Author/s:
Whittle, D;Brazil, M;Grossman, PA;Rubinstein, JH;Thomas, DA

Title:
Solving the prize-collecting Euclidean Steiner tree problem

Date:
2022-05-01

Citation:
Whittle, D., Brazil, M., Grossman, P. A., Rubinstein, J. H. & Thomas, D. A. (2022). Solving the prize-collecting Euclidean Steiner tree problem. International Transactions in Operational Research, 29 (3), pp.1479-1501. https://doi.org/10.1111/itor.12853.

Persistent Link:
https://hdl.handle.net/11343/276912

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. xx (2020) 1–21
DOI: xx.xxxx/itor.xxxxx

Solving the prize collecting Euclidean Steiner tree problem

David Whittle[a,*], Marcus Brazil[b], Peter A Grossman[a], J Hyam Rubinstein[c] and Doreen A Thomas[a]

[a]*Dept. Mechanical Engineering, The University of Melbourne, Parkville, Vic 3010, Australia*
[b]*Dept. Electrical & Electronic Engineering, The University of Melbourne*
[c]*Dept. Mathematics and Statistics, The University of Melbourne*
E-mail: david@whittle-dg.com [Whittle]; brazil@unimelb.edu.au [Brazil]; peterag@unimelb.edu.au [Grossman]; joachim@unimelb.edu.au [Rubinstein]; doreen.thomas@unimelb.edu.au [Thomas]

Received DD December 2019; received in revised form DD MMMM YYYY; accepted DD MMMM YYYY

Abstract

The prize collecting Euclidean Steiner tree (PCEST) problem is a generalisation of the well-known Euclidean Steiner tree (EST) problem. All points given in an EST problem instance are connected by a shortest possible network in a solution. A solution can include additional points called Steiner points. PCEST differs from EST in that given points each have assigned weights and a PCEST solution connects a subset of the given points in order to maximise the net value of the network (the sum of the selected point weights, less the length of the network).
We present an algorithmic framework for solving the PCEST problem. Included in the framework are efficient methods to determine subsets of points that must be in every solution, and subsets of points that cannot be in any solution. Also included are methods to generate and concatenate full Steiner trees.

Keywords: Prize collecting Steiner tree; Prize collecting Euclidean Steiner tree; Node weighted geometric Steiner tree

1. Introduction

An instance of a Prize collecting Euclidean Steiner Tree (PCEST) problem is a set of points in the plane, each with an associated weight. The aim is to construct the highest value connected network on some subset of these points. The value of the network is calculated as the sum of the weights of points included as vertices in the network minus the sum of the lengths of the edges in the network. The network can include additional vertices called *Steiner points* (each with zero weight) if their inclusion yields a shorter network. There are essentially two parts to the problem: choosing the set of points to be included in the network, and constructing an interconnection network of minimum length (necessarily a tree) on the chosen points.

*Author to whom all correspondence should be addressed (e-mail: david@whittle-dg.com).

This paper considers two variants of the PCEST problem: the rooted problem, which includes one mandatory point in the problem instance; and the unrooted problem, in which no points are initially set as mandatory. Both problems are formally defined in Section 3.

Our interest, in this paper, is in exact solutions to the PCEST problem. Such solutions are absent from the literature, however approximation schemes have been discussed in Remy and Steger (2009). The range of problems covered by their methods includes the PCEST problem. Remy and Steger give a polynomial time approximation scheme for the 2-dimensional version of this problem.

An important motivation for solving the PCEST problem comes from the mining industry. Mining is a major global industry, and a highly capital intensive one. With respect to new mines, and the expansion of operating mines, the mine designs should be optimised to maximise the economic value over the life of the mine. An improvement in a design's dollar value is an improvement in the value of an underlying asset of the mining venture. Most minerals are mined by methods falling into the categories of *open pit* (an open excavation from the surface) and *underground* (a network of tunnels and/or shafts giving access to the minerals underground). The application of optimisation to open pit mine design is well-developed and provides significant improvements in the value of mineral assets. The application of optimisation to underground mine design is less mature, representing an opportunity to increase the value of the mine in the planning stages. A decomposition of the underground mine plan optimisation problem can include a series of PCEST problems associated with the network of access tunnels on each level of the mine. More details of such decompositions are given in Whittle (2019).

The solution to the PCEST problem in this paper is built on a rich body of literature on the classic Euclidean Steiner tree (EST) problem and, to some extent, on a solution to the related prize collecting Steiner trees in graphs problem. These topics are reviewed in Section 2. Section 3 provides an overview of the PCEST problem and some of the properties of its solution. A significant contribution in this paper is in the development of efficient methods for point selection. In particular we define methods to determine which of the points given in a problem instance are definitely in all solutions (ruled in), or definitely not in any solutions (ruled out). This work is included in Sections 4 and 5. Section 6 describes the first known exact algorithm for solving the PCEST problem, an algorithm that promises to be much more efficient than a naïve exhaustive approach.

2. Mathematical Background

The problems studied in this paper are closely related to the following two classical problems. Here we are interested in versions of these problems where all lengths are measured using the Euclidean metric.

SPANNING TREE PROBLEM
Given: A finite set of points N in the plane
Find: A connected network $T = (V(T), E(T))$ embedded in the plane, such that $N = V(T)$, and such that the length of T, $L_T := \sum_{e \in E(T)} |e|$, is minimised.

A solution to the spanning tree problem is necessarily a tree, and is referred to as a *minimum spanning tree* (MST). In the Euclidean plane, MSTs can be constructed efficiently in $O(n \log n)$ time, where n is the cardinality of N (Preparata and Shamos (1988)).

STEINER TREE PROBLEM

Given: A finite set of points N in the plane

Find: A connected network $S = (V(S), E(S))$ embedded in the plane, such that $N \subseteq V(S)$, and such that the length of S, $L_S := \sum_{e \in E(S)} |e|$, is minimised.

We include below a selection of characteristics of MStTs that are important in this paper. Readers interested in a comprehensive coverage of MStTs should consult the book *Optimal Interconnection Trees in the Plane* (Brazil and Zachariasen (2015)). A solution to the Steiner tree problem is a tree, and is referred to as a *minimum Steiner tree* (MStT). An MStT may include vertices not in N. The points, N, in the problem instance are referred to as *terminals* and any additional vertices in S are referred to as *Steiner points*. We assume that the minimum number of Steiner points required to minimise the length of S are included. This implies that the degree of each Steiner point in S is at least 3. In fact, in the Euclidean plane it is known that all Steiner points are degree 3 and that the angle between each pair of edges incident to the Steiner point is $\frac{2\pi}{3}$. The angle between any pair of edges incident to a terminal is greater than or equal to $\frac{2\pi}{3}$. An MStT in which all terminals are leaves (that is, are degree 1) is referred to as a *full Steiner tree* (FST). If a sub-tree of an MStT S has the property that all Steiner points are degree 3 and the set of terminals of S in the sub-tree coincides with the set of leaves of the sub-tree, then the sub-tree is referred to as a *full component* of S. Any MStT can be uniquely decomposed into full components intersecting only at terminals.

An MStT is at most as long as an MST, but can be considerably shorter. The question of how much shorter has been studied, with the *Steiner ratio conjecture*[1] claiming that the length of an MStT can be no less than $\frac{\sqrt{3}}{2}$ times the length of an MST on the same set of terminals. More formally, we define the *Steiner ratio* ρ as

$$\rho := \inf_{N} \frac{|\text{MStT for} N|}{|\text{MST for} N|}$$

where the infimum is taken over all finite sets of points in the plane. The Steiner ratio conjecture claims that $\rho = \frac{\sqrt{3}}{2}$. This has been confirmed for cases with eight or fewer terminals (Kirszenblat (2014), De Wet (2009), Rubinstein and Thomas (1991)). For cases with more than eight terminals, the best known lower bound for the Steiner ratio is 0.82416874... (Chung and Graham (1985)).

Garey et al. (1977) have shown that the Euclidean Steiner tree problem is NP-complete. Despite this, exact algorithms have been implemented that are able to solve large instances of the problem. The most notable of these is GeoSteiner, a software package that can solve several variations of the Steiner tree problem, including the Euclidean Steiner tree problem for up to about 10,000 points (see, for example, Brazil and Zachariasen (2015) and Warme et al. (2000)). Geosteiner decomposes Steiner tree problems into two sub-problems: *Generation*, in which FSTs for all possible subsets of N are efficiently generated and *Concatenation* in which a subset of generated FSTs that interconnect N with minimum length is identified. We will discuss the proposed extension of GeoSteiner to the PCEST problem in Section 6.

