GENETIC APPROACH TO THE MINIMUM WEIGHT TRIANGULATION

I.Kolingerová [1]

Department of Computer Science
University of West Bohemia
Plzeň, Czech Republic
kolinger@kiv.zcu.cz

ABSTRACT

The weight of a triangulation is the sum of the Euclidean lengths of its edges. A minimum weight triangulation is such a triangulation which minimizes the weight among all triangulations of the given point set. Finding the MWT is difficult as for general data sets, no exact polynomial algorithm is known. This paper describes how to find an approximate solution to the MWT by a genetic approach, i.e., by a probabilistic method which maintains a set of potential solutions and tries to improve them so that at the end at least one triangulation is close or equal to the optimum. The operators used for improvement are quite general and were derived from genetics.

The results obtained by this approach are compared either with optima if available or with results of already existing heuristic algorithms.

KEYWORDS

Computational gemetry – computer graphics - minimum weight triangulation – genetic algorithm – optimalization - greedy triangulation

1. INTRODUCTION

Thanks to efforts to find the most appropriate mesh data representation for visualization purposes, the concept of minimum weight triangulation (MWT) appeared many years ago as the triangulation with the minimal total light energy.

The MWT problem can be defined as follows [BKO97]:

Let S be a set of points in the plane. **A triangulation T(S) of S** is a maximal planar subdivision whose vertex set is S. (That means that no edge connecting two vertices can be added to S without intersecting one of the existing edges.)

The weight of a triangulation is the sum of the Euclidean lengths of its edges.

A minimum weight triangulation of S MWT(S) is such a triangulation that has minimum weight among all triangulations [HeP94].

Finding MWT is a difficult problem as it is not known whether a polynomial solution exists. So far, the problem is suspected to be NP-hard.

Therefore, three main approaches exist :

[1] This work was supported by The Ministry of Education of the Czech Republic – projects PG 97193, PG 97185 and VS 97155

1. To find an exact solution of polynomial complexity for special instances of data sets - e.g., [MeR92], [AnC93].
2. To find an exact solution of polynomial complexity for general data sets, producing a subgraph of MWT – e.g., [DiM96], [Kei94], [KMF97].
3. To find an approximate solution – e.g., [BFM96], [PlH87], [HeP94].

This paper presents a method which belongs into group 3 – tries to minimize the length of a triangulation by genetic approach. An inspiration for such a solution was found in [Mra96] where a genetic algorithm is used for image recognition and in [BFM96] and [MFN97] where simulated annealing was used for total edge length optimization.

The genetic method used in the paper was taken from the literature, its application to the minimum weight triangulation problem is original.

The paper is organized as follows: Section 2 describes the genetic approach in general, section 3 gives details about its modification for the MWT problem, section 4 presents results and section 5 concludes the paper.

2. GENETIC APPROACH

Genetic approach (GA) is a heuristic problem solving method. Its fundamentals can be found, e.g., in [Mic96] or [Gol89]. GA is a general, but weak method – to be successful in real application, needs to incorporate knowledge of the solved problem.

According to [Mic96], we understand genetic approach as a probabilistic method which maintains a **population** P(t) of individuals x_i^t : $P(t)=\{x_1^t,x_2^t...,x_n^t\}$ for an iteration t. Each individual is a potential solution to the given problem and is implemented as some (usually complex) data structure.

In each iteration, each individual is evaluated to measure its suitability to solve the given problem. This suitability is called **fitness**. Then a new population is formed by selecting individuals from the old population with respect to their fitness. Some of these individuals are changed by genetic operators. Two types of operators are used: an unary one called **mutation** which makes a small change in a single individual and a binary one called **crossover** which combines two individuals into two new individuals. All the process is repeated for some number of generations and then the best individual is chosen in belief it converged close enough to optimum.

There are several **key points in this process** :

- representation of individuals
- initial population
- evaluation function
- genetic operators
- proper parameters of GA

Representation of individuals

Members of population have to be represented in such a way to allow genetic operators to be used effectively. Very often, binary representation is used as it brings simplicity of operations; mutation is carried out by a bit inversion, crossover by random choice of a crossing point (a number of bit where to divide the binary number into two) and swapping two parts of two binary numbers.

Initial population

Before starting evolution, a starting population of potential solutions has to be generated in such a way that diversity of individuals is provided.

