Benchmark Problem Generators and Results for the Multiobjective Degree-Constrained Minimum Spanning Tree Problem

Joshua D. Knowles and David W. Corne
Department of Computer Science, University of Reading, UK
J.D.Knowles@reading.ac.uk, D.W.Corne@reading.ac.uk
FAX: +44(0) 118 975 1994, TEL: +44 (0) 118 931 8983
http://www.rdg.ac.uk/~ssr97jdk

Abstract

Finding a minimum-weight spanning tree (MST) in a graph is a classic problem in operational research (OR) with important applications in network design. In this paper, we consider the degree-constrained multiobjective MST problem, which is NP-hard. We present several different parameterized problem generators for producing MST instances with different problem features, including any number of objectives, varying degrees of convexity and non-convexity in the Pareto front, and edge weight combinations that mislead greedy approaches. As well as being useful for the OR community, these generators are well-suited to provide problems to form part of a wider (evolutionary) multiobjective test problem suite, where constrained and NP-hard combinatorial problems are sometimes poorly represented. Fifteen instances are generated using the presented methods, and benchmark results on these instances for a multiobjective EA, AESSEA, are presented. These are compared with results from two different non-EA methods. All of our problem instances, generators, and solution sets will be made available for use by other researchers.

1 Introduction

For many years, minimum spanning tree (MST) problems have been of great interest to the operational research community. More recently, the multiobjective minimum spanning tree (mc-MST)[1] problem, in which there are multiple weights defined on each edge, and which is NP-hard, has become subject of increasing interest. Several papers on this subject [3, 5, 14] have proposed approximate polynomial algorithms, and exact methods, for tackling the problem. With growing interest in the evolutionary algorithm (EA) community in multiobjective optimization, it seems likely that this application should now become the focus of more EA approaches also. The first genetic algorithm (GA) for the mc-MST (in which a Prüfer number encoding is used) was proposed by Zhou and Gen [15]. The problems used to test their algorithm were, however, very simple and could be much better solved using good exact methods [14] (smaller problems), or heuristic approximation methods [3, 5] (larger problems). Nonetheless, some variants of the mc-MST may not be easily or efficiently solved by exact methods. One example is when the number of objectives is greater than two. It is then relatively straightforward for a multiobjective EA to be applied, whereas some exact methods [14] and heuristics have only been developed for the bi-objective case. Furthermore, when other constraints need to be incorporated then this may be achieved relatively easily in an evolutionary algorithm but not in some of the pure heuristic approaches. Unfortunately, these more difficult problems are not generally available and have not yet been considered in algorithm studies.

In order to aid in the further development of good evolutionary algorithms and other metaheuristics for a full range of variants of the mc-MST, it would be useful to have a collection of parameterized problem generators to provide benchmark problems that could be used in comparative studies. In this paper we present such a collection of simple problem generators that can provide problems with correlated and anti-correlated weights (which critically affect the shape of the Pareto front, and therefore the applicability of different methods), problems with large regions that have no solutions on the convex hull of the Pareto front, and problems which are difficult to solve when a degree constraint is additionally imposed. All of the gener-

[1] It is often called the multi-criterion MST, hence mc-MST.

ators can generate problems of different sizes, with a differing degree of sparsity, and with any number of objectives (except for the concave graph generator).

The generators proposed may also be used for testing the general strengths and weaknesses of multiobjective EAs, as part of a wider test problem suite. Much progress has been made recently in improving the variety, difficulty, and actual use of, test problem suites by researchers in the evolutionary multiobjective optimization (EMO) community. However, there is still a shortage of parameterized combinatorial problems, constrained problems, and problems with large numbers of objectives. The generators proposed here can provide problems with all these features, separately as well as together.

In this paper, the different generators are used to provide an initial set of 15 different problem instances, each with just two objectives. These instances are then tackled using a multiobjective EA called AESSEA, which is described in detail in a recent paper by us [9], where it was used to solve some much simpler random-weight mc-MST problems. The AESSEA results are also compared with an enumerative algorithm (smaller problems) and a polynomial-time iterated heuristic approach (larger problems) to provide the first benchmark results over a wide range of mc-MST problem types with constraints. All of the problem instances, generators, and results sets are available for others to use by e-mail contact with the first author.

