SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NEW YORK 14853

TECHNICAL REPORT NO. 818

September 1988

BEAUTIFICATION IS HARD

By

Esther M. Arkin [1]

[1] Partially supported by National Science Foundation grant ECSE-8857642.

Beautification is Hard

Esther M. Arkin

School of Operations Research and Industrial Engineering
Cornell University
Ithaca, NY 14853

Abstract

We describe an elementary version of the beautification problem in one dimension and show that it is NP-complete.

1. Introduction

An important problem in the design of a computer-aided design system is that of "beautifying" the user's input. Geometric information is input through a sequence of mouse clicks which are not perfectly positioned. The result is that figures are not as "neat" or "perfect" as they may have been intended. Indeed, most designed objects have a great deal of symmetry, and different objects are usually aligned. The beautification problem is to interpret user input in a way that best approximates his "intentions".

This problem has been addressed by many researchers and a common approach taken is the design of heuristics that try to maximize the "beauty" of an illustration by maximizing one particular measure or a combination of several measures. One simple and often used heuristic is the so-called "snapping" method. For example, "snap to grid" methods place a grid in the plane, and points input by the user are interpreted as (snapped to) the closest grid point to them. This method, although simple, has many drawbacks (e.g. not preserving colinearity). A more sophisticated approach was taken by Pavlidis and Van Wyk [PV]. They considered the lengths and angles between the line segments of an illustration and tried to create drawings that contain many equal-length segments, right angles, and parallel line segments. Their heuristics were designed and tested using clustering techniques. In this note we try to address the problem from the computational complexity point of view, analyzing how hard is it to beutify a drawing according to one simple measure.

We make our problem more precise by specifying that the automatic beautifier is allowed to alter the location of a point by at most a distance ϵ. (This can be thought of as the error tolerance.) Subject to this tolerance, the problem is to maximize the "beauty" of the input drawing. Several definitions of beauty are possible, leading to a variety of objective functions. Here, we focus on the problem of inputting a set of points, and we concentrate on the one-dimensional case (which must be solved in order to solve the more interesting two-dimensional case). Our definition of beauty is the (negative of the) number of distinct lengths of intervals between consecutive points. (We negate the number since we maximize beauty, but we want to minimize the number of interval lengths.) Our main result (given in section *2)* is a proof that minimizing the number of interval lengths is NP-complete. This implies that many of the problems we may want to solve in two dimensions are also NP-complete. In particular the corresponding two-dimensional problem would be defined as the number of different lengths of edges in the Delaunay triangulation, a measure suggested by Huttenlocher and Mitchell [HM].

2. The main theorem

In this section we present our main result, namely that the one-dimensional beautification problem is NP-complete. For the definition of NP-completeness see [GJ].

Problem: 1-Dimensional Beautification

Instance: Given N rational points on the line $x_1, x_2, \ldots, x_N$ such that $x_1 < x_2 < \ldots < x_N$, a rational $\epsilon > 0$ and an integer k.
Question: Do there exist rational points $y_1, y_2, \ldots, y_N$ such that $y_1 \leq y_2 \leq \ldots \leq y_N$ $|x_i - y_i| \leq \epsilon$ for all $1 \leq i \leq N$, and the number of different size intervals of the form $y_{i+1} - y_i$ is less than or equal to k? We refer to the minimum such k for which the answer is "yes" as the ***beautification number.***

Theorem *I-Dimensional Beautification* is *NP-complete.*

Proof: It is not hard to see that the problem is in NP. Simply "guess" a set of N labels (e.g., "a, b, a, **a,** *c, c,* b, b, a") that indicate which intervals will be of the same lengths, and then check the resulting system of linear equations and inequalities for feasibility.

To show completeness we reduce One-In-Three 3SAT, a problem known to be NP-complete [GJ], to the problem at hand:

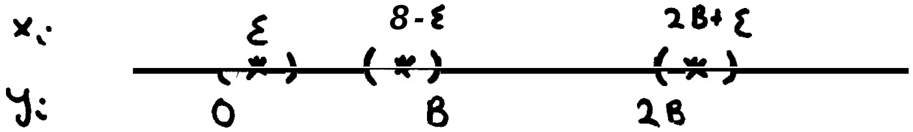
Problem: One-In-Three 3SAT

Instance: Set U of variables, collection C of clauses over U such that each clause $c \in C$ has $|c| = 3$.
Question: Is there a truth assignment for U such that each clause in C has exactly one true literal?

Let n be the number of variables in U. Set $k = 2n + 3$ and fix $0 < \epsilon < 1/3$. Next, we describe the gadgets that go into our construction. Each gadget will consist of several consecutive x_i's. For simplicity, gadgets will be described as if the first x_i is set to some arbitrary location, or else we will give the interval lengths $x_{i+1} - x_i$ instead of the location of the individual points. The gadgets will all be placed on the line one after the other so we need only shift each gadget in its entirety by the appropriate amount.