Evaluation function

Each individual has to be evaluated how good solution of the given problem it represents. Unlucky choice of this function may drastically reduce results of iteration and the speed of convergence as improper decisions are done.

Genetic operators

Members of a population are changed in two ways : by crossover when two individuals – parents are combined to develop two other individuals – children, and by mutation which is an alternation of a single individual. Both operators have to be designed properly to bring diversity into the population and at the same time not to lead out of the state space of valid solutions if possible. If a new individual – the result of crossover or mutation – is not a valid solution, some kind of correction has to be done to return back to the state space.

Proper parameters of genetic approach

Success of GA convergence is strongly influenced by a proper choice of its parameters, namely population size, termination condition, probabilities of applying genetic operators etc.

All these key points have to be solved in conditions of a particular application.

3. THE PROPOSED GENETIC ALGORITHM FOR THE MINIMUM WEIGHT TRIANGULATION

The overall structure of GA for the MWT is given in Algorithm 1 :

Input: a set S of n points in a plane
Output: a triangulation that is a suboptimal solution to MWT(S)

Variables :

CH - the convex hull
nCH - the number of edges in the convex hull
P - a population, i.e. a set of triangulations of the current generation stored in a temporary file
t - the number of generation
T_best - the triangulation with the highest fitness among all triang. over all generations.

---

1. Compute the convex hull CH(S) of S and store its number of edges in nCH;
2. Generate an initial population P(0) as a set of triangulations of S;
3. Evaluate fitness of all triangulations in P(0) and store the triangulation with the highest evaluation in T_best;
4. t := 1;
5. **while** not termination_condition **do**
6. **begin**
7. Select a new population P(t) from P(t-1);
8. Transform some members of P(t) by crossover and mutation;
9. Evaluate fitness of all triangulations in P(t);
10. **if** a triangulation with higher fitness than T_best is found **then** update T_best;
11. t := t + 1
12. **end**;
13. Return T_best as the resulting triangulation;

Algorithm 1 : General scheme of a genetic algorithm for the minimum weight triangulation

Let‘s explain the steps of the algorithm in detail.

Compute the convex hull

Convex hull is necessary for two reasons: first, its edges belong into any triangulation of the given data set, second, the number of edges in the convex hull is needed to know the number of other edges in a triangulation. Let‘s recall that the total number of edges in all triangulations over the given point set is the same, given by Euler's formula and equal to 3n - nCH - 3, where n is the number of points and nCH the number of convex hull edges.

Generate an initial population

To start evaluation, we need to generate an initial population of triangulations. First step is to prepare a list of all candidate edges which are possible in the given set of points S.

The first member of the initial population is the greedy triangulation (GT) generated according to the Algorithm 2.

Input: candEdges - a list of all possible edges in the point set S
Output: the greedy triangulation of S

Variables :

e - the active edge
i - an index of an edge in the candEdges
T - the resulting triangulation

---

1. Sort candEdges according to the length of edges into an increasing order;
2. T := {candEdges[1]}; (* the shortest candidate edge is accepted *)
3. i := 2;
4. **while** T is not finished **do**
5. **begin**
6. e := candEdges[i];
7. **if** {e} ∩ T = ∅ **then** T := T + {e};
8. i := i + 1;
9. **end**;
10. Return T;

Algorithm 2 : The greedy triangulation

The rest of the initial population is generated in a similar way - by Algorithm 2 without sort. The candidate edges are hashed into a random order for each particular triangulation.

The size of the population could be derived according to the number of points but for the time being was taken up to 150.

Using a GT as one of initial individuals has its advantage and disadvantage. Advantage is that the GT is one of heuristics for the MWT and therefore can help to improve the quality of the population. It also ensures that the genetic algorithm never finds a worse solution than the GT. On the other hand, GT may cause a premature convergence into a local maximum.

