No Steiner Point Subdivision Simplification is NP-Complete

Regina Estkowski*

June 30, 1998

1 Introduction

Planar subdivisions arise in Geographic Information Systems, where a variety of operations are performed on them, collectively known as map generalization. In a general context, a planar subdivision is a collection of nonintersecting polygons, polygonal chains, and points in the plane. For a variety of reasons (rendering purposes, size of database, transfer of data), we may want to reduce the number of vertices in an input subdivision. We would also like to maintain some sort of "fidelity" to the original. A number of methods have been developed for reducing the number of vertices in a subdivision while meeting various constraints [1], [2], [3], [6], a process known as subdivision simplification.

Many do not consider the issue of intersections in the reduced subdivision. In those that do, there is no guarantee, except in restricted cases, that the resulting subdivision will contain anywhere near the optimal number of vertices. However, ideally the reduced subdivision should not contain intersections.

A version of the problem, in which we are given a polygonal region containing a subdivision and it is required that the simplified subdivision lie within the polygon while allowing Steiner Points was shown to be NP-complete in [5]. We consider a particular version, as defined below, in which Steiner Points are not allowed and show that this is NP-complete using a reduction from planar MAX 2SAT.

We will define a *planar subdivision* to be a collection of simple polygonal chains in the plane such that if two chains intersect, then any intersection point between the two chains is an endpoint of both chains.

A *No Steiner Point simplification* (NSP simplification) of the polygonal chain $c = v_1 \dots v_j$ is a polygonal chain $c' = v_{i_1} \dots v_{i_r}$ such that $i_1, \dots, i_r$ is a subsequence of $1, \dots, j$ with $i_1 = 1$ and $i_r = j$. A NSP simplification of the planar subdivision S is a collection of polygonal chains S' such that if $c' \in S'$ then c' is a NSP simplification of some $c \in S$ and if $c \in S$, then S' contains exactly one simplification of c.

No Steiner Point Subdivision Simplification (NSP SUBSIMP) and planar MAX 2SAT are defined as follows.

NSP SBSIMP :

Instance : *A planar subdivision S a constant $\epsilon \geq 0$, and a constant $N \geq 0$.*

Question : *Is there a NSP simplification S' of S such that*

*regina@ams.sunysb.edu, Dept. of Applied Mathematics and Statistics, State University of New York, Stony Brook, NY 11794-3600.

- *If $c = v_1 \ldots v_j$ is a chain in S, $v_{i_l}, v_{i_{l+1}} \in c'$, where c' is the simplification of c, $v_s \in c'$, and $i_l \leq s \leq i_{l+1}$, then v_s is contained in the ϵ-neighborhood of the line segment $v_{i_l}v_{i_{l+1}}$. The distance function is Euclidean distance.*

- *If c' and c' are chains in S' then c and c' do not intersect except possibly at endpoints, and every chain in S' is simple.*

- *S' contains at most N segments.*

Planar MAX 2SAT :

Instance : *A set of variables $\mathcal{V}$ and a set of disjunctive clauses $\mathcal{C}$, each containing at most two literals, and a constant $N \geq 0$. If G is the bipartite graph $G = \mathcal{C} \bigcup \mathcal{V}$ with an edge between $c \in \mathcal{C}$ and $v \in \mathcal{V}$ if and only if either v or $\neg v$ is in c, then G is planar.*

Question : *Is there a truth assignment for $\mathcal{V}$ which satisfies at least N clauses in $\mathcal{C}$.*

We have that NSP SUBSIMP is in NP since if S' is an arbitrary NSP simplification of S, we can determine in polynomial time if S' meets the criteria given for NSP SUBSIMP.

2 The Reduction

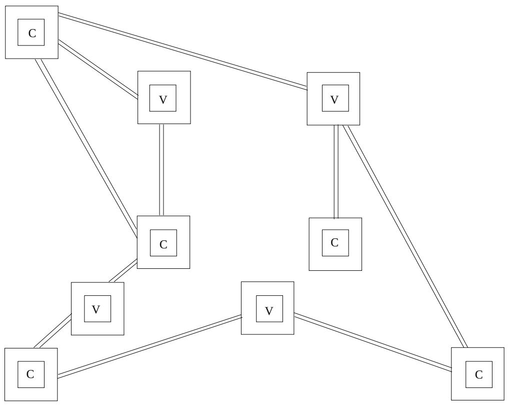

Figure 1: Variable nodes are denoted by V, and clause nodes by C.