Another well-studied variant of the Steiner Tree Problem relevant to formulating solutions for the PCEST problem is the prize collecting Steiner tree in graphs problem.

[1]Originally given in Gilbert and Pollak (1968). A brief history is given in Brazil and Zachariasen (2015) pp. 23-24

PRIZE COLLECTING STEINER TREE IN GRAPHS (PCSTG) PROBLEM

Given: An undirected, connected, vertex-weighted and edge-weighted graph $G = (V, E)$ where $c(e) \geq 0$ denotes the weight of edge $e \in E$ and $w(v) \geq 0$ denotes the weight of vertex $v \in V$.

Find: A connected subgraph $T = (V(T), E(T))$ of G such that $\sum_{v \in V(T)} w(v) - \sum_{e \in E(T)} c(e)$ is maximised.

The PCSTG problem is sometimes also treated as an equivalent minimisation problem where the objective function is the sum of the weights of all edges included in T and all vertices not included in T. The problem has received much more attention in the literature than its geometric counterpart. Early work focussed on approximation schemes (e.g. Johnson et al. (2000), Goemans and Williamson (1995)). Fast heuristics have application in some settings and a recent example is given in Sun et al. (2019). However, for the purposes of solving the concatenation sub-problem for PCEST, an exact solution to PCSTG is ideal. Ljubic et al. (2005) devised and demonstrated a branch and cut algorithm to solve large PCSTG problem instances to optimality. Gamrath et al. (2017) published details of a solver "SCIP-Jack". The solver is capable of solving a wide variety of Steiner tree problems in graphs, including PCSTG and the rooted variant thereof. Gamrath's approach is discussed in more detail in Section 6.2.

3. Prize collecting Euclidean Steiner tree problem: overview and preliminaries

For the remainder of this paper we assume that all geometric networks discussed are embedded in the Euclidean plane.

PRIZE COLLECTING EUCLIDEAN STEINER TREE (PCEST) PROBLEM

Given: A finite set of points $N = \{n_i\}$ in the plane, each with a corresponding weight $w_i \in \mathbb{R}$.

Find: A subset $\hat{N} \subseteq N$ and a connected network $P = (V(P), E(P))$ embedded in the plane, with $\hat{N} \subseteq V(P)$, such that $\text{val}(P) := \sum_{i:n_i \in \hat{N}} w_i - \sum_{e_j \in E(P)} |e_j|$ is maximised, and where there is no other such P^* with $\hat{N}^* \subset \hat{N}$ and $\text{val}(P^*) = \text{val}(P)$.

The last condition for the PCEST problem ensures a *parsimonious* network, that is, a maximum value network without unnecessary elements of N. Here, the points in N are referred to as *possible terminals* and the points in $\hat{N} \subseteq N$ as *terminals*. Note that the parsimonious condition does not guarantee that the solution will have the least possible number of terminals. This is because it is possible to have two maximal solutions with different numbers of terminals, where one set of terminals is not a subset of the other.

The PCEST problem is also referred to as the *un-rooted PCEST problem*. A variant that we will consider, the *rooted* PCEST problem, has a single mandatory terminal.

ROOTED PCEST PROBLEM

Given: A finite set of points $N = \{n_i\}$ in the plane each with a corresponding weight $w_i \in \mathbb{R}$; and a mandatory terminal $n_0 \in N$.

Find: A subset $\hat{N} \subseteq N$ with $n_0 \in \hat{N}$ and a connected network $P = (V(P), E(P))$ embedded in the plane, with $\hat{N} \subseteq V(P)$, such that $\text{val}(P) := \sum_{i:n_i \in \hat{N}} w_i - \sum_{e_j \in E(P)} |e_j|$ is maximised, and where there is no other such P^* with $n_0 \in \hat{N}^* \subset \hat{N}$ and $\text{val}(P^*) = \text{val}(P)$.

Fig. 1. The two trees are maximum PCESTs on $N = \{n_0, n_1, n_2, n_3\}$, but with different terminal sets.

As for the un-rooted PCEST problem, the last condition in the rooted PCEST problem ensures a parsimonious solution.

Definition 1. A *maximum (rooted) PCEST* is a network embedded in the plane that solves the (rooted) PCEST problem.

We will denote a maximum rooted or un-rooted PCEST by P. Observe that either version of the PCEST problem can be used to solve the other. A solution to the un-rooted PCEST problem can be found by solving the rooted PCEST problem card (N) times, where each time a different point is assigned as the mandatory terminal, and the resultant network with the highest value is chosen (a more efficient approach is given in Section 6.3). A solution to the rooted PCEST problem for N can be found by assigning a sufficiently high weight to the mandatory terminal and solving the un-rooted PCEST problem for N.

In both the un-rooted PCEST problem and the rooted PCEST problem, $V(P)$ may include a set of *Steiner points* $St(P)$ (vertices not in N), if their inclusion contributes to the maximisation of val(P). Accordingly, $V(P) = \hat{N} \cup St(P)$. It is assumed that Steiner points have no weight associated with them.

The relationship between the PCEST problem and the Steiner tree problem is given by the following lemma.

Lemma 1. *A maximum PCEST P on N is an MStT on $\hat{N}$.*

Proof. Suppose there exists a maximum PCEST P with terminals $\hat{N}$ that is not an MStT on $\hat{N}$. Then the length of P must be greater than the length of an MStT on $\hat{N}$. Since Steiner points in a maximum PCEST have no weight, it follows that all edges and Steiner points can be removed from P and replaced with the edges and Steiner points from an MStT, thereby increasing the value of P and leading to a contradiction. □

The Steiner tree problem can be thought of as a special case of a PCEST problem in which all weights of possible terminals are set *sufficiently high* to ensure that $\hat{N} = N$. For example: setting all weights to the diameter of the set N will ensure that $\hat{N} = N$. Thus, a PCEST problem is a generalisation of the Steiner tree problem.

It follows from this observation and Lemma 1 that many of the geometric properties of an MStT also hold for a maximum PCEST. For example, the fact that for a given set of terminals there may be more than one distinct MStT, means that for a given set of points N there may be more than one maximum PCEST. Furthermore, it is possible that two different maximum PCESTs for N have different terminal sets; an example is shown in Figure 1. It is worth noting, however, that for instances with randomly generated point locations and/or point weights, such non-uniqueness occurs with vanishingly small probability. For example, perturbing the position or weight of n_2 or n_3 in Figure 1 by any amount leads to there being only one solution to the PCEST problem. This suggests that cases of multiple solutions are likely to be rare for real-world data sets.

3.1. A naïve algorithm for solving the rooted PCEST problem

Some subset of points in N forms the set of terminals in the solution to the PCEST problem. Let $m =$ card $(N \setminus \{n_0\})$ and let $N^{'}$ denote the set of 2^m subsets of $N \setminus \{n_0\}$ (that is, the power set of $N \setminus \{n_0\}$). Let $N_k^{'}$ denote the k^{th} element of $N^{'}$. The naïve approach to finding a solution to the PCEST problem is:

1. Generate $N^{'}$.
2. For $k = \{1, 2, ..., 2^m\}$:
   (a) Find an MStT S for $N_k^{'} \cup \{n_0\}$
   (b) Calculate val(S).
3. Find all cases of $N_k^{'} \cup \{n_0\}$ with the highest associated val(S). Choose from these cases one with the fewest terminals. The MStT S for this terminal set corresponds to a maximum PCEST.

Finding an MStT in step 2(a) is an NP-Hard problem (Brazil et al. (2000)) that must be performed an exponential number of times. Even if an efficient exact program for solving the Steiner tree problem, such as GeoSteiner, is used, processing times quickly explode. For example, consider a PCEST problem with 30 points. The GeoSteiner documentation (Warme et al. (2017)) reports a 0.02 second computing time for a sample 20 terminal Euclidean Steiner tree. Suppose this is (optimistically) taken to be the mean time to conduct step 2(a) when $m = 30$. Then the total processing time for step 2 would be of the order of $0.02 \times 2^{30} \simeq 21$ million seconds, or around eight months.