Evaluate P(t)

General formulae for evaluation were taken from [Mic96] and adjusted for our purposes as follows :

1. For the given generation :

For each triangulation T_i, i = 1,...,pop_size compute its evaluation :

$$eval(T_i) = 1/\sum_e l_e \quad , e=1,...,nE$$

where nE is the total number of edges of T_i,
l_e is the length of edge e in T_i
pop_size is the size of the population

2. The total fitness of the generation is :

$$F = \sum_{i=1}^{pop_size} eval(T_i)$$

3. For each T_i , i=1,...,pop_size compute probability p_i and cummulative probability q_i of its choice :

$$p_i = eval(T_i) / F$$

$$q_i = \sum_{j=1}^{i} p_j$$

Select a new population P(t) from P(t-1)

Selection is done by Algorithm 3 :

Input: a generation of triangulations P(t-1),
cummulative probabilities q_i, i=1,...,pop_size, of a generation P(t-1)
Output: a new generation P(t)

Variables :
i - an index of an active triangulation
r - a random number

---

1. P(t) := ∅;
2. **for** i := 1 **to** pop_size **do**
3. **begin**
4. Generate a random value r ∈ <0;1>;
5. **if** r < q_1 **then**
6. P(t) := P(t) + T_1
7. **else**
8. P(t) := P(t) + T_i :
   $q_{i-1} < r \le q_i$: $2 \le i \le pop_size$;
9. **end;**
10. Return P(t);

Algorithm 3 : Selection of a new population

A modification of this step is to complete this selected population with the best individual found so far but we do not use this approach.

Transform some members of P(t) by crossover and mutation

First, select individuals for crossover from the new generation. The probability of choice for crossover is given by p_c (e.g., p_c=0.25). Selection is done by Algorithm 4 :

Input: a new generation P(t), the probability of crossover p_c
Output: a list of potential parents Par and the size of Par par_size

Variables :

i - an index of an active triangulation
r - a random number

---

1. par_size := 0; Par := 0;
2. **for** i := 1 **to** pop_size **do**
3. **begin**
4. Generate a random value r ∈ <0;1>;
5. **if** r < p_c **then**
6. **begin** (* Select T_i for crossover*)
7. Par := Par + T_i ;
8. par_size := par_size + 1;
9. **end;**
10. **end;**
11. **if** odd (par_size**) then**
12. **begin**
13. Par := Par – $T_{parsize}$;
14. par_size := par_size - 1
15. **end**;
16. Return Par, par_size;

Algorithm 4 : The parents selection

Selected triangulations in Par are randomly paired. Each couple of triangulations can be understood as "mother" and "father" that will have together two children. Then, for each couple, a random value c_p ∈ <1;nE> is generated. This value is the number of the edge in the father triangulation where is a point of the crossover – that means, where to divide the list of edges of "father" triangulation and combine it with "mother" edges into a new "child". The "father" and "mother" are then swapped and the other child triangulation is generated, see Fig.1.

In reality, however, crossover is not so simple because by combining two triangulations in this way we obtain a general set of edges which usually do not form a triangulation as some edges intersect each other. If we accept only nonintersecting edges from both parents, we are not done, anyway, as the child triangulation may have holes which have to be retriangulated.

Therefore, the crossover operation of one couple is done as follows (see Algorithm 5) :

Input: two triangulations, "mother" and "father" and the set of all candidate edges
Output: two "child" triangulations

Variables :

cp - the crossover point

---

1. Generate a random value c_p ∈ <1;nE>;
2. Accept c_p edges from the father triangulation into the child one;
3. From the mother triangulation, start at (c_p + 1)-th edge and accept all those edges which do not intersect already accepted edges in the child triangulation;
4. If the child triangulation is not finished, complete it by inserting such edges from the candidate set which do not intersect already accepted edges;
5. Swap mother and father and repeat steps 2-4 once more;
6. Return both child triangulations;

Algorithm 5 : The crossover

Mutation is done on an edge base. Its probability is given by a value p_m (e.g., p_m = 0.01). See Algorithm 6 below.

Input: P(t) and the set of all candidate edges, the probability of mutation p_m
Output: P(t) with some mutations

Variables :

e - the active edge

---

1. **for** e := 1 **to** nE*pop_size **do**
   (* for each edge in P(t) *)
2. **begin**
3. Generate a random value r ∈ <0;1>;
4. **if** r < p_m **then**
5. Delete the edge e from its triangulation and replace it by another candidate edge that does not intersect any other edge present in the triangulation;
6. **end**;
7. Return P(t);

Algorithm 6 : The mutation

Termination condition

At our implementation, the termination condition is the total number of generations. We used 150 generations for smaller data sets (up to 25 points) and 300 generations for larger data sets. Other condition could be derived from the distance between the desired length of a triangulation and the smallest weight achieved during evolution.