Given an arbitrary instance I of planar MAX 2-SAT we will produce an instance I' of NSP SUBSIMP such that there is a simplification of I' having at most N' segments if and only if there is a truth assignment for I which satsifies at least N clauses. We may assume that no clause in I contains both v and $\neg v$ for any variable v. Suppose there are k clauses in I containing both a variable and its negation and I_k is the instance containing all variables in I and all clauses in I except for these k clauses. Then asking if there is a truth assignment for I satisfying at least N clauses is equivalent to asking if there is a truth assignment for I_k satisfying at least $N - k$ clauses.

We will let $\mathcal{V}$ denote the set of variables, and K the number of clauses in I. Let ϵ and $\mu < \epsilon$ be positive constants. If G is the graph associated with I, then embed G on a $3K$ by $3K$ grid where any two grid points have a distance of $40K^2(2\epsilon + \mu)$ from each other [7].

At each node of G, we temporarily center two nested squares. The outer square has side length $25K^2(2\epsilon + \mu)$ and the inner square has side length $(14K + 8)(2\epsilon + \mu)$. The area between the inner and outer square is called a *frame*. These squares will help to guide our construction and are not part of I'. Also, to guide the construction, put a *tube* of width 2ϵ around the portion of each edge lying outside of any square. See Figure 1.

2.1 Variable and Clause Gadgets

For each variable, we construct a composite gadget $\mathcal{VG}$ to represent the variable. This consists of two polygonal chains, $\mathcal{VG}_1$ and $\mathcal{VG}_2$, placed in the inner surrounding square in a T-formation, as shown in Figure 2. We will call the two squares surrounding $\mathcal{VG}$ along with $\mathcal{VG}$ a *variable node*.

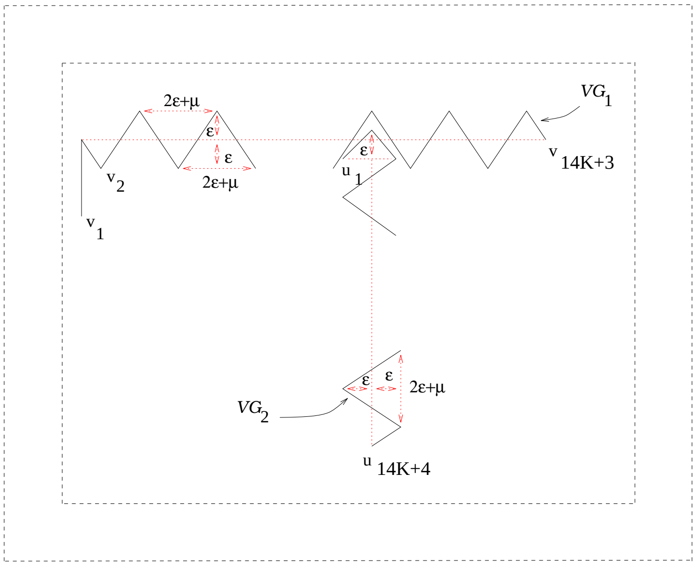

Figure 2: A composite variable gadget with surrounding frame.

The only possible simplification of $\mathcal{VG}_1$ is $c_1 = v_1v_2v_{14K+3}$. The two possible simplifications of $\mathcal{VG}_2$ are $c_{21} = u_1u_2u_{14K+4}$ and $c_{22} = u_1u_3\dots u_{14K+4}$, where c_{22} contains all vertices between u_3 and u_{14K+4}. If $\mathcal{VG}_1$ is simplified, then $\mathcal{VG}_2$ must be simplified by c_{22} and if $\mathcal{LG}_2$ is simplified by c_{21} then $\mathcal{VG}_1$ cannot be simplified. Simplifying $\mathcal{VG}_1$ will correspond to setting the variable represented by $\mathcal{VG}$ to FALSE, while simplifying $\mathcal{VG}_2$ by c_{21} will correspond to setting the variable to TRUE. In either case, there are a total of $14K + 4$ segments in the simplified composite gadget.

