
Presented by:
Kyu-Wook Cho
11/02/05

Outline
- Why Trickle?
- Trickle Algorithm
- Maintenance
- Propagation
- Related Work & Conclusion

Why Trickle?
- Must operate unattended for months or years
  - Changing environment and requirements need retasking

Cost for Retasking
- Propagation code is costly
  - Instead of full data transmit metadata
- Maintenance also cost a lot
  - Periodic broadcasting for maintenance
- Learn when to update the code costs more than actually propagating that code
Three Needed Properties

- Low Maintenance overhead
- Rapid Propagation
- Scalability

Outline

- Why Trickle?
- Trickle Algorithm
- Maintenance
- Propagation
- Related Work & Conclusion

Trickle Algorithm

- Time interval $\tau$, Threshold $k$, Counter $c$, Time $t$ in the range of $[0, \tau]$.
- At time $t$, broadcast metadata if $c < k$.
- Interval of size $\tau$ completes, reset $c = 0$.
- Increment $c$ when a mote hears identical metadata.
- Transmit update when it hears older metadata.
- If a mote hears new, broadcast its own summary.

Outline

- Why Trickle?
- Trickle Algorithm
- Maintenance
- Propagation
- Simulation
- Related Work & Conclusion
Maintenance

- Evaluating Trickle with relaxing each assumption
  - Ideal: No packet loss, perfect interval synchronization, single hop
  - Packet loss
  - Without synchronization: Short-listen problem
  - Multi-hop Network: Hidden Terminal problem

Packet Loss Relaxation

Without Synchronization (1/4)

- Scales $O(\infty)$
- Short listen problem

Without Synchronization (2/4)

- Short listen problem
Maintenance
Without Synchronization (3/4)

- Listen-only period: \([0, r] \rightarrow [r/2, r]\)

Maintainance
Without Synchronization (4/4)

Hidden Terminal Problem

Multi-hop Network Scalability

(a) Total Transmissions per Interval
(b) Receptions per Transmission

Hidden Terminal Problem

Regions of Possible Hidden Terminal Nodes

Uniformly distributed in 50'x50' square area
Maintenance
Multi-hop Network with Redundancy

- Redundancy
  - Increase overall communication, per-mote transmission rate shrinks
  - Each mote’s expected redundancy
    - \((c + s)/k - 1\), where \(s\) is 1 if mote transmit, 0 if not

Outline
- Why Trickle?
- Trickle Algorithm
- Maintenance
- Propagation
- Simulation
- Related Work & Conclusion

Propagation
- Large \(\tau\) (gossiping interval)
  - Low communication overhead
  - Slower propagation
- Small \(\tau\)
  - Higher communication overhead
  - Quick propagation

Trickle Pseudocode
- Lower bound \(\tau_l\), upper bound \(\tau_u\)

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\tau) Expires</td>
<td>Double (\tau), up to (\tau_u). Reset (c), pick a new (t).</td>
</tr>
<tr>
<td>(t) Expires</td>
<td>If (c &lt; k), transmit.</td>
</tr>
<tr>
<td>Receive same metadata</td>
<td>Increment (c).</td>
</tr>
<tr>
<td>Receive newer metadata</td>
<td>Set (\tau) to (\tau_l). Reset (c), pick a new (t).</td>
</tr>
<tr>
<td>Receive older metadata</td>
<td>Set (\tau) to (\tau_l). Reset (c), pick a new (t).</td>
</tr>
<tr>
<td>Receive older metadata</td>
<td>Send updates.</td>
</tr>
</tbody>
</table>

\(t\) is picked from the range \([\frac{2}{3}, \tau]\).
Propagation Simulation (1/2)

- Graph showing signal strength over time for different intervals (5', 60 seconds).

Propagation Simulation (2/2)

- 5' spacing
- Hidden Terminal Problem occurred
- Wave-like propagation in sparser network

Propagation Empirical Deployment Layout

Propagation Empirical Network Propagation (1/2)

(a) \( \tau_h \) of 1 minute, \( k = 1 \)
Empirical Network Propagation (2/2)

Outline
- Why Trickle?
- Trickle Algorithm
- Maintenance
- Propagation
- Related Work & Conclusion

Related Work
- Epidemic algorithm for maintaining data consistency in distributed systems
  - Delivering a piece of data to as many nodes as possible within a certain time period
- PlanetP project, gossiping protocol, uses unicast link to random member of a neighbor set
- SPIN
  - Propagation protocol, does not address maintenance cost
- TinyDB
  - Query system uses an epidemic style of code forwarding
  - However, it depends on periodic data collection with embedded metadata

Conclusion
- Rapid propagation with low maintenance
  - Dynamically change the time interval
- Scales logarithmically over different density and different network size
- Nameless protocol
  - No name management such as neighbor lists