Journal of the Operational Research Society (2017) 68, 1223–1235

www.palgrave.com/journals

A greedy heuristic for the capacitated minimum spanning tree problem

M. Kritikos[1]* and G. Ioannou[1]

[1]*Management Science Laboratory, Department of Management Science and Technology, Athens University of Economics and Business, Athens, Greece*

This paper develops a greedy heuristic for the capacitated minimum spanning tree problem (CMSTP), based on the two widely known methods of Prim and of Esau–Williams. The proposed algorithm intertwines two-stages: an enhanced combination of the Prim and Esau–Williams approaches via augmented and synthetic node selection criteria, and an increase of the feasible solution space by perturbing the input data using the law of cosines. Computational tests on benchmark problems show that the new heuristic provides extremely good performance results for the CMSTP, justifying its effectiveness and robustness. Furthermore, excluding the feasible space expansion, we show that we can still obtain good quality solutions in very short computational times.

Journal of the Operational Research Society (2017) **68(10),** 1223–1235. doi:10.1057/s41274-016-0146-7;
published online 21 December 2016

Keywords: capacitated minimum spanning tree problem; Esau-Williams heuristic; Prim's algorithm

1. Introduction

The Capacitated Minimum Spanning Tree problem (CMSTP) can be defined as the design of a minimum cost tree which spans over all vertices of an undirected graph G, so that the sum of demands of every main subtree does not exceed a given capacity Q. The CMSTP plays an important role in the design of backbone telecommunications networks, as well as in distribution, transportation, and logistics. Gavish (1991), formulated telecommunication network design problems as CMSTPs. In addition a CMSTP solution provides a lower bound on the capacitated vehicle routing problem (CVRP) defined on G (Toth and Vigo, 1995).

The CMSTP can be divided into two categories based on whether the weights of vertices are identical or not. The first is the homogeneous CMSTP where all vertices have the same weight. The second is the heterogeneous CMSTP where different vertices have different weights. When the weights of all vertices are equal to unity the problem reduces to finding a minimum cost rooted spanning tree in which each subtree contains at most Q (the capacity) vertices; this unit demand case is usually referred to as the CMSTP in the literature (Oncan and Altinel, 2009). The general homogeneous problem can be transformed into a unity problem by dividing the weights of the vertices and the capacity Q by the common vertex weight. In our work, we propose an algorithm for the CMSTP when the weights of all vertices are equal to unity.

*Correspondence: M. Kritikos, Management Science Laboratory, Department of Management Science and Technology, Athens University of Economics and Business, Athens, Greece.
E-mail: kmn@aueb.gr

The CMSTP is a difficult combinatorial optimization problem. It has been shown to be NP-hard even in the case of unit demand (Papadimitriou, 1978); thus, the solution of the CMSTP with exact methods is very time consuming and even impossible even for moderate size instances (Ruiz *et al*, 2015), and as a result heuristics are widely used in practice. Due to the importance of the problem, there is a vast literature that addresses modelling and solutions aspects of the CMSTP. Several mathematical formulations and exact algorithms have been proposed for the CMSTP. Exact algorithms are based on branch and bound and dynamic programming methods. The existing exact algorithms only solve small scale CMSTPs or find lower bounds on the optimal solution—see e.g., Chandy and Russell (1972), Chandy and Lo (1973), Gavish (1982), Gouveia and Paixao (1991), Malik and Yu (1993), Hall (1996), Han *et al* (2002), Gouveia and Martins (2005), and Uchoa *et al* (2008).

Early heuristics based on the greedy paradigm are those of Esau and Williams (1966)—the most widely known and the one used as a benchmark in computational tests—and the unified algorithm of Kershenbaum and Chou (1974). More sophisticated heuristics have been developed by Amberg *et al* (1996), Sharaiha *et al* (1997), Patterson *et al* (1999), Ahuja *et al* (2001, 2003) and Souza *et al* (2003), that employ techniques such as local search. The problem with these approaches is that in each iteration the new solution does not always improve the objective function, thus they are quite slow in converging to high quality trees. More recent metaheuristics include: (a) the hybrid ant colony algorithm of Raimann and Laumanns (2006) that solves the CVRP and applies an implementation of Prim's algorithm to obtain a feasible CMST solution; (b) the work of Martins (2007) that proposes an enhanced version of the second order (SO) algorithm

originally described by Karnaugh (1976), one of the first metaheuristics applied for the CMSTP; (c) the tabu search heuristic of Rego *et al* (2010) introducing dual and primal–dual RAMP algorithms, (d) the filter and fan algorithm by Rego and Mathew (2011); and (e) the biased random—key genetic algorithm (BRKGA) of Ruiz *et al* (2015).

The enhancements of construction algorithms, play a great role in producing very good solutions to complex classical combinatorial problems such as the VRP and the CMSTP. Altinel and Oncan (2005) reported that to improve the accuracy of the Clarke and Wright (1964) heuristic for the VRP problem without harming its speed and simplicity is an interesting question and proposed a new enhancement of the savings criterion; the new method was both fast and very accurate. Bruno and Laporte (2002) suggested a simple enhancement of the Esau–Williams heuristic for the CMSTP removing the longest edge on the path linking vertex j to the root against removing the first edge in the path linking j to the root of the tree according to the Esau–Williams heuristic. Oncan and Altinel (2009) proposed three parametric enhancements of the Esau–Williams heuristic for the CMSTP: In the first enhancement they parametrized the classical saving criterion of Esau–Williams; in the second enhancement, they added a term expressing the asymmetry between two vertices with respect to the central vertex. In the third enhancement the authors took into account the fact that CMSTP is a combination of the minimum spanning tree and of the bin packing problem; so they added a third term that included demand information over the joining process.

Battara *et al* (2012) justified the popularity of the Esau–Williams heuristic in practice and the motivation behind its enhancements, recognizing the problem that the best metaheuristic implementations outperform classical heuristics but they require long computational times and many are not very easy to implement. Additionally, they claimed that the parameters involved in the Esau–Williams enhancements improve their competition with the best metaheuristics and proposed a genetic algorithm procedure to tune efficiently a three-parameter enhancement of the latter algorithm. The proposed evolutionary approach produced high quality results without affecting its simplicity in a limited amount of computing time.

In our work we proposed a two-stage algorithm for the CMSTP: First, motivated by the results of Oncan and Altinel (2009), we develop a new greedy function that measures the cost of linking vertex j to the partial -under construction- capacitated minimum spanning tree. This composite greedy function combines the effects of several metrics and the heuristic follows Prim's optimization framework. Then we increase the space of feasible solutions by perturbing the input data using the law of cosines, in order to explore multiple and possibly not easily reachable solutions by applying the previous framework. Computational tests on benchmark problems from the literature show that the new heuristic provides extremely good performance results for the CMSTP. Furthermore, excluding the feasible space expansion, we show that we can still obtain good quality solutions in very short computational times.

The remainder of the paper is organized as follows: Section 2 provides an overview of the steps embedded within the proposed heuristic, the selection criteria, and the cosine-based engine for the expansion of the feasible solution space. Section 3 offers the computational results while the conclusions are summarized in Section 4.

