A Localized Algorithm for Bi-Connectivity of Connected Mobile Robots

Ad Hoc Networks - Seminar, WS 08/09

Freiburg, 16. February 2009

Computer Networks and Telematics
University of Freiburg

Speaker: Alexander Schätzle
Organizer: Prof. Dr. Christian Schindelhauer
Introduction

- Multi-robot systems have become more and more attractive in the past few years due to the significant advancements in robotics technology.

- Generally wireless ad hoc networks are used for communication in such systems.

- But most of the existing algorithms are only suitable for robots with very low or even no failure rates!

- This is not very practical because robots are susceptible to failures!

Conclusion

In a fault-tolerant network there should be at least two node-disjoint communication paths between each pair of robots in order to handle communication faults.
A network is **bi-connected** if there exist two node-disjoint paths between any pair of nodes in the network.

**Conclusion:** Networks is still connected if one node fails!
A node is called a **critical node** if the network is disconnected without the node.

**Conclusion:** There are no critical nodes in bi-connected networks!
Problem Definition

- Communication links in mobile Networks can easily fail!
  (e.g. hardware damage, energy depletion, harsh environments, malicious attacks)

- There should be at least two node-disjoint paths between any two nodes

- Network should be bi-connected

**Task:** Given a connected but not bi-connected network, move the robots such that the network becomes bi-connected

**Objective:** Minimize total movement of robots
Lokal vs. Global

- so far only **globalized** Algorithm exists
- at least one node has to know the entire topology of the Network
  Applicable only for small size Networks

- **localized** Algorithm is executed on each node of the Network
- uses only $p$-hop neighbor information
  more practical for large size Networks
p-Hop Neighborhood

\[ p = 1 \]
p-Hop Neighborhood

\[ p = 2 \]
Globalized Algorithm

- Divides the network into bi-connected blocks
- Every node knows the entire topology of the network
- Blocks are iteratively merged to form a single bi-connected block
Localized Movement Control Algorithm

- First localized movement control algorithm to achieve bi-connectivity
- Executed at each node iteratively (every iteration consists of two phases)
- Significant improvement when compared to the globalized algorithm

Assumptions:
- All nodes have a common communication range $r$
- Each node knows its $p$-hop neighborhood (HELLO messages)
- Network is connected but not bi-connected

Drawbacks:
- Does not guarantee bi-connectivity
- May stop at connected but not bi-connected stage
- May even partition the network
- Can cause coverage holes
Phase 1 - Initialization

- Each node checks whether it is a \( p \)-hop critical node

**Definition**

A node is called a \( p \)-hop critical node if and only if its \( p \)-hop subgraph is disconnected without the node

- Every \( p \)-hop critical node broadcasts a critical announcement packet to all its direct neighbors

**Remarks:**

- A \( p \)-hop critical node is not necessarily globally critical but every globally critical node is a \( p \)-hop critical node for any \( p \)

- Experiments showed that over 80% of locally estimated critical nodes are indeed globally critical
Phase 1 - Initialization

- Critical nodes should **not be moved** because breaking some current links of a critical node can disconnect the network.

- So the basic idea is to move only **non-critical nodes** such that critical nodes become non-critical.
Phase 1 - Initialization

- Critical nodes should **not be moved** because breaking some current links of a critical node can disconnect the network.

- So the basic idea is to move only **non-critical nodes** such that critical nodes become non-critical.
Phase 2 – Node Movement

- After the initialization phase a $p$-hop critical node knows which of his neighbors are also $p$-hop critical nodes!

- Critical nodes try to make their neighborhood bi-connected by moving some of the nodes in their neighborhood which are non-critical

3 Cases

1. There are no critical neighbors
2. There exists exactly one critical neighbor
3. There exist two or more critical neighbors
2.1 - No critical neighbor

- Select two neighbors from disjoint sets of the $p$-hop subgraph and move them towards each other until they become neighbors.

- Every node should move $(d-r)/2$ when $d$ is the distance between them.

- To minimize the movement choose the pair of nodes with minimal distance.

After movement any node that loses a neighbor or finds a new neighbor broadcasts a topology update packet.

