The Emergence of a Networking Primitive in Wireless Sensor Networks

P. Levis, E. Brewer, D. Culler, D. Gay, S. Madden,
N. Patel, J. Polastre, S. Shenker, R. Szewczyk and A. Woo

CACM Volume 51, Number 7

Presenter - Bob Kinicki

WPI

Internet of Things

Fall 2015

Outline

- Introduction
- WSN
- WSN Protocols
  - Dissemination Protocols
  - Collection Protocols
- Trickle Algorithm
- Maté Case Study
- Trickle Discussion
- Conclusions

Introduction

- Sensors are normally on a mote platform with a low-power CMOS radio.
- Mote constraints include:
  - Power supply
  - Limited memory (a few kilobytes of RAM)
  - Unattended operation
  - Lossy and transient wireless communications
- WSNs (a collection of motes) are typically embedded in physical environment associated with their application.

Wireless Sensor Networks

- Physical device location often dictated by application and physical constraints.
- Any retransmission, response or ACK contributes to contention, interference and loss.
- Redundancy is essential to reliability!
- The variety of WSN topologies and densities calls for a polite, density-aware, local retransmission policy.

Networking Protocols

- Network protocols focus on minimizing transmissions and providing reliability (namely, making sure transmitted packets arrive successfully).
- Most sensor networks rely on two multi-hop protocols:
- a collection protocol for pulling data out of a network.
- A dissemination protocol for pushing data into a network.

Dissemination

- WSN administrators need to adjust how the network collects data by changing the sampled sensors, the sampling rate or the code running on the nodes.

- Administrator needs to disseminate these changes to every node in the network.

Dissemination

- Early systems used flooding to disseminate.
- Flooding can be unreliable and many, concurrent packet broadcasts yield a broadcast storm.
- Adaptive flooding uses an estimate of node density to limit the flooding rate.
  - Getting this to work across network densities is tricky!

Dissemination Protocols

- Another view – dissemination protocols ensure every node eventually has a consistent version of a shared state.
- Casting dissemination as a data consistency problem means it does not provide full reliability.
- Eventual consistency only implies delivery of the most recent version to connected nodes.

Dissemination Protocols

- An effective dissemination protocol needs to bring nodes up to date quickly while sending few packets when every node has the most recent version.

- Hence, this is a requirement for the underlying consistency mechanism.

