“Shortest Path” Option for Flight Line Style (Net Bar)

Introduction

There is the recent improvement in Expert Layout Editor, the “Shortest path” option was added to Flight line styles (Net Bar). This option allows to build the shortest tree (network) connecting all selected nodes.

“Tree” means the network of vertices (nodes) connected with undirected edges (lines) which has two basic properties:

1. Any two vertices are connected with some path.
2. There are no unnecessary edges, in other words if any edge is removed then the first property will be violated.

“Shortest tree” means the tree with minimal summarized length of edges. It is also called the Minimum Spanning Tree (MST). Minimum Spanning Tree of the given set of points P on Euclidean plane is called Euclidean Minimum Spanning Tree (EMST).

Algorithm

There are a lot of algorithms for finding the Minimum Spanning Tree (one of them is well-known Prim’s algorithm, which is described below), but their common problem is bad productivity.

Most of these algorithms require at least O(n^2) time (where n is number of vertices), because the first step for these algorithms is building of the complete graph with all possible edges, where each vertex is directly connected with any other vertex. This operation takes O(n^2) time.

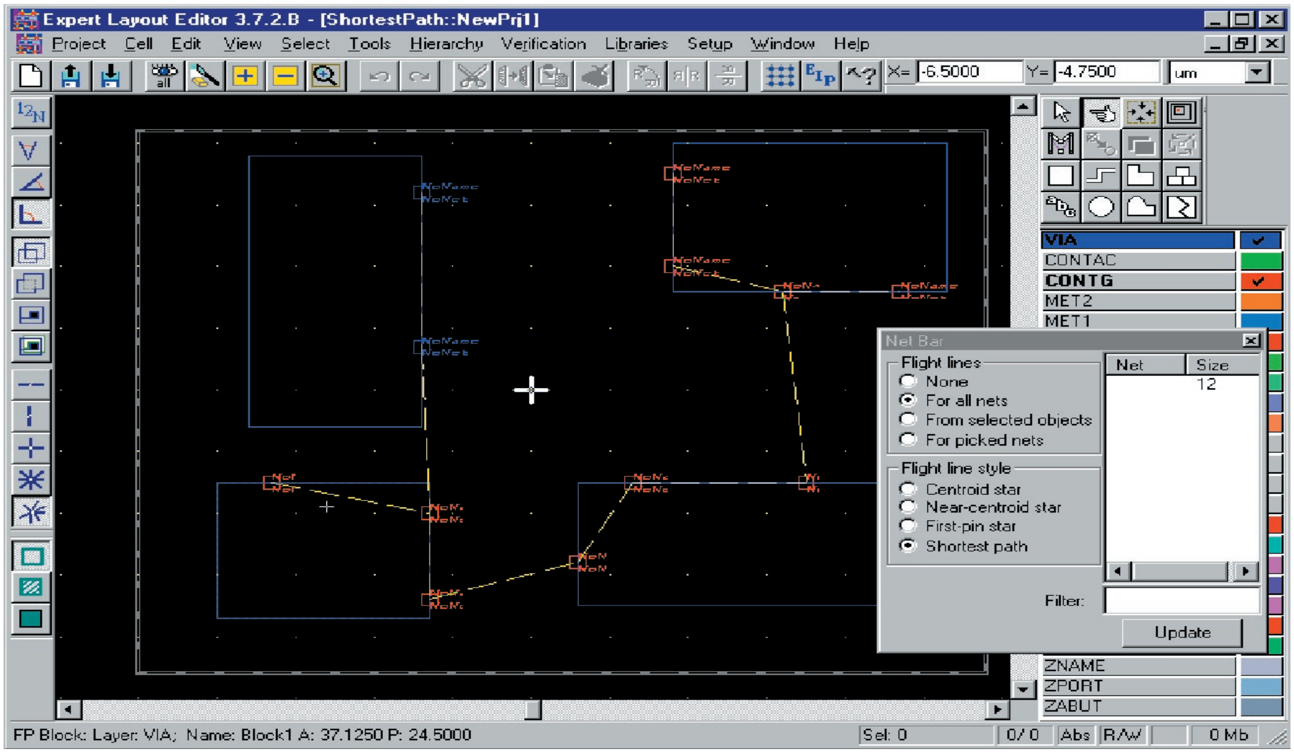
Such slow algorithms can be used with small number of vertices but are absolutely inefficient with huge number of vertices. For example, processing of 10,000 vertices will be about 1,000,000 times longer than processing of 10 vertices.

To avoid this problem the set P of points on Euclidean plane is processed in two steps.

The first step is finding the Delaunay triangulation (Figure 3) for n points on Euclidean plane using the “divide and conquer” algorithm (presented in [1]).

During second step we run the Prim’s algorithm on the Delaunay triangulation (Figure 4) to find the Minimum Spanning Tree.

The total processing time is O(n·log(n)), so processing time grows almost linearly with number of processed points.


Expert Layout Editor 3.7.2.B - [ShortestPath::NewPrj1]
Project Cell Edit View Select Tools Hierarchy Verification Libraries Setup Window Help
VIA
CONTAC
CONTG
MET2
MET1
ZNAME
ZPORT
ZABUT
Net Bar
Net
Size
Filter:
Update

X= -6.5000 Y= -4.7500 um

Flight lines
○ None
● For all nets
○ From selected objects
○ For picked nets

Flight line style
○ Centroid star
○ Near-centroid star
○ First-pin star
● Shortest path

12

FP Block: Layer: VIA; Name: Block1 A: 37.1250 P: 24.5000 Sel: 0 0/0 Abs R/W 0 Mb

Figure 1. Layout Expert Editor screenshot.

