Distributed Fine-Grained Node Localization in Ad-Hoc Networks

A Scalable Location Service for Geographic Ad Hoc Routing

Presented by An Nguyen
Distributed Fine-Grained Node Localization in Ad-Hoc Networks

By Andreas Savvides and Mani Srivastava
UCLA
Problem Setting
Simple idea: Atomic Multilateration

The position of a node can be determined when 3 beacons are within its range.
Slightly more complicated: Iterative Multilateration

Once a node’s position is known, it becomes a beacon
Main Result: Collaborative Multilateration

- Works for more general settings
- 3 phases
  - Formation of collaborative subtrees
  - Computation of initial estimates
  - Position refinement
Phase #1: Formation of collaborative subtrees

Goal:
- Well-determined or over-determined system of equations
- Facilitate distributed computation model

Approach: add nodes one by one
What nodes to add?

Condition 1: An unknown node that is connected to 3 nodes that are beacons or have tentatively unique position
What nodes to add?

- Condition 2: An unknown node uses at least one reference point that is not collinear with the rest of its reference points
What nodes to add?

- Condition 3: Each node has at least one link that connects to a different node from the nodes used as references by the other nodes.
Phase #2: Computation of initial estimates

- Find bounding box for each unknown node
- Set initial estimate of the unknown node as the center of its bounding box
Phase #3: Position Refinement (Centralized)

- Minimize the sum of edge error squares
- Use Kalman Filters

\[
\begin{align*}
    f_{2,3} &= R_{2,3} - \sqrt{(x_2 - e_{x_3})^2 + (y_2 - e_{y_3})^2} \\
    f_{3,5} &= R_{3,5} - \sqrt{(e_{x_3} - x_5)^2 + (e_{y_3} - y_5)^2} \\
    f_{4,3} &= R_{4,3} - \sqrt{(e_{x_4} - e_{x_3})^2 + (e_{y_4} - e_{y_3})^2} \\
    f_{4,5} &= R_{4,5} - \sqrt{(e_{x_4} - x_5)^2 + (e_{y_4} + y_5)^2} \\
    f_{4,1} &= R_{4,1} - \sqrt{(e_{x_4} - x_1)^2 + (e_{y_4} - y_1)^2}
\end{align*}
\]

\[
F(x_3, y_3, x_4, y_4) = \min \sum f_{i,xj}^2
\]
Phase #3: Position Refinement (Distributed)

- Repeatedly estimate node position using estimated positions of neighbors
- Yield approximately the same result as centralized approach
Experimental Results (1)

Different between distributed and centralized estimates
Experimental Results (2)

Cost of estimating positions
Experimental Results (3)

Localization accuracy
Experimental Results (4)

Communication cost

Convergence latency
End of 1\textsuperscript{st} paper
A Scalable Location Service for Geographic Ad Hoc Routing

By Jinyang Li, John Jannotti, Douglas De Couto, David Karger, Robert Morris, MIT
Problem Setting

- Geographic forwarding
- Each node knows its position
- Location service: given an ID, find the position of a node with that ID
Constraints

- No node should be a bottleneck.
- Work should be spread evenly.
- Failure of a node should not affect much the location service.
- Queries for nearby nodes should be local.
- Low storage and communication.
GLS Idea (1)

Partition the world
GLS Idea (2)

A node selects location servers “close” to itself
Location servers of a node are well sampled
## GLS Query

<table>
<thead>
<tr>
<th>70</th>
<th>1,5,6,10,12</th>
<th>19,35,37,45</th>
</tr>
</thead>
<tbody>
<tr>
<td>82</td>
<td>38</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>39</td>
<td>45</td>
</tr>
<tr>
<td>63</td>
<td>1,5,16,37,62</td>
<td>1,2,16,37,62</td>
</tr>
<tr>
<td>90</td>
<td>70,90,91</td>
<td>70,90,91</td>
</tr>
<tr>
<td></td>
<td>32,35</td>
<td>32,35</td>
</tr>
<tr>
<td>61</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>62</td>
<td>62,91,98</td>
<td>62,91,98</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>87</td>
<td>2,17,23,63</td>
<td>2,17,23,63</td>
</tr>
<tr>
<td></td>
<td>31,32,62</td>
<td>31,32,62</td>
</tr>
<tr>
<td>98</td>
<td>63</td>
<td>63</td>
</tr>
<tr>
<td></td>
<td>41</td>
<td>41</td>
</tr>
<tr>
<td>81</td>
<td>31,32,81,87</td>
<td>1,2,5,21,76</td>
</tr>
<tr>
<td></td>
<td>90,91</td>
<td>84,87,90,91</td>
</tr>
<tr>
<td>98</td>
<td>61</td>
<td>61</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>31</td>
<td>2,12,14,17</td>
<td>2,12,14,17</td>
</tr>
<tr>
<td></td>
<td>23,26,28,32</td>
<td>26,31,32,35</td>
</tr>
<tr>
<td>81</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>43</td>
<td>43</td>
</tr>
<tr>
<td>98</td>
<td>A: 76</td>
<td>A: 76</td>
</tr>
<tr>
<td></td>
<td>84</td>
<td>84</td>
</tr>
</tbody>
</table>
GLS is nice...

- No node is a bottleneck,
- Work is spread evenly
- Failure of a node does not affect much the location service
- Queries for nearby nodes is local
- Low storage and communication
Dealing with motion

- Update its location server from time to time
  - Higher level location server are updated less frequently
Experimental Results (1)

Query success rate vs. Number of nodes

Number of packets passing through a node vs. Number of nodes
Experimental Results (2)

Query path vs communication path
Experimental Results (3)

Storage per node
Experimental Results (4)

Query success rate vs node speed
Experimental Results (5)

Query success rate vs node failure rate
Experimental Results (6)

Delivery rate

GLS + Data traffic

Packets per node
Summary

- Localization
  - Distributed
  - Accurate

- Location Service for Geographic Forwarding
  - Local
  - Balanced
  - No bottle neck nodes
  - Handle node failures gracefully
  - Low storage/bandwidth requirement