Collection Protocols

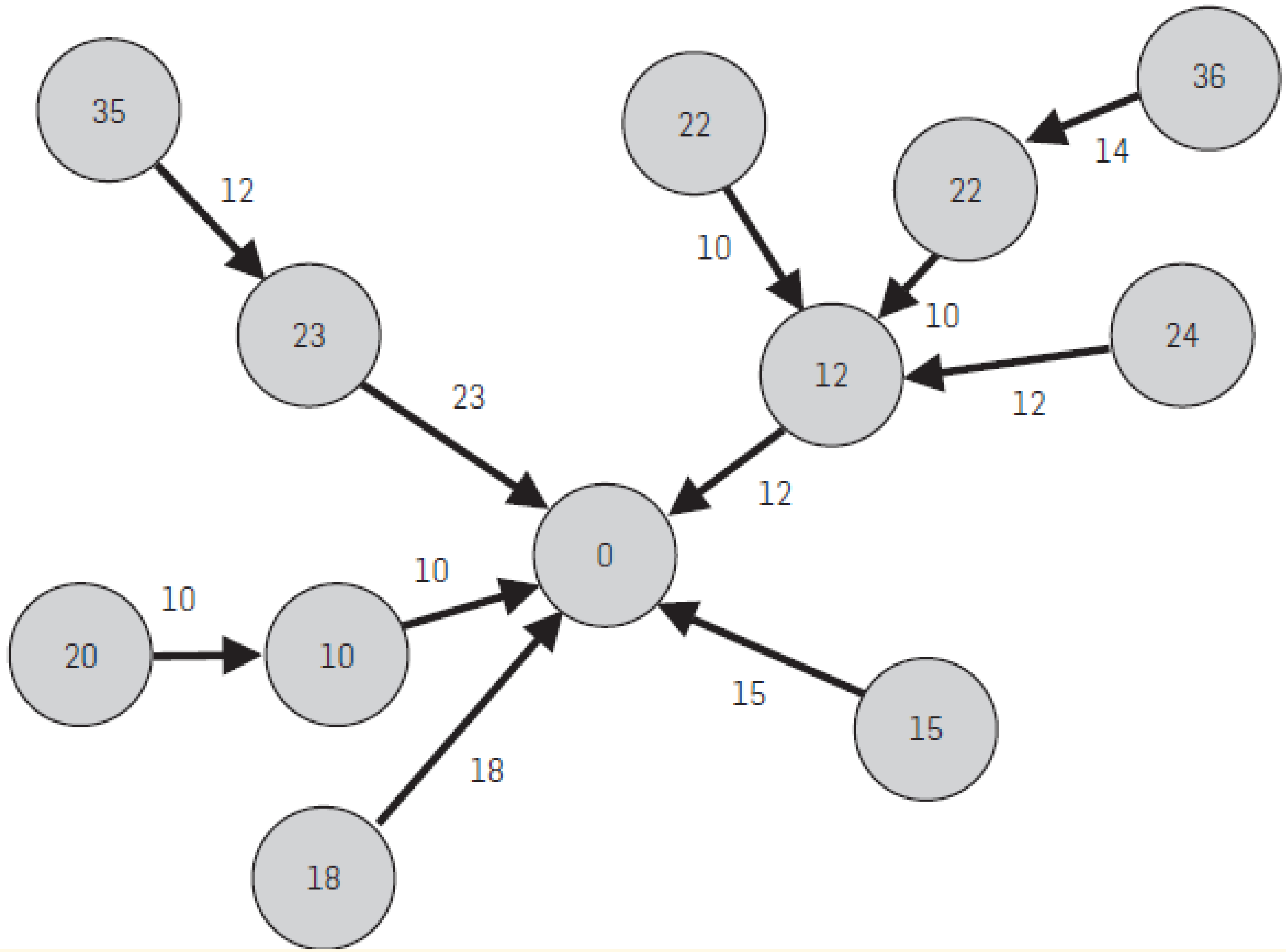
- WSNs report observations on a remote environment and thereby need a collection protocol.
- Collection protocols provide unreliable datagram delivery to a collection point using a minimum-cost routing tree.
- Typically cost measured in ETX (expected number of transmissions) which is related to packet delivery rate.
  - Nodes send packets on the route that requires the fewest transmissions to reach a collection point.

Collection Protocols

- An early collection protocol, directed diffusion, used collection trees based on data-specific node requests.
- Experiences with low-power wireless networks moved strategies towards a simpler approach where each node decides on a single next hop for all forwarded traffic.
  → creating routing trees to fixed collection points.
  - Tree built using a routing cost gradient.
  - Collection point has a cost of 0.

Figure 2: Collection Tree

Figure 2: Sample collection tree, showing per-link and node costs. The cost of a node is its next hop's cost plus the cost of the link.



Collection Protocols

- Collection variation includes:
  - How to quantify and calculate link costs.
  - The number of links in the tree.
  - How link state changes are propagated.
  - How frequently to re-evaluate link costs and switch parents.

Collection Protocols

- Early collection protocols used link costs.
- Second generation (similar to AODV and DSDV) used periodic broadcasts to estimate transmissions per delivery.
- Third generation added physical layer quality data.
- Current generation (e.g., Collection Tree Protocol (CTP)) gets information from multiple layers.

Collection Protocols

- Newer protocols reduce control traffic to increase efficiency.
- However, they need to send link-layer broadcasts to their local neighbors to advertise their presence and routing cost.
- Transmission frequency of routing advertisements causes design tension.
- Protocols reduce this tension by converting routing gradient to a data consistency problem.

Data Consistency Mechanism

- To address both dissemination and collection protocols as a problem of maintaining data consistency, the data consistency requirements are:
  - Resolve inconsistencies quickly.
  - Send few packet when data is consistent.
  - Require very little state.
- Trickle meets these three requirements.

