An Introduction to Steiner Minimal Trees on Grids

Frederick C. Harris, Jr.
fredh@cs.unr.edu

Department of Computer Science
University of Nevada
Reno, Nevada 89557

Abstract

The optimization problem is simply stated as follows: Given a set of N cities, construct a connected network which has minimum length. The problem is simple enough, but the catch is that you are allowed to add junctions in your network. Therefore the problem becomes how many extra junctions should be added, and where should they be placed, so as to minimize the overall network length.

This intriguing optimization problem is also known as the Steiner Minimal Tree Problem, where the junctions that are added to the network are called Steiner Points. What is known about the general problem will be presented first and then the focus will turn from the general problem to the problem on a lattice of points called a grid.

The characterization of the Steiner Minimal Tree (SMT) for a $2 \times m$ grid is generally known, while the only other conjectured characterizations for grids previously known were for square grids. We will present the characterization of SMT's for $2 \times m$ grids, as well as characterizations of SMT's for grids up through $7 \times m$.

Keywords: Steiner Minimal Trees, Grids

1 The Problem

Minimizing a network's length is one of the oldest optimization problems in mathematics and, consequently, it has been worked on by many of the leading mathematicians in history. In the mid-seventeenth century a simple problem was posed: Find the point P that minimizes the sum of the distances from P to each of three given points in the plane. Solutions to this problem were derived independently by Fermat, Torricelli and Cavalieri. They all deduced that either P is inside the triangle formed by the given points and that the angles at P formed by the lines joining P to the three points are all 120°, or P is one of the three vertices and the angle at P formed by the lines joining P to the other two points is greater than or equal to 120°.

In the nineteenth century a mathematician at the University of Berlin named Jakob Steiner studied this problem and generalized it to include an arbitrarily large set of points in the plane. This generalization created a star when P was connected to all the given points in the plane and is a geometric approach to the 2-dimensional center of mass problem.

In 1934 Kössler and Jarník generalized the network minimization problem even further [20]: Given n points in the plane find the shortest possible connected network containing these points. This generalized problem, however, did not become popular until the book *What is Mathematics* by Courant and Robbins [11] was published in 1941. Courant and Robbins linked the name Steiner with this form of the problem proposed by Kössler and Jarník, and it became known as the Steiner Minimal Tree (SMT) problem. The general solution to this problem allows multiple points to be added, each of which is called a Steiner Point, creating a tree instead of a star.

Much is known about the exact solution to the Steiner Minimal Tree problem. Those who wish to learn about some of the spin-off problems are invited to read the introductory article by Bern and Graham [1], the excellent survey paper on this problem by Hwang and Richards [18], or the recent volume in The Annals of Discrete Mathematics devoted completely to Steiner Tree problems [19]. Some of the basic pieces of information about the Steiner Minimal Tree problem that can be gleaned from these articles are: (i) all of the original n points will be of degree 1, 2, or 3, (ii) the Steiner Points are all of degree 3, (iii) any two edges meet at an angle of at least 120° in the Steiner Minimal Tree, and (iv) at most $n-2$ Steiner Points will be added to the network.

In Section 2 we review the first fundamental algorithm generally known for calculating SMT's. In Section 3 we look at the current computational results for finding SMT's. In Section 4 we present what is currently known about SMT's on grids and then introduce SMT's for other grid networks. Finally we present some ambitious future work in Section 5.

2 The First Solution

A typical problem-solving approach is to begin with the simple cases and expand to a general solution. As we saw in Section 1, the trivial three point problem had already been solved in the 1600's, so all that remained was the work toward a general solution. As with many interesting problems this is harder than it appears on the surface.

The method proposed by the mathematicians of the mid-seventeenth century for the three point problem is illustrated in Figure 1. This method stated that in order to calculate the Steiner Point given points A, B, and C, you first construct an equilateral triangle (ACX) using the longest edge between two of the points (AC) such that the third (B) lies outside the triangle. A circle is circumscribed around the triangle, and a line is constructed from the third point (B) to the far vertex of the triangle (X).

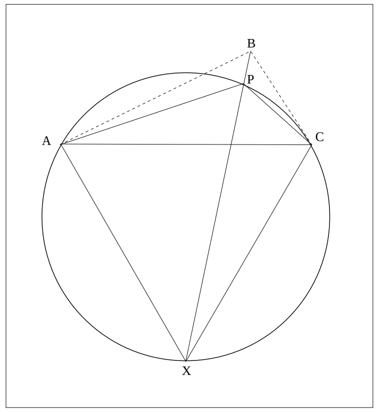

Figure 1: AP + CP = PX.

