A Genetic Algorithm for the Steiner Minimal Tree Problem

Joseph Jones
Department of Computer Science
University of Nevada
Reno, NV 89557
jones@cs.unr.edu

Frederick C. Harris, Jr.
Department of Computer Science
University of Nevada
Reno, NV 89557
fredh@cs.unr.edu

Abstract

We present a Genetic Algorithm to solve the Steiner Minimal Tree Problem. The Steiner Minimal Tree Problem is a network optimization problem in which we are able to add points to the network in order to minimize its length. The Genetic Algorithm searches for the number and location of these new points. Preliminary results are promising.

keywords: Genetic Algorithms, Steiner Minimal Tree

1 Introduction.

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°.

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

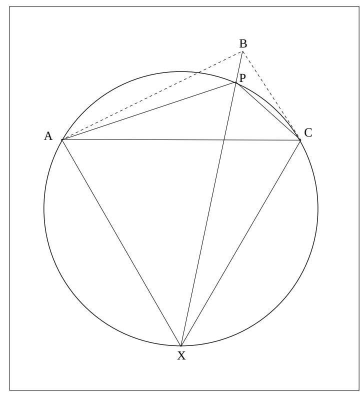

Figure 1: AP + CP = PX.

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 [10]: 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 [2], appeared 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 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 [8], or the recent volume in The Annals of Discrete Mathematics devoted completely to Steiner Tree problems [9]. Some of the basic pieces of information about the Steiner Minimal Tree problem that can be gleaned from these articles are: (i) the fact that 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.

While the Steiner Minimal Tree (SMT) problem is itself quite interesting, it also has a number of application areas. Some of these include: Laying telephone cable between a number of cities; laying pipe between points; connecting a number of computers together using the least amount of cable necessary.

The current algorithms in use for solving the SMT problem take a large amount of time to compute a non-trivial solution [5, 6]. It is for this reason that we have developed a Genetic Algorithm (GA) to solve the problem. The GA we developed has provided a great improvement in the time it takes for computation of SMTs, dropping the time from many hours to under 20 minutes. We are able to produce an acceptable answer for the 100 point problem, where acceptable means a SMT of shorter length than the Minimal Spanning Tree over the same points. For those readers interested in GAs, Goldberg [4] and Davis [3] are the important works in this field to read, while previous work in applying GAs to different aspects of the Steiner Tree Problem are presented in Hesser [7] and Julstrom [11].

In Section 2 we go into depth on the Genetic Algorithm we developed. In Section 3 Results are presented. Conclusions and Future Work are presented in Section 4.

2 The Genetic Algorithm

A Genetic Algorithm is an algorithm based upon the process of natural selection and genetics. It combines survival of the fittest with random, but structured, information exchange among members of the population. There are four distinct points in which GAs differ from more traditional search techniques. These are:

1. They work with an encoding of the problem parameters, not the parameters themselves.

2. They search through a population of points, not a single point.

3. They use objective function information (payoff), not derivatives or other secondary information.

4. They use probabilistic transition rules, not deterministic.

When using a GA, the parameters are usually encoded as a bit string. A GA begins with an instantiated population of individuals, in which each individual's encoded string is generated at random. The members of the population then undergo Reproduction, Crossover, and Mutation. It is these three processes that provide the strength of the GA as a search technique.

Reproduction selects members of the current generation and mates them. Selection is based on fitness of an individual, where fitness is a measure of how close each member of this population is to the correct solution. This provides the next generation with members that incorporate a better solution then the previous generation and is modeled after survival of the fittest. The members selected in this manner are called parents, and the new members of the next generation are called children

Crossover mates the encodings of the parents to generate children. A traditional GA uses one-point crossover, where a point on the chromosome (the bit string representing the encoding) is selected at random and the parents swap their strings from the selected point to the end of the chromosome. This swapping of material models chromosome crossover in the nucleus of cells.