Trickle

- Trickle algorithm establishes a density-aware local broadcast with an underlying consistency model that guides when a node communicates.
- The algorithm controls the send rate such that each node hears a trickle of packets (enough to stay consistent).
- Trickle relies only on local broadcasts and its basic mechanism is a randomized suppressive broadcast.

Trickle Algorithm [Bjamaa 15]

Trickle variables:

- c consistency counter
- I Trickle interval
- t transmission time

Trickle configuration parameters:

- I_{min} minimum interval size (time units)
- I_{max} determines maximum interval size as: $I_{min} \times 2^{I_{max}}$
- k redundancy constant

Trickle Algorithm

0. Initialization:

   I ← random value in range [$Imin$, $Imin \times 2^{Imax}$];

1. c ← 0;

   Pick t uniformly at random from range [I/2, I];

   start timer;

   {anytime during interval, if node hears a consistent message increment c;}

2. Once timer reaches t

   iff c < k node transmits message;   suppression possible!

3. When I expires, double interval length until I reaches $Imin \times 2^{Imax}$; go to 1.

   {anytime during interval, if node hears an inconsistent message and if I > $Imin$ then (I ← $Imin$; go to 1) }

Message Suppression

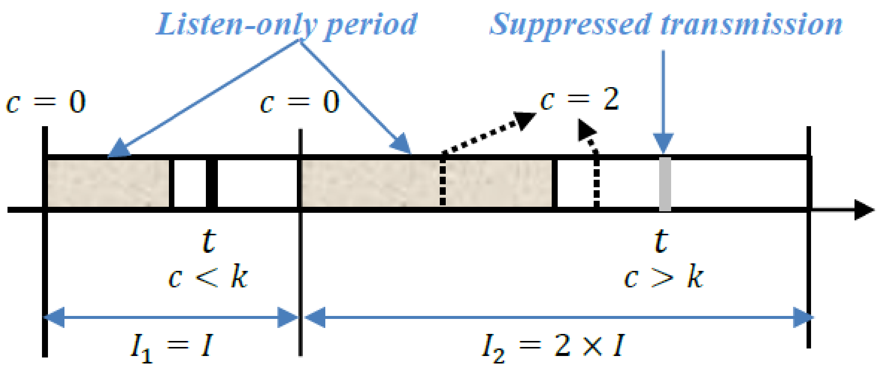


Figure 1 Trickle algorithm over two intervals with $k = 1$. Black line is a transmission, grey one is a suppressed transmission and dotted lines are receptions

- **Listen-only period addresses short-listen problem.**
- **When c > k , state is consistent and other transmissions are suppressed.**

Short-listen Problem

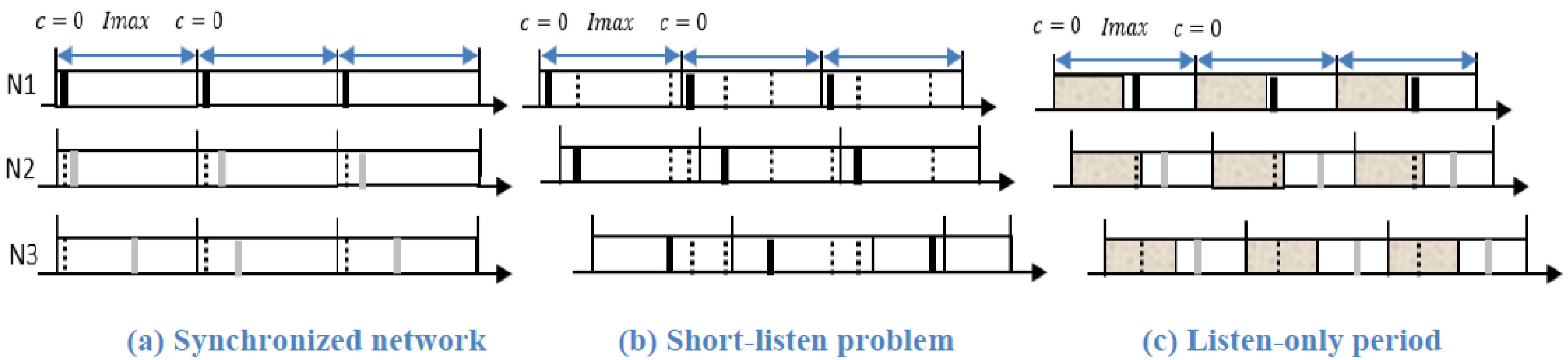


Figure 2 Short-listen problem and listen-only period with k = 1. Black boxes are transmissions, dotted lines are receptions and grey boxes are suppressed transmissions.