For each clause, we also construct a composite gadget $\mathcal{CG}$ to represent the clause. For a binary clause, this consists of three polygonal chains placed in the inner surrounding box in a H-formation; two *clause-literal gadgets*, $\mathcal{CG}_1$ and $\mathcal{CG}_2$, and one *toggle gadget*, $\mathcal{CG}_3$ as shown in Figures 3 and 4. $\mathcal{CG}_1$ and $\mathcal{CG}_2$ represent the literals in the clause and $\mathcal{CG}_3$ is a device which keeps track of the TRUE/FALSE count in the clause. A simplified clause-literal gadget corresponds to a TRUE literal, while an unsimplified clause-literal gadget corresponds to a FALSE literal.

The only possible simplification of $\mathcal{CG}_1$ is $d_1 = r_1r_4r_5$, with a similar simplification for $\mathcal{CG}_2$. It can be seen in Figures 3 and 4 that if both $\mathcal{CG}_1$ and $\mathcal{CG}_2$ are simplified, then $\mathcal{CG}_3$ must be simplified by $d_{33} = s_1s_3s_4s_5s_7$, if exactly one of $\mathcal{CG}_1$ and $\mathcal{CG}_2$ is simplified then $\mathcal{CG}_3$ must be simplified by one of $d_{31} = s_1s_6s_7$ or $d_{32} = s_1s_2s_7$, and if neither is simplified then $\mathcal{CG}_3$ remains unsimplified.

If at least one of $\mathcal{CG}_1$ or $\mathcal{CG}_2$ are simplified, then the simplified $\mathcal{CG}_3$ contains eight segments. This is the minimum number of segments for any simplification of the gadget. If neither of the

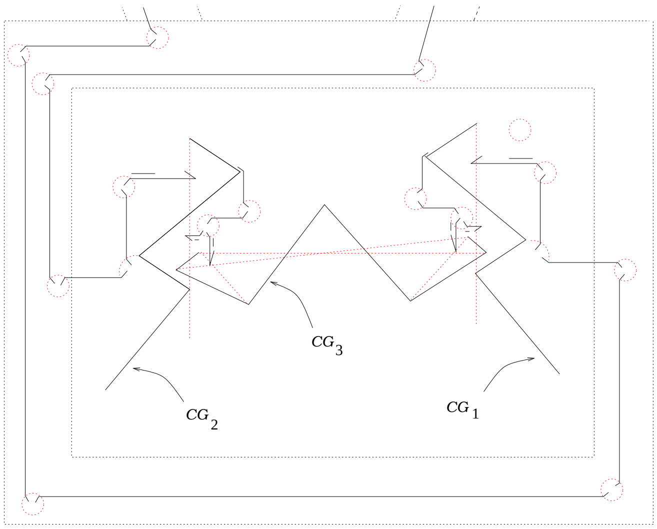

Figure 3: A binary composite clause gadget with surrounding frame and incoming hinge-link sequences. Hinge-link sequences are explained in Sections 2.2 and 2.3.

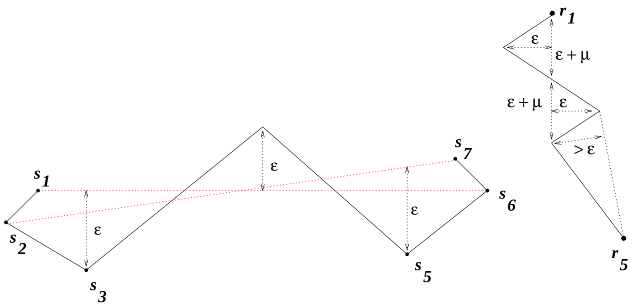

Figure 4: A toggle gadget and a clause-literal gadget.

clause-literal gadgets is simplified, then the simplified composite gadget contains fourteen segments.

For a unary clause, $\mathcal{CG}$ consists of one clause-literal gadget $\mathcal{UCG}_1$ and one *correction gadget* $\mathcal{UCG}_2$ as shown in Figure 5. $\mathcal{UCG}_1$ has only one simplification and it contains one segment, and $\mathcal{UGC}_2$ does not simplify. If $\mathcal{UCG}_1$ is simplified then the composite gadget contains eight segments, and if it is not simplified, then the composite gadget contains fourteen segments. A simplified $\mathcal{UCG}_1$ corresponds to a TRUE literal and $\mathcal{UCG}_2$ has no simplification.