Mutation provides for the generation of new information or the regeneration of possibly lost information due to selection. This process is modeled after radiation mutation of chromosomes in the cells nucleus [4].

In developing the GA for this problem, we made many design changes to the traditional GA. Specifically we changed the style of the encoding, altered the way crossover and mutation work and provided for population regeneration due to sub-optimal convergence.

Encoding: Our encoding consisted of an array of x and y coordinates representing the location of the Steiner Points and some extra flags used in Steiner Point generation and tree validity testing. Instead of crossing over at random bit positions, we decided to crossover the chromosomes at point boundaries. This provided us with the means of keeping known Steiner

Points, as opposed to creating random non-Steiner points.

Roulette wheel selection proved to be too random, causing the GA to not converge to any specific solution. For this reason we chose to use CHC selection with a probability of crossover (Pc) = 0.95 and a probability of mutation (Pm) = 0.05. These settings provided us with the best results.

Crossover: Simple one-point crossover proved to be a little too inefficient, so we decided to move to two-point crossover. The idea behind two-point crossover is that we select two points on the chromosome randomly and then swap the information between the two points. We selected this method to provide us with some control over locality problems concerning the Steiner Points.

Mutation: Mutation was another area in which modification was necessary. Since our encoding did not allow us to just mutate simple on/off bits, it was necessary to alter it. In our GA, mutation does two things: 1. If the chromosome is full, select a point at random and remove it. 2. Generate a new point and add it to the chromosome.

Population Regeneration: Our first tests with the GA showed that we were unable to generate an exact solution. We realized that not as many different possible Steiner Points as we needed were being generated. As an attempt to fix this problem, we implemented two different population regeneration schemes, both based on the length of population convergence. The two types of regeneration are (1) Random regeneration and (2) Cataclysmic regeneration. Random regeneration is when the bottom 80% of the population regenerates their encoded strings at random. Cataclysmic regeneration is when the bottom 95% of the population regenerates their encoded strings as mutations of the top 5%. While random regeneration proved useful and will be kept as a viable tool, cataclysmic regeneration provided little or no help in generating an exact solution.

3 Results

The parameters that we found to work the best are given in Table 1. We found these population size settings by comparing the information provided by evaluating the maximum performance while running this on a 4 x 4 Grid. The input and spanning tree for the 4 x 4 Grid are presented in Figure 2, and the GA output is given in Figure 3. We then used these settings to generate our solution to a 100-point problem. Our results are presented in Table 2. It is important to note that our GA results were computed from non-relaxed trees (the steienr points are not in the exact position, but the tree is structured properly). This is due to the fact that in the crossover operations trees are combined and placement of points is not re-calculated. Relaxation of the trees would provide an even larger decrease in length.

Probability of Crossover	0.95
Probability of Mutation	0.05
Population Size	75
Generations	100-200

Table 1: Best Parameter Settings

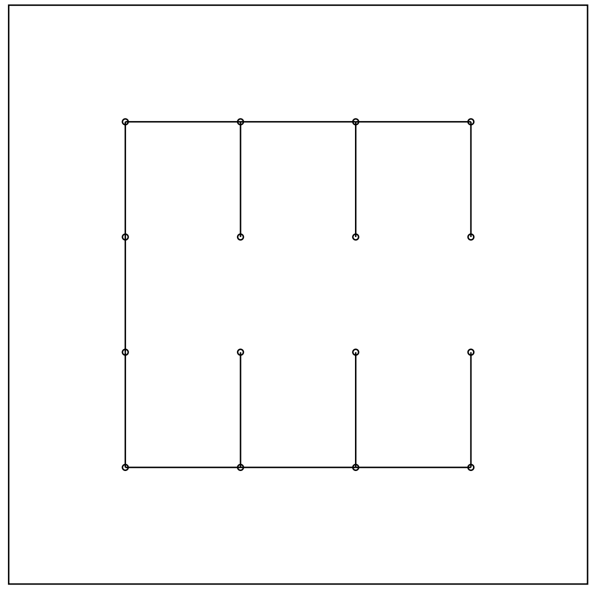

Figure 2: 4 x 4 Spanning Tree

GA solutions for test data sets. (Given as Tree length, % of MST Length)				
	16 point Grid problem		100 point Random problem	
MST Solution	3.00000	100%	6.44869	100%
SMT Solution	2.73205	91%	6.25546	97%
GA Solution	2.85757	95%	6.37021	98.8%