- **Short-listen comes from non-synchronized intervals among neighbors.**
- **N2 and N3 in (b) suffer from short-listen problem.**
- **(c) shows listen-only impact.**

Figure 4: Transmissions vs Nodes

These graphs assume nodes are synchronized.

Figure 4: Trickle's transmissions per interval scales logarithmically with density. The base of the logarithm is a function of the packet loss rate (the percentages)

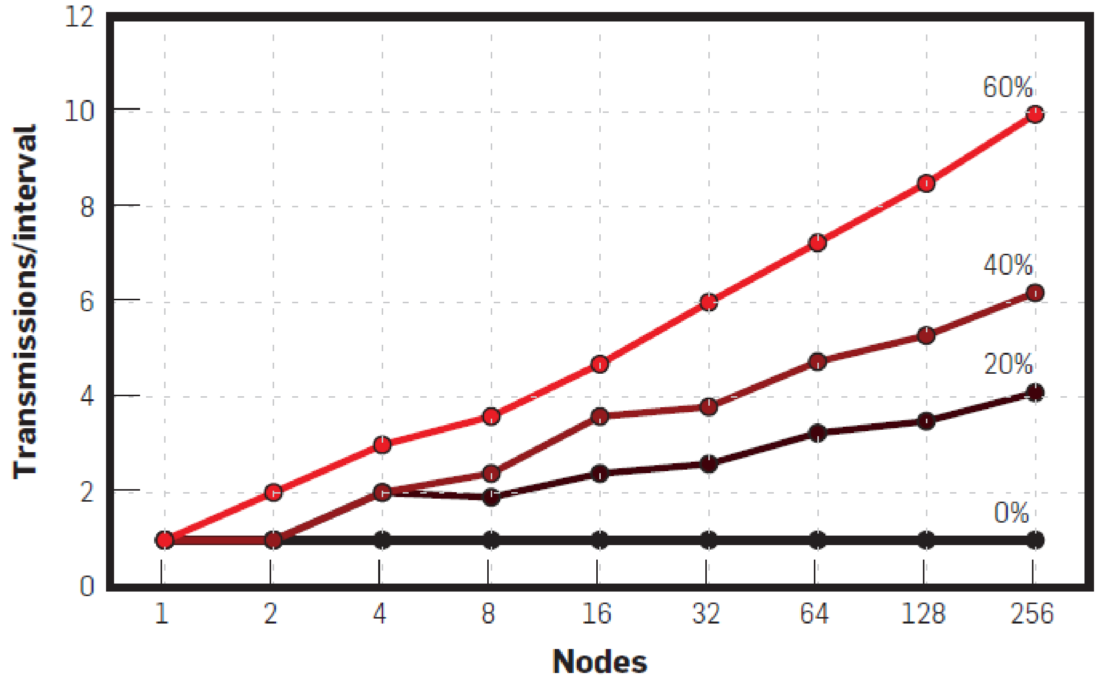
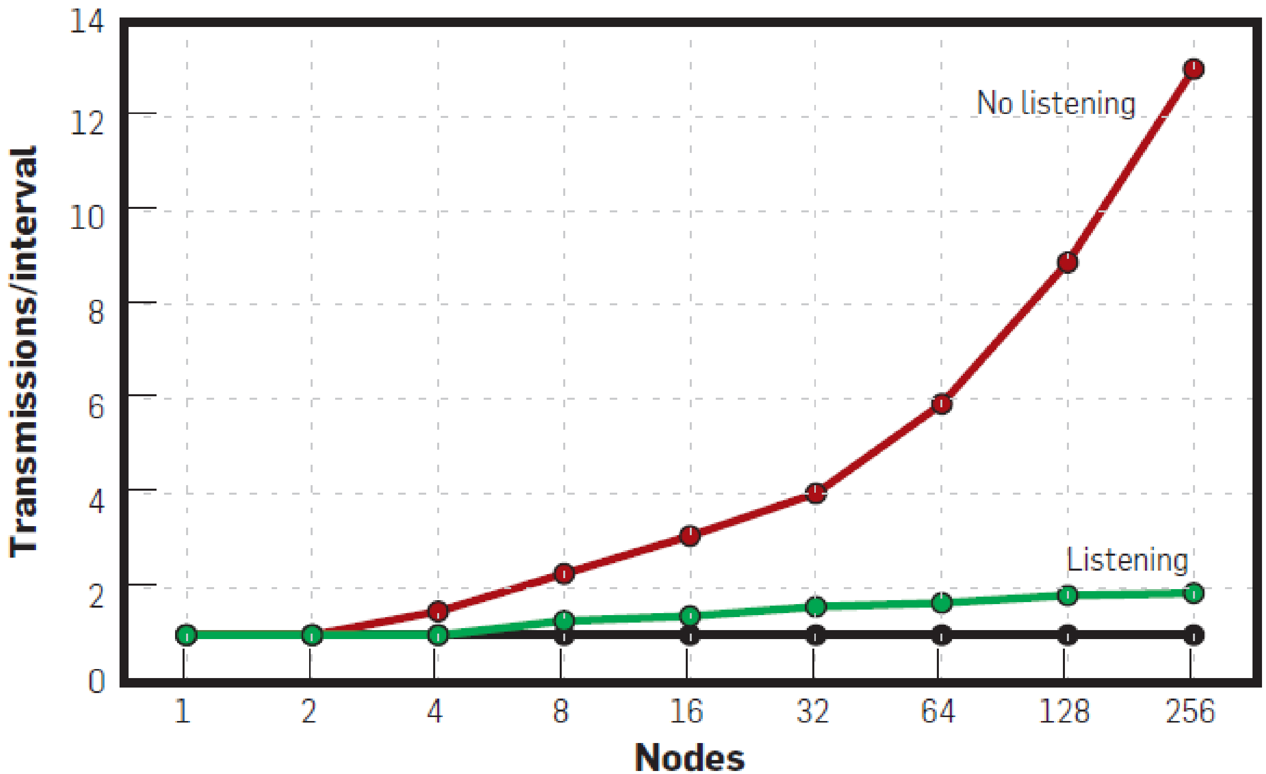