2. The heuristic and the feasible solution space expansion mechanism

Let $G = (V, E)$ be an undirected graph, where $V = \{0, 1, 2, \ldots, n\}$ is the vertex set, with 0 as the root vertex, and $E = \{(i, j): i, j \in V, i \neq j\}$ is the edge set. A nonnegative weight or demand q_i is associated with each vertex $i \in V\text{-}\{0\}$, and a length or cost c_{ij} is associated with each edge $\{i, j\}$. Given a spanning tree, any subtree linked to the root by a single edge is called a main subtree. Given a vertex $i \in V\text{-}\{0\}$ the main subtree containing j is called the subtree of j. We refer to the first vertex in the subtree of j as the gate vertex $g(j)$ of the subtree of j.

Our work is based on the pioneering algorithms of Prim (1957) and Esau–Williams (1966). Prim's algorithm starts only with the root vertex in the spanning tree and at each iteration, the vertex whose distance or cost to any vertex already in the tree is minimal is brought into the tree. Esau–Williams's algorithm, on the other hand, starts with each vertex and the root vertex in separate components; then they define a tradeoff function C_{ij} (different from c_{ij}) as the minimum cost of connecting the component containing vertex i to the root vertex minus the cost of connecting vertices i and j. Thus, at each stage, the algorithm finds $C_{i^*j^*} = \max(C_{ij})$ and brings in line (i^*j^*), forming a new component without exceeding the capacity constraint.

The algorithm we propose requires the definition of the "shortest point" for a vertex i, i.e., the root vertex or a vertex in the tree that has the minimum direct distance with i and the linking is feasible (does not violate the capacity constraint); we denote the shortest point for vertex i by $s(i)$. Furthermore, let $C^i_{s(i)}$ be the value of the composite selection criterion that is associated with the selection of vertex i; $C^i_{s(i)}$ will be properly defined later in this section.

Given all the previous definitions and notation, the steps of the new heuristic are as follows:

PEW Heuristic

Step 0 Initialization. Read n, c_{ij}, Q $\forall$ $i, j = 0, \ldots, n$

Step 1 Select the vertex nearest to the root to start the spanning tree

Step 2 Find the feasible vertex j that minimizes the composite criterion $C^j_{s(j)}$:

- Step 2a Select the shortest point $s(j)$ and calculate $C^j_{s(j)}$
- Step 2b Repeat step 2a for all vertices not linked to the spanning tree
- Step 2c Select vertex j with the minimum $C^j_{s(j)}$

Step 3 Link the selected vertex j to its shortest point on the current spanning tree and update the spanning tree; set vertex j as a tree vertex

Step 4 If there are vertices remaining to be linked to the spanning tree, return to Step 2; otherwise proceed to Step 5

Step 5 Terminate; get sequence of vertices in each subtree and total distance

The overall loop is performed until all vertices have been assigned to the spanning tree. The solution procedure is straightforward adopting a very simple execution mechanism. However, it is based on new criteria for vertex selection and linking, which are motivated by the minimization function of the new enhancement heuristics for the CMSTP.

2.1. Selection criteria

To form our new composite selection function, we combine six metrics through a weighted linear relationship.

The first metric is $C^1_{s(j)j}$ defined as the direct distance of vertex j to the subtree $s(j)$ (the shortest point of vertex j), namely $c_{s(j)j}$.

The second metric is $C^2_{s(j)j}$ defined as the direct distance of vertex j to the gate node (or vertex) of $s(j)$, that is $c_{jg(s(j))}$.

The third metric is C^3_{ij} i.e., the distance of the shortest point $s(j)$ to the gate vertex of $s(j)$, named $c_{g(s(j))s(i)}$.

The fourth metric is C^4_{ij} defined as the inverse of the following parameterized saving expression (Oncan and Altinel, 2009), adjusted for the component case with one vertex inside: $C^4_{s(j)j} = (c_{dj} - \alpha \times c_{s(j)j})^{-1}$, where d is the root vertex and α is the positive tree shape parameter. This formula is the inverse of the parameterized extension of the savings formula of the Esau–Williams heuristic because our composite vertex selection criterion ensures that a vertex j selected for subtree connection will minimize the selection criterion.

The fifth metric is motivated by Paessens (1988) who introduced a new term to the savings expression of the Vehicle Routing Problem solution algorithm, that was the asymmetry between customers i and j with respect to their distances to the depot. Oncan and Altinel (2009) used the same term to extent the first enhancement of the Esau–Williams heuristic for the CMSTP. The inverse of the asymmetry is included in our selection criterion by the metric: $C^5_{ij} = |c_{ds(j)} - c_{dj}|^{-1}$ where d is again the root vertex.

The sixth metric involves the notion of a "moving vertex" m, i.e., a vertex that is second, third, fourth, etc. in distance from the root vertex; this is to capture additional information about the spatial distribution of vertices in a space expanding mechanism logic embedded within the selection criteria. In the first iteration of the metric calculations, the moving vertex is the second in distance vertex nearest to the root; in the next iteration, m is the third in distance and so on so forth. Consequently, the calculation of this new metric involves an iterative scheme based on the moving vertex m. The criterion is defined as $C^6_{s(j)j} = |c_{j,f} - \beta \times c_{j,m}|$, where β is a tuning parameter taking values between 0 and 1, f is the nearest to the root vertex, and the final value of $C^6_{s(j)j}$ is the minimal one along all possible m's examined.

To summarize, we can state the following:

- Criteria C^1, C^2 and C^3 are simple distance-based functions we define to link vertex j to other points of the network.
- Criterion C^4 is the inverse of the parametrized extension of the saving formula of Esau–Williams proposed by Oncan and Altinel (2009).
- Criterion C^5 is the inverse of the third term within the second enhancement of the saving formula of Esau–Williams proposed by Oncan and Altinel (2009).
- Criterion C^6 is a new composite criterion we propose for the first time.

Now we can define the overall vertex selection criterion which accounts for all previously defined metrics. We use a simple linear relationship to merge the effects of the six metrics. The greedy function that measures the cost of connecting vertex j to its shortest point $s(j)$ in the minimum spanning tree under construction is denoted as $C^j_{s(j)}$. This composite greedy function is defined as follows:

$$C^j_{s(j)} = b_1 C^1_{s(j)j} + b_2 C^2_{s(j)j} + b_3 C^3_{s(j)j} + b_4 C^4_{s(j)j} + b_5 C^5_{s(j)j} + b_6 C^6_{s(j)j} \quad (1)$$

In (1) $b_1, b_2, b_3, b_4, b_5, b_6 \geq 0$. Note that the weights b_1, b_2, b_3, b_4, b_5, b_6 define the relative importance of the associated metric in the selection of vertex j. It is important to note that a

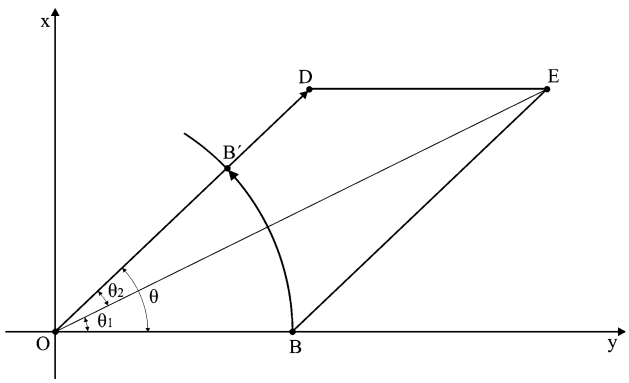

Figure 1 The low of cosines embedded in the heuristic (LCP).