Table 2: GA Solution Comparisons

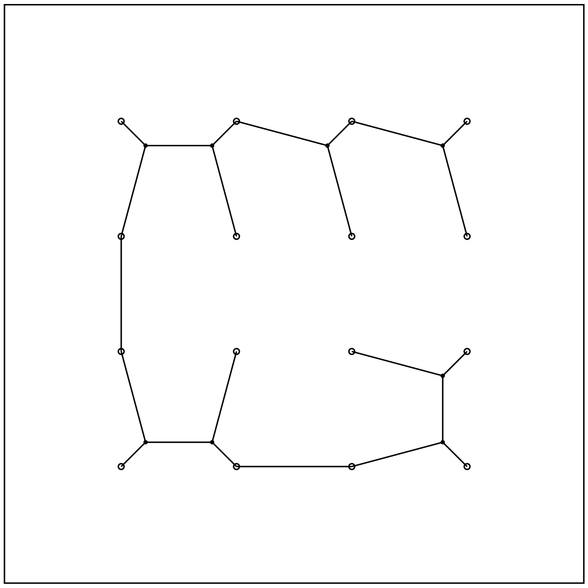

Figure 3: 4 x 4 Genetic Algorithm

Figure 4 presents the 100 point input set for the problem while Figures 5 and 6 show the output of the Genetic Algorithm for 100 points and the exact solution for this 100 point problem.

While looking at the information from the GA, we began to realize that it was usually generating sub-optimal solutions. We believe this is because we are not generating as many different Steiner Points as are necessary to find an exact solution. This is why we chose two-point crossover and population regeneration. Unfortunately these techniques provided only marginal performance improvement. We believe that the problem lies in how we generate Steiner Points, not in how the GA operators work. If this is true, we can gain considerably in run time by the elimination, or at least scaling back of, population regeneration.

4 Conclusions and Future Work

While the solutions provided from our GA for the test data sets are not exact, We believe that our GA holds much promise for finding good solutions to the SMT problem. With more work and an improvement in the Steiner Point generation algorithm, we hope we can produce near exact solutions in the future.

The future work for this problem extends in many areas. First we want to try to fix the problems that we are having with our Steiner Point generation. This could be done through a few different ways. The best (and probably the hardest) would be to treat the