The location of the Steiner Point (P) is the intersection of this line (BX) with the circle.

For the next thirty years after Kössler and Jarník presented the general form of the SMT problem, the only algorithms that existed were heuristics. The heuristics were typically based upon the Minimum-Length Spanning Tree (MST), which is a tree that spans or connects all vertices whose sum of the edge lengths is as small as possible, and tried in various ways to join three vertices with a Steiner Point. In 1968 Gilbert and Pollak [14] linked the length of the SMT to the length of a MST. It was already known that the length of an MST is an upper bound for the length of an SMT, but their conjecture stated that the length of an SMT would never be any shorter than $\frac{\sqrt{3}}{2}$ times the length of an MST. This conjecture was recently proved [12] and has led to the MST being the starting point for most of the heuristics that have been proposed in the last 20 years [16, 19].

In 1961 Melzak developed the first known algorithm for calculating an SMT [21]. Melzak's Algorithm was geometric in nature and was based upon some simple extensions to Figure 1. The insight that Melzak offered was the fact that you can reduce an n point problem to a set of $n-1$ point problems. This reduction in size is accomplished by taking every pair of points, A and C in our example, calculating where the two possible points, X_1 and X_2, would be that form equilateral triangles with them, and creating two smaller problems, one where X_1 replaces A and C, and the other where X_2 replaces A and C. Melzak's algorithm can then be run on the two smaller problems. This recursion, based upon replacing two points with one point, finally terminates when you reduce the problem from three to two vertices. At this termination the length of the tree will be the length of the line segment connecting the final two points. This is due to the fact that $BP + AP + CP = BP + PX$. This is straightforward to prove using the law of cosines, for when P is on the circle, $\angle APX = \angle CPX = 60°$. This allows the calculation of the last Steiner Point (P) and allows you to back up the recursive call stack to calculate where each Steiner Point in that particular tree is located.

This reduction is important in the calculation of an SMT, but the algorithm still has exponential order, since it requires looking at every possible reduction of a pair of points to a single point. The recurrence relation for an n-point problem is stated quite simply in the following formula:

$$T(n) = 2 * \binom{n}{2} * T(n-1).$$

This yields what is obviously a non-polynomial time algorithm. In fact Garey, Graham, and Johnson [13] have shown that the Steiner Minimal Tree problem is NP-Hard (NP-Complete if the distances are rounded up to discrete values).

3 Current Results

In 1967, just a few years after Melzak's paper, Cockayne [6] clarified some of the details from Melzak's proof and proposed a new term, the "steiner hull" [7]. The steiner hull was the foundation for the first decomposition algorithm for the SMT problem. This decomposition, along with a clarified algorithm, proved to be the basis for the first computer program to calculate SMTs, which appeared in [10], and which could compute an SMT for any placement of up to 7 vertices.

The biggest breakthrough after Cockayne's algorithm came from Winter in 1985 [22] who was able to establish some geometric results that enabled one to eliminate (prune out) a great many of the reconstruction sequences required by Melzak's algorithm. Using this, Winter was able to compute SMTs for 15 or fewer vertices. Winter's algorithm has been the basis for most of the enhancements in SMT algorithms since then.

In 1986 another major computational breakthrough was made. Cockayne and Hewgill [8] were able to calculate the SMT for up to 30 vertices 80% of the time. They achieved this through the use of what they termed an incompatibility matrix, which took the sub-trees left after Winter's pruning and determined whether trees i and j could appear together in the SMT.

Most of the rest of the current results for exact computation of SMTs belong to Cockayne and Hewgill. In 1992 they developed better pruning techniques that have allowed them to calculate SMTs for up to 100 vertices 80% of the time. Their paper describing these results [9] recently appeared in a special issue of *Algorithmica* devoted exclusively to the Steiner Problem.

The last major breakthrough belongs to the author. His work involved complete parallelization of the construction of SMTs. While it is still under modification, this method has reduced the computation time by at least an order of magnitude and methods under examination may lead to further improvements [15, 17]. Table 1 provides a summary of the major programs written to solve the SMT problem and their capabilities.