The desired weight of the exact MWT is, of course, unknown, but can be approximated in two ways :

1. by the length of the greedy triangulation
2. by the length of the shortest edge multiplied by nE

Case 1: In a general case it can provide a good estimate [Lin86]. It was proved, however, that the greedy triangulation is in some special cases as bad approximation of the MWT as any other triangulation [MaZ79].

In our implementation, we cannot use GT as a measure as it is a member of the initial population.

Case 2 : This lower bound is simple but evidently too strict measure.

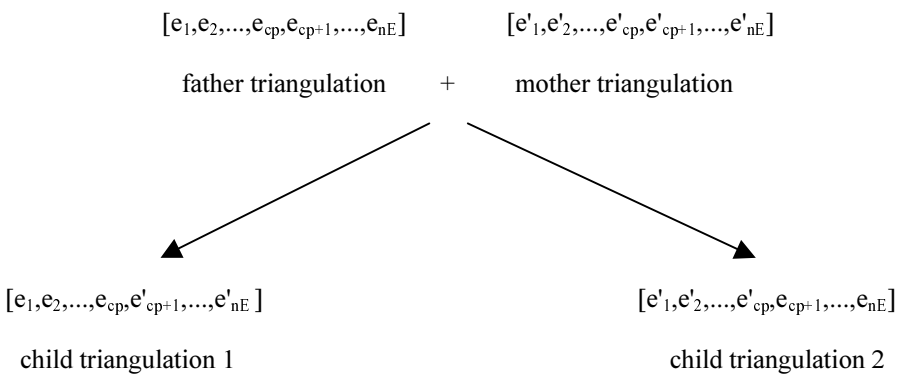

Fig.1 : Crossover of triangulations

4. RESULTS

The proposed method was implemented in Delphi 2 on Pentium Pro HP Vectra 128 MB/ 200 MHz and tested with sets of sizes 10, 15, 20 and 25 with 400 sets of each size and of sets 30, 40, 50 and 100 with 200 sets of each size. The points were generated randomly with uniform distribution. The sizes of data sets and type of experiments were chosen to be at least partly comparative with very good MWT heuristics in [Hep94] where spanning trees are used for the MWT- approximation construction.

Three types of comparison were made:

1. For the two smallest sizes, the results of GA were compared with the exact solution, computed by brute force method (for larger data sets is, however, our brute force program too slow), see Table 1.

2. For all the sizes, the results of GA were compared with the GT, see Table 2.

3. For the sizes 15, 20 and 25 points, the results of [HeP94] were also used for comparison. Larger data sets cannot be compared as in these cases [HeP94] compares itself with another heuristics and neither comparison with the greedy triangulation nor with the exact solution are available. See Table 2.

Notation: GT - greedy triangulation
GA - genetic approach
HP - the heuristics from [HeP94]

n	GT [%]	GA[%]
10	6.00	2.00
15	12.25	3.25

Table 1 : Number of failures to find an optimal triangulation in per cents for small data sets

n	Number of cases when GA is better then GT [%]	Number of cases when HP is better then GT [%]
10	4.25	-
15	10.00	11.00
20	8.75	15.00
25	10.00	17.00
30	12.50	-
40	11.50	-
50	10.25	-
100	9.00	-

Table 2 : Comparison of the genetic approach, greedy triangulation and the reference heuristics from [HeP94]

The dependence of the results on the size of the data set can be seen in Fig. 2.

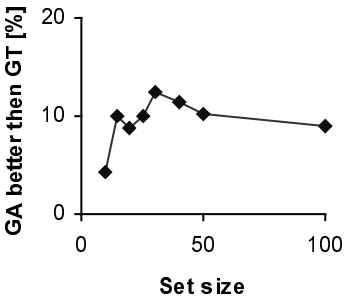

Fig. 2 : Dependence of success of GA on the set size

From the tables can be seen that the genetic approach pays itself in comparison with the greedy triangulation as in many cases it improves its results but, on the other hand, the heuristics from [HeP94] based on spanning trees seems stronger.

Nevertheless, the genetic approach seems worth further investigation and improvement, namely

- to start with more cleverly chosen initial population (not so much random) to improve convergence
- to derive the total number of generations and the total size of one generation from the total number of triangulations possible in the given data set
- to improve the terminal condition of the genetic evaluation
- to experiment with larger data sets
- to improve genetic operators so that a proper edge for completion of a triangulation is recognized without expensive intersection tests.