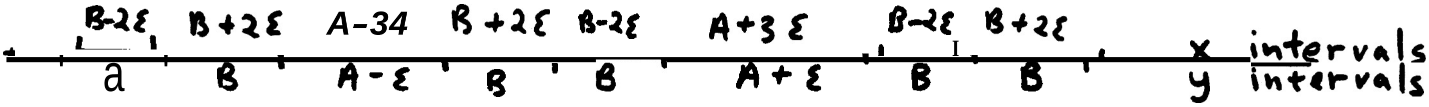
Gadget "set endpoint":

This gadget consists of 3 points x_i, x_{i+1} and x_{i+2} such that $x_i = \epsilon$, $x_{i+1} = B - \epsilon$ and $X_{i+2} = 2B + \epsilon$, where B is an integer greater than $2n$. This gadget will appear in all other gadgets and its purpose is to ensure that the endpoints of all other gadgets are "rigid". This is accomplished if we notice that the two intervals in this gadget can be of equal **y** length (meaning that $y_{i+1} - y_i = y_{i+2} - y_{i+1}$) if and only if we set $y_i = 0$, $y_{i+1} = B$ and $y_{i+2} = 2B$. Thus we can think of the endpoints x_i and x_{i+2} as setting the points y_i and y_{i+2} at a fixed position on the line. In particular, both endpoints will be shifted by E to the left, meaning that $y_i = x_i - E$ and $y_{i+2} = x_{i+2} - E$.



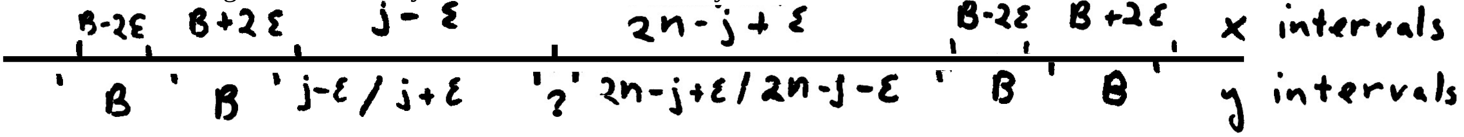
Gadget "set interval lengths A $+e$ and $A - \epsilon$":

Here A is an integer greater than $2n$ but not equal to B. This gadget consists of three set endpoint gadgets. The distance between the first and second set endpoint gadgets is A $- 3\epsilon$ (meaning that if x_i is the third point of the first set endpoint gadget and x_{i+1} is the first point of the second gadget then x_{i+1} – xi= $A - 3e$.) Similarly, the distance between the second and third set endpoint gadget is A+ 3ϵ. Note that there is no way to make the two corresponding **y** intervals between the first and second set endpoint gadgets equal the distance between the third and second ones (and clearly neither can be made equal to B, an interval length that already exists.) Thus, the best we can do so far is to have three different interval lengths. If in addition we reverse the second set endpoint gadget so that the two x_i intervals in it are $B + 2\epsilon$ and $B - 2\epsilon$ in that order (rather then the reverse order which was defined), we ensure that the three **y** interval lengths we get are B, A $+e$ and $A - E$. (In addition we will have an interval length corresponding to each variable and its negation yielding the $k = 2n + 3$ required.)

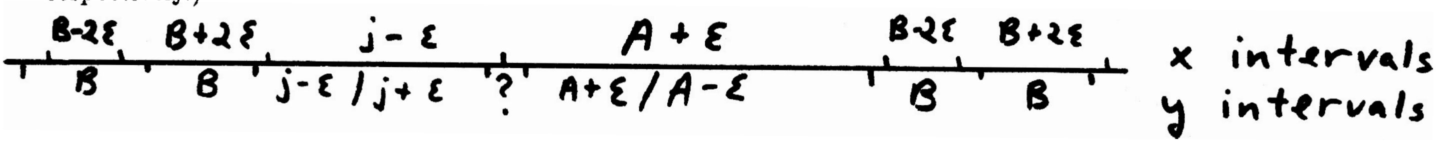


Variable gadgets:

For each variable u_j we describe two gadgets. The first will consist of two set endpoint gadgets with a distance of $2n$ between them, and one additional point between the two set endpoints at distance j $- E$ from the first (and hence $2n - j + E$ from the second). This gadget determines whether the variable is true or false. The next gadget we will describe shortly will ensure that there are only two possible placements for the **y** point corresponding to the additional **x**: One yields **y** interval lengths **j** $- E$ and $2n -$ j $+ E$, which we will interpret as variable u_j being true; and the other placement yields lengths j $+ E$ and $2n -$ j $- E$, which we will interpret as the variable u_j being false. (Recall that, because of the set endpoint gadgets, the sum of lengths of the two **y** intervals must be exactly $2n$.)