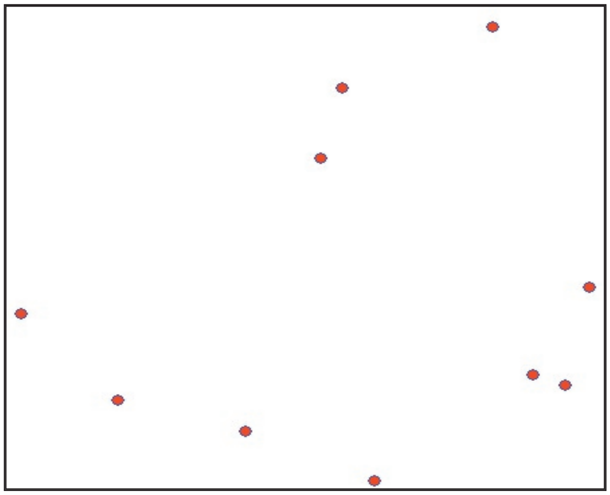


Figure 2. There are no edges.

The results of described two-steps algorithm is EMST because of well-known theorem

EMST of given set of points P on plane is a subgraph of Delaunay triangulation of P.

Delaunay Triangulation and “Divide and Conquer” Algorithm

The Delaunay triangulation is a triangulation in which every circumcircle of a triangle is an empty circle. Figure 3 shows the example of Delaunay triangulation.

- The common idea of the “divide and conquer” algorithm is to process the set of vertices recursively.
- The original set of vertices $V = \{v_1, \ldots, v_n\}$, $v_i = (x_i, y_i)$ is sorted by x and y coordinates so that $v_i < v_{i+1}$, that means $x_i \leq x_{i+1}$, and if $x_i = x_{i+1}$ then $y_i < y_{i+1}$.
- The total set of vertices V is divided into two parts $V_1 = \{v_1, \ldots, v_k\}$ and $V_2 = \{v_{k+1}, \ldots, v_n\}$, $V_1 \cup V_2 = V$ according to sort order (left part and right part).

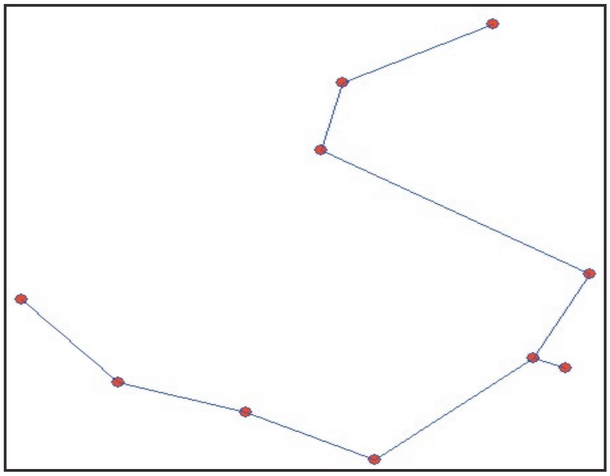


Figure 4. Step 2 – finding the Minimum Spanning Tree using Prim’s algorithm

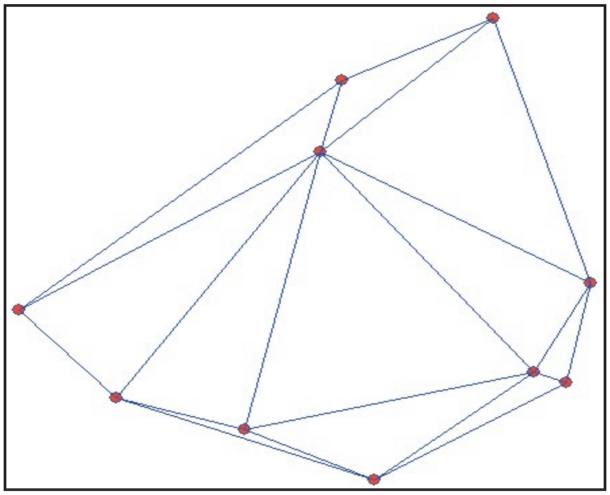


Figure 3. Step 1 – creating the Delaunay triangulation using “divide and conquer” algorithm

- Then “divide and conquer” algorithm is recursively applied to both parts (independently).
- The resulting Delaunay triangulations of two parts are joined (zipped) into one Delaunay triangulation.

Prim’s Algorithm

Prim’s algorithm processes graph G = (V, E) of n vertices $V = \{v_1, \ldots, v_n\}$ and m undirected edges $E = \{e_1, \ldots, e_m\}$ and removes unnecessary edges.

- First step is to choose an arbitrary vertex $v_1 \in V$ and to build the set of processed vertices $P = \{v_1\}$, consisting only of v_1.
- Then on every step one new vertex v_i is added to the set of processed vertices P. To choose this vertex we look for smallest edge $e = (v_j, v_k)$ connecting the processed vertices with unprocessed, $v_j \in P$, $v_k \in V \backslash P$.
- Process is repeated until there are no unprocessed vertices more.

Figure 4 shows the results of Prim’s algorithm.

References

[1] Guibas, L. and Stolfi, J., “Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams”, ACM Transactions on Graphics, Vol.4, No.2, April 1985, pages 74-123.

[2] Okabe, A.; Boots, B.; and Sugihara, K. Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. New York: Wiley, 1992.

[3] Lee, D. T. and Schachter, B. J. “Two Algorithms for Constructing a Delaunay Triangulation.” Int. J. Computer Information Sci. 9, 219-242, 1980.