Program	Author(s)	Location	Points
STEINER [10]	Cockayne & Schiller	Univ of Victoria	7
STEINER72 [2]	Boyce & Serry	ATT Bell Labs	10
STEINER73 [5]	Boyce & Serry	ATT Bell Labs	12
GEOSTEINER [22]	Winter	Univ of Copenhagen	15
EDSTEINER86 [8]	Cockayne & Hewgill	Univ of Victoria	30
EDSTEINER89 [9]	Cockayne & Hewgill	Univ of Victoria	100
PARSTEINER94 [15, 17]	Harris	Clemson University	100

Table 1: Computation Results

4 Grids

The problem of determining SMTs for grids was mentioned to the author by Ron Graham. In this context we are thinking of a grid as a regular lattice of unit squares. The literature has little of information regarding SMTs on grids, and most of the information that is given is conjectured and not proven. In Section 4.1 we will look at what is known about SMTs on grids. In the following sub-sections we will introduce new results for grids up through $7 \times m$ in size. The new results presented are computational results from PARSTEINER94 [15, 17] which was discussed in the previous section.

4.1 $2 \times m$ and Square Grids

The first proof for anything besides a 2×2 grid came in a paper by Chung and Graham [4] in which they proved the optimality of their characterization of SMTs for $2 \times m$ grids. The only other major work was presented in a paper by Chung, Gardner, and Graham [3]. They argued the optimality of the SMT on 2×2, 3×3, and 4×4 grids and gave conjectures and constructions for those conjectures for SMTs on all other square lattices.

In their work Chung, Gardner, and Graham specified three building blocks from which all SMTs on square ($n \times n$) lattices were constructed. The first, labeled $\mathcal{I}$, is just a K_2, or a path on two vertices. This building block is given in Figure 2-A. The second, labeled $\mathcal{Y}$, is a Full Steiner Tree (FST) (n vertices and $n-2$ steiner points) on 3 vertices of the unit square. This building block is given in Figure 2-B. The third, labeled $\mathcal{X}$, is an FST on all 4 vertices of the unit square. This building block is given in Figure 2-C. For the generalizations we are going to make here, we need to introduce one more building block, which we will label $\mathcal{S}$. This building block is an FST on a 3×2 grid and appears in Figure 2-D.

SMTs for grids of size $2 \times m$ have two basic structures. The first is an FST on all the vertices in the $2 \times m$ grid. An example of this for a 2×3 grid is given in Figure 2-D. The other structure is constructed from the building blocks previously described. We hope that these building blocks, when put in conjunction with the generalizations for $3 \times m$, $4 \times m$, $5 \times m$, $6 \times m$, and $7 \times m$ will provide the foundation

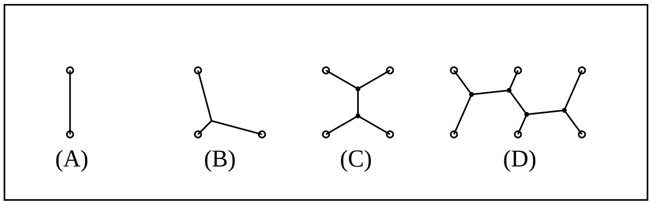

Figure 2: Building Blocks

for a generalization of $m \times n$ grids in the future.

In their work on ladders ($2 \times m$ grids) Chung and Graham established and proved the optimality of their characterization for $2 \times m$ grids. Before giving their characterization, a brief review of the first few $2 \times m$ SMTs is in order. The SMT for a 2×2 grid is shown in Figure 2-C, the SMT for a 2×3 grid is shown in Figure 2-D, and the SMT for a 2×4 grid is given in Figure 3.

Chung and Graham [4] proved that SMTs for ladders fell into one of two categories. If the length of the ladder was odd, then the SMT was the FST on the vertices of the ladder. The SMT for the 2×3 grid in Figure 2-D is an example of this. If the length of the ladder was even, the SMT was made up of a series of $(\frac{m}{2} - 1)$ $\mathcal{XI}$'s followed by one last $\mathcal{X}$. The SMT for the 2×4 grid in Figure 3 is an example of this.

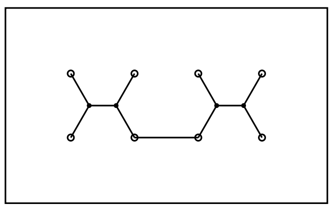

Figure 3: SMT for a 2×4 Grid

4.2 $3 \times m$ Grids

The SMT for $3 \times m$ grids has a very easy characterization which can be seen once the initial cases have been presented. The SMT for the 3×2 grid is presented in Figure 2-D. The SMT for the 3×3 grid is presented in Figure 4.