The remainder of the paper is organized as follows: Section 2 defines the mc-MST problem and a degree-constrained variant. In Section 3, the different problem generators are described. Section 4 then describes AESSEA, the encoding and operators used, and the other non-EA algorithms. Section 5 provides details of the experimental method including all parameter settings, and describes the statistical method used to analyze the results. Section 6 presents the results, and Section 7 concludes.

2 Multiobjective Degree-Constrained Minimum Spanning Tree Problem

A spanning tree of an undirected, connected graph, $G = (V, E)$, is a subgraph $T = (V, E_T), E_T \subset E$ that contains all vertices in V and connects them with exactly $|V| - 1$ edges, so that there are no cycles. If G is complete, then the set S of spanning trees T of G has $|S| = |V|^{|V|-2}$ members. If each edge $(i, j) \in E$ has $K > 1$ associated non-negative real numbers, representing K attributes defined on it and denoted with $\mathbf{w}_{i,j} = (w^1_{i,j}, w^2_{i,j}, \ldots, w^K_{i,j})$, then the mc-MST problem may be defined as:

$$\begin{aligned} \text{“minimize” } \mathbf{W} &= (W^1, W^2, \ldots, W^K) \\ \text{with } W^k &= \sum_{(i,j)\in E_T} w^k_{i,j}, \; k \in 1..K \end{aligned} \tag{1}$$

where the term ‘minimize’ is in quotation marks to indicate that it may not be possible to find a single solution that is minimal on all the components of $\mathbf{W}$. Instead, one is required to find a set of spanning trees $S^* \subset S$, called the Pareto optimal set, with the property that:

$$\forall T^* \in S^* \bullet \; \nexists T \in S \bullet T \prec T^* \tag{2}$$

where $T \prec T^* \iff \forall k \in 1..K \bullet W^k \leq {W^k}^* \wedge \exists k \in 1..K \bullet W^k < {W^k}^*$. The expression $T \prec T^*$ is read as T *dominates* T^*, and solutions in the Pareto optimal set are also known as efficient or admissible solutions.

If there is, in addition, a constraint d on the maximum vertex degree in the spanning tree, then the problem is called the multiobjective degree-constrained minimum spanning tree (mc$d$-MST) problem.

3 Problem generators

We propose generators for the following types of non-Euclidean problems:

Random Random (uncorrelated) integer or real number weighted graphs;

Correlated: Random correlated real number weighted graphs;

Anti-Correlated: Random anti-correlated real number weighted graphs;

M-Correlated: Correlated real number weighted graphs with high vertex degree in the underlying MST;

Concave: Real number weighted graphs that have a large concave region in their Pareto front.

All of the above can be generated as either sparse graphs, or complete graphs. We consider only complete graphs in this paper. Similarly, all but the concave graphs can be generated with an arbitrary number of objectives, K, although here we restrict our attention to bi-objective problems only.

Random

The random graph generator simply sets each component of each edge weight vector to a value drawn from a uniformly random distribution within some

range, $\mathcal{U}(min, max)$. In this paper we do not consider graphs of this type but we have already shown [9] that our AESSEA algorithm using a direct coding and specialized operators is superior to the AESSEA algorithm using a Prüfer encoding [12, 15] on problems of this type. Here we restrict our attention to more difficult problem types.

Correlated and Anti-Correlated

The correlated (and anti-correlated) graphs are generated by using the algorithm given in Figure 1. The procedure takes the required correlation $\alpha \in [-1, 1]$ as an argument and returns a weight vector of K weights where the first component is drawn from a uniform distribution, and all subsequent weights are either positively or negatively correlated with respect to the *first* component, and lie within the same range of values. Note that since the correlation exists between the first and each other component of the weight vector, a correlation of $|\alpha|$ exists between all pairs of components w^k, w^l $k, l \in 2..K$. The correlation between the components of a weight vector affects the shape of the associated Pareto front of the MST problem of the graph. A zero correlation gives a smooth, convex Pareto front with a fairly constantly varying gradient along its length. In contrast, a large positive correlation gives a convex Pareto front with more of a discontinuous change in the gradient. With a correlation approaching +1, the front becomes smaller until in the limit, it will only contain one optimal point. With a strong negative correlation, the front is convex but approaches a straight line or flat surface in objective space as the correlation approaches -1. Because of this there tend to be a large number of non-supported efficient solutions (those that do not lie on the convex hull of the Pareto front). This shape of Pareto front might make it difficult for methods based on the use of weighted sum aggregation of objectives, to find a good approximation to the Pareto front since they tend to find it difficult to discover non-supported solutions, and also rely on a changing gradient in the Pareto front to find a good range of points on it.