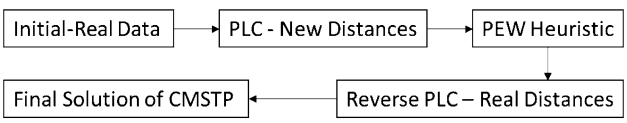

Figure 2 The PEW-PLC heuristic.

determinant factor for the effective deployment of the proposed greedy heuristic is the selection of appropriate weights and the metrics' parameters embedded in the composite greedy function. The tuning of these parameters requires statistical experimentation. We discuss the determination of the intervals of weights and parameters of the metrics in the computational results section.

The composite greedy function allows for the exploration of a large solution space, and thus, we expect the criterion by itself to lead an unsophisticated method to solutions of high quality; such expectations are proven by the application of the proposed heuristic to literature benchmarks. Furthermore, the diversity of the individual selection metrics can capture the specific and the unique characteristics of each problem instance, thus leading a simple greedy approach to evolve in a manner similar to that of meta-heuristics.

2.2. Mechanism for feasible solution space expansion

Hart and Shogan (1987) suggested the perturbation of the problem's data and the reapplication of the relevant algorithm as one way to improve the performance of a heuristic. Thus, instead of applying a heuristic only to the original data, they claimed that improvements can be achieved when several minor perturbations of the data are used as starting points for the algorithm's execution. The best of the solutions obtained can then be implemented using the original data.

Based on this observation, to improve the performance of any proposed greedy heuristic for the CMSTP, we can proceed to data perturbation by altering the distance between the root vertex and the second in distance vertex nearest to the root. We opt to move this particular vertex because it is included in the selection criterion of our algorithm ($C^6_{s(j)j}$); furthermore,

Table 1 Comparison of the results for benchmark instances te40 with $n = 40$ vertices

Sets	*Q*	*Best*	*PEW*	*Iterations*	*PEW CPU*	*PEW % deviations*	*EW3 % deviations*	*EW3 CPU*	
te40-1	3	1190	1201 (8)	44820	13	0.924	0.558	59	–
te40-2	3	1103	1109 (35)	222524	63	0.543	0.544	60	*
te40-3	3	1115	1117 (7)	36789	11	0.179	0.897	59	*
te40-4	3	1134	1139 (6)	31527	9	0.440	0.176	61	–
te40-5	3	1115	1112 (7)	38654	11	0.000	0.000	57	*
te40-1	5	830	841 (12)	71900	21	1.325	3.614	52	*
te40-2	5	792	806 (8)	42485	13	1.767	1.894	51	*
te40-3	5	797	807 (14)	84256	25	1.254	2.509	56	*
te40-4	5	814	838 (12)	70837	20	2.948	1.843	52	–
te40-5	5	784	810 (7)	35195	11	3.316	1.403	54	–
te40-1	10	596	598 (3)	12832	4	0.335	5.705	49	*
te40-2	10	573	590 (5)	22603	7	2.966	6.108	47	*
te40-3	10	568	572 (15)	90727	28	0.704	3.697	50	*
te40-4	10	596	598 (3)	11680	4	0.335	0.671	46	*
te40-5	10	572	584 (3)	8924	3	2.097	2.797	46	*
Average					16.2	1.275	2.163	53.3	
Improvements					69.60%	41.05%			

Table 2 Comparison of the results for benchmark instances tc40 with n = 40 vertices

Sets	*Q*	*Best*	*PEW*	*Iterations*	*PEW CPU*	*PEW % deviations*	*EW3 % deviations*	*EW3 CPU*	
tc40-1	3	742	749 (3)	10218	3	0.943	0.809	29	*
tc40-2	3	717	721 (18)	107602	32	0.557	2.092	30	*
tc40-3	3	716	722 (7)	35171	11	0.837	1.257	27	*
tc40-4	3	775	783 (4)	15326	5	1.032	0.516	29	
tc40-5	3	741	746 (5)	24156	7	0.674	0.135	28	
tc40-1	5	586	590 (24)	146617	51	0.682	0.341	26	
tc40-2	5	578	580 (14)	81908	27	0.346	1.730	25	*
tc40-3	5	577	579 (9)	48523	17	0.346	1.906	24	*
tc40-4	5	617	618 (12)	70810	25	0.162	1.135	27	*
tc40-5	5	600	604 (17)	10353	35	0.666	0.833	26	
tc40-1	10	498	502 (7)	38734	12	0.803	0.000	24	
tc40-2	10	490	490 (5)	25207	9	0.000	0.000	26	–
tc40-3	10	500	504 (3)	8914	4	0.800	1.600	25	*
tc40-4	10	512	520 (5)	22046	8	1.562	0.000	24	
tc40-5	10	504	512 (5)	21958	8	1.587	0.000	26	
Average					16.93	0.733	0.824	26.40	
Improvements					35.87%	11.04%			

Table 3 Comparison of the results for benchmark instances te80 with n = 80 vertices

Sets	*Q*	*Best*	*PEW*	*Iterations*	*PEW CPU*	*PEW % deviations*	*EW3 % deviations*	*EW3 CPU*	
te80-1	5	2544	2588 (18)	107329	212	1.729	0.825	766	–
te80-2	5	2551	2582 (8)	41650	82	1.215	1.568	780	*
te80-3	5	2612	2666 (49)	313666	623	2.067	1.914	740	–
te80-4	5	2558	2567 (36)	227626	451	0.351	1.368	759	*
te80-5	5	2469	2528 (49)	314649	680	2.389	0.486	794	–
te80-1	10	1631	1713 (20)	124145	279	5.027	4.598	748	–
te80-2	10	1639	1673 (4)	19079	43	2.074	2.807	760	*
te80-3	10	1687	1743 (31)	196308	401	3.319	2.964	763	–
te80-4	10	1629	1669 (4)	18324	38	2.455	2.824	741	–
te80-5	10	1603	1668 (7)	35207	73	4.054	5.303	740	*
te80-1	20	1275	1312 (13)	77462	163	2.901	0.784	721	–
te80-2	20	1224	1242 (2)	5201	11	1.470	3.676	710	*
te80-3	20	1267	1288 (4)	18248	39	1.657	5.130	732	*
te80-4	20	1265	1294 (5)	24190	51	2.292	4.901	704	*
te80-5	20	1240	1247 (7)	37925	78	0.564	0.887	718	*
Average					114.93	2.237	2.670	745.10	
Improvements					71.15%	16.22%			

Table 4 Comparison of the results for benchmark instances tc80 with n = 80 vertices

