A Self-Stabilizing and Local Delaunay Graph Construction

Riko Jacob$^1$, Stephan Ritscher$^1$, Christian Scheideler$^2$, Stefan Schmid$^2$

$^1$ Institut für Informatik, Technische Universität München
jacob@in.tum.de, ritsches@in.tum.de

$^2$ Department of Computer Science, University of Paderborn
scheideler@upb.de, schmiste@mail.upb.de

18. December 2009
## Contents
Motivation

Peer-to-Peer Network

- Decentralized
- Recover from faults
  ⇒ Self-stabilizing
- Nice geometric properties
Motivation

Peer-to-Peer Network

- Decentralized
- Recover from faults

⇒ Self-stabilizing

- Nice geometric properties

Related Work

- 1-D graphs: Line, ring, and skip graphs
- 2-D graphs: Delaunay graph in wireless systems
Delaunay Graphs

Assume nodes in general position.

**Definition (Delaunay Graph $G_D$)**

$\{u, v\} \in G_D$ iff some circle $C$ contains $u, v$ but no other nodes.
Delaunay Graphs

Assume nodes in general position.

Definition (Delaunay Graph $G_D$)

$\{u, v\} \in G_D$ iff some circle $C$ contains $u, v$ but no other nodes.

Properties

- Triangulation
- Geometric spanner
- Allows greedy routing
Model (1)

- Only direct neighbors known
Model (1)

- Only direct neighbors known

![Diagram of a graph with nodes a, b, c, d, e, f, g, and h, with directed edges showing only direct neighbors known.]

Jacob, Ritscher, Scheideler, Schmid: Self-Stabilizing Delaunay Graphs
Model (2)

- Updates in rounds

Initial graph
Model (2)

- Updates in rounds

Initial graph

Local updates
Model (2)

- Updates in rounds

```
Initial graph                  Local updates                  Update graph
```

Jacob, Ritscher, Scheideler, Schmid: Self-Stabilizing Delaunay Graphs
Problem Statement

Situation

- Start with weakly connected graph
- Nodes in general position
Problem Statement

Situation

- Start with weakly connected graph
- Nodes in general position

Task

- Converge to Delaunay graph
- Little resources (number of edges)
- Monotonous behaviour
Trivial Strategy

1. If local incorrectness appears, calculate complete graph: $O(n)$
2. Compute Delaunay graph.
Trivial Strategy

1. If local incorrectness appears, calculate complete graph: $O(n)$
2. Compute Delaunay graph.

Criticism

- Too many edges
- Not monotonous
Local Update

Compute Delaunay graph of neighbors of $v$. 

```
h    a
   / \
  v   b
 /    \
 d    c
 \
 e
 g

f
```
Local Update

Compute Delaunay graph of neighbors of $v$. 

![Diagram of a Delaunay graph with nodes a, v, c, d, e, and g. The node v is highlighted in red.]
Rule I (1)

Select edges to local neighbors of $v$. 
Rule 1 (2)

Select circular edges around $v$. 

![Diagram showing a graph with nodes and edges]
Rule II

Connect local non-neighbors (nearest neighbor strategy).
”Temporary edges”
Main Theorem

Theorem
Using this algorithm, any weakly connected graph $G$ converges in $O(n^3)$ rounds to the Delaunay graph.

If $G$ is a super graph of the Delaunay graph, it converges in $O(n)$ rounds.
Proof Idea

- Delaunay edges don't disappear.
- Temporary edges (Rule II) shorten in each step.
- Define potential $\phi(G)$ which
  - has only $O(n^2)$ values,
  - decreases at least every $n$ steps, and
  - is zero if and only if $G$ is a super graph of the Delaunay graph.
- Superfluous edges disappear in $n - 1$ rounds.

$\Rightarrow$ Convergence in $O(n^3)$ rounds
Example Proof

Lemma

The only 'stable' graph is the Delaunay graph.

Proof.

- 'Stable' means 'locally triangulated'.
- 'Locally triangulated' implies planar.
- Planar and 'stable' implies triangulated.
- From any triangulation, edge flips lead to the Delaunay graph.

⇒ 'Stable' implies Delaunay graph.
Experiments

Setting

- Random graphs
- Up to 500 nodes
- Various topologies:
  - Random tree
  - Maximum spanning tree
  - Clique
  - Slightly disturbed circle
Experiments

Setting

- Random graphs
- Up to 500 nodes
- Various topologies:
  - Random tree
  - Maximum spanning tree
  - Clique
  - Slightly disturbed circle

Measurements

- Number of rounds
- Number of edges
Statistics: Runtime

The graph shows the runtime (in rounds) for different methods as a function of the number of nodes. The methods compared are Circle, Clique, Max Tree, and Rand Tree. The x-axis represents the number of nodes, ranging from 0 to 500, and the y-axis represents the number of rounds, ranging from 0 to 40.

- **Circle**: The runtime increases gradually with the number of nodes.
- **Clique**: The runtime increases at a moderate rate.
- **Max Tree**: The runtime increases at a faster rate compared to the other methods.
- **Rand Tree**: The runtime increases slowly at first, then increases more sharply as the number of nodes grows.

This chart helps to visualize how each method performs under varying node densities.
Statistics: Number of edges

![Graph showing the number of edges over rounds for different algorithms: Circle, Clique, Max Tree, and Rand Tree.](image-url)
Results

- Run time much better than analysis
- Number of edges can rise in first rounds, but not too much
- Results stable for randomly distributed nodes
- High variance for CIRCLE topology
Conclusion

- Self-stabilizing Delaunay graph construction
- Worst case time bound: $O(n^3)$ rounds
- Experiments on random graphs:
  - Less than $n$ rounds
  - Moderate edge growth
Open Problems

- Better bound on number of rounds
- Bounds for edge growth
- Construct worst case examples
- Alternative strategies (Rule II)
- Resolve round scheduling
Thank you!