4. Reducing the number of possible terminals

A key strategy for reducing the complexity of the naïve approach outlined above is to find efficient methods for decreasing the number of initial possible vertices whose membership in $V(P)$ is uncertain. Our approach involves partitioning N into three subsets: $N = N_{\mathrm{I}} \uplus N_{\mathrm{O}} \uplus N_{\mathrm{P}}$, defined as follows.

Definition 2 (Ruled in terminals). A point in N is said to be *ruled in as a terminal* ("ruled in" for short) if it has been shown to be in *every* maximum PCEST for the given instance. The set of all ruled in terminals in N is denoted by N_{I}.

Definition 3 (Ruled out terminal). A point in N is said to be *ruled out as a terminal* ("ruled out" for short) if it has been shown that it is not a vertex of *any* maximum PCEST for the given instance. The set of all ruled out terminals in N is denoted by N_{O}.

We next refine the definition of a *possible terminal*.

Definition 4 (Possible terminal). A *possible terminal* is a point in N that has neither been *ruled in* nor *ruled out*. The set of all possible terminals in N is denoted by N_{P}.

Note that these definitions depend on the state of knowledge of properties of the elements of N at any point in time, and hence these sets can change during the course of running an algorithm. When a point is ruled in or out, the number of possible terminals reduces by one, thereby halving the cardinality of the aforementioned power set and consequently halving the number of computations needed to solve the PCEST problem using the naïve approach. In this section we develop and test some methods for ruling points in or out.

Throughout the remainder of this paper, if p and q are points in the plane, let pq represent the line segment connecting the two points, and let $|pq|$ be the Euclidean distance between them.

4.1. Ruling in

Our focus here is on the rooted PCEST problem. Some of the lemmas apply to both maximum rooted PCESTs and maximum un-rooted PCESTs, in which case the term maximum PCEST is used.

First note, by the definition of the rooted PCEST problem, that the mandatory terminal is ruled in. The following lemma shows that points sufficiently close to ruled in terminals can also be ruled in.

Lemma 2. *Let n_i be a ruled in terminal and let n_j be a possible terminal with weight w_j. If $w_j > |n_j n_i|$ then n_j can be ruled in.*

Proof. Suppose to the contrary n_j is not a member of some maximum PCEST. Then it can be added to the network with an edge (n_j, n_i), increasing the network's value and giving a contradiction. □

Define the *diameter* of N to be $\text{diam}(N) := \max\{|n_i n_j| : n_i, n_j \in N\}$. Then, we have the following useful corollary to Lemma 2.

Corollary 3. *Let n_i be a possible terminal with weight w_i. If $w_i > \text{diam}(N)$ then n_i can be ruled in.*

For any set of points $\tilde{N}$, let $S(\tilde{N})$ denote an MStT with terminal set $\tilde{N}$. A stronger version of Lemma 2 applies to multiple possible terminals in the vicinity of a ruled in terminal.

Lemma 4. *Consider a ruled in terminal n_i and possible terminals n_j and n_k with weights w_j and w_k respectively. If $\min\{w_j, w_k\} - |n_j n_k| > 0$ and if $w_j + w_k > |S(\{n_i, n_j, n_k\})|$ then both n_j and n_k can be ruled in.*

Proof. Suppose to the contrary that the two conditions of the lemma are met, and that n_j and n_k are not both in a maximum PCEST, P. The first condition, along with Lemma 2, implies that neither point belongs to P. Then clearly the points can be appended to P via an MStT on $\{n_i, n_j, n_k\}$, such that the sum of the edge lengths in the MStT is less than the sum of the point weights, contradicting that P was a maximum PCEST. □

Lemma 4 provides a method to rule in pairs of terminals when an attempt to rule them in individually using Lemma 2 would fail. Consider, for example, a case in which n_i is a ruled in terminal and n_j and n_k are possible terminals, with the properties $|n_i n_j| = |n_i n_k| = 13$, $|n_j n_k| = 10$ and $w_j = w_k = 11$. It is an easy exercise to show that the MStT on $\{n_i, n_j, n_k\}$ has length $12 + 5\sqrt{3} \approx 20.66$, so the points can be ruled in by Lemma 4.

Remark 1. [Merging possible terminals] A consequence of Lemma 4 is that when ruling in, a *merging* step can be effected if two possible terminals n_j and n_k have distance between them less than the smaller of the two point weights ($\min\{w_j, w_k\} - |n_j n_k| > 0$). The merging assigns to a single point a combination of the characteristics of the original two points, which are then discarded. Denote the new point by n_{j*} with point weight $w_{j*} = w_j + w_k$. The distance between n_{j*} and any n_i (for $i : n_i \in N_\text{I} \cup N_\text{P} \backslash \{n_j, n_k\}$) is given by $|n_{j*} n_i| = |S(\{n_i, n_j, n_k\})|$. This step is not particularly computationally

Fig. 2. Case illustrating difficulty of ruling out.

burdensome as an MStT can be computed in constant time for a small number of terminals. If point n_{j*} is subsequently ruled in by Lemma 2, then it can be interpreted as the original points n_j and n_k being both ruled in. This process can be iterated, although it does involve computing MStTs on increasingly large sets of points; the iteration effectively allows multiple points to be merged into clusters.

It is now straightforward to design a ruling in algorithm to give effect to Lemmas 2 and 4. Such an algorithm first computes all possible mergings of points in N_{P}, then rules in as many points as possible through the repeated application of Lemma 2.

4.2. Ruling out

In general, ruling out points as terminals in a maximum PCEST is considerably harder than ruling them in. Part of the reason for this is illustrated in Figure 2. In the figure there is a mandatory terminal n_0 and two possible terminals n_1 and n_2 with weights 200 and 3 respectively. The distances between points are as indicated in the figure. Clearly n_1 can be ruled in by Lemma 2. Point n_2 should also be part of a maximum PCEST, together with n_1, since its inclusion increases the value of the network by 1. This is despite the fact that the weight of n_2 is small compared to its distance from any other possible terminal. However, the addition of other points in the problem instance could change this. For example, additional points could provide an alternative and preferable path between n_0 and n_1 that does not involve n_2. This suggests that it is not easy to use local conditions to determine when a point can be ruled out.

Despite this, we have been able to develop a number of tests for ruling out points as terminals in a maximum PCEST. The methods are presented in their intended order of implementation.

Let n_q denote some point in N_{P} with weight w_q that is to be subject to a ruling out test. Note that if $w_q \leq 0$, then n_q can immediately be ruled out. Hence, in the following we assume that $w_q > 0$.

Also note that if the merging and ruling in algorithms described above have been applied, a process that we refer to as *pre-screening*, then there may be one or more clusters of possible terminals, which were not ruled in. For non-trivial clusters (clusters with two or more elements), none of the elements can be successfully ruled out by the ruling out methods described in this paper. This is because each of the methods rely on $w_q < d$, where d is the distance between the point n_q being tested and its nearest neighbour in $N_{\mathrm{I}} \cup N_{\mathrm{P}}$. Furthermore, the methods cannot be applied to the new merged points, as the methods are reliant on the Euclidean geometry of the embedding plane, which is not preserved under the merging process. Accordingly, only possible terminals that are *unmerged* (that is, do not belong to a non-trivial cluster) need be the subject of ruling out tests.

Here we establish conditions that allow ruling out certain points that are on the boundary of the convex hull of $N_{\mathrm{I}} \cup N_{\mathrm{P}}$. Recall that in an MStT the angle between any two edges meeting at a vertex is at least $\frac{2\pi}{3}$. It follows that if the internal angle of the convex hull at a boundary vertex is less than $\frac{2\pi}{3}$, then that vertex cannot be degree 2 or 3 in any maximum PCEST.

Let $n_q \in N_{\mathrm{P}}$ denote a boundary vertex of the convex hull of $N_{\mathrm{I}} \cup N_{\mathrm{P}}$, such that the internal angle of the convex hull at n_q is less than $\frac{2\pi}{3}$. This implies that n_q has degree 1 if it is in a PCEST. Let Γ denote a

Fig. 3. Artefacts relevant to Theorem 5.

disk centred on n_q with radius w_q. As in Gilbert and Pollak (1968), we define a *wedge* to be a translation of a convex cone in the coordinate plane. For any point z on the boundary of Γ, let $W = W(n_q, z)$ denote an open wedge with vertex z and angle $\frac{2\pi}{3}$, such that each boundary ray of W makes an angle of $\frac{2\pi}{3}$ with $\overline{n_q z}$. This is illustrated in Figure 3.