M-Correlated

The M-Correlated graph generator is based on a graph generator developed by us [6] for producing 'misleading' or M-graph problems for the standard (single-objective) d-MST problem, and combining this with the correlated graph generator described above. The M-Correlated graphs are designed to be particularly difficult to solve when a low maximum vertex degree constraint must be satisfied.

Algorithm: *Gen_correlated_wts*

$\alpha \in [-1, 1]$ is the correlation, provided by the user
β is the offset, calculated from α
γ is the variation, calculated from α and β
$\mathcal{U}(min, max)$ is a uniformly distributed random deviate $\in [min, max)$

if $(\alpha \geq 0)$
  $\beta \leftarrow 1/2(1 - \alpha)$
  $\gamma \leftarrow \beta$
else
  $\beta \leftarrow 1/2(1 + \alpha)$
  $\gamma \leftarrow \beta - \alpha$
foreach edge $(i, j) \in E$
  $w^1_{i,j} \leftarrow \mathcal{U}(0, 1)$
  foreach objective $k \in 2..K$
    $w^k_{i,j} \leftarrow \alpha w^1_{i,j} + \beta + \gamma \mathcal{U}(-1, 1)$

Figure 1: An algorithm for generating a graph with correlated weight vectors

In theory, the maximum vertex degree of a MST in a graph of random edge weights is $|V| - 1$. However, in practice, when reasonably large uniformly random weight graphs are generated, the maximum vertex degree of the graph's MST rarely exceeds four or five. Due to this fact, some researchers [1, 10] have developed methods for generating biased random graphs where the graphs' MSTs have a high maximum vertex degree. Knowles and Corne [6] further developed the graph generator of Boldon *et al.* to bias the edge weights in such a way as to mislead any algorithm that greedily chooses edges of low weight in an attempt to grow a low-weight spanning tree. The M-graph generator, as it is called, requires four parameters to be set:

$\|V\|$	the number of vertices in the graph;
f	the number of vertices with large degree;
ld	the lower bound on the degree of large-degree vertices; and
ud	the upper bound on the degree of large-degree vertices,

with the constraints that $f.ud < |V|$ and $ld < ud$. The generator has two main stages. In the first stage a spanning tree that will be the MST of the graph is formed. In the second stage other edges are added to form a graph of the required density. The first stage begins by forming f different 'star' graphs with the degree of each star centre vertex chosen uniformly at random in $[ld, ud]$. These disconnected components are then connected by adding $f - 1$ edges at random, to form a tree. The set of connected vertices, V_T, in the tree will have fewer than $|V|$ members, so that the tree is not spanning. To span the whole graph, additional vertices in $V \setminus V_T$, not in any of the stars,

will be added. However, first the edges in the (non-spanning) tree formed so far, are all assigned uniformly random weights in $[0, \chi)$. Next, all remaining unconnected vertices are connected to the tree by adding an edge between them and exactly one of the star centre vertices. The weight of these edges is assigned a uniformly random weight in $[\phi, \chi)$. Additional edges are then added between any pair of non-adjacent vertices, until the graph reaches the required density of connectedness. The weights assigned to these additional edges are uniformly random in $[\chi, \omega]$ for any pair of vertices where both are members of V_T, and uniformly random in $[\psi, \omega]$ for all other vertex pairs. If the weight parameters are set so that $0 < \phi < \chi \ll \psi < \omega$ then the resulting graph will be a misleading graph, that is the graph's structure will successfully mislead algorithms that favour choosing low-weight edges. This can be understood by first noticing that the edges incident to the vertices in $V \setminus V_T$ have two different ranges of values. Those edges whose other incident vertex is a star-centre have a low weight, whereas all others have a high weight. However, often a greedy-style algorithm will be forced to choose the higher weighted edge to connect these vertices because in earlier choices it will favour the edges that are incident to a star centre vertex (because these have the lowest range of weights in the graph) thereby causing the star centre vertex to reach its maximum allowed degree, and so preventing the connection of it to one of the vertices in $V \setminus V_T$. Because $\psi \gg \chi$ this will lead to a heavier graph, overall.

