CS603 Project: Topology Discovery in Wireless Sensor Networks

Presented by Mark Terwilliger
April 8, 2004

Topology Discovery

- **The problem**: Find the location of *all* the nodes in a system given the location of a small subset of the nodes
- The nodes that are aware of their location will be referred to as **anchor nodes**
- Anchor nodes are either physically placed in a known position or are equipped with a positioning technology such as GPS

Why is topology discovery an important problem to solve?
- WSN protocols and applications simply assume that all nodes in the system are location-aware
- If a sensor is reporting a critical event or data, we must know the location of that sensor
- If a WSN is using a geographical routing technique, all of the nodes must be aware of their location
Problems with existing algorithms

- Most existing topology discovery algorithms consist of two phases:
  1. Estimate Position
  2. Iterative Refinement
- The second phase consists of approximately 25 iterations of every node sending its location to all of its neighbors
- This process must be repeated when nodes move or other changes to the topology occur

Project Goal

- **My question**: Is it possible to develop an accurate positioning algorithm without expending all of this energy?
- **The plan**: Compare the accuracy and energy consumption of existing and new topology discovery algorithms
A new strategy: Use Genetic Algorithms

- A base station issues a TOPOLOGY_DISCOVERY request that floods the network
- All nodes use a ranging technique to estimate the distance to each of their neighbors
- It is assumed that errors will occur during the ranging phase
- All nodes forward their neighbor/distance pairs to the base station
- Base station uses a genetic algorithm to compute all locations and finally sends each node its location

How do Genetic Algorithms work?

Generate $N_1$ random potential solutions (population of chromosomes)

While not done {
  Perform mutations on each of the $N_1$ chromosomes
  Evaluate the fitness level of each chromosome
  Sort the $2 \times N_1$ chromosomes according to fitness level
  The $N_1$ chromosomes with the highest fitness level survive
}

- Chromosomes may be chosen for mutation based on fitness level
- A crossover operator combines the chromosomes of two parents
- $B \times N_1$ new chromosomes may be generated during each iteration
- Genetic algorithms may terminate by: (1) Fixed number of generations, (2) Time limit exceeded, (3) Given fitness level achieved, or (4) Convergence

The Program

- The program was written in Java so that it could integrate easily with TinyOS and Berkeley motes
- It also serves as a simulator in which the user enters the ranging error as an input parameter
- Assumptions:
  - Signals are omni directional and symmetric
  - All nodes have the same radio transmission range
  - All nodes have at least one neighbor
Scenario #1
- A total of 20 nodes, 5 of which are anchors
- Distributed in 50 x 80 square foot rectangular area
- Nodes have 25-foot transmission range
- GA parameters
  - Population size = 1000
  - Number of generations = 300
  - Fitness function computes distance errors based on position
  - Mutation operators:
    1. Two random nodes switch x coordinates
    2. Two random nodes switch y coordinates
    3. A random node moves between 0..100 in the x direction
    4. A random node moves between 0..100 in the y direction

Software Interface - Simulation

Current and future work
- Currently implementing other algorithms
- Need to compare accuracy and energy consumption of algorithms
  - Variables in comparison: (1) Number of nodes, (2) Number of anchor nodes, (3) Density of nodes, (4) Ranging error
- Based on results, either choose an algorithm or develop/modify an algorithm to put into practice
- Finally, will implement topology discovery on Berkeley motes