In both the binary and unary cases, we have eight segments in the simplified composite gadget if at least one clause-literal gadget is simplified and fourteen segments if no clause-literal gadget is simplified. We will call $\mathcal{CG}$ along with the surrounding frame a *clause node*. If at least one clause-literal gadget in the composite clause gadget is simplified, we will say that the composite clause gadget is satisfied.

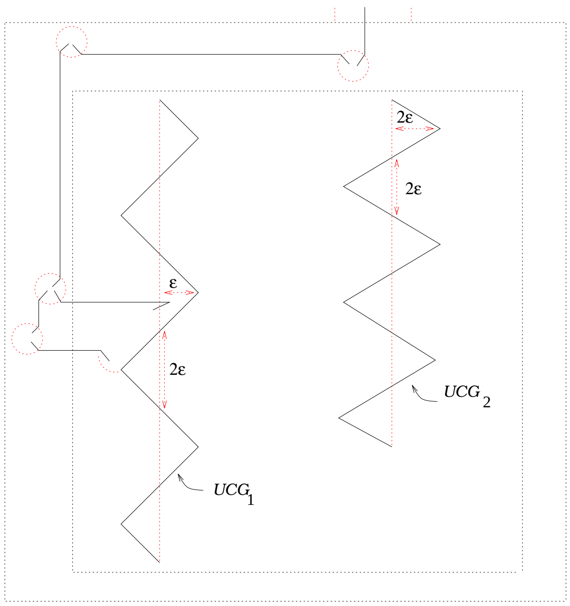

Figure 5: A unary composite clause gadget with surrounding frame and incoming hinge-link sequence.

2.2 Hinge-Link Sequences

We will use hinge-link sequences to communicate between variable and clause gadgets. A *hinge-link sequence* is an alternating sequence of two types of objects, called *hinges* and *links*. See Figure 6. Hinges are used to turn corners and consist of a number of two segment chains, called *hats*, arrayed in a circle. Links are used to travel straight lines and consist of one three segment chain.

Each chain in a hinge simplifies to a one segment chain. An isolated link simplifies to either a one or two segment chain, while a link in a hinge-link sequence can only simplify to a one segment chain. If any chain in a hinge link sequence is simplified, then all chains in a hinge-link sequence must be simplified with links simplified to one segment chains.

Associated with hinge-link sequences, there are *modified links*, *half-hinges*, and *switches*. See Figures 7, and 8

A modified link consist of a three segment chain along with a one segment chain to prevent a two segment simplification. Modified links simplify to two one segment chains. A half-hinge consists of a portion of a hinge.

A switch takes two input hinge-link sequences, one which is simplified and one which is not, and outputs two "switched" hinge-link sequences. So that if the simplified sequence enters on the left, it exits on the right, and visa versa. In Figure 8 the input sequences are denoted by in_1 and in_2, and the output sequences by out_1 and out_2.

A switch consists of two toggle gadgets, $\mathcal{S}w_1$ and $\mathcal{S}w_2$, along with connecting hinges and links. One toggle gadget is as shown in Figure 4 and the other is a scaled version.

To see how the sequences are switched, suppose that the sequence entering on the left is simplified. Then $\mathcal{S}w_1$ and $\mathcal{S}w_2$ must both be simplified to $s_1 s_2 s_7$, and the sequence exiting on the left cannot be simplified while the sequence exiting of the right must be simplified. In Figure 8, the entering sequences are in_1 and in_2 with in_1 on the left, and the exiting sequences are out_1 and

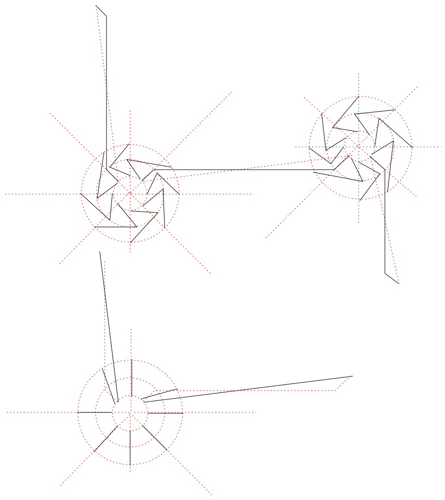

Figure 6: A portion of a hinge-link sequence and a portion of a simplified hinge-link sequence.

out_2.