Sets	*Q*	*Best*	*PEW*	*Iterations*	*PEW CPU*	*PEW % deviations*	*EW3 % deviations*	*EW3 CPU*	
tc80-1	5	1099	1137 (18)	110096	256	3.457	4.732	423	*
tc80-2	5	1100	1142 (32)	201544	462	3.818	2.818	430	
tc80-3	5	1073	1120 (24)	148931	332	4.380	3.728	419	
tc80-4	5	1080	1134 (11)	63517	135	5.000	3.056	425	
tc80-5	5	1287	1325 (30)	188180	421	2.952	3.263	421	*
tc80-1	10	888	902 (26)	159829	338	1.576	3.266	413	*
tc80-2	10	877	894 (9)	48208	102	1.938	2.281	428	*
tc80-3	10	878	898 (19)	116089	244	2.277	2.506	409	*
tc80-4	10	868	878 (15)	90565	186	1.152	3.571	411	*
tc80-5	10	1002	1023 (11)	65087	134	2.095	4.691	420	*
tc80-1	20	834	840 (6)	30832	66	0.719	0.959	399	*
tc80-2	20	820	824 (3)	12526	27	0.487	0.732	405	*
tc80-3	20	828	832 (3)	8950	19	0.483	0.483	395	–
tc80-4	20	820	824 (2)	5309	12	0.487	0.488	412	*
tc80-5	20	916	938 (2)	2203	5	1.528	3.275	404	*
Average					124.53	2.156	2.657	414.27	
Improvements					55.92%	18.85%			

Table 5 Average PEW improvements on EW3's solutions

Data set	*Solution improvement (%)*	*Fraction of CPU time (%)*
te-40	41.05	30.40
tc-40	11.05	64.13
te-80	16.22	28.85
tc-80	18.85	44.08

according to Kershenbaum and Chow (1974), the second nearest feasible neighbor leads to significant benefits for the overall solution.

To perturb the CMSTP data, we move the second in distance nearest to the root vertex B cyclically by an angle θ, while also changing its distance from vertex O (which is the root vertex *0*) as shown in Figure 1. Thus, first B moves to B' and then B' moves to D, and using the law of cosines we can calculate the distance DE (where E is any random vertex) as follows:

$$EB^2 = OE^2 + OB^2 - 2 \times OE \times OB \times \cos\angle BOE = >$$

$$\cos\angle BOE = \left(OE^2 + OB^2 - EB^2\right)/(2 \times OE \times OB)$$

so the angle BOE is known. Because the angle BOD is known by the cyclic move of vertex B, we conclude that $\angle\mathrm{DOE} = \angle\mathrm{BOD} - \angle\mathrm{BOE}$. The law of cosines also states:

$$DE^2 = OD^2 + OE^2 - 2 \times OD \times OE \times \cos\angle DOE$$

As a result, the distance between new location of B (which is D now) and the random vertex E is known.

Table 6 PEW vs other heuristics on existing best solutions

Instance set	*EWBF3 (2008)*		*EW3 (2009)*		*PEW*	
	Dev. (%)	*Time (s)*	*Dev. (%)*	*Time (s)*	*Dev. (%)*	*Time (s)*
tc40	1.14	371.22	0.82	396.00	0.73	253.95
te40	2.91	372.42	2.16	799.00	1.27	243.00
tc80	2.67	3102.33	2.65	6214.00	2.16	1867.95
te80	5.01	3180.34	2.67	11176.00	2.24	1723.95
Average	2.93	1756.58	2.07	4646.25	1.60	1022.21
Total		7026.31		18585.00		4088.85

Table 7 Comparison between heuristics on EW solutions

Instance set	*EWBF3 (2008)*		*EWR (2012)*		*MEW (2002)*		*EW3 (2009)*		*PEW*	
	Imp. (%)	*Time (s)*	*Imp. (%)*	*Time (s)*	*Imp. (%)*	*Time (s)*	*Imp. (%)*	*Time (s)*	*Imp. (%)*	*Time (s)*
tc40	1.87	371.22	2.13	12.72	0.96	2.7	2.47	396.00	2.56	253.95
te40	2.86	372.42	3.22	12.81	0.69	2.7	2.39	799.00	3.22	243.00
tc80	3.76	3102.33	3.95	104.81	0.90	6.00	2.16	6214.00	2.58	1867.95
te80	2.55	3180.34	2.75	101.91	0.31	10.20	2.44	11176.00	2.85	1723.95
Average	2.76	1756.58	3.01	232.25	0.72	5.4	2.36	4646.25	2.80	1022.21
Total		7026.31		58.06		21.6		18585		4088.85

Table 8 Solution quality for PEW-PLC on te40 test problem

Sets	*Q*	*BEST*	*EW*	*EW3% DEV*	*PEW-PLC*	*PEW-PLC % DEV*	*PEW-PLC %IMP*	
te40-1	3	1190	1215	0.588	1194	0.336	1.728	–
te40-2	3	1103	1134	0.544	1104	0.090	2.646	*
te40-3	3	1115	1146	0.897	1115	0.000	2.705	*
te40-4	3	1134	1153	0.176	1139	0.440	1.214	–
te40-5	3	1110 (1115)	1147	0.000	1110	0.000	3.226	**
te40-1	5	830	857	3.614	839	1.084	2.100	*
te40-2	5	792	839	1.894	806	1.767	3.933	*
te40-3	5	797	820	2.509	807	1.254	1.585	*
te40-4	5	814	854	1.843	829	1.843	2.927	*
te40-5	5	784	816	1.403	801	2.169	1.838	–
te40-1	10	596	658	5.705	598	0.335	9.118	*
te40-2	10	573	632	6.108	589	2.792	6.804	*
te40-3	10	568	596	3.697	572	0.704	4.027	*
te40-4	10	596	638	0.671	598	0.335	6.270	*
te40-5	10	572	597	2.797	582	1.748	2.513	*
Average				2.163		0.993	3.509	
Improvement						54.09%		

Our mechanism of feasible solution space expanding calculates all the new distances between the vertices of the graph and the new location of the second nearest to the root vertex. Our proposed greedy algorithm implemented to new distance matrix producing a new spanning tree under the capacity restriction. Then, we can recalculate the cost of the solution using the real distance matrix. The final results should lead to improvements in the objective function value for data sets, as per the claim of Hart and Shogan (1987).

The sequential steps of this expanded procedure, denoted as PEW-PLC heuristic, are shown in Figure 2. Note that PLC refers to the recalculation of distances after the perturbation via the angle rotation.

3. Computational results

The proposed heuristic in both its versions (PEW and PEW-PLC) has been implemented in FORTRAN 90, using the Fortran PowerStation 4.0 compiler. The computational experiments have been performed on a PC with an Intel Core i5 processor. The heuristics were tested on the classical unit demand data sets from the OR-Library (http://people.brunel.