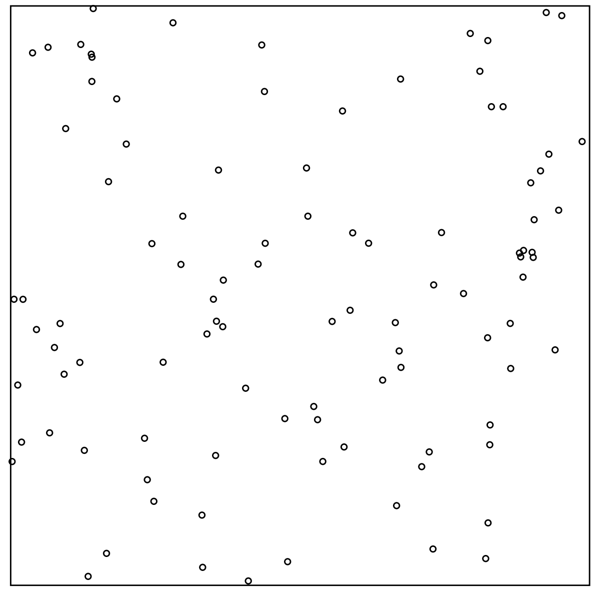

Figure 4: 100 point Input

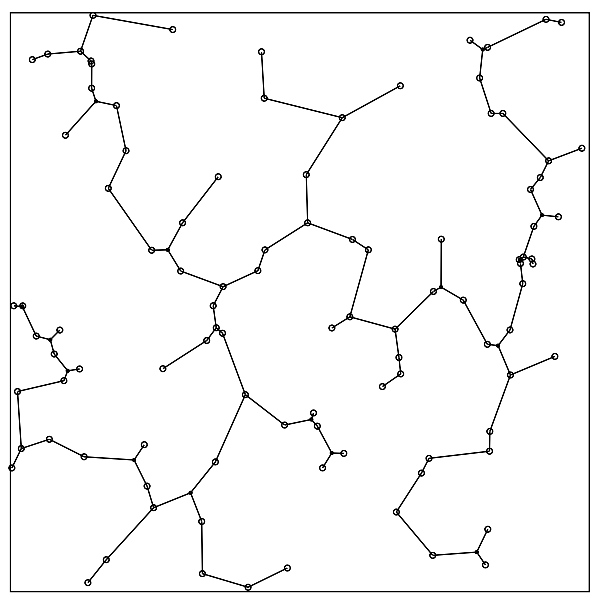

Figure 5: 100 point Genetic Algorithm

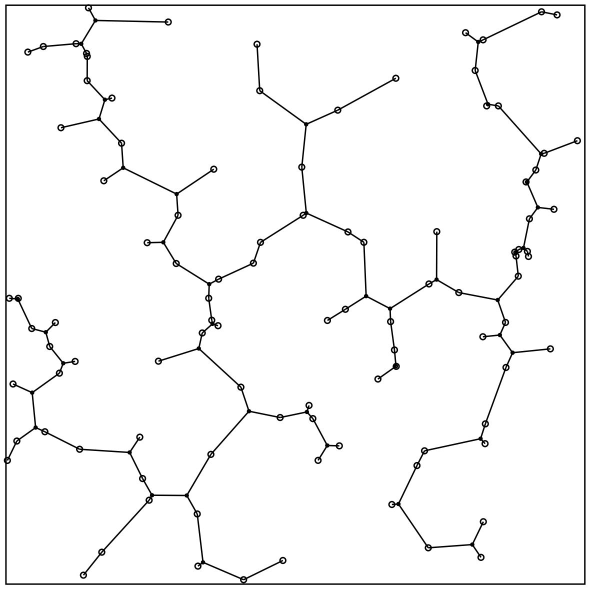

Figure 6: 100 point exact solution

points as logical points in a tree structure, where they logically connect other vertices, but do not have a coordinate location until the length of the tree is calculated. This would take care of the relaxation problem discussed earlier. The second item of future work would be to extend the GA to a parallel environment, as was described in [12], so that we can attack larger problems. The third direction for future work, which is probably the hardest, is to extend the SMT problem to 3 dimensions. This attack, while being the hardest, will probably yield some interresting communication networks for building construction, and other 3 dimensional spaces.

References

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

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

[3] L. Davis, editor. *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York, 1991.

[4] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

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

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

[7] J. Hesser, R. Manner, and O. Stucky. Optimization of steiner trees using genetic algorithms. In *International Conference on Genetic Algorithms*, pages 231–237, 1989.

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

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

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

[11] B.A. Julstrom. A genetic algorithm for rectilinear steiner problem. In *Fifthe International Conference on Genetic Algorithms*, pages 474–480, 1993.

[12] Marat Zhaksilikov and F.C. Harris, Jr. Comparison of different implementations of parallelization of genetic algorithms. In Frederick C. Harris, Jr., editor, *Proceedings of the ISCA Int. Conf. on Intelligent Systems (IS '96)*, Reno, NV, June 1996. ISCA.