Theorem 5. *If there exists a point z on the boundary of Γ such that W contains $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$ then n_q can be ruled out as a terminal in a maximum PCEST.*

Proof. Suppose that W contains $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$ and that contrary to the theorem, n_q is a member of a maximum PCEST P. Consider first the case that n_q is adjacent to a terminal. Then its incident edge must have length greater than w_q since all terminals other than n_q are inside W. The removal of n_q and its incident edge will yield a network with increased value, contradicting the maximality of P. Now consider the case that n_q is adjacent to a Steiner point s. Then $|n_q s| \leq w_q$ otherwise P is not maximal, implying s is inside Γ and adjacent to two other vertices (either Steiner points or terminals). Since all edges at s meet with angle $\frac{2\pi}{3}$ this implies that at least one of these adjacent vertices, say s', is further from the boundary of the convex hull of $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$ than s. We can repeat this argument with s', to conclude that there is a path from s in the network that continually moves away from the boundary of the convex hull of $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$ and hence terminates at some terminal outside the convex hull, which is a contradiction.

□

Some details of an efficient ruling out test based on Theorem 5 follow. The vertices of a convex hull can be determined by several methods including Graham's scan (Graham (1972)) with time complexity $O(n \log n)$, where n is the cardinality of $N_{\mathrm{I}} \cup N_{\mathrm{P}}$. As before, n_q is a boundary vertex of the convex hull of $N_{\mathrm{I}} \cup N_{\mathrm{P}}$, such that the internal angle of the convex hull at n_q is less than $\frac{2\pi}{3}$. Let L_1 and L_2 denote lines overlapping the two rays of W and observe that they intersect at z. We will determine a *suitable* L_1, which implies a z and an L_2. In this context *suitable* means that if n_q can be ruled out for this implied z, it can be ruled out for every z. Let v denote an element of $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$. Observe that given the coordinates of n_q, its corresponding weight w_q and the coordinates of a v it is possible to determine the coordinates of a corresponding z. For some v there exists a line $\overleftrightarrow{vz}$ such that n_q is not on the same side of the line as any elements of $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$. Such a $\overleftrightarrow{vz}$ is a suitable L_1. The range of candidates for v for a suitable L_1 is small and an efficient search order can be established commencing with a convex hull vertex adjacent to n_q. Accordingly, a suitable L_1 can be found in low order polynomial time. Given L_1 (and z) it is straightforward to find the corresponding L_2. If n_q is not on the same side of L_2 as any element of $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$, then the conditions of Theorem 5 are satisfied and n_q can be ruled out.

If any possible terminals are ruled out, the procedure can be iterated until none of the elements of N_{P} on the boundary of the convex hull satisfy the condition of the theorem.

5. Further ruling out methods based on local connections

Let n_q be an element of $N_{\rm P}$ with weight w_q; we want to determine whether n_q can be ruled out as a terminal in a maximum PCEST $P = (V(P), E(P))$, based on what is known about n_q's possible local connections and some easily determinable facts about all the ruled in and possible terminals. In this section we develop two *replacement arguments* that allow us to make such a determination.

5.1. Preliminaries and definitions

In the definitions below some networks embedded in the plane are described in a logical sequence. Each network is either an MStT (denoted with S), a spanning tree (T) or a forest (F). The sequence of dependencies is given by superscripts, so S^1 is followed by S^2, which is followed by T^3 etc. In order to decide whether some point $n_q \in N_{\rm P}$ can be ruled out as a terminal in a maximum PCEST, without calculating multiple MStTs, we investigate the possible local structure of the network in the vicinity of n_q.

Definition 5 (Steiner tree S^1). A tree $S^1 = (V(S^1), E(S^1))$ is an MStT where $V(S^1) = \hat{N}' \cup St(S^1)$ for some $\hat{N}' \subseteq N$.

It follows that for some $\hat{N}' \subseteq N$, S^1 is a maximum PCEST on N.

We wish to determine whether $n_q \in \hat{N}'$ is consistent with S^1 being a maximum PCEST. For example: if n_q and all parts of edges within a small neighbourhood of n_q can be replaced by a new set of edges which maintain connectivity but increase the value of the tree, then it would follow that n_q can be ruled out as a terminal in a maximum PCEST on N. The next definition gives a suitable neighbourhood.

Definition 6 (Disk $\mathbb{D}$ and $\delta(\mathbb{D})$). We denote by $\mathbb{D} = \mathbb{D}(n_q, r)$ a disk centred on n_q with a radius r, where $r + \mu$ is equal to the distance between n_q and its nearest neighbour in $N_{\rm I} \cup N_{\rm P}$ for some $\mu << r$. $\delta(\mathbb{D})$ denotes the boundary of $\mathbb{D}$.

Note that the only terminal inside $\mathbb{D}$ is n_q. If $n_q \in \hat{N}^1$, then there must be at least one edge passing through the boundary of $\mathbb{D}$ in S^1. There can be Steiner points inside $\mathbb{D}$. An example is shown in Figure 4. In this example, there are four edges passing through the boundary of $\mathbb{D}$ including one connecting n_q to its nearest neighbour through a Steiner point.

Fig. 4. Example of an MStT S^1 showing the disk $\mathbb{D}$. The terminals are indicated by open blue diamonds, and the Steiner points by solid black circles. The open blue circles indicate the Rubin points required for the construction of S^2.

Definition 7 (Rubin points). Let $R^* := S^1 \cap \delta(\mathbb{D})$ (where S^1 is treated as an embedded network in the plane). Note that R^* is a finite set of points. By adjusting the value of μ in the definition of $\mathbb{D}$ if necessary, we may assume that each element of R^* lies on the interior of an edge of S^1. Let $R \subseteq R^*$ be the set of all points of R^* for which each corresponding edge of S^1 has an end point inside $\mathbb{D}$. We refer to the elements of R as *Rubin points*[2].

[2]These were named by the corresponding author after Professor J H Rubinstein, who suggested this approach.

Definition 8 (Steiner tree S^2). The Steiner tree S^2 is obtained from S^1 by inserting Rubin points R into the tree as new terminals. The Rubin points are assumed to have weight 0, so transforming S^1 to S^2 does not change the length or value of the tree.

Rubin points and an example of the tree S^2 are illustrated in the example in Figure 4. Note that, by definition, every Rubin point has degree 2 in S^2.

Definition 9 (Steiner tree $\mathbb{S}$). The Steiner tree $\mathbb{S}$ is defined to be the connected component of $S^2 \cap \mathbb{D}$ containing n_q.

It follows from the minimality of S^2 that $\mathbb{S}$ is an MStT on $R \cup \{n_q\}$ with Steiner points exactly corresponding to the Steiner points of S^2 in $\mathbb{S}$. By the construction of $\mathbb{S}$, all Rubin points in $\mathbb{S}$ have degree 1 in $\mathbb{S}$, and hence $\mathbb{S}$ is a union of FSTs meeting only at n_q. It follows that the number of FSTs in $\mathbb{S}$ is equal to the degree of n_q.

Definition 10 (Steiner tree $\mathbb{U}$). We define $\mathbb{U}$ to be an MStT on all the Rubin points in R that are terminals of $\mathbb{S}$.

Remark 2. $L_{\mathbb{U}} \leq L_{\mathbb{S}}$.

As a basic property of an MStT, any vertex of such a tree must have degree 1, 2 or 3. We now develop two replacement methods for ruling out a possible terminal n_q based on the degree of n_q in $\mathbb{S}$. If the degree of n_q is 1, then Replacement Argument A applies; if n_q has degree 2 or 3 then we require Replacement Argument B. If the degree of n_q in $\mathbb{S}$ cannot be predetermined for a given n_q, it can only be ruled out if the conditions for both replacement arguments are met.