To make a multiobjective version of an M-graph, we use the M-graph procedure to set the weights of the first component of all the edge weight vectors. The others components of the edge weight vectors are then set using the correlation procedure outlined above. If a large positive correlation is used then the graph will be misleading in all of its components, and it will be difficult for a greedy approach to find a low-weight solution if the degree constraint is much lower than the parameter ud.

Concave

The Concave problem generator can only be used to make bi-objective problems at present. It works by setting the edge weights of three 'special' vertices (labelled 1, 2, and 3) in such a way that a large concave region in the Pareto front will result. If we restrict all edge weights to lie in [0,1], then the weights that can be used are the following: $\mathbf{W}_{0,1} = (\zeta, \zeta)$, $\mathbf{W}_{0,2} = (0, 1-\zeta)$, and $\mathbf{W}_{1,2} = (1-\zeta, 0)$. All other edges are $\mathbf{W}_{i,j} = (\mathcal{U}(\zeta, \eta), \mathcal{U}(\zeta, \eta))$ for $i, j > 3$ and $\mathbf{W}_{i,j} = (\mathcal{U}(1-\zeta, 1), \mathcal{U}(1-\zeta, 1))$ if i xor $j \leq 3$, with $i, j \in V$, ζ a small positive value of the order of $1/|V|$, $\zeta < \eta \ll 1 - \zeta$, and $\mathcal{U}(\min, \max)$ giving a uniformly random deviate in $[\min, \max]$.

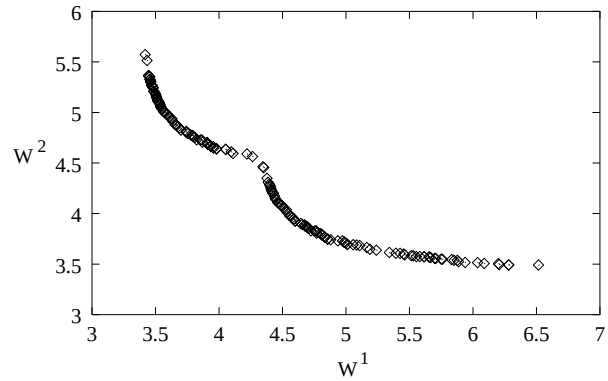

Figure 2: A plot showing true nondominated points on a 25 vertex concave problem

An example of (an approximation of) the Pareto front of a concave graph problem is given in Figure 2. The graph has 25 vertices and the values of the parameters for generating it were: $\zeta = 0.05$ and $\eta = 0.2$.

4 Algorithms

In a recently submitted paper [9], we studied the performance of two different encodings within a multi-objective evolutionary algorithm called AESSEA, on the mc-MST problem without degree-constraints. In that work, it was found that a mixed method based on a decoder-style encoding for initialization, and a direct encoding with specialized crossover and mutation operators was superior to the Prüfer encoding, in the same evolutionary algorithm. We also concluded in the paper that for un-constrained random-weight problems, a simple approach based on iterating Prim's algorithm [11] for different weighted-sum aggregations of the objectives in the given mc-MST, actually produces results that are far better than the Prüfer encoded evolutionary algorithm, and about the same as the mixed encoded evolutionary algorithm, but in far less time. However, we also noted that for various constrained problems and for situations where the Pareto front may contain many non-supported solutions, the Prim's algorithm approach would fail to perform well.

Our aim here is to test those ideas by applying AESSEA with a mixed encoding to a range of degree-constrained problems with different shaped Pareto fronts and once again we compare our results with our iterated mc$d$-Prim algorithm procedure, and also against a complete enumeration of the space on the smaller 10-vertex problems. The algorithms mc$d$-Prim and AESSEA are described briefly below, including the encodings and operators used in the latter.

Algorithm mc$d$-Prim

Prim's algorithm [11] is a well-known polynomial time constructive algorithm for solving the (single-objective, unconstrained) MST problem. It can be adapted to the d-MST by changing it so that at each step, in its construction of a tree, it checks for a degree-constraint violation before adding in the next edge. We call this constrained version of Prim's algorithm, d-Prim. Of course, since d-MST is NP-hard, d-Prim does not guarantee an optimal solution.

