Reliability-based Multihop Routing for Sensor Networks

Alec Woo
David Culler

NEST Winter Retreat
January 16th, 2003
Problem Statement

• Design an ad hoc routing protocol
  – creates a many-to-one spanning tree topology
  – self-organizing through local operations
  – simple and efficient
  – explore quality/reliable routing paths to the base station
Hypothesis

- Shortest path routing can yield unreliable paths

- Build reliability statistics of each neighbor through link estimations
  - Even coarse estimations are better than none

- End-to-end reliability guarantee is unlikely
  - Even if it exists, local actions are building blocks
    - Limited retransmissions per hop
    - Explore reliable paths to route packets

- $E(\text{transmissions})$ along a path captures both “distance” and “reliability” for routing
  - ARPANET available bandwidth metric leads to congestion
Shortest Path is Good?

- Exp num. trans. from A->B->Sink
  - Assume link reliability is symmetric
  - $1/(90\% \times 90\%) + 1/(90\% \times 90\%) = 2.47$

- Routing objective:
  - Minimize $\sum(1/(p_{f,i} \times p_{r,i}))$
  - where $p_{f,i}$ and $p_{r,i}$= forward and reverse link reliability at each i along the path

<table>
<thead>
<tr>
<th>Path</th>
<th>Hop Count</th>
<th>Exp. Num. Trans.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-&gt;Sink</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>A-&gt;B-&gt;Sink</td>
<td>2</td>
<td>2.47</td>
</tr>
</tbody>
</table>
Empirical Measurement => Lossy Links

- Data = Connectivity matrix of 20 Micas on a line at a particular trans. power setting.
- Each dot represents reliability of a directional link at a given distance.

Reception Probability vs. Distance at Power Setting 50

clear   transitional   silent
Pitfalls

• Assumption that links are “inherently” good  
  – assumes link layer abstraction provides good links and pay little attention below network layer

• Reverse link quality is “as good”  
  – routing (DSR, Diffusion, AODV) using reverse path routing

• Fixed consecutive # of failures = link failures  
  – “semi-good” links are easily treated as failures  
  – create instability
Routing Techniques to Evaluate

- Directed Diffusion, DSR, AODV
  - Essence:
    - reverse path routing by flooding
    - non-shortest path routing (do switch if a shorter path is found)
  - DSR, AODV (On-demand routing)
    - but every node is a source
    - source initiated route request broadcasts
      - “local storm” of replies
  - Evaluation
    - Sink initiated flooding to approximate reverse path routing
Routing Techniques to Evaluate (Cont.)

• DSDV
  – Essence:
    • “fresher” path overrides “shorter” path
    • fixed number of failure = link failure detection

• Distance vector based routing
  – Link estimations
    • Simple moving average link estimator
      – Sniff link seq. number in packets
      – Exchange link estimations via route messages

  – Two cost metrics
    • Shortest path
      – Only consider links with 75% reliability or better
    • Exp. number of transmissions along a path
Simulation Platform

- Matlab simulator
  - packet level simulation
  - incorporates loss model based on empirical traces
  - implements TinyOS network stack
  - visualization of the tree evolution
Simulation Results

- 49 nodes layout on a grid
  - grid interval = length of “effective” com. range
  - simulation time 1000s
  - data rate: 10s/msg
  - route rate:
    - 10s/msg (0 - 200s)
    - 50s/msg (200s – 1000s)

- Simple, coarse link estimation is much better than none

<table>
<thead>
<tr>
<th></th>
<th>Success to B.S.</th>
<th># Retrans. /msg recv. by B.S.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sink Broadcast</td>
<td>35.8%</td>
<td>5.4</td>
</tr>
<tr>
<td>DSDV</td>
<td>84.8%</td>
<td>1.42</td>
</tr>
<tr>
<td>Shortest Path (75%)</td>
<td>94.7%</td>
<td>0.7</td>
</tr>
<tr>
<td>Exp. # Trans.</td>
<td>96.0%</td>
<td>0.54</td>
</tr>
</tbody>
</table>
Link Failures

- Link Failure =
  - 9 consecutive transmission failures

- DSDV:
  - selection of bad links are frequent without link estimations
Link Estimator Study

• A sampled window average with an exponentially weighted moving average filter
  – yields a simple yet efficient link estimator
  – constant memory footprint for any tuning

• Reliability ~ 50%
  – largest variance
  – requires about 100 packet opportunities to reach ±10% accuracy

• See paper for details
Network Partition

• When partition is detected, use negative reinforcement to prune down paths
  – Partition arises when no potential parents are available
    – Reset routes
    – Stop forwarding
    – Stop acking received multihop packet
  – Many-to-one traffic naturally creates this recursive pruning
Loops

• stale/incorrect information leads to loops

• finding a loop?
  1. a packet goes into the loop and comes back
  2. check entire routing path/network
  3. none, hopefully, if protocol is “loop free” in practice
     • Internet
       – 1) alone will lead to long term cycles
     • Sensor networks
       – many-to-one traffic
       – every node is a router and a data source
       – short-term cycles: 1) is adequate to signal route changes
         » relatively immobile topology
         » loop-free guarantee protocol has higher cost and seems unnecessary
High Cell Density

- Limited memory and bandwidth
  - can only learn and interact a subset of neighbors at high cell density

- Common case assumption
  - < 100 neighbors
  - Memory
    - 4kB => statistics ~ 100 neighbors may take 10% to 20%
  - Bandwidth
    - Scarce resource (especially for multihop traffic)
    - Select a subset of neighbors for link estimation exchange in each route message

- Resource allocation problem
  - Given limited slots in each route messages
  - Which neighbors to feedback?
Multiple-Winner Lottery

• Hold lotteries to select N neighbors to be included in each route message

• Ticket allocation scheme
  – No tickets to nodes with smaller routing metrics
    • no exchange with potential parents

  – Tickets = (number of children of that neighbor + 1) * (Δrouting_metric) * f(link estimations)

  – f should be an inverted U function mapping [0,100%] to tickets

<table>
<thead>
<tr>
<th>Tickets</th>
<th>reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>100%</td>
</tr>
</tbody>
</table>
Initial Result

- Same experimental setting as before
- Ignore $f()$
- Average cell density is 12 nodes
- Route message has max. 6 neighbors
- Results comparable with shortest path (75%)

<table>
<thead>
<tr>
<th></th>
<th>Success to B.S.</th>
<th># Retrans. /msg recv. by B.S.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exp. # Trans. (N=6)</td>
<td>92.8%</td>
<td>0.73</td>
</tr>
<tr>
<td>DSDV</td>
<td>84.8%</td>
<td>1.42</td>
</tr>
<tr>
<td>Shortest Path (75%)</td>
<td>94.7%</td>
<td>0.7</td>
</tr>
<tr>
<td>Exp. # Trans.</td>
<td>96.0%</td>
<td>0.54</td>
</tr>
</tbody>
</table>
Slower Convergence Time

Exchange with at most 6 neighbors in each route message.

Exchange with all neighbors in each route message.
Forwarding Decisions

• Given there are multiple “good” parents
  – guidance from upper layer (aggregation or scheduling) may provide better forwarding decisions

  • aggregation example

    • ensures each distinct packet is destined to the same parent
Conclusion and future work

- Simulation results support hypothesis

- Link estimator can be coarse but essential

- Further investigations
  - Tickets allocation scheme and parameter N
  - Expose interface for forwarding decisions

- Get real measurements