5.2. Replacement argument A

Suppose the possible terminal n_q (with weight w_q) has degree 1 in $\mathbb{S}$. Assume that the disk $\mathbb{D}$ has radius r. Then we define the following constant (over all possible sets of Rubin points):

$$\phi_1 := \frac{\inf(L_{\mathbb{S}} - L_{\mathbb{U}})}{r}.$$

In Whittle (2019) it is proven that $\phi_1 = 2 - \sqrt{3}$. The proof is long and technical, and so is not repeated here. What is important here for Replacement Argument A is that if n_q has degree 1 then ϕ_1 has a known strictly positive value. This is not true if n_q has degree 2 or 3, which is why a separate argument is required for those cases.

Theorem 6 (Replacement argument A)**.** *If n_q has degree 1 in $\mathbb{S}$ and if $w_q \leq \phi_1 r$ where r is the distance between n_q and its nearest neighbour in $N_{\mathrm{I}} \cup N_{\mathrm{P}}$, then n_q can be ruled out as a terminal in a maximum PCEST.*

Proof. If $w_q \leq \phi_1 r = \inf(L_{\mathbb{S}} - L_{\mathbb{U}})$ then $w_q - L_{\mathbb{S}} \leq -L_{\mathbb{U}}$. This implies that replacing $\mathbb{S}$ with $\mathbb{U}$ does not decrease the value of the tree but clearly maintains connectivity, hence n_q can be ruled out. □

Suppose it is possible to determine that a given possible terminal must have degree 1 if it is a member of a maximum PCEST. Then replacement argument A can be used as a ruling out test. One way to make such a determination is by constructing the convex hull on all ruled in and possible terminals.

Recall that if the internal angle of the convex hull at a boundary vertex is less than $\frac{2\pi}{3}$, then that boundary vertex cannot be degree 2 or 3 in any maximum PCEST. Any such vertex that is also a possible terminal is a candidate for a ruling out test with replacement argument A. This approach can potentially be improved by using a tighter, not necessarily convex region, known as a *Steiner hull*, the construction of which is described in Winter (2002).

5.3. Replacement argument B

This replacement argument applies when n_q has degree 2 or 3 in $\mathbb{S}$, if it is in a maximum PCEST. Before presenting the main theorem, we require some additional definitions.

Definition 11 (chord). For the purposes of this paper, we restrict the term *chord* to exclusively denote any straight line segment between Rubin points that are adjacent on the boundary of $\mathbb{D}$.

Definition 12 (forest $\mathbb{F}$). We define $\mathbb{F}$ to be a forest on R with edges corresponding to all chords on R other than the two longest chords (where ties in length are broken arbitrarily).

Definition 13 (forest F^3). We define F^3 to be the forest that results from replacing $\mathbb{S}$ with $\mathbb{F}$ in S^2.

Observe that the forests $\mathbb{F}$ and F^3 each have exactly two distinct connected components, and that $\mathbb{F}$ is a minimum forest (with no Steiner points) on R.

Definition 14 (edge e and edge-length bound $\mathcal{U}e$). Let e denote a shortest length line segment between terminals of F^3 that connects the two connected components of a forest F^3. As before, we denote the length of e as $|e|$. For a given (N, n_q), $\mathcal{U}e$ is an upper bound for $|e|$.

A method to efficiently calculate $\mathcal{U}e$ for a given (N, n_q) is given in Section 5.3.1.

Definition 15 (tree T^4). We define T^4 to be a tree resulting from the addition of e to the edges of F^3.

Observe that T^4 is a tree that spans all the terminals that S^2 spans other than n_q.

Replacement argument B relies on the observation that $L_{\mathbb{S}} - L_{\mathbb{F}}$ is strictly positive, and the claim that a good lower bound for this difference can be usefully exploited. Let

$$\phi_2 := \frac{\inf(L_{\mathbb{S}} - L_{\mathbb{F}})}{r},$$

where, as before, we assume that r is the radius of the disk $\mathbb{D}$.

In Whittle (2019) it is conjectured that $\phi_2 = \sqrt{2}(\sqrt{3} - 1) \approx 1.035$, and a proof of this is being prepared for a forthcoming paper Whittle et al. (2020).

Theorem 7 (Replacement argument B)**.** *If n_q has degree 2 or 3 in $\mathbb{S}$ and if $w_q \leq \phi_2 r - \mathcal{U}e$ where r is the distance between n_q and its nearest neighbour in $N_{\mathrm{I}} \cup N_{\mathrm{P}}$, then n_q can be ruled out as a terminal in a maximum PCEST.*

Proof. Assume, contrary to the statement of the theorem, that $w_q \leq \phi_2 r - \mathcal{U}e$ holds and n_q is a member of a maximum PCEST. Such a maximum PCEST meets the definition of a tree S^1 (Definition 5). Now consider a set of modifications that changes S^1 into a tree T^4 (Definition 15) and the corresponding changes in tree length:

1. Insert Rubin points into S^1 resulting in a tree S^2 (Definition 8). Observe that S^2 still interconnects the same terminals as S^1 and:

$$L_{S^2} = L_{S^1}. \tag{1}$$

2. Replace $\mathbb{S}$ with $\mathbb{F}$ in S^2 resulting in a forest F^3 (Definition 13). Observe that F^3 has two components and:

$$L_{F^3} = L_{S^2} - (L_{\mathbb{S}} - L_{\mathbb{F}}). \tag{2}$$

3. Add an edge e (of shortest possible length) to F^3, connecting its two components, resulting in a tree T^4. Observe that:

$$L_{T^4} \leq L_{F^3} + \mathcal{U}e. \tag{3}$$

Now substitute for L_{S^2} and L_{F^3} from Equations 1 and 2 into Inequality 3:

$$L_{T^4} \leq L_{S^1} - (L_{\mathbb{S}} - L_{\mathbb{F}}) + \mathcal{U}e. \tag{4}$$

Observe that T^4 spans the same terminals as S^1, other than n_q. If S^1 is a maximum PCEST, then:

$$w_q > L_{S^1} - L_{T^4}. \tag{5}$$

If Equation 5 did not hold then this would imply that T^4 is a network with greater PCEST value than S^1 contradicting that S^1 is a maximum PCEST. Now substituting for L_{T^4} from Inequality 4 into Inequality 5 gives:

$$w_q > (L_{\mathbb{S}} - L_{\mathbb{F}}) - \mathcal{U}e \geq \phi_2 r - \mathcal{U}e,$$

contradicting the initial assumption. □

5.3.1. A method to calculate an upper bound $\mathcal{U}e$ for a given N and n_q

In order to use Theorem 7 as part of a practical algorithm, a method for calculating the upper bound $\mathcal{U}e$ is required. The difficulty is that for any given N the terminals of F^3 are not initially known. The only thing known is that the terminal set is of the form $N_{\mathrm{I}} \cup N_{\mathrm{P}}^{'}$ where $N_{\mathrm{P}}^{'} \subseteq N_{\mathrm{P}} \setminus \{n_q\}$.

The objective is to find a subset of $N_{\mathrm{P}} \setminus \{n_q\}$ such that the length of the longest edge in an MST on N_{I} and this subset is maximised. Clearly this length can be taken as $\mathcal{U}e$.

A naïve approach to computing $\mathcal{U}e$ is to generate MSTs for every $N_{\mathrm{I}} \cup N_{\mathrm{P}}^{'}$ where $N_{\mathrm{P}}^{'} \subset N_{\mathrm{P}} \setminus \{n_q\}$ and to choose the longest edge among all of these. However, this requires the computation of $2^{(n-1)}$ MSTs, where n is the number of possible terminals. Furthermore, the $\mathcal{U}e$ calculated with this approach may be unnecessarily large (because of improvements that can be made by taking clustering into account) leading to inefficient ruling out.

Improvements to the naïve approach flow from the following observations:

- Consider an MST T_1 with longest edge e_1:

— An MST T_2 with longest edge e_2 can be obtained by removing a degree 1 vertex and its incident edge from T_1. In this case $e_2 \leq e_1$.
— An MST T_2 with longest edge length e_2 can be obtained by removing a vertex with degree 2 or higher from T_1, then generating a new MST. In this case, by the greedy properties of minimum spanning trees, $e_2 \geq e_1$.

Now consider an MST on $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$ and suppose some of the possible terminals are degree 1 in the MST. When considering MSTs on $N_{\mathrm{I}} \cup N_{\mathrm{P}}^{'}$ with the objective of finding an upper bound $\mathcal{U}e$, only these degree 1 possible terminals need to be considered for inclusion in $N_{\mathrm{P}}^{'}$. Furthermore, if there are no degree 1 possible terminals in the aforementioned MST, then it suffices to consider just an MST on ruled in terminals for the determination of $\mathcal{U}e$.