Table 9 Solution quality for PEW-PLC on te80 test problem

Sets	*Q*	*BEST*	*EW*	*EW3% DEV*	*PEW-PLC*	*PEW-PLC %DEV*	*PEW-PLC % IMP*	
te80-1	5	2544	2604	0.825	2588	1.730	0.614	–
te80-2	5	2551	2633	1.568	2576	0.980	2.165	*
te80-3	5	2612	2723	1.914	2663	1.953	2.203	–
te80-4	5	2558	2624	1.368	2551	0	2.782	**
te80-5	5	2469	2593	0.486	2505	1.458	3.394	–
te80-1	10	1631	1746	4.598	1708	4.721	2.176	–
te80-2	10	1639	1748	2.807	1670	1.891	4.462	*
te80-3	10	1687	1828	2.964	1735	2.891	5.088	*
te80-4	10	1629	1685	2.824	1644	0.921	2.433	*
te80-5	10	1603	1712	5.303	1666	3.930	2.687	*
te80-1	20	1275	1330	0.784	1312	2.902	1.353	–
te80-2	20	1224	1289	3.676	1242	1.471	3.646	*
te80-3	20	1267	1340	5.130	1288	1.657	3.881	*
te80-4	20	1265	1343	4.901	1294	2.292	3.649	*
te80-5	20	1240	1334	0.887	1247	0.565	6.521	*
Average				2.669		1.957	3.137	
Improvement						26.68%		

Table 10 Solution quality for PEW-PLC on tc40 test problem

Sets	*Q*	*BEST*	*EW*	*EW3 % DEV*	*PEW-PLC*	*PEW-PLC % DEV*	*PEW-PLC % IMP*	
tc40-1	3	742	774	0.809	747	0.673	3.488	*
tc40-2	3	717	749	2.092	720	0.418	3.872	*
tc40-3	3	716	728	1.275	722	0.838	0.824	*
tc40-4	3	775	804	0.516	781	0.774	2.861	–
tc40-5	3	741	760	0.135	746	0.674	1.842	–
tc40-1	5	586	595	0.341	590	0.682	0.840	–
tc40-2	5	578	588	1.730	580	0.346	1.361	*
tc40-3	5	577	602	1.906	579	0.346	3.821	*
tc40-4	5	617	645	1.135	617	0.000	4.341	*
tc40-5	5	600	615	0.833	604	0.667	1.789	*
tc40-1	10	498	516	0.000	500	0.402	3.101	–
tc40-2	10	490	505	0.000	490	0.000	2.970	*
tc40-3	10	500	517	1.600	504	0.800	2.515	*
tc40-4	10	512	524	0.000	518	1.172	1.145	–
tc40-5	10	504	540	0.000	506	0.396	6.296	–
Average				0.824		0.546	2.738	
Improvement						33.74%		

ac.uk/∼mastjjb/jeb/info.html). The tc instances in this library have the central vertex in a central position with respect to the other ones. The te instances have the central vertex in a corner with respect to the other ones. The problems include ten instances with fully connected graphs of 40-vertices with arc capacities 3, 5 and 10, and ten instances with fully connected graphs of 80-vertices with arc capacities 5, 10 and 20. Thus, a total of 60 problem instances are examined solved.

One requirement in our approach is the determination of the intervals of weights b_1, b_2, b_3, b_4, b_5, and b_6 in the selection formula, as well as the setting of parameters α and β in the metrics of the cost function. In our experiments, the weights b_1, b_2, b_3, b_4, b_5, and b_6 are chosen within the interval [0.0, 1.0] in an incremental manner with increment set to 0.5. Parameters α and β are set in the same manner.

The experimental results of the proposed greedy heuristic PEW are reported in Tables 1, 2, 3 and 4. The test problems reference in OR Library and its capacity are shown in columns one and two of each table. The optimal known (literature) solutions for each instance or the relevant lower bound if the optimal is not reached are listed in the third column, while the best solutions produced by PEW are reported in the fourth column (distance of the capacitated minimum spanning tree and moving vertex in parenthesis). In the fifth column, we provide the number of capacitated spanning trees produced up to point when the best solution is obtained. The sixth column of each table offers the CPU time in seconds that PEW requires to find the best solution.

The seventh column depicts the percentage deviation of each instance with respect to the best known solution; they are

Table 11 Solution quality for PEW-PLC on tc80 test problem

Sets	*Q*	*BEST*	*EW*	*EW3% DEV*	*PEW-PLC*	*PEW-PLC %DEV*	*PEW-PLC % IMP*	
tc80-1	5	1099	1182	4.732	1134	3.185	4.061	*
tc80-2	5	1100	1170	2.818	1129	2.636	3.504	*
tc80-3	5	1073	1131	3.728	1115	3.914	1.415	–
tc80-4	5	1080	1151	3.056	1118	3.519	2.867	–
tc80-5	5	1287	1338	3.263	1325	2.953	0.972	*
tc80-1	10	888	920	3.266	900	1.351	2.174	*
tc80-2	10	877	917	2.281	894	1.938	2.508	*
tc80-3	10	878	916	2.506	892	1.595	2.620	*
tc80-4	10	868	915	3.571	878	1.152	4.044	*
tc80-5	10	1002	1069	4.691	1022	1.996	4.397	*
tc80-1	20	834	856	0.959	840	0.719	1.869	*
tc80-2	20	820	836	0.732	820	0.000	1.914	*
tc80-3	20	828	856	0.483	832	0.483	2.804	*
tc80-4	20	820	866	0.488	820	0.000	5.312	*
tc80-5	20	916	971	3.275	922	0.655	5.046	*
Average				2.657		1.740	3.034	
Improvement						34.51%		

Table 12 Aggregate comparison between literature best enhancements of EW heuristic and PEW/PEW-PLC

Set	*Q*	*EW3*	*PEW*	*PEW-PLC*
tc40	3	0.962	0.809	0.675
	5	1.189	0.440	0.408
	10	0.320	0.950	0.554
tc80	5	3.519	3.921	3.241
	10	3.263	1.808	1.606
	20	1.187	0.741	0.371
te40	3	0.435	0.417	0.173
	5	2.253	2.122	1.623
	10	3.796	1.287	1.183
te80	5	1.232	1.550	1.224
	10	3.699	3.386	2.871
	20	3.076	1.777	1.777
Average		2.078	1.601	1.309

Table 13 Comparison between EW3 and PEW-PLC heuristics and optimal solutions

Instances	*Best/LB*	*EW3*	*PEW-PLC*
te40-1 (10)	596 (LB)	630	598 (New Best)
te40-5 (3)	1104 (LB)	1115*	1110 (New Best)
tc40-2 (10)	490 (LB, Best)	490	490 (LB, Best)
tc40-4 (5)	617(LB, Best)	624	617 (LB, Best)
te80-4 (5)	2558 (Best)	2593	2551 (New Best)
tc80-2 (20)	820 (LB, Best)	826	820 (LB, Best)
tc80-4 (20)	820 (LB, Best)	824	820 (LB, Best)

* Patterson *et al* (1999)

calculated according to the formula $100 \times (z - z^*)/z^*$, where z is the objective value obtained by proposed heuristic and z^* is the best known value reported in the literature that is listed in the third column. The eighth column entitled EW3 reports the percentage deviation of the best enhancement of Esau–Williams heuristic according to Oncan and Altinel (2009) from the best-known solutions. The ninth column presents the CPU times in seconds of the Oncan and Altinel (2009) heuristic. Note that we cannot directly compare computational times between our approach and that of Oncan and Altinel (2009) since different processors were used and no scaling has been applied. In the last column, we indicate instances for which our heuristic outperforms (or is equal to) the best enhancement of Esau–Williams with '*'. The solution derived by our heuristic