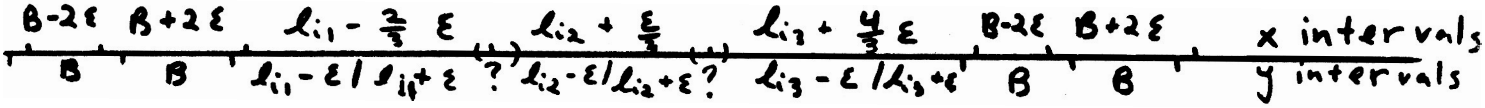


The second gadget (that ensures only two possible placements) also consists of two set endpoint gadgets with an additional point between them. The set endpoints are distance j $+$ A apart. The additional point is located at distance $j - \epsilon$ from the first set endpoint (and hence $A + \epsilon$ from the second). Recall that we already described one gadget involving an interval of length almost A. In fact we argued that the lengths of almost-A intervals are limited to $A + \epsilon$ and $A - \epsilon$. Unless we are to create more almost-A y intervals (which we can not afford to do), we are forced to have the y interval whose length is almost j be equal to $j - \epsilon$ or $j + \epsilon$. (This, in turn, will force the y interval of length almost $2n - j$ to be $2n - j + \epsilon$ or $2n - j - \epsilon$ respectively.)



Clause Gadgets:

Let C_i be the clause (l_{i1}, l_{i2}, l_{i3}) where l_{ij} is a variable or a negation of one of length almost l_{ij} (meaning that if l_{ij} is a variable u_t, its length is almost t, and if l_{ij} is the negation of a variable u_t, then its length is almost $2n - t$). The gadget corresponding to C_i consists of two set endpoint gadgets that are at distance $l_{i1} + l_{i2} + l_{i3} + \epsilon$. There are two additional x points, the first at distance $l_{i1} - (2\epsilon)/3$ from the first set endpoint gadget, and the second x point at distance $l_{i2} + \epsilon/3$ from the first x point (and hence $l_{i3} + (4\epsilon)/3$ from the second set endpoint). Recall that the x endpoints of a set endpoint gadget will determine y points that are shifted by ϵ to the left. After this shift we have created three intervals of length $l_{ij} + \epsilon/3$ each. So as not to create new y interval lengths, we are forced to have two of the literals be of length $l_{ij} + \epsilon$ and one of length $l_{ij} - \epsilon$, meaning that exactly two of the literals in the clause are false and one is true.



To complete our construction we "string" all the gadgets (two per variable, one for each clause and the "set interval lengths $A + \epsilon$ and $A - \epsilon$") along the line by making the last point of one gadget be the first point of the next. The order in which we put these gadgets does not matter. Clearly this construction can be completed in polynomial time.

As we have argued throughout, the only possible way to get a beautification number at most (in fact exactly) $2n + 3$ is to have y interval sizes B, $A + \epsilon$, $A - \epsilon$, and for each variable u_j two lengths: $j - \epsilon$ and $2n - j + \epsilon$, or $j + \epsilon$ and $2n - j - \epsilon$. In the first case we set variable u_j to be true and in the second case to be false. This assignment is well defined (i.e., no variable is assigned both true and false). If indeed there are only $2n + 3$ different length y intervals we can conclude that the gadgets corresponding to clauses did not generate any new lengths (beyond those obtained by the variable gadgets already mentioned), implying that each clause has one true literal and two false ones. This is the case because the total y length of a clause gadget must be $l_{i1} + l_{i2} + l_{i3} + \epsilon$. Thus we have shown that at most $2n + 3$ different length intervals implies a truth assignment as required by One-In-Three 3SAT. The converse is easy to see: Given a truth assignment, set the lengths of intervals corresponding to literals appropriately and the resulting beautification number will be $2n + 3$. I

3. Conclusion

It is important to note that although we have shown that the l-dimensional beautification problem is NP-complete, many simple heuristics come to mind that will probably perform quite satisfactorily in practice. In particular, the construction used to show the problem NP-complete by proving that a formula is satisfiable if and only if we can restrict the number of different length intervals to 2n + 3 (n is the number of variables) can be beautified using 2n + 5 different length intervals whether the formula is satisfiable or not!

A trivial lower bound on the beautification number is the following: Define $a_i = x_i - x_{i-1}$ and solve the problem of finding new numbers $b_1, \ldots, b_k$ minimizing k, such that for each a_i there exists a b_j for which $|a_i - b_j| \leq 2\epsilon$ (i.e., find a minimum cardinality cover of the a_i's). Clearly this problem can be solved optimally in linear time after sorting the ai's. Furthermore, k is a lower bound on the original beautification problem. If we drop the restriction that we must have $y_1 \leq y_2 \leq \ldots \leq y_N$ (i.e., we allow the y's to "cross over") then we have a trivial upper bound on the beautification number $k + n/2$. It remains to be seen what better bounds can be obtained, how simple heuristics perform in practice, and how they may extend to the more interesting 2-dimensional case.

Acknowledgment

This research was partially supported by National Science Foundation grant ECSE-8857642

Bibliography

[GJ] M. R. Garey and D. S. Johnson, ***Computers and Intractability: A Guide to the Theory of NP-Completeness,*** Freeman, 1979.

[HM] D. Huttenlocher and J. S. B. Mitchell, private communication, 1988.

[PV] T. Pavlidis and C. J. Van Wyk, "An Automatic Beautifier for Drawings and Illustrations", ***Computer Graphics*** 19(3):225-234, SIGGRAPH San Francisco, CA, July 1985.