Each switch is associated with a variable node, as will be seen in Section 2.3, and must be simplified if the composite variable gadget is simplified. Any two simplified switches always contain the same constant number of segments.

Extra segments are segments which are added to variable nodes so that the number of segments contained in the combined total of all extra segments, simplified hinges, links, modified hinges and links outside of switches is the same for any simplification in which all composite variables are simplified. See Figure 9 and Section 2.3.

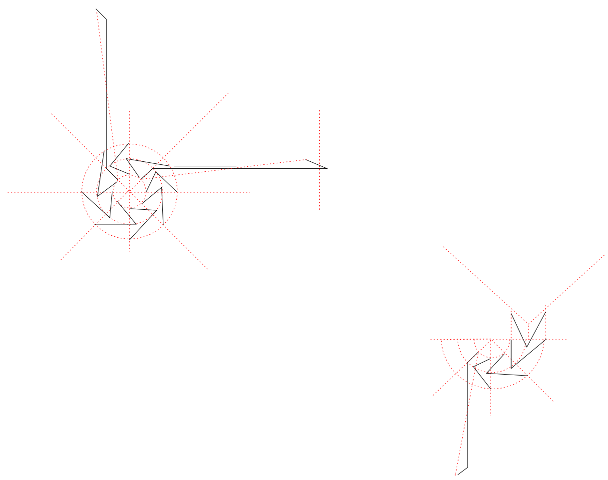

Figure 7: A modified link and a half-hinge along with a connecting link.

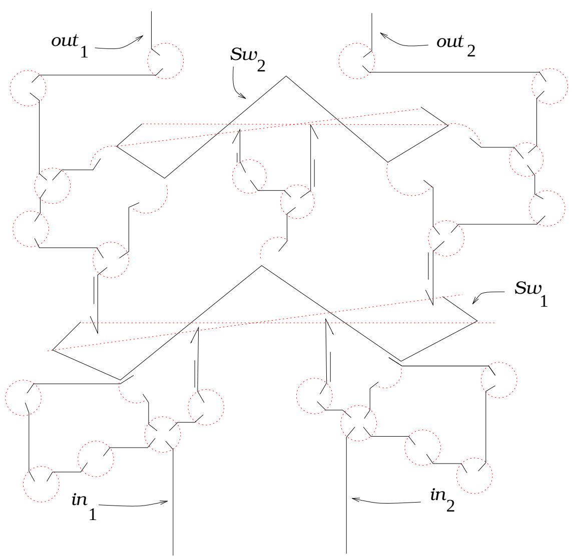

Figure 8: A switch.

2.3 Comunication Between Variable and Clause Gadgets

Hinge-link sequences travel between variable and clause nodes. If any chain in a hinge-link sequence is simplified then all chains in the sequence must be simplified. This forces clause-literal gadgets to be simplified or to remain unsimplified, depending on whether the literal represented by the gadget is TRUE or FALSE.

Suppose that V is a variable node representing the variable v. See Figure 9. If there are clauses in I containing both the literals v and $\neg v$, then there is a hinge-link sequence starting in a half-hinge and modified link at each of $\mathcal{VG}_1$ and $\mathcal{VG}_2$. If there is no clause containing a $\neg v$, then there is only one hinge-link sequence and it starts at $\mathcal{VG}_2$, and if no clause contains a v, then the sequence starts at $\mathcal{VG}_1$. For $s = 1, 2$, the sequence beginning at $\mathcal{VG}_s$ is simplified if and only if $\mathcal{VG}_s$ is simplified. These two sequences travel in a counterclockwise direction through the frame of V, with the sequence that started at $\mathcal{VG}_2$ on the outside, and branch off to travel through the tubes incident on V. The sequences begin branching off on the top edge of V. If a tube leads to a negated variable and clauses contain both negated and unnegated variables, then the sequences pass through a switch before branching off at that tube, so that the sequence that travels through the tube is the sequence that started at $\mathcal{VG}_1$. At the next tube leading to an unnegated variable, the sequences once again pass through a switch before branching off. So that if a sequence leads to a negated variable then the sequence starts at $\mathcal{VG}_1$ and if a sequence leads to an unnegated variable then the sequence starts at $\mathcal{VG}_2$. Also the sequence starting at $\mathcal{VG}_1$ is simplified if and only if the variable is FALSE and the sequence starting at $\mathcal{VG}_2$ is simplified if and only if the variable is TRUE. So that a sequence leading to a literal is simplified if and only if the literal is TRUE.

