For every procedure the computational time increases as the size of the instance (number of nodes) increases. In 61.86% of the instances (133 from 215 instances), the computational time of the LB heuristic is smaller than the computational time of the HP procedure. The set of instances with the highest computational time is, again, the instance set E. Using the HP procedure it was not possible to obtain an upper bound within the time limit imposed for two E instances.

6 Conclusions

We describe two heuristic procedures to the WMST problem, the FP (feasibility pump) and the LB (local branching) heuristics. The FP heuristic is a constructive heuristic and uses a relax-and-fix strategy to obtain a good feasible solution. The LB heuristic uses a local branch strategy to improve a feasible solution. Our computational results show that the FP heuristic is fast in obtaining feasible solutions for the WMST problem, and the LB heuristic can be used to improve the obtained feasible solution. Both heuristics can be used independently, however the best strategy is to use the heuristics together, the FP followed by the LB. This process is faster than the HP procedure, a branch-and-cut procedure used to access the quality of the described heuristics, and obtains good valued solutions.

The FP and the LB heuristics are a good choice in obtaining feasible solutions for the WMST problem and can be used together for better quality solutions and with very competitive computational times.

Acknowledgements. The research of the authors has been partially supported by Portuguese funds through the *CIDMA (Center for Research and Development in Mathematics and Applications)* and the FCT, the Portuguese Foundation for Science and Technology, within project UID/MAT/04106/2013.

References

1. Achterberg, T., Berthold, T.: Improving the feasibility pump. Discrete Optim. **4**(1), 77–86 (2007)
2. Aggarwal, V., Aneja, Y.P., Nair, K.P.K.: Minimal spanning tree subject to a side constraint. Comput. Oper. Res. **9**, 287–296 (1982)

3. Agra, A., Cerveira, A., Requejo, C., Santos, E.: On the weight-constrained minimum spanning tree problem. In: Pahl, J., Reiners, T., Voß, S. (eds.) INOC 2011. LNCS, vol. 6701, pp. 156–161. Springer, Heidelberg (2011). doi:10.1007/978-3-642-21527-8_20
4. Agra, A., Requejo, C., Santos, E.: Implicit cover inequalities. J. Combin. Optim. **31**(3), 1111–1129 (2016)
5. Bertacco, L., Fischetti, M., Lodi, A.: A feasibility pump heuristic for general mixed-integer problems. Discrete Optim. **4**(1), 63–76 (2007)
6. Fischetti, M., Glover, F., Lodi, A.: The feasibility pump. Math. Program. **104**(1), 91–104 (2005)
7. Fischetti, M., Lodi, A.: Local branching. Math. Program. **98**(1–3), 23–47 (2003)
8. Hansen, P., Mladenović, N., Urošević, D.: Variable neighborhood search and local branching. Comput. Oper. Res. **33**(10), 3034–3045 (2006)
9. Hassin, R., Levin, A.: An efficient polynomial time approximation scheme for the constrained minimum spanning tree problem using matroid intersection. SIAM J. Comput. **33**(2), 261–268 (2004)
10. Henn, S.: Weight-constrained minimum spanning tree problem. Master's thesis, University of Kaiserslautern, Kaiserslautern, Germany (2007)
11. Hong, S., Chung, S., Park, B.H.: A fully polynomial bicriteria approximation scheme for the constrained spanning tree problem. Oper. Res. Lett. **32**, 233–239 (2004)
12. Magnanti, T.L., Wolsey, L.A.: Optimal trees. In: Ball, M., Magnanti, T.L., Monma, C., Nemhauser, G.L. (eds.) Network Models, Handbooks in Operations Research and Management Science, vol. 7, pp. 503–615. Elsevier Science Publishers, North-Holland (1995)
13. Mladenović, N., Hansen, P.: Variable neighborhood search. Comput. Oper. Res. **24**(11), 1097–1100 (1997)
14. Pisinger, D.: Where are the hard knapsack problems? Comput. Oper. Res. **32**(9), 2271–2284 (2005)
15. Ravi, R., Goemans, M.X.: The constrained minimum spanning tree problem. In: Karlsson, R., Lingas, A. (eds.) SWAT 1996. LNCS, vol. 1097, pp. 66–75. Springer, Heidelberg (1996). doi:10.1007/3-540-61422-2_121
16. Requejo, C., Agra, A., Cerveira, A., Santos, E.: Formulations for the weight-constrained minimum spanning tree problem. AIP Conf. Proc. **1281**, 2166–2169 (2010)
17. Requejo, C., Santos, E.: Lagrangian based algorithms for the weight-constrained minimum spanning tree problem. In: Proceedings of the VII ALIO/EURO Workshop on Applied Combinatorial Optimization, pp. 38–41 (2011)
18. Shogan, A.: Constructing a minimal-cost spanning tree subject to resource constraints and flow requirements. Networks **13**, 169–190 (1983)
19. Sourd, F., Spanjaard, O.: A multiobjective branch-and-bound framework: application to the biobjective spanning tree problem. INFORMS J. Comput. **20**(3), 472–484 (2008)
20. Xue, G.: Primal-dual algorithms for computing weight-constrained shortest paths and weight-constrained minimum spanning trees. In: Proceedings of the IEEE International Conference on Performance, Computing, and Communications, pp. 271–277 (2000)
21. Yamada, T., Watanabe, K., Kataoka, S.: Algorithms to solve the knapsack constrained maximum spanning tree problem. Int. J. Comput. Math. **82**, 23–34 (2005)