Self-Organizing and Self-Stabilizing Sensor Network

Students: Amir Menczel, Radmila Fishel
Advisors: Shlomi Dolev, Armin Shmilovici
Outline

- Problem Domain
- System Architecture
- System Class Diagram
- Solution
- GUI
- Testing
- Future Tasks
Outline

- Problem Domain
- System Architecture
- System Class Diagram
- Solution
- GUI
- Testing
- Future Tasks
Sensor networks are used to sample the environment for sensory information (E.g. temperature) and propagate this data to a central point.

Problem Domain

- Sensor networks are used to sample the environment for sensory information (E.g. temperature) and propagate this data to a central point.
Problem Domain (cont'd)

• Wireless sensors have a limited energy that keeps them working.
• The only power supply of outdoor sensors is solar energy.
• Efficient algorithms must be developed to save as much energy as possible.
Outline

• Problem Domain
• **System Architecture**
• System Class Diagram
• Solution
• GUI
• Testing
• Future Tasks
System Architecture

Graphical User Interface

SHAWN
Self-Stabilizing and Self-Organizing Algorithms for Sensor Network
Outline

- Problem Domain
- System Architecture
- **System Class Diagram**
- Solution
- GUI
- Testing
- Future Tasks
NetworkData <<singleton>>
- numberOfSensors : int
- numberOfIterations : int
- parameterX : int
- toAnimate : bool

NetworkFileHandler
- fileLoaded : string
+ saveNetwork(in networkData : NetworkData) : void
+ loadNetwork() : NetworkData

ShawnCommunicator
+ startSimulation(in confName : string) : void
+ stopSimulation() : void
+ addSensors(in numOfSensors : int) : void
+ removeSensors(in numOfSensors : int) : void
+ saveNetwork(in filename : string) : File
+ loadNetwork(in filename : string) : File

RectangleNetworkData
- height : int
- width : int

CircleNetworkData
- radius : int

NetworkController
+ runNewSimulation() : void
+ runExistingSimulation() : void

GUI
- gui manages

ConfigurationCreator
+ createConfigurationFile(in networkData : NetworkData) : File
Outline

- Problem Domain
- System Architecture
- System Class Diagram
- **Solution**
  - GUI
  - Testing
- Future Tasks
Proposed solution
Leader Election

X = 2
Proposed solution
Leader Election

No other leaders, all is O.K
Proposed solution
Leader Election
A close leader exists, relinquish leadership.
Proposed solution
Leader Election

processor
leader
link

X = 2
No leaders can be found, declare candidacy and take a new snapshot - $c'$. 
Proposed solution
Leader Election

- processor
- leader
- link
- candidate

\[ X = 2 \]
Proposed solution
Leader Election

If no leaders or candidates were found in c’, declare leadership.
Proposed solution
Leader Election

If a leader/candidate was found, fight it for domination.
Proposed solution
Leader Election

Predicates:
leader(C_p) :=
\[ \exists q \in C_p \mid q \neq p \land \text{leader}(q) \]

1 \((\text{leader}_p \oplus \text{leader}(C_p)) = \text{true}:\)
   /* do nothing (stable). */

2 \(\text{leader}_p = \text{false} \land \text{leader}(C_p) = \text{false}:\)
3 \(\text{rtp}_p \leftarrow \text{random}()\)
4 \(\text{candidate}_p \leftarrow \text{true}\)
5 \(C'_p \leftarrow \text{new snapshot}\)
6 \(\textbf{if} \ \text{leader}(C'_p) = \text{true} \ \textbf{then}\)
7     \(\text{candidate}_p \leftarrow \text{false}\)
8     \(\text{leader}_p \leftarrow \text{false}\)
9 \(\textbf{else if} \ \forall q \in C'_p \ \text{candidate}_q = \text{true} \rightarrow\)
10     \((\langle \text{rtp}_q, \text{id}_q \rangle < \langle \text{rtp}_p, \text{id}_p \rangle) \ \textbf{then}\)
11     \(\text{leader}_p \leftarrow \text{true}\)
12 \(\textbf{else}\)
13     \(\text{candidate}_p \leftarrow \text{false}\)
14     \(\text{leader}_p \leftarrow \text{false}\)
15 \(\textbf{end}\)

16 \((\text{leader}_p = \text{true} \land \text{leader}(C_p) = \text{true}):\)
17 \(\text{candidate}_p \leftarrow \text{false}\)
18 \(\text{leader}_p \leftarrow \text{false}\)
Proposed solution
Cluster Partitioning

- The entire network is divided into smaller clusters of random size.
- Every cluster contains a group of sensors and one leader.
Proposed solution
Hierarchy Construction

• Layer 1
  – Original network is divided into clusters.
Proposed solution
Hierarchy Construction (cont'd)

• Layer 2 – layer $\log(n)$
  – Leaders of previous layer form clusters of their own.
  – Every cluster has its leader.
Proposed solution
Topology Changes

• Adding / Removing sensors is possible.
• Every sensor continuously executes the **snapshot algorithm** in order to gain information about its environment.
• Every sensor can change its routing table according to new information received.
Outline

- Problem Domain
- System Architecture
- System Class Diagram
- Solution
- **GUI**
- Testing
- Future Tasks
GUI

Network Example
GUI

Network Example

How many sensors to add?
GUI

Please choose the type of data you want:

Data

- Animation
- Pictures

Finish
Outline

• Problem Domain
• System Architecture
• System Class Diagram
• Solution
• GUI
• **Testing**
• Future Tasks
Testing

- Different topologies:
  - Constructing a small/medium/large network:
    - Sensors will be randomly scattered
    - Sensors will form a circle/rectangle
  - For each type of Network:
    - Adding/removing sensors
    - Different $X$ parameters
      - The optimal $X$ parameter will be chosen for maximum performance of the algorithm.
Outline

- Problem Domain
- System Architecture
- System Class Diagram
- Solution
- GUI
- Testing
- **Future Tasks**
## Future Tasks

<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Topology Construction</td>
<td>12/04/2010</td>
<td>21/04/2010</td>
<td>10d</td>
</tr>
<tr>
<td>2</td>
<td>Find Leader (layer 1)</td>
<td>22/04/2010</td>
<td>03/05/2010</td>
<td>12d</td>
</tr>
<tr>
<td>3</td>
<td>Construct Hierarchy</td>
<td>04/05/2010</td>
<td>20/05/2010</td>
<td>17d</td>
</tr>
<tr>
<td>4</td>
<td>Interface between c# and SHAWN</td>
<td>21/05/2010</td>
<td>28/05/2010</td>
<td>8d</td>
</tr>
<tr>
<td>5</td>
<td>Additional Required Documents</td>
<td>12/04/2010</td>
<td>01/06/2010</td>
<td>51d</td>
</tr>
<tr>
<td>6</td>
<td>Windows GUI</td>
<td>29/05/2010</td>
<td>03/06/2010</td>
<td>6d</td>
</tr>
<tr>
<td>7</td>
<td>Testing</td>
<td>03/06/2010</td>
<td>30/06/2010</td>
<td>28d</td>
</tr>
</tbody>
</table>