A Compact Data Structure to Represent the Delaunay Triangulation

Clément Maria

Intern in GEOMETRICA,
INRIA Sophia-Antipolis Méditerranée.

TGDA Workshop (July, 8-10).

Context

- We have an implementation of the incremental algorithm for constructing Delaunay triangulations in any dimension.
- constructs step by step the 1-skeleton of the triangulation.
- runs out of RAM quickly.

Aim

- Compress the 1-skeleton's graph to reach bigger dimensions and/or triangulate more vertices.
- queries supported :
  - neighbor listing
  - addition and deletion of edges

  in order to work in parallel to the previous implementation.

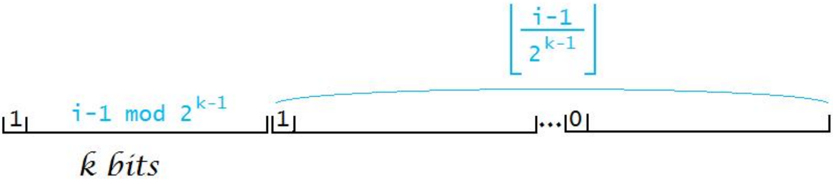
Strategy

Assuming the fact that two vertices which are close in the graph have close labels.

For a vertex v :

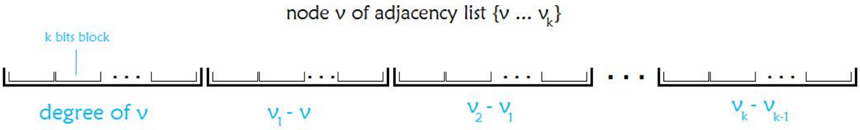
- with sorted adjacency list $\{v_1, ..., v_k\}$
- we store the successive differences $[v_1 - v \ ; \ v_2 - v_1 \ ; \ ... \ ; \ v_k - v_{k-1}]$.

- we code integer i as a sequence of k-bit blocks
- each block begins with a continue bit
- if $i \leq 2^{k-1}$:
  - continue bit $=$ zero
  - we store $i-1$ in the $k-1$ free bits of the block
- else
  - continue bit $= 1$
  - we store $(i-1)\text{mod}[2^{k-1}]$ in the current block
  - continue recursively coding $\lfloor \frac{i-1}{2^{k-1}} \rfloor$



Then, to code a vertex v :

- we code contiguously :
  - the degree of v
  - the differences $[v_1 - v \, ; \, v_2 - v_1 \, ; \, ... \, ; \, v_k - v_{k-1}]$ (with a sign bit for the first).
- We form an adjacency table by concatenating the code of the vertices in the order of their labels.

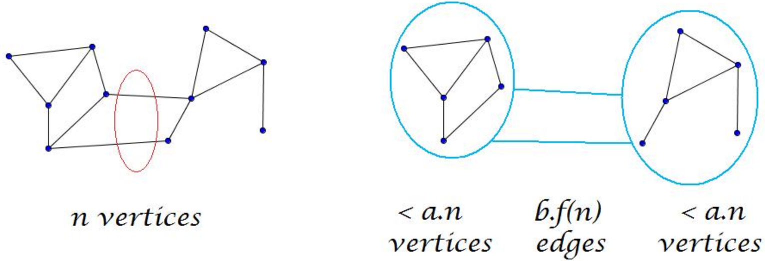


Sorting the points along a Hilbert Curve

- Already implemented in the Delaunay triangulation construction.
- Divides the space in cubes and labelizes them successively.
- Done only one time in a pre-processing phase.
- It garantees that two nodes with close labels are close in space.
- We hope that “close in the space” implies “close in the graph” for a Delaunay triangulation.

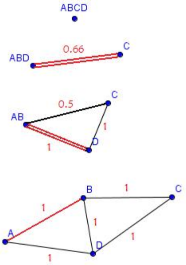
Using Edge-Separator Tree

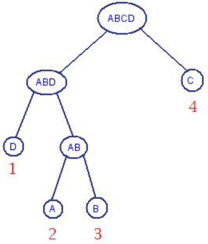
- We hope that the graph of the 1-skeleton of a Delaunay triangulation satisfies good edge-separator properties
- *i.e.* the graph may be partitionned into two subgraphs, with approximatively the same number of vertices, by deleting few edges and the two subgraphs satisfies this property.



Using Edge-Separator Tree

To separate the graph, we give a priority to edges and we merge multivertices *n* times to construct a separator tree.





$$\text{priority of edge } AB = \frac{W(AB)}{s(A).s(B)}$$

$W(AB)$ number of edges between multivertex A and multivertex B

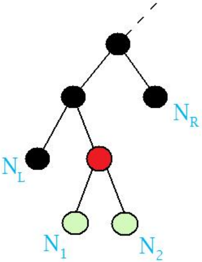
$s(A)$ number of nodes in multivertex A

Child-Flipping

For all node v,

- if N_1 and N_2 are respectively the left and the right son of v
- if N_L and N_R are respectively the left child of v's left ancestor and the right child of v's right ancestor
- if $E_{A,B}$ = number of edges between multivertices A and B

we insure that $E_{N_L,N_1} + E_{N_R,N_2} \geq E_{N_L,N_2} + E_{N_R,N_1}$.



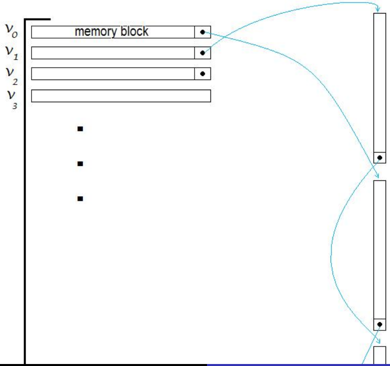
Results for the static data structure

For the triangulation of a uniform repartion of points :

- we compress 4.5 times the structure
- separator tree and Hilbert sort methods seem equivalent

The Dynamic Data Structure

- A memory block of fixed size for each vertex
- An array of blocks
- A pool of spare memory blocks
- A pointer to another block at the end of each block



The Dynamic Data Structure

We keep uncompressed the last used vertices in a cache.
To add/delete a edge, we look in the cache

- if need be, we uncompress the nodes of the edge and store them in the cache
- we treat the adjacency lists

We can treat independantly each node.

References

About the implementation of the algorithm for constructing Delaunay triangulations :
J.-D. Boissonnat, O. Devilliers and S. Hornus. *Incremental construction of the Delaunay triangulation and the Delaunay graph in medium dimension.*

About k-bit code and separator tree :
D. Blandford, G. Belloch and I. Kash. *Compact representations of separable graphs.*
D. Blandford, G. Belloch and I. Kash. *An experimental analysis of a compact graph representation.*