For tackling the mc$d$-MST, a multiobjective version of d-Prim is easily devised. By simply replacing the vector of edge weights in the graph by a weighted sum scalarization of them, optimization can be carried out in one 'direction' of the objective space. In mc$d$-Prim, this procedure is iterated for many different weightings of the objectives, giving a whole range of solutions approximating the Pareto front.

The scalarization of the objectives is achieved by replacing the vector weights defined on each edge in G by a scalar weight b formed by taking the inner product of a normalized scalarizing vector, $\boldsymbol{\lambda}$, and $\mathbf{w}$, $b = \boldsymbol{\lambda}.\mathbf{w}$. To obtain different weightings, we use every normalized scalarizing vector, $\boldsymbol{\lambda}$, with components equal to $l/s, l = 0..s$ where s is a parameter controlling the number of different vectors that will be generated. This gives $\binom{s+K-1}{K-1}$ different, evenly distributed, scalarizing vectors. For each different $\boldsymbol{\lambda}$ vector, the constrained version of Prim's algorithm (d-Prim) is applied.

For a large number of different scalarizing vectors (in our experiments we set $s = 1000$, giving 1001 different $\boldsymbol{\lambda}$ vectors), mc$d$-Prim may generate an approximation to S^*, the set of Pareto optimal spanning trees, that is satisfactory in many cases. However, with a low degree constraint the algorithm may not generate many (or any) optimal solutions on a single run. Therefore, it may be useful to run the algorithm several times with a different start vertex. In the experiments reported below, we always run mc$d$-Prim five times, each with a different start vertex.

AESSEA

The multiobjective evolutionary algorithm, AESSEA, is based closely on procedures already defined for the Pareto archived evolution strategy (PAES) [7], and is described more fully in [9]. AESSEA is a steady-state EA, that is, only one new solution is evaluated per 'generation'. It keeps a set of non-dominated solutions in an archive, and uses this set of solutions to estimate the quality of newly generated solutions. The algorithm is elitist in the sense that parents and offspring compete, but the overall selection pressure of the algorithm is not too strong since selection for mating is purely random, and offspring only replace one of their parents, rather than the weakest member of the population. Some testing of this algorithm and comparison with PESA [2] suggest that it is both an effective and computationally efficient, multiobjective EA.

RPM decoder encoding

The randomized primal method was put forward in [6] (where it is described in detail) as an encoding for solving the d-MST problem using any metaheuristic search method. It is a decoder type of representation, that is the chromosome encodes for choices that are made when a constructive algorithm builds a tree. The problem with this type of encoding is that it does not exhibit good locality, and it has super-linear growth in complexity for linear increase in the graph size $|V|$. However, it is good for initialization where it used only a small number of times.

Direct encoding and operators

The direct encoding and operators used in AESSEA are multiobjective versions of those put forward by Raidl in [13] for the d-MST problem. We describe how these operators were adapted for the mc$d$-MST problem in detail in [9]. In summary, the operators are adapted so that they bias the choice of edges towards those that are the minima on some weighted-sum single objective evaluation of the multiobjective tree weight. The weights in the weighted-sum are also encoded for by the chromosome and are subject to mutation and crossover. The method ensures that good solutions are found across the whole Pareto front.

5 Experimental method

Generated problems

Three graphs for each of the problem types: Correlated, Anti-Correlated, M-Correlated, and Concave were generated; one each at sizes of 10, 25, and 50 vertices, giving 12 graphs in all, from which 15 problems are created by setting degree constraints of 3 on all of the problems, and an additional, lighter degree constraint of 5 on the three M-correlated problems.

The correlations for the Anti-Correlated graphs, 10vAC, 25vAC, and 50vAC were set at -0.7. For the Correlated graphs, 10vC, 25vC, and 50vC, the correlation was set at 0.7. For the M-Correlated graphs the correlation was also set at 0.7, and the other parameters were $f = 1, 2, 5$, $ld = 6, 6, 7$, $ud = 8, 10, 9$, for the 10vM-C, 25vM-C, and 50vM-C, respectively. There is

no correlation between the edge weight components in the concave graphs, and the other parameters used for generating these graphs were $\zeta = 0.1, 0.05, 0.03$ and $\eta = 0.25, 0.2, 0.125$ for the 10vConc, 25vConc and 50vConc graphs respectively.