From here we can characterize all $3 \times m$ grids. Except for in the 3×2 grid, which is an $\mathcal{S}$ building block, there will be only two basic building blocks present, $\mathcal{X}$'s and $\mathcal{I}$'s. There will be exactly two $\mathcal{I}$'s and $(m-1)\mathcal{X}$'s. The two $\mathcal{I}$'s will appear on each end of the grid. The $\mathcal{X}$'s will appear in a staggered checkerboard pattern, one on

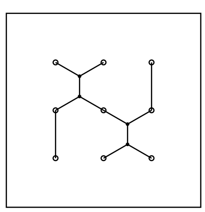

Figure 4: SMT for a 3×3 Grid

each column of the grid the same way that the two $\mathcal{X}$'s are staggered in the 3×3 grid. The 3×5 grid is a good example of this and is shown in Figure 5.

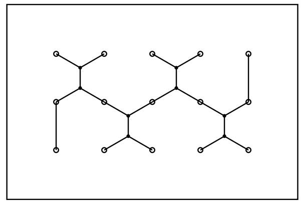

Figure 5: SMT for a 3×5 Grid

4.3 $4 \times m$ Grids

The foundation for the $4 \times m$ grids has already been laid. In their most recent work, Cockayne and Hewgill presented some results on Square Lattice Problems [9]. They looked at $4 \times m$ grids for $m = 2$ to $m = 6$. They also looked at the SMTs for these problems when various lattice points in that grid were missing. What they did not do, however, was characterize the structure of the SMT's for all $4 \times m$ grids.

The 4×2 grid is given in Figure 3. From the work of Chung, Gardner, and Graham [3], we know that the SMT for a 4×4 grid is a checkerboard pattern of 5 $\mathcal{X}$'s. This layout gives us the first two patterns we will need to describe the $4 \times m$ generalization. The first pattern, which we will call pattern $\mathcal{A}$, is the same as the 3×4 grid without the two $\mathcal{I}$'s on the ends. This pattern is given in Figure 6. The second pattern, denoted as pattern $\mathcal{B}$, is the 2×4 grid in Figure 3 without the connecting $\mathcal{I}$. This is shown in Figure 7.

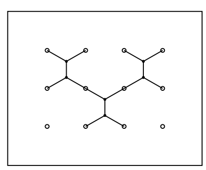

Figure 6: $4 \times m$ Pattern $\mathcal{A}$

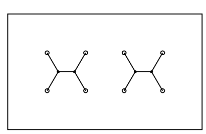

Figure 7: $4 \times m$ Pattern $\mathcal{B}$

Before the final characterization can be made, two more patterns are needed. The first one, called pattern $\mathcal{C}$, is a 4×3 grid where the pattern is made up of two non-connected 2×3 SMTs, shown in Figure 8. The next pattern, denoted pattern $\mathcal{D}$, is quite simply a $\mathcal{Y}$ centered in a 2×4 grid. This is shown in Figure 9. The final pattern, denoted $\mathcal{E}$, is just an $\mathcal{I}$ on the right side of a 2×4 grid. This is shown in Figure 10.

Now we can begin the characterization. The easiest way to present the characterization is with some simple string rewriting rules. Since the 4×2, 4×3, and 4×4

patterns have already been given, the rules will begin with a 4×5 grid. This grid has the string $\mathcal{AC}$. The first rule is that whenever there is a $\mathcal{C}$ on the right end of your string replace it with $\mathcal{BDB}$. Therefore a 4×6 grid is $calABDB$. The next rule is that whenever there is a $\mathcal{B}$ on the right end of your string replace it with a $\mathcal{C}$. The final rule is whenever there is a $\mathcal{DC}$ on the right end of your string replace it with an $\mathcal{EAB}$. These rules are summarized in Table 2. A listing of the strings for m from 5 to 11 is given in Table 3.

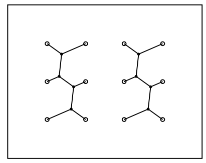

Figure 8: $4 \times m$ Pattern $\mathcal{C}$

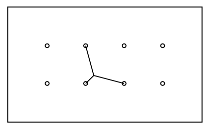

Figure 9: $4 \times m$ Pattern $\mathcal{D}$

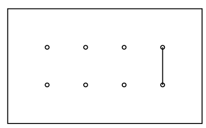

Figure 10: $4 \times m$ Pattern $\mathcal{E}$

1	$\mathcal{B} \rightarrow \mathcal{C}$
2	$\mathcal{C} \rightarrow \mathcal{BDB}$
3	$\mathcal{DC} \rightarrow \mathcal{EAB}$