- Recall from Remark 1 in Section 4 that points can be clustered in such a way that if one member of the cluster is ruled in then all members of the cluster are ruled in. This implies that if any members of a cluster of possible terminals $N_{\mathrm{P}} \setminus \{n_q\}$ are ruled out, they must all be ruled out. Let C_{P} denote the set of clusters of possible terminals not including n_q. The implication of this observations in terms of improving the naïve approach to finding an upper bound $\mathcal{U}e$ is: Rather than generating an MST for every $N_{\mathrm{P}}^{'} \cup N_{\mathrm{I}}$, it suffices to generate an MST for every $C_{\mathrm{P}}^{'} \cup N_{\mathrm{I}}$, where $C_{\mathrm{P}}^{'} \subset C_{\mathrm{P}}$. Combining this with the first observation: it suffices to consider for inclusion in $C_{\mathrm{P}}^{'}$ only clusters of possible terminals that include at least one degree 1 vertex in the MST on $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$.

5.4. Joint application of replacement arguments A and B

If, for a given possible terminal n_q, it cannot be predetermined if it would be degree 1, 2 or 3 in a maximum PCEST, then it can be tested for ruling out by applying both replacement arguments A and B. Refer to Figure 5 for an example of such a combined application of replacement arguments. Observe n_q's nearest neighbour is n_1 and all distances and the node weight w_q are normalised, so that $|n_q n_1| = 1$. This example has been constructed in such a way that the ruling out methods other than replacement arguments A and B (Section 4.2) are all incapable of ruling out n_q.

Fig. 5. Illustration of the application of replacement arguments A and B.

We first apply replacement argument A, with the assumption that $\phi_1 = 2 - \sqrt{3}$. By Theorem 6 it is easy to see that n_q cannot be degree 1 in a maximum PCEST since $w_q = 0.26 \leq 2 - \sqrt{3} \approx 0.268$.

For replacement argument B, we assume that $\phi_2 = \sqrt{2}(\sqrt{3} - 1) \approx 1.035$. Attention turns first to the calculation of $\mathcal{U}e$. In Figure 5 the points shown in solid green are ruled in terminals. The points shown with blue outlines on the right are possible terminals in a cluster C_1. The points shown on the left with blue outlines are possible terminals that are not in a cluster. Consider first an MST on $N_{\mathrm{I}} \cup N_{\mathrm{P}} \setminus \{n_q\}$. Observe that for the possible terminals on the left, three have degree > 1 and can be ignored. It follows that $\mathcal{U}e$ is the length of the longest edge in MSTs on the N_{I} in union with each element of the power set of $\{C_1, \{n_2\}, \{n_3\}, \{n_4\}\}$. In this case it is easy to see that the longest edge in an MST is obtained for MSTs on $\{N_{\mathrm{I}} \cup \{n_3\}\}$ and also on $\{N_{\mathrm{I}} \cup C_1 \cup \{n_3\}\}$. It follows that $\mathcal{U}e = 0.42$.

We now apply Theorem 7. Using the values $r := |n_q n_1| = 1$, and $\mathcal{U}e = 0.42$ we obtain:

$$\phi_2 r - \mathcal{U}e > 1.035 - 0.42 = 0.615 > w_q = 0.26. \tag{6}$$

Hence, it follows that n_q can be ruled out as a degree 2 or 3 terminal in a maximum PCEST.

6. An exact algorithm for solving the PCEST problem

This section describes an algorithmic method to exactly solve the PCEST problem for a given instance. The method is a modification to an existing framework implemented for the Euclidean Steiner tree problem in the software package GeoSteiner (Warme et al. (2017)).

In solving the Steiner tree problem, GeoSteiner divides the task into two phases: a Generation phase and a Concatenation phase. In this section we consider how each of the phases in turn can be adapted to the PCEST problem.

6.1. The Generation phase for the PCEST problem

In the Steiner tree problem, the Generation phase of GeoSteiner efficiently enumerates a set of FSTs, such that it is guaranteed that at least one MStT can be constructed from a union of some of the FSTs in the set. Similarly, for the PCEST problem the Generation phase involves enumerating a set of FSTs such that it is guaranteed that at least one maximum PCEST can be constructed from a union of some of the FSTs. In each case, efficient enumeration relies on the use of various pruning rules that can be used to discard FSTs or classes of FSTs that are not feasible as candidate components of an MStT. The pruning rules used in GeoSteiner Generation in solving the Steiner tree problem rely on having a fixed set of terminals. A particular challenge in the PCEST problem is that the set of terminals is not initially known; instead, after running the ruling in and ruling out tests, there is a set of ruled in terminals (N_{I}) and a set of possible terminals (N_{P}) that must be considered.

A naïve approach to generating a sufficient set of FSTs for the PCEST problem is to run the original GeoSteiner Generation for $N_{\mathrm{I}} \cup \tilde{N}$ for all $\tilde{N} \subseteq N_{\mathrm{P}}$ and then to eliminate duplicate FSTs. However, this would be computationally burdensome since there are 2^n possibilities for $\tilde{N}$ where n is the cardinality of N_{P}.

A more efficient approach, is to construct all feasible FSTs with terminals in $N_{\mathrm{I}} \cup N_{\mathrm{P}}$ in a bottom-up manner while applying modified pruning rules which take into account the fact that some possible terminals may not be required in the final solution. The bottom-up construction means that a single pruning test may be able to eliminate a whole family of candidate FSTs at once.

There are three types of pruning tests that have been shown to be particularly effective for the Euclidean Steiner tree problem: projection tests; tests based on the bottleneck Steiner distance bound; and lune tests (see Brazil and Zachariasen (2015), Section 1.4.3). The role of the projection tests is to ensure that each generated tree is full - in other words, to avoid degeneracy. Because of this, the projection tests can be applied, without any changes, to the PCEST problem. The two other types of tests, however, need to be considered in more detail.

6.1.1. PCEST bottleneck Steiner distance bound

With respect to the MStTs, the *Bottleneck Steiner Distance* (BSD) bound provides a useful way to eliminate some FSTs from consideration. The BSD as defined for MStTs cannot be used directly for the maximum PCEST, but modified forms are presented here.

Let $Z_T(n_i, n_j)$ denote the unique path between n_i and n_j in some MST T for N.

Definition 16 (MStT Bottleneck Steiner Distance (MStT BSD)). The *bottleneck Steiner distance* $BSD(n_i, n_j)$ for two terminals n_i and n_j in T, is equal to the length of the longest edge in $Z_T(n_i, n_j)$.

Lemma 8. *(Brazil and Zachariasen (2015)) Given two terminals $n_i, n_j \in N$, let $Z_S(t_i, t_j)$ be the path between two terminals in some MStT S for N. For any edge $e \in Z_S(n_i, n_j)$, $|e| \leq BSD(n_i, n_j)$.*

We now consider the use of the BSD for solving the PCEST problem. During the Generation phase of GeoSteiner, when it would be useful to make use of the BSD, some of the terminals in the solutions are known (N_{I}), but it is not known which of the points in N_{P} are in the solution. Accordingly, an MST on the terminals in the solution for the purposes of calculating BSDs cannot be constructed beforehand. However, the following lemma shows that useful bottleneck distances can still be computed without needing to know the terminal set for the maximum PCEST.

Consider the set of terminals $\hat{N}$ for a maximum PCEST, its corresponding MST $\hat{T}$ and the bottleneck Steiner distance $BSD(n_i, n_j)$ for $n_i, n_j \in \hat{N}$.