One example of the MWT approximation generated by the genetic approach is shown in Fig.3.

5. CONCLUSION

The paper presents a genetic approach for the MWT approximation. Results show improvement over the greedy triangulation but the heuristics of [HeP94] looks better. As the genetic algorithm was implemented in a very simple way, the results seem to be encouraging to devote further investigation to this subject, namely to incorporate more knowledge of the solved problem.

ACKNOWLEDGEMENT

I would like to thank to my colleagues from the Comenius University in Bratislava, namely to dr.A.Ferko, dr.L.Niepel and I.Magová whose ideas and programs stimulated this work and to uknown reviewers whose recommendations helped to improve the level of presentation.

REFERENCES

[AnC93] E.Anagnostou, D.Corneil : Polynomial-time Instances of the MWT Problem, *Computational Geometry, Theory and Applications* 3, 1993, pp.247-259

[BFM96] M.Bartánus, A.Ferko, R.Mag, L.Niepel, T.Plachetka, E.Šikudová : New Heuristics for Minimum Weight Triangulation, In : *WSCG 96, Conference Proceedings*, Pilsen, 1996, pp.31-40

[BKO97] M.de Berg, M.van Kreveld, M.Overmars, O.Schwarzkopf : *Computational Geometry. Algorithms and Applications*, Springer-Verlag Berlin Heidelberg, 1997

[DiM96] M.T.Dickerson, M.H.Montague : *A (Usually ?) Connected Subgraph of the Minimum Weight Triangulation*, In : ACM Symp.on Comp.Geom., 1996, pp.192-201

[Gol89] D.E.Goldberg : *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Pub., 1989

[HeP94] L.S.Heath, S.V.Pemmaraju : New Results for the MWT Problem, *Algorithmica* 12, 1994, pp.533-552

[Kei94] J.M.Keil : Computing a Subgraph of the Minimum Weight Triangulation, *Comput.Geom.*4, 1994, pp.13-26

[KMF97] I.Kolingerová, I.Magová, A.Ferko, L.Niepel : Better Subgraph of Minimum Weight Triangulation, In : *Spring Conference on Computer Graphics SCCG 1997, Conference Proceedings*, Bratislava, 1997, pp.49-56

[Lin86] A.Lingas : The Greedy and Delauney Triangulations Are not Bad in the Average Case, *Inf. Proc. Lett*. 22, 1986, pp.25-31

[MaZ79] G.K.Manacher, A.L.Zobrist : Neither the Greedy Nor the Delaunay Triangulation of a Planar Point Set Approximates the Optimal Triangulation, *Inf. Proc. Lett*.9, 1979, pp.31-34

[MeR92] H.Meijer, D.Rappaport : Computing the MWT of a set of linearly ordered points, *Inf. Proc. Lett.*42, 1992, pp.35-38

[MFN97] I.Magová, A.Ferko, L.Niepel : On Edges Elimination for the Shortest Mesh, In : *WSCG'97 Conference Proceedings*, Pilsen, 1997, pp.396-403

[Mic96] Z.Michalewitz *: Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag Berlin Heidelberg New York, 1996

[Mra96] P.Mrázek : Genetic Algorithms and Image Search, In : *WSCG 96, Conference Proceedings*, Pilsen, 1996, pp.278-284

[PlH87] D.A.Plaisted, J.Hong : A Heuristic Triangulation Algorithm, *J.Algorithms* 8, 1987, pp.405-437

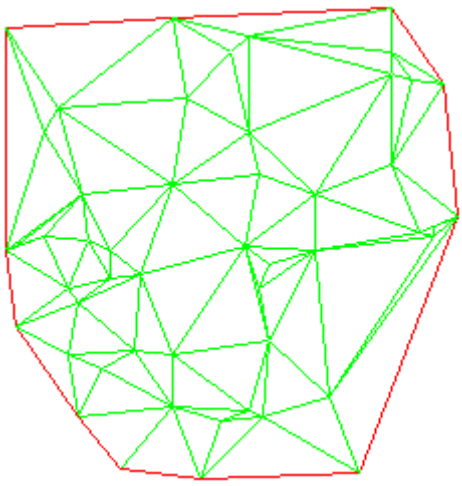

Fig. 3 : An example of the MWT approximation (N=50)