Table 2: Rewrite rules for $4 \times m$ Grids.

m =	5	6	7	8
String	$\mathcal{AC}$	$\mathcal{ABDB}$	$\mathcal{ABDC}$	$\mathcal{ABEAB}$

m =	9	10	11
String	$\mathcal{ABEAC}$	$\mathcal{ABEABDB}$	$\mathcal{ABEABDC}$

Table 3: String Representations for $4 \times m$ Grids

4.4 $5 \times m$ Grids

For the $5 \times m$ grids there are 5 building blocks (and their mirror images which are donated with an $'$) that are used to generate any $5 \times m$ grid. These building blocks appear in Figure 11, Figure 12, Figure 13, Figure 14, and Figure 15.

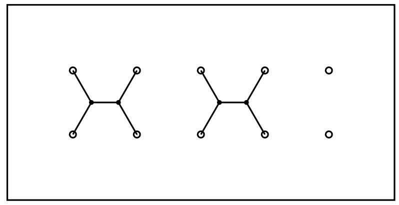

Figure 11: $5 \times m$ Pattern $\mathcal{A}$

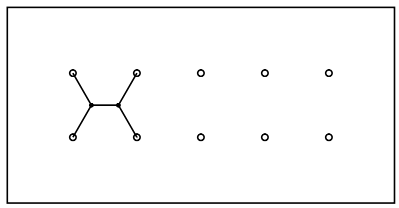

Figure 12: $5 \times m$ Pattern $\mathcal{B}$

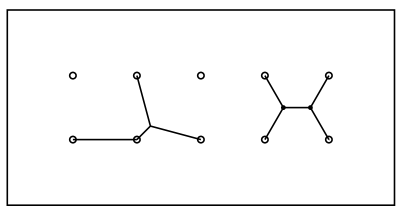

Figure 13: $5 \times m$ Pattern $\mathcal{C}$

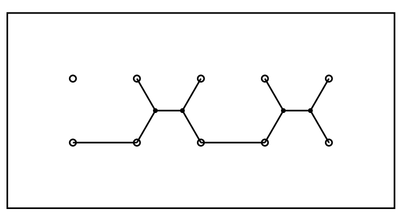

Figure 14: $5 \times m$ Pattern $\mathcal{D}$

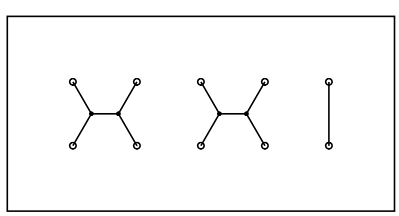

Figure 15: $5 \times m$ Pattern $\mathcal{E}$

With the building blocks in place, the characterization of $5 \times m$ grids is quite easy using grammar rewrite rules. The rules used for rewriting strings representing a $5 \times m$ grid are given in Table 4. The SMTs for 5×2, 5×3, and 5×4 have already been given. For a 5×5 grid the SMT is made up of the following string: $\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{D}$. As a reminder, the $\mathcal{A}'$ signifies the mirror of building block $\mathcal{A}$. A listing of the strings for m from 5 to 11 is given in Table 5.

1	$\mathcal{C} \rightarrow \mathcal{B}'\mathcal{D}'$
2	$\mathcal{D} \rightarrow \mathcal{A}'\mathcal{E}$
3	$\mathcal{E} \rightarrow \mathcal{A}\mathcal{C}$
4	$\mathcal{C}' \rightarrow \mathcal{B}\mathcal{D}$
5	$\mathcal{D}' \rightarrow \mathcal{A}\mathcal{E}'$
6	$\mathcal{E}' \rightarrow \mathcal{A}'\mathcal{C}'$

Table 4: Rewrite rules for $5 \times m$ Grids

$m =$	5	6	7	8
String	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{D}$	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{E}$	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{C}$	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{D}'$

$m =$	9	10	11
String	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{E}'$	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{A}'\mathcal{C}'$	$\mathcal{E}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{A}'\mathcal{B}\mathcal{D}$

Table 5: String Representations for $5 \times m$ Grids

4.5 $6 \times m$ Grids

For the $6 \times m$ grids there are 5 building blocks that are used to generate any $6 \times m$ grid. These building blocks appear in Figure 16, Figure 17, Figure 18, Figure 19, and Figure 20.