Algorithm parameters and experiments

The parameters used for AESSEA are given in Table 1. AESSEA is run 30 times independently on each of the 15 problems and the nondominated archive returned by it from each run is stored for statistical analysis, and comparison with the non-EA approaches. For each problem, mc*d*-Prim is run 5 times with a different start vertex, and the combined nondominated solution set is stored. On the ten-vertex problems we also use an enumerative procedure to give us the entire true Pareto front for comparison.

Parameter	AESSEA
population size, $\|P\|$	200
nondominated solutions archive size, *arcsize*	200
initialization method	RPM
mutation type	edge-mutation
crossover type	edge-crossover
total number of function evaluations, *num_evals*	20k/50k/50k
# of grid squares used for 'crowding' strategy [7]	1024

Table 1: Parameter settings for AESSEA. The three figures for number of evaluations relate to the three different problem sizes, 10, 25, and 50 vertices, respectively

Statistical Analysis of Data

When a multiobjective EA or other approximate method is run on a multiobjective problem, it returns a set of (mutually nondominated) solutions that approximate the true Pareto optimal set. Each of the solutions also has an image in the multi-dimensional objective space, consisting of an objective vector. The set of objective vectors approximate the true Pareto front. How well the discovered objective vectors (points) approximate the true Pareto front is usually more important than the proportion of the Pareto optimal solutions that have been found. But measuring the quality of an approximation to the Pareto front is not a straightforward problem because several dimensions of quality in an approximation can be identified. Coupled with this problem is the need for some statistical analysis of the performance of the (stochastic) approximate method over multiple runs, to provide useful summarising data for the expected performance of the algorithm.

Our methods rely on a technique developed by Fonseca and Fleming [4], and later implemented and extended by us [7]. The technique relies on the notion that the nondominated points from any approximation to a Pareto front define a surface (called the attainment surface), that divides up the objective space into a region that is dominated by the discovered nondominated points, and a region that is not dominated by them. Over multiple runs, an approximate algorithm will generate multiple different attainment surfaces. By sampling the distance of these surfaces from an origin at many different angles, one can obtain statistical information about the expected position of the attainment surface along each different angled direction (or sampling line). For example, one can calculate the median attainment surface, or the quartile attainment surfaces of an algorithm. One can also compare directly the *whole* distribution of positions of the attainment surfaces obtained from multiple runs of two or more different algorithms. This is particularly useful when comparing the performance of two or more EAs (e.g. see [7]). Here, where we just have biobjective problems, our first method is to simply plot the median attainment surface of AESSEA and compare it with the combined nondominated points found by mc*d*-Prim.

The second method is quantitative. For a single algorithm's sets of nondominated points, the method first computes (a sampling of) the attainment surfaces as above, and then calculates the (sampled) median and quartile surfaces. Now, to convert these surfaces into a simple figure of merit, the size of the dominated region of the surfaces can be calculated. For a biobjective minimization problem (as we have here) this is simply the area above and to the right of the attainment surface up to some bounding rectangle. How the bounding rectangle is best set is open to debate. Here we calculate the weight of the worst feasible solution (heaviest spanning tree) for each of the objectives in turn, giving us a point (z^1, z^2) that is used as the upper right corner of our bounding rectangle. (The points were calculated using Prim's algorithm set to maximize). In our results we report the absolute (unnormalized) size of the dominated region of the median surface, and (to get an idea of the variation over different independent runs), the interquartile dominated region size. One disadvantage of this approach is that concave regions of the Pareto front are under-represented in the statistics. This is seen in the results section where there is a seeming difference in the results shown by viewing the plots and by the numerical results, for the concave graph problems.

6 Results

Problem	Total size Enum	mc*d*-Prim	Median (Interqt) AESSEA
10vAC	21.3837	20.4108	**21.357 (0.0032)**
25vAC		**246.256**	245.175 (1.428)
50vAC		**1240.27**	1224.07 (7.6)
10vC	36.0955	35.3919	**35.7672 (0)**
25vC		363.233	**363.329 (0.109)**
50vC		1868.59	**1869.3 (3.52)**
10vM-C-*d*3	26.8362	23.6895	**26.8342 (0)**
25vM-C-*d*3		281.923	**343.892 (0.55)**
50vM-C-*d*3		1302.33	**1493.89 (3.17)**
10vM-C-*d*5	40.0389	35.3428	**40.0365 (0)**
25vM-C-*d*5		345.62	**384.511 (0)**
50vM-C-*d*5		1496.8	**1615.49 (4.56)**
10vConc	37.3255	37.0848	**37.2367 (0)**
25vConc		**334.694**	334.644 (0.354)
50vConc		**2122**	2118.39 (1.67)