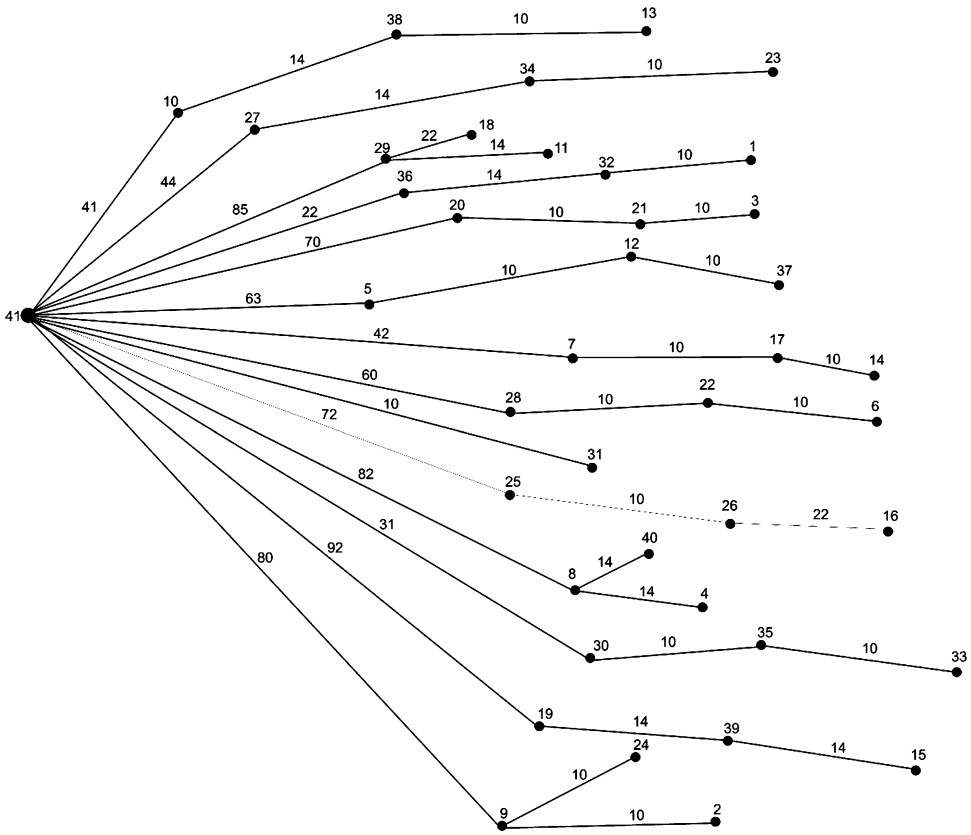

Figure 3 Best CMST for te40-5(3).

is equal to or better than those of the best enhancement heuristic in more than 50% of the data sets.

From Tables 1, 2, 3 and 4, it is evident that PEW improves upon classical heuristic approaches and provides results that are comparable to the ones produced by more computationally expensing metaheuristics. Tables 5, 6 and 7 further support this statement. Specifically, Table 5 shows the average improvement in the solution quality for each set of instances examined versus the fraction of the CPU time required to reach this improvement – the effectiveness of PEW is obvious.

In Table 6, we compare the PEW heuristic to the results obtained by EWBF3 and EW3 reported by Battara *et al* (2008) and Oncan and Altinel (2009) respectively. The EWBF3 is an enhancement of the Esau–Williams heuristic towards a single-stage genetic search procedure for finding the best parameter values of the savings expression for the three-parameter EW enhancement (EW3) of Oncan and Altinel (2009). Also, Oncan and Altinel (2009) determine the best values of the parameters using a brute force evaluation procedure within given intervals. The overall average percentage deviation from the best known solution values is 1.60% for our proposed PEW (2016) heuristic compared to 2.39% for the EWBF3 (2008) and to 2.07% for the EW3 (2009). Also, our approach reduces the computational time considerably. For example, the total time to find the best solution for all problems is 4088.85 s against the 18585.00 s of EWBF3 and 7026.32 s of EW3 respectively.

In Table 7, we continue the comparison of PEW with previously developed approaches. The "Imp (%)" columns reports the average percentage improvement of a heuristic over Esau–Williams for each data set. We summarize the results obtained by consideration the following solution approaches when applied to the same data sets: (a) the enhancement of Battara *et al* (2008) EWBF3, (b) the new genetic enhancement of Battara *et al* (2012) EWR, the new genetic enhancement combined with local search and randomized prohibitions, (c) the modified enhancement of Bruno and Laporte (2002), MEW, (d) the third enhancement of Oncan and Altinel' s (2009) EW3, and (e) the suggested PEW. From the results of Table 7, it is clear that PEW outperforms all approaches apart from EWR, which is a more complex metaheuristic.

As an overall conclusion, we can state that PEW outperforms heuristics and is close to metaheuristics with respect to solution quality since it examines a larger portion of the solution space than simple heuristics. Note that the total number of runs using one moving vertex without using the expanding mechanism is $3^8 = 6561$. The total number of runs for the recent enhancement heuristic (Oncan and Altinel, 2009) is 7600 runs. Using all the 39 vertices (the nearest

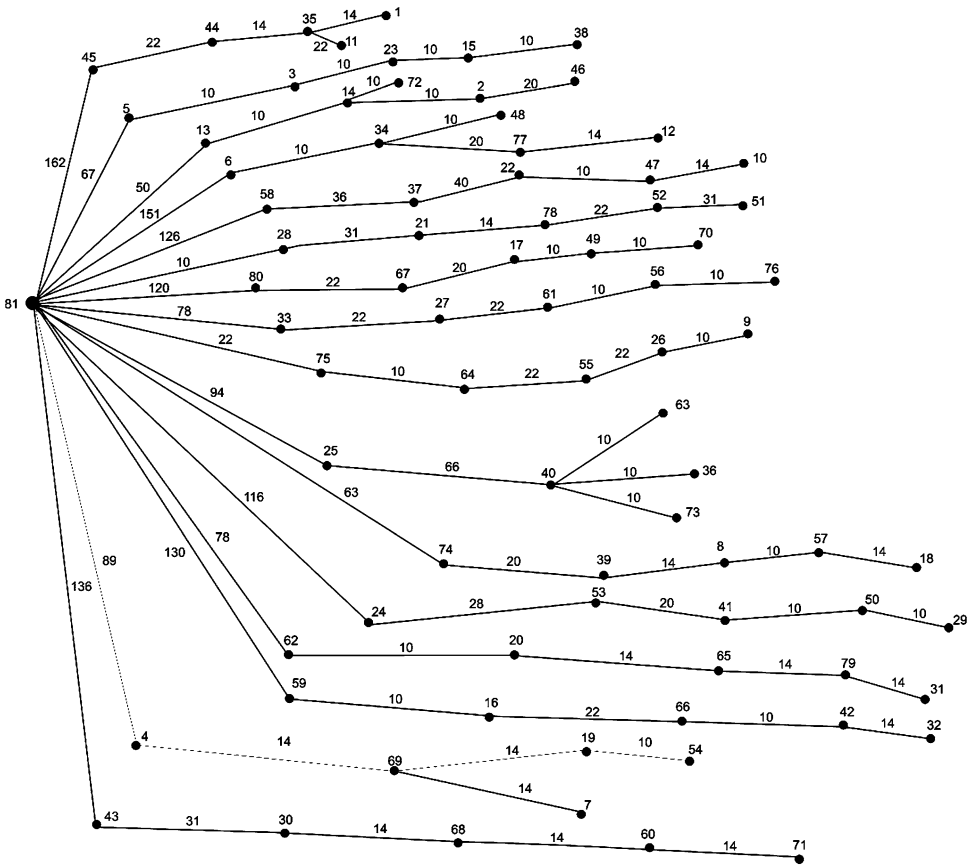

Figure 4 Best CMST for te80-4(5).

neighbor vertex not included), the number of capacitated spanning trees produced is $6561 \times 39 = 255879$ for the 40-vertex instances, and $6561 \times 79 = 518319$ for the 80-vertex instances. The total CPU time in seconds spent to perform all iterations of our heuristic per instance is approximately 70 s for the 40 vertex instances and 1000 s for the 80 vertex instances.