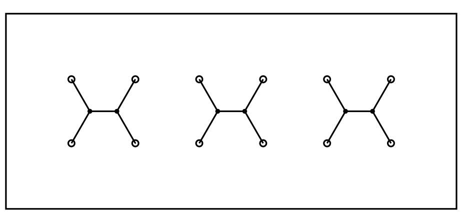

Figure 16: $6 \times m$ Pattern $\mathcal{A}$

The solution for $6 \times m$ grids can now be characterized by using grammar rewrite rules. The rules used for rewriting strings representing a $6 \times m$ grid are given in Table 6. The basis for this rewrite system is the SMT for the 6×3 grid which is $\mathcal{A}\mathcal{C}$. It is also nice to see that for the $6 \times m$ grids there is a simple regular expression which can characterize what the string will be. That regular expression has the form: $\mathcal{A}(\mathcal{B}\mathcal{E})^*(\mathcal{C}|\mathcal{B}\mathcal{D})$. A listing of the strings for m from 6 to 11 is given in Table 7.

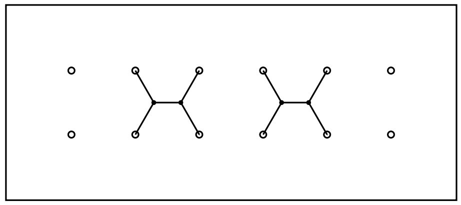

Figure 17: $6 \times m$ Pattern $\mathcal{B}$

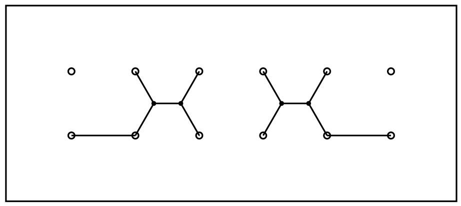

Figure 18: $6 \times m$ Pattern $\mathcal{C}$

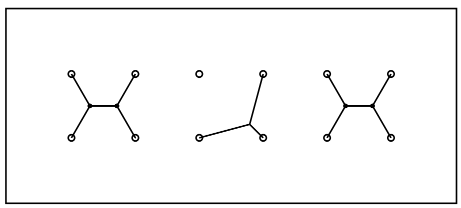

Figure 19: $6 \times m$ Pattern $\mathcal{D}$

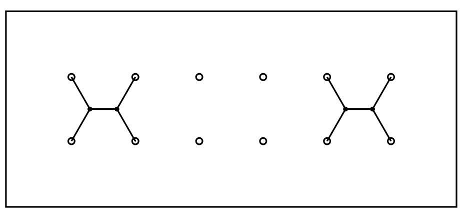

Figure 20: $6 \times m$ Pattern $\mathcal{E}$

1	$\mathcal{C} \rightarrow \mathcal{BD}$
2	$\mathcal{D} \rightarrow \mathcal{EC}$

Table 6: Rewrite rules for $6 \times m$ Grids

$m =$	6	7	8
String	$\mathcal{ABEBD}$	$\mathcal{ABEBEC}$	$\mathcal{ABEBEBD}$

$m =$	9	10	11
String	$\mathcal{ABEBEBEC}$	$\mathcal{ABEBEBEBD}$	$\mathcal{ABEBEBEBEC}$

Table 7: String Representations for $6 \times m$ Grids

4.6 $7 \times m$ Grids

For the $7 \times m$ grids there are 6 building blocks that are used to generate any $7 \times m$ grid. These building blocks appear in Figure 21, Figure 22, Figure 23, Figure 24, Figure 25, and Figure 26.

The grammar rewrite rules for strings representing a $7 \times m$ grid are given in Table 8. The basis for this rewrite system is the SMT for the 7×5 grid which is $\mathcal{F}\mathcal{A}'\mathcal{E}'\mathcal{F}'$. A listing of the strings for m from 6 to 11 is given in Table 9.