Table 2: Size of the median and interquartile dominated regions for 30 runs of AESSEA, and the total combined size of the dominated region found using five runs of mc*d*-Prim. For the ten-vertex problems the true size of the dominated region is represented by the results of the Enumeration algorithm

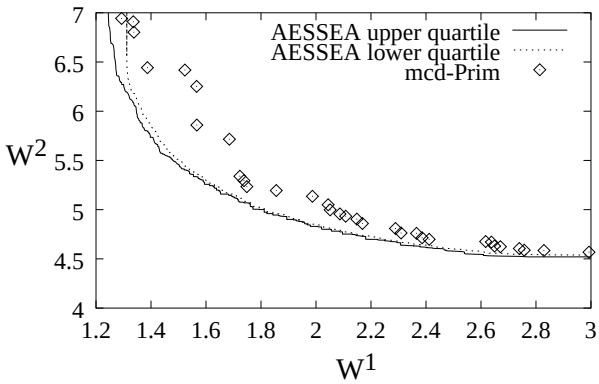

Figure 3: AESSEA's quartile attainment surfaces, and mc*d*-Prim's combined discovered vectors, on the 50 vertex correlated problem with degree constraint 3

From the results given in Table 2, we can see that mc*d*-Prim performs very well compared to AESSEA on the correlated and anti-correlated problems, and is considerably faster (but see Figure 3 for further help visualizing the Pareto fronts discovered). However, on the M-correlated graphs it clearly struggles. This is as expected because on these problems the degree constraint has a real effect on the difficulty of finding good solutions. We can see that AESSEA is clearly superior here; observe the size of the region discovered by it, compared to the enumeration method on the 10 vertex M-correlated problem for both d=3 and d=5. This is further shown in a plot given in Figure 4. Finally, on the (larger) concave problems, it appears that mc*d*-Prim does better than the AESSEA but in fact is unable to find any solutions in the concave region of the Pareto front. Its larger dominated region is due to it finding the very edge of the Pareto Front, which the EAs do not achieve on every run. This is illustrated in a plot showing the median attainment surface for AESSEA, and the points found from 5 runs of mc*d*-Prim is given in Figure 5.

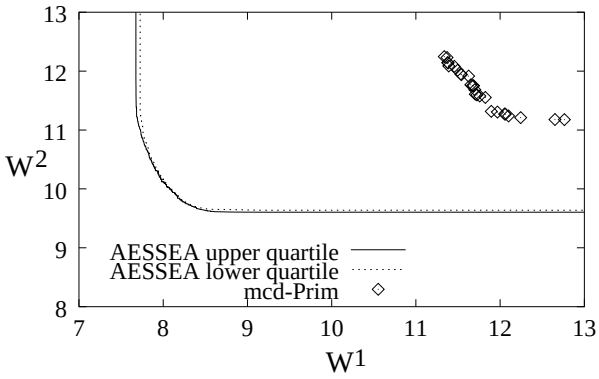

Figure 4: AESSEA's quartile attainment surfaces, and mc*d*-Prim's combined discovered vectors, on the 50 vertex M-correlated problem with degree constraint 3

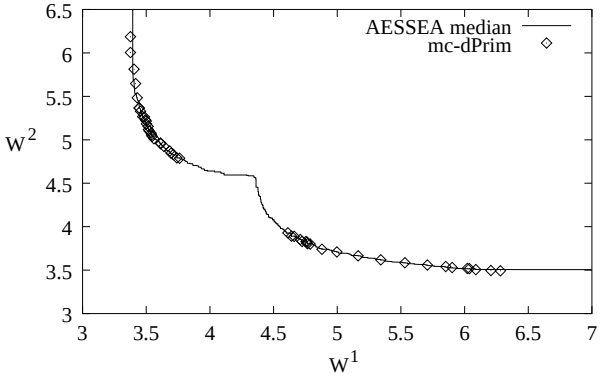

Figure 5: Nondominated points found from 5 runs of mc*d*-Prim, and the median attainment surface achieved by AESSEA. Note how AESSEA finds points in the concave region of the Pareto front

7 Conclusion

We have presented a number of graph generators that can produce a range of challenging graph types for the mc*d*-MST problem. The generators can be used to make problems for the OR community to test exact and heuristic approaches to the mc-MST, and for generating benchmark problems to form part of broader test problem suites for evaluating multiobjective EAs.