The experimental results of the proposed PEW-PLC heuristic are reported in Tables 8, 9, 10 and 11. The PEW-PLC heuristic was compared to the results obtained by EW3. Also, for each test set, the average percentage improvement with respect to EW are reported. The test problems are shown in columns one and two, the best known solution for each instance is listed in the third column. The Esau–Williams solutions are reported in the fourth column. The fifth column depicts the average percentage deviations of the EW3. The sixth column presents the best solution produced by the proposed PEW-PLC heuristic. The seventh columns reports the average percentage deviation of each instance from the best solution. In the last column we present the average percentage deviation of the PWE-PLC heuristic from the Esau–Williams heuristic. The final column shows a "*" whenever our result is better than EW3 and a "**" whenever our result is the best known to-date.

Table 14 Aggregate comparison of PEW-PLC with EWR

Data set	*EWR Imp %*	*PEW-PLC Imp %*	
tc40	2.13	2.74	*
te40	3.22	3.51	*
tc80	3.95	3.03	–
tc80	2.75	3.14	*
Average	3.01	3.11	

When we apply the PLC procedure, a crucial point is the determination of the increments of the angle in the space [0, 3.14] and of the length of the change interval in the space [0.1, 2.2], both with an increment of 0.5. This means that we permit the distance of the moving vertex m to vary between $0.1 \times c_{m,0}$ and $2.2 \times c_{m,0}$, where $c_{m,0}$ is the actual distance of the moving vertex m to the root vertex 0. As a result the total number of iterations to cover all possible parameter values is 35, while the number of CMST's produced is $6561 \times 35 = 229635$.

When the increments reduce or/and the intervals increase the required computational effort becomes excessive. On the other hand, we often may obtain better solutions. Note that as

Table 15 Non-zero b_i's from metrics C^1–C^6 in the best solution of PEW

Data Sets	C^1	C^2	C^3	C^4	C^5	C^6
TE40(3)	29.8	12.4	28.8	63.8	11.0	83.8
TE40(5)	29.0	13.4	34.6	66.4	47.0	70.4
TE40(10)	18.4	8.8	39.6	39.4	12.6	58.6
TC40(3)	20.4	21.6	31.8	58.2	24.8	72.8
TC40(5)	21.6	16.2	33.2	66.0	22.8	68.8
TC40(10)	17.8	9.4	44.6	70.8	19.6	74.0
TE80(5)	23.0	9.0	50.0	65.8	37.6	61.8
TE80(10)	19.8	3.4	48.8	58.4	39.6	67.0
TE80(20)	19.2	9.4	58.0	62.0	20.6	77.6
TC80(5)	21.0	9.0	48.8	56.2	21.2	81.6
TC80(10)	13.8	2.8	54.8	57.6	24.4	75.6
TC80(20)	13.6	2.8	36.6	48.6	21.4	72.6
AVERAGE	20.6	9.8	42.4	59.4	25.2	72.0

Table 16 Aggregate ranking of criteria

Criteria	*Low (0.0)*	*Middle to hight (0.5 or 1.0)*	*Ranking*
C^1	79.4	20.6	5
C^2	90.22	9.8	6
C^3	57.6	42.4	3
C^4	40.6	59.4	2
C^5	74.8	25.2	4
C^6	28.0	72.0	1

moving vertex m, in our implementation of PLC, we use the best according to PEW applied before.

In Table 12, we provide some aggregate comparison between the results derived by PEW and PEW-PLC vis-à-vis the best enhancements of the EW heuristic. Specifically, we report the percentage deviation from the optimal solution or lower bound of EW3, PEW and PEW-PLC heuristics aggregated across the individual data sets with identical capacities. From the results, it is obvious that PEW-PLC provides the best solutions overall, followed by PEW and EW3. This result was expected since Table 12 aggregates the results of Tables 1, 2, 3, 4, 8, 9, 10 and 11.

Subsequently, we compare the results of our approaches to existing optimal solutions reported by Ruiz *et al* (2015) and Osman and Atikson (2009).The comparison is provided in Table 13. The optimal solutions or lower bounds are reported in the second column from Ruiz *et al* (2015) for the data sets mentioned in the first column. The solutions produced by Osman and Atikson's heuristic (2009) and PEW-PLC are shown in the third and fourth column, respectively.

From Table 13, it is evident that PEW-PLC reaches the optimal solution or provides a new best solution with respect the cost of the spanning tree for all data sets examined. Specifically, the results indicate that we produce a new best literature solution for data sets te40-5(3) and te80-4(5); the structure of the resulting minimum spanning trees is shown in Figures 3 and 4, respectively.

Table 14 provides an aggregate comparison of the results obtained using the metaheuristic EWR of Battara *et al* (2012) and the ones reached by PEW-PLC. The results indicate that our simple heuristic outperforms a metaheuristic in 3 of the 4 groups (indicated by an asterisk in the last column of Table 14) as well as in the global average. This is an indication of the unique strength of our approach.

Finally, in Tables 15 and 16 we have further explored the contribution of each metric in the derivation of the best solution obtained by our proposed heuristic PEW. They show the values of the b_i weights in the respective best solutions found.

By examining the percentage of instances in each data set where the relevant weights were non-zero, and aggregating instances with b_i's taking values of 0.5 and 1.0, it becomes clear the our proposed metric (C^6), together with the Essay-Williams metric modified by Oncan and Altinel (C^4) are the two dominant metrics in terms of contribution to the objective function value in the best solutions found. On the opposite side (and quite expected we would add) are metrics C^1 and C^2, which are simple distance-based criteria that could be eliminated. Finally, although C^3 is a distance-related metric, it is associated with the shortest point and gate vertex that we have introduced in our approach and provide significant value to the solution quality.

4. Conclusions

In this paper we have developed a new heuristic with two versions to solve the capacitated minimum spanning tree problem. Our approach, which is based upon Prim's (1957) and Esau and Williams (1966) heuristics for the minimum spanning tree and capacitated minimum spanning tree problems respectively, utilizes several metrics exploiting the interrelationships between vertices introduced and dictates the sequence in which the vertex linking takes place.