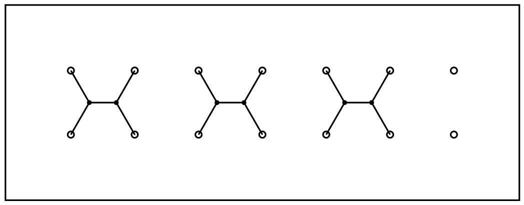

Figure 21: $7 \times m$ Pattern $\mathcal{A}$

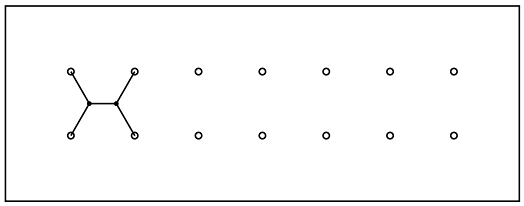

Figure 22: $7 \times m$ Pattern $\mathcal{B}$

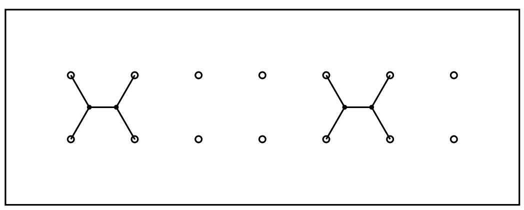

Figure 23: $7 \times m$ Pattern $\mathcal{C}$

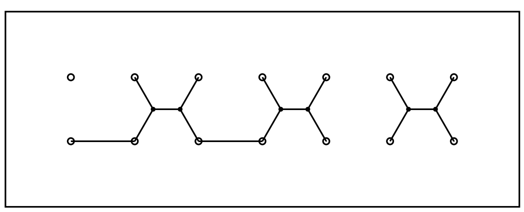

Figure 24: $7 \times m$ Pattern $\mathcal{D}$

5 Conclusions and Future Work

In this work we reviewed what is known about SMTs on grids and then presented results from PARSTEINER94 [15, 17] which characterize SMTs for $3 \times m$ to $7 \times m$ grids. The next obvious question is what is the characterization for an $8 \times m$ grid,

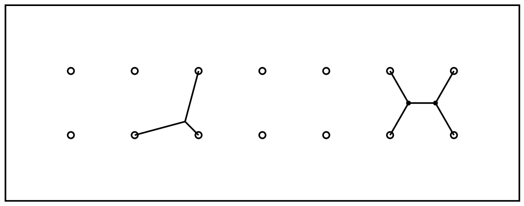

Figure 25: $7 \times m$ Pattern $\mathcal{E}$

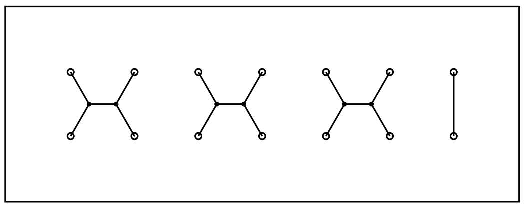

Figure 26: $7 \times m$ Pattern $\mathcal{F}$

1	$\mathcal{E}'\mathcal{F}' \rightarrow \mathcal{B}\mathcal{A}'\mathcal{F}$
2	$\mathcal{F} \rightarrow \mathcal{C}\mathcal{D}$
3	$\mathcal{C}\mathcal{D} \rightarrow \mathcal{A}\mathcal{E}\mathcal{F}$
4	$\mathcal{E}\mathcal{F} \rightarrow \mathcal{B}'\mathcal{A}\mathcal{F}'$
5	$\mathcal{F}' \rightarrow \mathcal{C}'\mathcal{D}'$
6	$\mathcal{C}'\mathcal{D}' \rightarrow \mathcal{A}'\mathcal{E}'\mathcal{F}'$

Table 8: Rewrite rules for $7 \times m$ Grids

$m =$	6	7	8	9
String	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{F}$	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{C}\mathcal{D}$	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{E}\mathcal{F}$	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{F}'$

$m =$	10	11	12
String	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{C}'\mathcal{D}'$	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{A}'\mathcal{E}'\mathcal{F}'$	$\mathcal{F}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{A}\mathcal{B}'\mathcal{A}\mathcal{A}'\mathcal{B}\mathcal{A}'\mathcal{F}$

Table 9: String Representations for $7 \times m$ Grids

or an $n \times m$ grid? Well, this is where things start getting nasty. Even though PARSTEINER94 cuts the computation time of the previous best program for SMTs by an order of magnitude, the computation time for an NP-Hard problem blows up sooner or later, and $8 \times m$ is where we run into the computation wall.

We have been able to make small chips into this wall though, and have some results for $8 \times m$ grids. The pattern for this seems to be based upon repeated use of the 8×8 grid which is shown in Figure 27. This grid solution seems to be combined with smaller $8\times$ solutions in order to build larger solutions. However, until better computational approaces are developed further characterizations of SMTs on grids will be very hard, and tedious.