Now we consider clause nodes. See Figures 2 and 5. A binary clause node has two hinge-link sequences entering through tubes. These sequences travel in a counterclockwise direction through the frame of the node to end on one of $\mathcal{CG}_1$ or $\mathcal{CG}_2$ in a half-hat and modified link. For $s = 1, 2$, the gadget $\mathcal{CG}_s$ is simplified if and only if the sequence ending on it is simplified, and $\mathcal{CG}_2$ is simplified if and only if the sequence ending on it is simplified. Also, a sequence is simplified if and only if

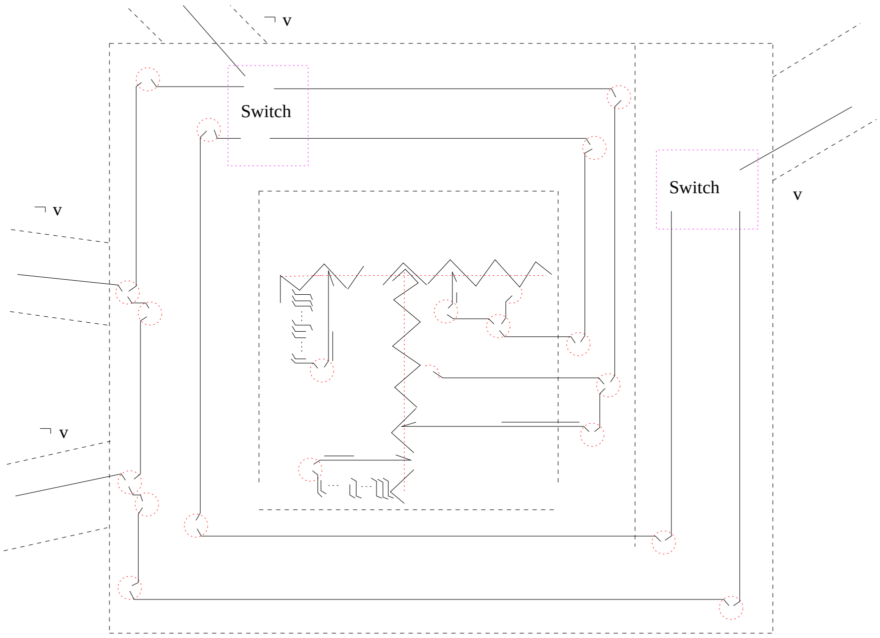

Figure 9: A variable node along with outgoing hinge-link sequences and extra segments.

the literal it ends on is TRUE, so that for $s = 1, 2$, the gadget $\mathcal{CG}_s$ is simplified if and only if the literal it represents is TRUE.

2.4 Correctness of the Reduction

There is no overlap among the variable and clause nodes since each node is centered at a grid point, each node has side length $25K^2(2\epsilon + \mu)$, and any two grid points are at least $40K^2(2\epsilon + \mu)$ apart. Also, there is no overlap between tubes since the distance between the two sides of a tube when meeting the outer square of a node is at most $2\sqrt{2}\epsilon$, any tube is centered on an edge, and the distance between any two edges when meeting a node is at least 4ϵ.

Embedding the graph on the grid can be done in time linear in K [7]. Each composite variable gadget contains $28K + 5$ segments, each clause gadget contains fourteen segments, and each switch contains a constant number of segments. For each literal occurring in a clause, there are at most fifteen hinges and switches, and at most fifteen links. Hinges and links contain a constant number of segments. The number of extra segments associated with variable gadgets is the total number of segments contained in all hinges and links. This shows that the reduction is polynomial.