Lemma 9. *For $n_i, n_j \in \hat{N}$, the addition of a point to $\hat{N}$ does not increase $BSD(n_i, n_j)$.*

Proof. Let $\hat{N}' := \hat{N} \cup \{n_k\}$. Denote the corresponding MST and bottleneck Steiner distances as $\hat{T}'$ and $BSD'(n_i, n_j)$ respectively. We will show that $BSD(n_i, n_j) \geq BSD'(n_i, n_j)$. Suppose to the contrary that $BSD(n_i, n_j) < BSD'(n_i, n_j)$. Then the longest edge e' in the path $Z_{\hat{T}'}(n_i, n_j)$ must be longer than the longest edge e in the path $Z_{\hat{T}}(n_i, n_j)$. If e' is removed from $\hat{T}'$, then two component networks remain, one of which contains n_i and the other containing n_j. Denote these two components A and B. Observe that $Z_{\hat{T}}(n_i, n_j)$ must include an edge f that connects a member of A to a member of B, with $|f| < |e'|$. It follows that if e' is replaced with f then a spanning tree on N' will have a shorter length, contradicting the minimality of $\hat{T}'$. □

Definition 17 (PCEST Bottleneck Steiner Distance (PBSD)). The *PCEST bottleneck Steiner distance* $PBSD(n_i, n_j)$ for points $n_i, n_j \in N_{\mathrm{I}} \cup N_{\mathrm{P}}$, is equal to the length of the longest edge in $Z_{\hat{T}}(n_i, n_j)$ for some MST of $N_{\mathrm{I}} \cup \{n_i, n_j\}$.

The following lemma is a direct corollary of Lemma 9.

Lemma 10 (PBSD bound)**.** *Given two points $n_i, n_j \in N_{\mathrm{I}} \cup N_{\mathrm{P}}$, let $Z_{\mathrm{P}}(n_i, n_j)$ be a path in some maximum PCEST P for N. For any edge $e \in Z_{\mathrm{P}}(n_i, n_j)$, $|e| \leq PBSD(n_i, n_j)$.*

All PCEST bottleneck Steiner distances can be rapidly computed in a preprocessing phase (immediately following the running of tests for ruling points in or out). These bottleneck distances can then be used during the Generation phase to limit the lengths of edges during the construction of FSTs, with the effect of substantially reducing the number of FSTs that are constructed. This same strategy has been used very successfully for the Euclidean Steiner tree problem.

6.1.2. PCEST lune and disk properties

The *lune property* was established by Gilbert and Pollak (1968) as a condition that must be satisfied by MStTs for a given terminal set.

Definition 18 (Lune). Given a line segment $\overline{bc}$, the *lune* $\mathbb{L}(b, c)$ is the intersection of open disks $\Gamma(b)$ and $\Gamma(c)$ centred at the points b and c respectively, each with radius $|bc|$.

Lemma 11. *(Gilbert and Pollak (1968)) If $\overline{bc}$ is an edge or a part of an edge of an MStT S, then the lune $\mathbb{L}(b, c)$ does not contain any points of S.*

In GeoSteiner for the Euclidean Steiner tree problem, the lune property (Lemma 11) is used to prevent the construction of any edge that contains a terminal in its lune. Similarly, for the PCEST problem it is an easy exercise to see that Lemma 11 can be combined with the argument used in the proof of Lemma 2 to give the following result.

Lemma 12. *Let $\overline{bc}$ be an edge or a part of an edge of a maximum PCEST P, and let a be an element of $N_{\mathrm{I}} \cup N_{\mathrm{P}}$. Let $\Gamma(a)$ denote a disk with its centre on a and with radius equal to the weight w_a of a.*

1. *If $a \in N_{\mathrm{I}}$, then the lune $\mathbb{L}(b, c)$ does not contain a.*
2. *If any part of $\overline{bc}$ lies in $\Gamma(a)$, then $\mathbb{L}(b, c)$ does not contain a.*

It is straightforward to ensure that both conditions in this lemma are maintained during the generation phase of GeoSteiner; this process has the effect of removing from consideration individual candidate FSTs, or indeed whole families of FSTs.

We also have the following related result.

Lemma 13. *Let $\overline{bc}$ be an edge or a part of an edge of a maximum PCEST P, and let a be an element of N_{P}. If the lune $\mathbb{L}(b, c)$ contains a, then a can be ruled out as a terminal of P.*

Lemma 13 does not reduce the set of candidate FSTs generated during the generation phase of GeoSteiner, but it could be used to add extra constraints to the concatenation phase, which has the potential to help speed up the concatenation phase (which is discussed further in the next subsection).

6.2. The Concatenation phase for the PCEST problem

Here, we first describe the standard integer programming formulation for the GeoSteiner Concatenation phases (for MStTs) and then present a method of revising the formulation in order to solve the Concatenation problem for the PCEST problem. We also outline an alternative method based on a prize collecting Steiner tree in graphs (PCSTG) problem, and note that there exist efficient implemented solutions to this latter problem.

6.2.1. Standard GeoSteiner concatenation formulation for MStT

Recall that for the Euclidean Steiner tree problem the aim of the concatenation phase of GeoSteiner is to find a subset of candidate FSTs (from the generation phase) that span the terminals such that the sum of lengths of these FSTs is minimum. The approach is to treat each candidate FST as a hyperedge on a

set of terminals in a hypergraph, where the cost of the hyperedge is the length of the FST, and to find the MST of the hypergraph.

Consider a hypergraph $G = (V, E)$, with vertices $v_i \in V$ and hyperedges $e_j \in E$ with each $e_j \subseteq V$. Here $V = N$, the given set of terminals, and each element of E corresponds to the terminal set of a candidate FST. Let c_j denote the weight of hyperedge e_j, which we define to be the length of the candidate FST represented by e_j. A minimum Steiner tree on N now corresponds to a minimum weight spanning tree on V, with hyperedges E_T. The decision variable $x_j = 1$ if $e_j \in E_T$, otherwise $x_j = 0$. Let $|e_j|$ and $|V|$ denote the cardinality of e_j and V respectively. The spanning tree formulation is as follows:

$$\text{minimise} \quad \sum_{j:e_j \in E} c_j x_j \tag{7}$$

$$\text{subject to} \quad \sum_{j:e_j \in E} (|e_j| - 1)x_j = |V| - 1 \tag{8}$$

$$\sum_{e_j \in E, e_j \cap W_k \neq \varnothing} (|e_j \cap W_k| - 1)x_j \leq |W_k| - 1, \qquad \emptyset \neq W_k \subset V \tag{9}$$

$$x_j \in \{0, 1\}, \qquad j : e_j \in E_T. \tag{10}$$

Constraint 8 ensures that T has the correct number of hyperedges of appropriate cardinality for a spanning tree on V. Constraint 9 ensures that T does not contain any cycles, and hence, by the previous constraint, is connected. For more details on this formulation, see Chapter 5 of Brazil and Zachariasen (2015).

It has been shown that this formulation can be solved efficiently using branch-and-cut methods, based in the work of Warme (1998) – this has been implemented as part of GeoSteiner for the Euclidean Steiner tree problem.

6.2.2. Concatenation formulations for the PCEST problem

In revising the previous formulation for the PCEST problem, the full components are again treated as hyperedges in a hypergraph. The revision involves adding decision variables for the inclusion of the terminals and weights for the terminals. The notation is the same as before, but with the following additions: let w_i denote the weight of vertex v_i; let V_T be the set of terminals in a minimum PCEST tree; and let the decision variables $y_i = 1$ if $v_i \in V_T$, otherwise $y_i = 0$. The formulation is as follows:

$$\text{maximise} \quad \sum_{i:v_i \in V} w_i y_i - \sum_{j:e_j \in E} c_j x_j \tag{11}$$

$$\text{subject to} \quad \sum_{j:e_j \in E} (|e_j| - 1)x_j - \sum_{i:v_i \in V} y_i = -1 \tag{12}$$

$$\sum_{e_j \in E, e_j \cap W_k \neq \varnothing} (|e_j \cap W_k| - 1)x_j \leq |W_k| - 1, \qquad \emptyset \neq W_k \subset V \tag{13}$$

$$\sum_{i:v_i \in e_j} y_i - x_j \geq 0, \qquad j : e_j \in E \tag{14}$$

$$y_i \in \{0, 1\}, \qquad i : v_i \in V \tag{15}$$

$$x_j \in \{0, 1\}, \qquad j : e_j \in E. \tag{16}$$

Constraints 12 and 13 have the same functions as 8 and 9 respectively, modified to ensure the correct number and cardinality of hyperedges for the resulting set of terminals V_T. Constraint 14 guarantees that every terminal belongs to at least one hyperedge of the final tree. That is, if $y_i = 1$, then there must exist a hyperedge e_j with $x_j = 1$ such that $v_i \in e_j$.