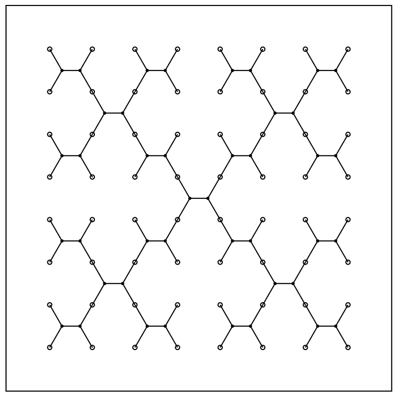

Figure 27: 8×8

References

[1] M.W. Bern and R.L. Graham. The shortest-network problem. *Sci. Am.*, 260(1):84–89, January 1989.

[2] W.M. Boyce and J.R. Seery. STEINER 72 – an improved version of Cockayne and Schiller's program STEINER for the minimal network problem. Technical Report 35, Bell Labs., Dept. of Computer Science, 1975.

[3] F.R.K. Chung, M. Gardner, and R.L. Graham. Steiner trees on a checkerboard. *Math. Mag.*, 62:83–96, 1989.

[4] F.R.K. Chung and R.L. Graham. Steiner trees for ladders. In B. Alspach, P. Hell, and D.J. Miller, editors, *Annals of Discrete Mathematics:2*, pages 173–200. North–Holland Publishing Company, 1978.

[5] E.J. Cockayne. Private communication.

[6] E.J. Cockayne. On the Steiner problem. *Canad. Math. Bull.*, 10(3):431–450, 1967.

[7] E.J. Cockayne. On the efficiency of the algorithm for Steiner minimal trees. *SIAM J. Appl. Math.*, 18(1):150–159, January 1970.

[8] E.J. Cockayne and D.E. Hewgill. Exact computation of Steiner minimal trees in the plane. *Info. Proccess. Lett.*, 22(3):151–156, March 1986.

[9] E.J. Cockayne and D.E. Hewgill. Improved computation of plane Steiner minimal trees. *Algorithmica*, 7(2/3):219–229, 1992.

[10] E.J. Cockayne and D.G. Schiller. Computation of Steiner minimal trees. In D.J.A. Welsh and D.R. Woodall, editors, *Combinatorics*, pages 52–71, Maitland House, Warrior Square, Southend-on-Sea, Essex SS1 2J4, 1972. Mathematical Institute, Oxford, Inst. Math. Appl.

[11] R. Courant and H. Robbins. *What is Mathematics? an elementary approach to ideas and methods.* Oxford University Press, London, 1941.

[12] D.Z. Du and F.H. Hwang. A proof of the Gilbert-Pollak conjecture on the Steiner ratio. *Algorithmica*, 7(2/3):121–135, 1992.

[13] M.R. Garey, R.L. Graham, and D.S Johnson. The complexity of computing Steiner minimal trees. *SIAM J. Appl. Math.*, 32(4):835–859, June 1977.

[14] E.N. Gilbert and H.O. Pollak. Steiner minimal trees. *SIAM J. Appl. Math.*, 16(1):1–29, January 1968.

[15] F.C. Harris, Jr. *Parallel Computation of Steiner Minimal Trees.* PhD thesis, Clemson, University, Clemson, SC 29634, May 1994.

[16] F.C. Harris, Jr. A stochastic optimization algorithm for steiner minimal trees. *Congr. Numer.*, 105:54–64, 1994.

[17] F.C. Harris, Jr. Parallel computation of steiner minimal trees. In David H. Bailey, Petter E. Bjorstad, John R. Gilbert, Michael V. Mascagni, Robert S. Schreiber, Horst D. Simon, Virgia J. Torczan, and Layne T. Watson, editors, *Proc. of the* 7^{th} *SIAM Conf. on Parallel Process. for Sci. Comput.*, pages 267–272, San Francisco, California, February 1995. SIAM.

[18] F.K. Hwang and D.S. Richards. Steiner tree problems. *Networks*, 22(1):55–89, January 1992.

[19] F.K. Hwang, D.S. Richards, and P. Winter. *The Steiner Tree Problem*, volume 53 of *Ann. Discrete Math.* North-Holland, Amsterdam, 1992.

[20] V. Jarník and O. Kössler. O minimálnich gratech obsahujicich n daných bodu [in Czech]. *Casopis Pesk. Mat. Fyr.*, 63:223–235, 1934.

[21] Z.A. Melzak. On the problem of Steiner. *Canad. Math. Bull.*, 4(2):143–150, 1961.

[22] P. Winter. An algorithm for the Steiner problem in the Euclidian plane. *Networks*, 15(3):323–345, Fall 1985.