Let N_s be the total number of segments in all simplified switches, N_h the total number of hinge and modified hinge segments outside of switches, N_l the total number of link segments outside of switches, and N_{ml} the total number of modified link segments outside of switches.

Suppose that there is a truth assignment for the instance I of PLANAR MAX 2-SAT which satisfies at least N clauses. If we simplify variable gadgets according to the truth assignment, then since each simplified composite variable gadget contains $14K + 4$ segments, $A_1 = |\mathcal{V}|(14K + 4)$ is the number of segments in all simplified composite variable gadgets. In simplifying the hinge-link sequences, extra segments, and switches, we obtain $A_2 = \frac{3}{2}N_h + \frac{4}{3}N_l + N_s + \frac{3}{2}N_{ml}$ segments.

Also, each satisfied composite clause gadget must be simplified to contain eight segments, and each unsatisfied composite clause must can be simplified to contain fourteen segments. So that the instance I' of NSP SUBSIMP has a simplification with $N' = A_1 + A_2 + 8N + 14(K - N)$ segments.

Now, suppose that I' has a simplification containing at most N' segments. Then all composite variable gadgets must be set to either TRUE or FALSE, since if some variable gadget is not simplified, then the total number of hinge, link, switch, and extra segments is greater than $A_2 + 2$, and each unsimplified composite variable gadget contains $28K+5$ segments. So that if some variable gadget is not simplified, then the number of segments in the simplified subdivision is greater than $A_1 + A_2 + 8K + 14K + 1 > A_1 + A_2 + 8K + 14(K - N)$ for any N with $0 \leq N \leq K$.

Since all composite variable gadgets are simplified, we have a corresponding truth assignment for I. Also, under this truth assignment, satisfied composite clause gadgets correspond to satisfied clauses and unsatisfied composite clause gadgets correspond to unsatisfied clauses.

Recall that a TRUE composite clause gadget contains eight segments, and a FALSE composite clause gadget contains fourteen segments.

Now, we have that the number of segments in simplified variable gadgets must be A_1, the number of segments in simplified extra segments, hinges, links, and switches must be A_2, and simplified clauses contain either eight or fourteen segments. So, that the number of segments in I' is $A_1 + A_2 + 8Y + 14X \leq A_1 + A_2 + 8N + 14(K - N)$, where Y is the number of satisfied composite clause gadgets and X is the number of unsatisfied composite clause gadgets. We now have that $Y \geq N$ and the corresponding truth assignment for the instance I of planar MAX 2SAT satisfies at least N clauses.

3 Conclusion

In this paper we have shown that the No Steiner Point Subdivision Simplification Problem is NP-Hard. Recently, we have also shown that this problem is MIN PB complete, obtaining $n^{\frac{1}{5}-\epsilon}$ for any $\epsilon > 0$, where n is the number of vertices in the input instance, as a lower bound for an approximation factor [4].

References

[1] K. W. Chan and Francis Y. L. Chin. Optimal multiresolution polygonal approximation. In *Computing and Combinatorics, Proc. Third Ann. Internat. Conf., COCOON'97, Shanghai, China, August 20-22 1997*, Lecture Notes Comput. Sci., Berlin, 1997. Springer-Verlag.

[2] M. de Berg, M. van Kreveld, and S. Schirra. A new approach to subdivision simplification. In *Proc. of Auto-Carto 12*, pages 79–88, 1995.

[3] David H. Douglas and Thomas K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Canadian Cartographer*, 10(2):112–122, December 1973.

[4] R. Estkowski. No steiner point subdivision simplification is hard to approximate. *in preparation.*

[5] L. J. Guibas, J. E. Hershberger, J. S. B. Mitchell, and J. S. Snoeyink. Approximating polygons and subdivisions with minimum link paths. *Internat. J. Comput. Geom. Appl.*, 3(4):383–415, December 1993.

[6] C. Jones, G. Bundy, and J. Ware. Map generalization with a triangulated data structure. *Cartography and Geographic Information Systems*, 22:317–331, 1995.

[7] W. Schnyder. Embedding planar graphs on the grid. In *Proc. 1st ACM-SIAM Sympos. Discrete Algorithms*, pages 138–148, 1990.