An alternative approach is to transform the concatenation phase of GeoSteiner for the PCEST problem into a prize-collecting Steiner tree problem on graphs (PCSTG) problem. The transformation process, following the generation phase of GeoSteiner, involves constructing a weighted graph $G = (V, E)$ as follows:

- Each possible terminal $n_i \in N_{\mathrm{P}}$ is represented as a vertex $v \in V$ with weight $p_v = w_i$.
- Each ruled in terminal (including the mandatory terminal) is represented as a vertex $v \in V$ with weight p_v set sufficiently high to ensure its inclusion in the solution. Alternatively, the decision variable for the inclusion of each such vertex is pre-set to 1.
- Each Steiner point s_i of each candidate FST is represented as a vertex $v \in V$ with weight $p_v = 0$.
- Each edge (v_i, v_j) of a candidate FST is represented by an edge $e \in E$ between the corresponding vertices in V with cost $c_e = |v_i v_j|$.

It is clear that a solution to the PCSTG problem for G corresponds to a maximum PCEST on N. The disadvantage of this approach is if the set of candidate FSTs is large, then the graph G becomes large and dense, meaning that this method may not scale well with the cardinality of N. On the other hand, a potential advantage of this approach is that there exists an efficient software solution to the PCSTG problem. An example is the solver "SCIP-Jack" (Gamrath et al. (2017)). The solver is capable of solving a wide variety of Steiner tree problems in graphs, including PCSTG and the rooted variant thereof. The general approach used by Gamrath et. al. is to transform a PCSTG problem into an equivalent Steiner arborescence problem and to solve that problem by using a sophisticated MIP framework that can be employed in massively parallel environments.

6.3. Solving the un-rooted PCEST problem

Note that the various ruling out techniques, FST generation and concatenation described in Sections 4.2 to 6.2 apply to both the rooted and un-rooted PCEST problems. On the other hand, the two main ruling in lemmas in Section 4.1 require the existence of a known ruled in point. Hence, the full suite of techniques can be applied to any case where there is at least one mandatory or ruled in terminal. We now briefly discuss the case of the un-rooted PCEST problem where there are no given ruled in points.

Let N denote the set of points from an instance of the PCEST problem, where none of the points is identified as mandatory or ruled in. Let c_i denote the i^{th} cluster resulting from the application of the merging algorithm to N. Let C denote the set of clusters. Now the ruling in algorithm is utilised in a new way.

For each cluster $c_i \in C$:

1. Select any one point in cluster c_i. For the purposes only of the next step, change the selected point's identification to n_0 (i.e. the mandatory terminal).
2. Run the ruling in algorithm. Denote the set of clusters ruled in as U_i (including the cluster containing n_0. Observe that $U_i \subseteq C$. Note that each U_i corresponds to points ruled in to a maximum rooted PCEST, with the selected point in c_i treated as the mandatory terminal.

Let U denote the set of all U_is.

Now the results of the aforementioned ruling in runs are analysed, first to check if a ruled in terminal for a maximum un-rooted PCEST can be identified by finding a point common to all the U_is, or else to find the smallest possible set of points, one or more of which must be a member of a maximum un-rooted PCEST.

Let Q_k denote a largest possible subset of U such that there is at least one cluster that is common to all U_i elements of Q_k. Let $|Q|$ denote the cardinality of Q. If $|Q| = 1$, then the common cluster to Q_k contains points that are all in a maximum un-rooted PCEST. If $|Q| > 1$, then each Q_k contains a cluster c_{Q_k} at least one of which must be in a maximum un-rooted PCEST.

7. Conclusions

In this paper, we have described a new algorithmic framework for solving PCEST problems. The broad strategy is to first find a solution to the rooted PCEST problem. Then, if a solution to the un-rooted PCEST problem is required, it can be obtained by judicious application of the rooted PCEST problem to a small set of cases. A key issue is that points provided in a problem instance for a PCEST problem are only possible terminals in the solution. The strategy for solving the rooted PCEST problem is to reclassify possible terminals to ruled in or ruled out, through a series of tests. Points that are ruled out are discarded. What remains is a set of ruled in terminals and a set of possible terminals.

The remaining topics covered in this paper concern GeoSteiner, a software package that includes efficient algorithms to solve Euclidean Steiner tree problems. We have described ways in which GeoSteiner's Generation and Concatenation functions can be adapted to the PCEST problem. The efficient implementation of all these changes, along with associated computational analysis, is a topic for future research.

Acknowledgement

David Whittle is the grateful recipient of the 2015 Gilbert Rigg Scholarship, which supported his graduate research in underground mine plan optimisation. In addition, David was awarded the 2016 John Collier Scholarship to support his research-related travel.

References

Brazil, M., Thomas, D.A., Weng, J.F., 2000. On the complexity of the Steiner problem. *Journal of combinatorial optimization* 4, 2, 187–195.

Brazil, M., Zachariasen, M., 2015. *Optimal Interconnection Trees in the Plane, Theory, Algorithms and Applications*. Algorithms and Combinatorics. Springer International Publishing Switzerland.

Chung, F.R., Graham, R.L., 1985. A new bound for Euclidean Steiner minimal trees. *Annals of the New York Academy of Sciences* 440, 1, 328–346.

De Wet, P.O., 2009. Geometric Steiner minimal trees. PhD thesis, University of South Africa, Mathematics.

Gamrath, G., Koch, T., Maher, S.J., Rehfeldt, D., Shinano, Y., 2017. SCIP-Jack — a solver for STP and variants with parallelization extensions. *Mathematical Programming Computation* 9, 2, 231–296.

Garey, M.R., Graham, R.L., Johnson, D.S., 1977. The complexity of computing Steiner trees. *SIAM Journal on Applied Mathematics* 32, 4, 835–859.

Gilbert, E., Pollak, H., 1968. Steiner minimal trees. *SIAM Journal on Applied Mathematics* 16, 1, 1–29.

Goemans, M.X., Williamson, D.P., 1995. A general approximation technique for constrained forest problems. *SIAM Journal on Computing* 24, 2, 296–317.

Graham, R.L., 1972. An efficient algorithm for determining the convex hull of a finite planar set. *Info. Pro. Lett.* 1, 132–133.

Johnson, D.S., Minkoff, M., Phillips, S., 2000. The prize collecting Steiner tree problem: theory and practice. In *SODA 2000*, Citeseer, p. 4.

Kirszenblat, D., 2014. The Steiner ratio conjecture for eight points. Thesis, University of Melbourne, Mathematics & Statistics.

Ljubic, I., Weiskircher, R., Pferschy, U., Klau, G.W., Mutzel, P., Fischetti, M., 2005. Solving the prize-collecting Steiner tree problem to optimality. In *ALENEX/ANALCO 2005*, pp. 68–76.

Preparata, F.P., Shamos, M.I., 1988. *Computational Geometry: An Introduction* (2nd edn.). Springer, New York.

Remy, J., Steger, A., 2009. Approximation schemes for node-weighted geometric Steiner tree problems. *Algorithmica* 55, 1, 240–267.

Rubinstein, J.H., Thomas, D.A., 1991. The Steiner ratio conjecture for six points. *Journal of Combinatorial Theory, Series A* 58, 1, 54–77.

Sun, Y., Brazil, M., Thomas, D., Halgamuge, S., 2019. The fast heuristic algorithms and post-processing techniques to design large and low-cost communication networks. *IEEE/ACM Transactions on Networking*

Warme, D., Winter, P., Zachariasen, M., 2017. *GeoSteiner 5.1 User's Guide and Reference Manual*.

Warme, D.M., 1998. Spanning Trees in Hypergraphs with Applications to Steiner Trees. Thesis, University of Virginia, Faculty of the School of Engineering and Applied Science.

Warme, D.M., Winter, P., Zachariasen, M., 2000. Exact algorithms for plane Steiner tree problems: A computational study. In *Advances in Steiner trees*. Springer, pp. 81–116.

Whittle, D., 2019. Optimisation of underground mine plans. PhD thesis, University of Melbourne, Engineering.

Whittle, D., Brazil, M., Grossman, P., Rubinstein, H., Thomas, D.A., 2020. Solving the prize collecting Euclidean Steiner tree problem - two universal constants.

Winter, P., 2002. Optimal Steiner hull algorithm. *Computational geometry* 23, 2, 163–169.