A node which receives a topology update packet updates its $p$-hop subgraph for the next iteration of the algorithm.
2.1 - No critical neighbor (Example)

- Nodes 4 and 8 have minimal distance
2.1 - No critical neighbor (Example)

- some connections may get lost
- perhaps new critical nodes are created
2.2 – Exactly one critical neighbor

- Two adjacent critical nodes (suppose A and B)
- Node with larger ID leads the movement control (suppose A)
  --> Node IDs are used to assign priorities
- A selects a non-critical neighbor with minimal distance to B and let it move towards B (from the disjoint set not including B)
- Selected neighbor should move $d-r$ when $d$ is the distance between them

After movement any node that loses a neighbor or finds a new neighbor broadcasts a *topology update packet*

A node which receives a *topology update packet* updates its $p$-hop subgraph for the next iteration of the algorithm
2.2 – Exactly one critical neighbor (Example)

- suppose node A has larger ID than node B
2.2 – Exactly one critical neighbor (Example)

- some connections may get lost
- perhaps new critical nodes are created
2.3 – Two ore more critical neighbors

Definitions

- A critical node is available if it has non-critical neighbors and non-available otherwise.
- A critical node is a critical head if and only if:
  - It is available and its ID is larger than the ID of any available critical neighbor.
  - Or it has no available critical neighbors.

- Basic idea is to use a pair wise merging strategy.
- Each critical head selects the available critical neighbor with largest ID it to pair with (or non-available if there is no available critical neighbor).
- Then for each pair the movement control algorithm for case 2 is called.
2.3 – Two or more critical neighbors (Example)

- Nodes 1, 5, 6 are critical heads
- Resulting pairs are (1,3), (5,4) and (6,4)
Example – 1. Iteration

- Nodes 0, 4, 9 are **critical nodes**
- For every critical node **case 1** holds

[A Localized Algorithm for Bi-Connectivity of Connected Mobile Robots, p. 7]
Example – 2. Iteration

- Nodes 0 and 5 are critical nodes
- For these nodes case 2 holds

[A Localized Algorithm for Bi-Connectivity of Connected Mobile Robots, p. 7]
Example – 3. Iteration

- Node 0 is the only critical node
- For this node case 1 holds

[A Localized Algorithm for Bi-Connectivity of Connected Mobile Robots, p. 7]
Example – Termination

- There is no **critical node** left

   ![Diagram](A Localized Algorithm for Bi-Connectivity of Connected Mobile Robots, p. 7)
Simulation Environment

- Varying number of nodes (from n=10 to n=100)

- Sensor field size scaled according to number of nodes
  (300 m² for n=10 up to 3000 m² for n=100)

- Network density $d \approx 10$ (i.e. an average of 10 neighbors per node)

- Communication range $r = 10$ m

- Various values of $p$

- Networks generated by random placement
  (100 different networks for each parameter setting)
Simulation Results

- Value of p affects the performance to a great extent!
  (if p is small many globally non-critical nodes are detected as p-hop critical)

![Graph showing the effect of p-value on distance traveled](image-url)
Comparison with globalized algorithm

Average Case Comparison

- Our Localized Algorithm
- Globalized Algorithm

Total Distance Traveled vs. Number of nodes

- $p = 3$

[A Localized Algorithm for Bi-Connectivity of Connected Mobile Robots, p. 10]
Performance on sparse networks

- On networks with smaller density (i.e. $d < 10$) the algorithm is not always successful!
Critical view on the Simulation Results

- Value of $p$ was set to 3 for most of the simulations
  - For $d \approx 10$ and $n = 100$ it can be expected that most of the network topology is within the 3-hop neighborhood of a node!
  - For smaller values of $d$ the algorithm was not always successful! (network not bi-connected or even disconnected)
  - What if $d \ll 10$ or $n \gg 100$?

Critical view on the algorithm

- For small values of $p$ the success rate of the algorithm sinks
  - Not applicable for small networks or networks with small density
  - Density of 10 is not very realistic for current applications of mobile robots

- Algorithm can cause "coverage holes" in the considered network area
  - Especially worse for sensor networks!
Thank you for your attention!