The second version of the new method allows the expansion of the feasible solution space using the Law of Cosines Procedure (LCP). This procedure allows the search to be

displaced to other regions of the feasibility set, producing additional spanning trees. The paper also emphasizes the importance of a good choice of weights and parameters in the cost function criterion.

The proposed algorithm is simple and easy to implement and to apply to capacitated minimum spanning tree problems with minimal computational effort. It performs very well on test problems from the literature, providing high quality solutions with respect to the cost of the capacitated minimum spanning tree within short computational times. The new method increases remarkably the accuracy of the classical heuristic of Esau–Williams and its enhancements for the capacitated minimum spanning tree problem, without increasing much the complexity and the speed because the new search effort.

The results presented indicate that the heuristic provides solutions that are competitive with the best solutions of metaheuristics for a large number of literature problems. Comparison with optimal solutions show that the new approach reaches near optimal solutions for most problem instances. Our purpose in developing the new method was not to compete with metaheuristics but to produce a simple and powerful heuristic that can match some metaheuristics in terms of solution quality.

In terms of further research we can state the following: (a) better exploitation of information gathering during the linking phase about the structure of the CMST problem; (b) the placement of vertices; (c) the weights and the parameters of the selection criteria in the heuristic; (d) the solution for variants of the large scale CMST problems; and (e) the use CMSTP approaches for vehicle routing; all are worth pursuing research directions.

Acknowledgements—The authors would like to thank the two anonymous referees and the Associate Editor for their acute comments and useful suggestions that helped improve the content and the presentation of the paper.

References

Ahuja RK, Orlin JB and Sharma D (2001). Multi-exchange neighborhood structures for the capacitated minimum spanning tree problem. *Mathematical Programming* **91**(1):71–97.

Ahuja RK, Orlin JB and Sharma D (2003). A composite very large-scale neighborhood structure for the capacitated minimum spanning tree problem. *Operations Research Letters* **31**(3):185–194.

Altinel IK and Oncan T (2005). A new enhancement of the Clarke and Wright savings heuristic for the capacitated vehicle routing problem. *Journal Operational Research Society* **56**(8):954–961.

Amberg A, Domschke W and Voss S (1996). Capacitated minimum spanning trees: algorithms using intelligent search. *Combinatorial Optimization: Theory and Practice* **1**:9–33.

Battara M, Golden BL and Vigo D (2008). Tuning a parametric Clarke–Wright heuristic via a genetic algorithm. *Journal Operational Research Society* **59**(11):1568–1572.

Battara M, Oncan T, Altinel IK, Golden B, Vigo D and Phillips E (2012). An evolutionary approach for tuning parametric Esau and Williams heuristics. *Journal Operational Research Society* **63**(3):368–378.

Bruno G and Laport G (2002). A simple enhancement of the Esau–Williams heuristic for the capacitated minimum spanning tree problem. *Journal Operational Research Society* **53**(5):583–586.

Chandy KM and Russell RA (1972). The design of multipoint linkages in a teleprocessing tree network. *IEEE Transactions on Computers* **21**(10):1062–1066.

Chandy KM and Lo T (1973). The capacitated minimum spanning tree. *Networks* **3**(2):173–181.

Clarke G and Wright J (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research* **12**(4):568–581.

Esau LR and Williams KC (1966). On teleprocessing system design, part II. *IBM System Journal* **5**(3):142–147.

Gavish B (1982). Topological design of centralized computer networks formulations and algorithms. *Network* **12**(4):355–377.

Gavish B (1991). Topological design of telecommunication networks-local access design methods. *Annals of Operations Research* **33**(1):17–71.

Gouveia L and Paixão J (1991). Dynamic programming based heuristics for the topological design of local access networks. *Annals of Operations Research* **33**(4):305–327.

Gouveia L and Martins P (2005). The capacitated minimum spanning tree problem: revisiting hop-indexed formulations. *Computers & Operations Research* **32**(9):2435–2452.

Hall L (1996). Experience with a cutting plane algorithm for the capacitated spanning tree problem. *INFORMS Journal on Computing*. **8**(3):219–234.

Han J, McMahon GB and Sugden SJ (2002). A branch and bound algorithm for capacitated minimum spanning tree problem. *Lecture Notes in Computer Science* **2400**:404–408.

Hart JP and Shogan AW (1987). Semi-greedy heuristics: an empirical study. *Operations Research Letters* **6**(3):107–114.

Karnaugh M (1976). A new class of algorithms for multipoint network optimization. *IEEE Transactions on Communications* **24**(5): 500–505.

Kershenbaum W and Chow A (1974). A unified algorithm for designing multidrop teleprocessing networks. *IEEE Transactions on Communications* **22**(11):1762–1772.

Malik K and Yu G(1993). A branch and bound algorithm for the capacitated minimum spanning tree problem. *Network* **23**(6): 523–532.

Martins P (2007). Enhanced second order algorithm applied to the capacitated minimum spanning tree problem. *Computer & Operations Research* **34**(8):2495–2519.

Oncan T and Altinel IK (2009). Parametric enhancements of the Esau–Williams heuristic for the capacitated minimum spanning tree problem. *Journal Operational Research Society* **60**(2):259–267.

Paessens H (1988). The savings algorithm for the vehicle routing problem. *European Journal of Operational Research* **34**(3):336–344.

Papadimitriou, CH (1978). The complexity of the capacitated tree problem. *Networks* **8**(3):217–230.

Patterson R, Rolland E and Pirkul H (1999). A memory adaptive reasoning technique for solving the capacitated minimum spanning tree problem. *Journal of Heuristics* **5**(2):159–180.

Prim RC (1957). Shortest connection networks and some generalizations. *Bell System Technical Journal* **36**(6):1389–1401.

Rego C, Mathew F and Glover F (2010). RAMP for the capacitated minimum spanning tree problem. *Annals of Operations Research* **181**(1):661–681.

Rego C and Mathew F (2011). A filter-and-fan algorithm for the capacitated minimum spanning tree problem. *Computers and Industrial Engineering* **60**(2):187–194.

Reimann M and Laumanns M (2006). Savings based ant colony optimization for the capacitated minimum spanning tree problem. *Computer & Operations Research* **33**(6):1794–1822.

Ruiz E, Albareda-Sambola M, Fernandez E and Resende MGC (2015). A biased random-key genetic algorithm for the capacitated minimum spanning tree problem. *Computer & Operations Research* **57**:95–108.

Sharaiha YM, Gendreau M, Laporte G and Osman IH (1997). A tabu search algorithm for the capacitated shortest spanning tree problem. *Networks* **29**(3):161–167.

Souza MC, Duhamel C and Ribeiro CC (2003). A GRASP heuristic for the capacitated minimum spanning tree problem using memory-based local search strategy. *Applied Optimization* **86**:627–658.

Toth P and Vigo D (1995). An exact algorithm for the capacitated shortest spanning arborescence. *Annals of Operations Research* **61**(1):121–141.

Uchoa E, Fukasawa R, Lysgaard J, Pessoa A, Poggi de Aragao M and Andrade D (2008). Robust branch-cut-and-price for the Capacitated Minimum Spanning Tree problem over a large extended formulation. *Mathematical Programming Ser A* **112**(2):443–472.

Received 25 July 2016;
accepted 18 October 2016