We have shown that on some problems it may not be necessary to use an evolutionary algorithm or other metaheuristic technique for tackling these problems, because a simple, iterative approach — mc*d*-Prim — can provide very good results in a fraction of the time. However, we have also demonstrated that for certain problems with constraints that are difficult to meet, an evolutionary algorithm, AESSEA, does obtain superior results. Furthermore, AESSEA is able to find points in the non-supported regions of the Pareto front, as was clearly demonstrated using the concave graph generator. In real problems of interest to the telecommunications industry, the number and variety of constraints that must be met will necessitate the use of evolutionary algorithms similar to that investigated here.

In our future work we will investigate the performance of a memetic algorithm, M-PAES [8], on these problems. Although M-PAES has been shown to perform well on other problems, it seems likely that in this application, a local-search element would be particularly useful. We base this conjecture on the observation made by Ehrgott and Klamroth [3] that from a sample of 50 random instances of a random weight bi-objective MST problem, all of them were ergodic with respect to a single exchange operator (although they prove this will not always be true). In light of this, we predict that further advances in tackling these difficult constrained mc-MST problems will come from techniques that incorporate a strong local search element, as used in M-PAES.

Acknowledgments

The authors would like to acknowledge the continuing sponsorship of British Telecommunications Plc., and thank the anonymous reviewers for useful comments.

References

[1] B. Boldon, N. Deo, and N. Kumar. Minimum-Weight Degree-Constrained Spanning Tree Problem: Heuristics and Implementation on an SIMD Parallel Machine. Technical Report CS-TR-95-02, Department of Computer Science, University of Central Florida, Orlando, FL 32816, 1995.

[2] D. W. Corne and J. D. Knowles. The Pareto-envelope based selection algorithm for multiobjective optimization. In *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature (PPSN VI)*, pages 839–848, Berlin, 2000. Springer-Verlag.

[3] M. Ehrgott and K. Klamroth. Connectedness of efficient solutions in multiple criteria combinatorial optimization. *European Journal of Operational Research*, 97:159–166, 1997.

[4] C. M. Fonseca and P. J. Fleming. On the Performance Assessment and Comparison of Stochastic Multiobjective Optimizers. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, Lecture Notes in Computer Science, pages 584–593, Berlin, Germany, September 1996. Springer-Verlag.

[5] H. W. Hamacher and G. Ruhe. On spanning tree problems with multiple objectives. *Annals of Operations Research*, 52:209–230, 1994.

[6] J. Knowles and D. Corne. A new evolutionary approach to the degree-constrained minimum spanning tree problem. *IEEE Transactions on Evolutionary Computation*, 4(2):125–134, July 2000.

[7] J. D. Knowles and D. W. Corne. Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[8] J. D. Knowles and D. W. Corne. M-PAES: A Memetic Algorithm for Multiobjective Optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC00)*, pages 325–332, Piscataway, NJ, 2000. IEEE Press.

[9] J. D. Knowles and D. W. Corne. A comparison of encodings and algorithms for multiobjective minimum spanning tree problems. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC'01)*. IEEE Press, 2001. (To appear).

[10] M. Krishnamoorthy, A. T. Ernst, and Y. M. Sharaiha. Comparison of algorithms for the degree constrained minimum spanning tree. Technical report, CSIRO Mathematical and Information Sciences, Clayton, Australia, 1999.

[11] R. Prim. Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, 36:1389–1401, 1957.

[12] H. Prüfer. Neuer beweis eines satzes über permutationen. *Arch. Math. Phys.*, 27:742–744, 1918.

[13] G. R. Raidl. An efficient evolutionary algorithm for the degree-constrained minimum spanning tree problem. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC00)*, pages 104–111, Piscataway, NJ, 2000. IEEE Press.

[14] R. M. Ramos, S. Alonso, J. Sicilia, and C. González. The problem of the optimal biobjective spanning tree. *European Journal of Operational Research*, 111:617–628, 1998.

[15] G. Zhou and M. Gen. Genetic Algorithm Approach on Multi-Criteria Minimum Spanning Tree Problem. *European Journal of Operational Research*, 114(1), April 1999.

[16] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.