Figure 5: Listen-only Effect

Figure 5: Without a listen-only period, Trickle's transmissions scale with a square root of the density when intervals are not synchronized. With a listen-only period of duration $\frac{\tau}{2}$, the transmissions per interval asymptotically approach 2*k*. The black line shows how Trickle scales when intervals are synchronized. These results are from lossless networks.



Maté Case Study

- Maté :: a lightweight, bytecode interpreter for WSNs.
- Maté uses Trickle as a propagation service to periodically broadcast version summaries.
  - 30-byte code fits into one frame.
  - Consistency mechanism – broadcast missing routines 1,3 and 7 seconds after hearing there is an inconsistency.
- Trickle resources are small (70 bytes RAM, 11 bytes counters, 1.8K executable).

TOSSIM Simulation Details

- TOSSIM, a TinyOS simulator, models wireless connectivity at the bit level and wireless loss.
- **Node density** modeled via spacing between nodes (5 to 20 ft. in 5 ft. increments).

  I_{min} = 1 second;

  $I_{min} \times 2^{I_{max}}$ = 1 minute;

- 400 sensor nodes "regularly spaced" in a 20 x 20 grid (note - graph shows distances between a 19 x 19).

Figure 7: Time to Consistency

- Crossing the network takes from 6 to 40 hops.
- Time to complete propagation varied from 16 sec (dense network) to 70 sec (sparce network).
- Minimum per hop delay is *Imin/2* with 1 sec broadcast time → best case delay is 1.5 sec/hop.
- Claim: graphs show nodes cooperate efficiently.

Figure 7: Time to consistency in 20 × 20 TOSSIM grids (seconds). The hop count values in each legend are the expected number of transmissions necessary to get from corner to corner, considering loss.

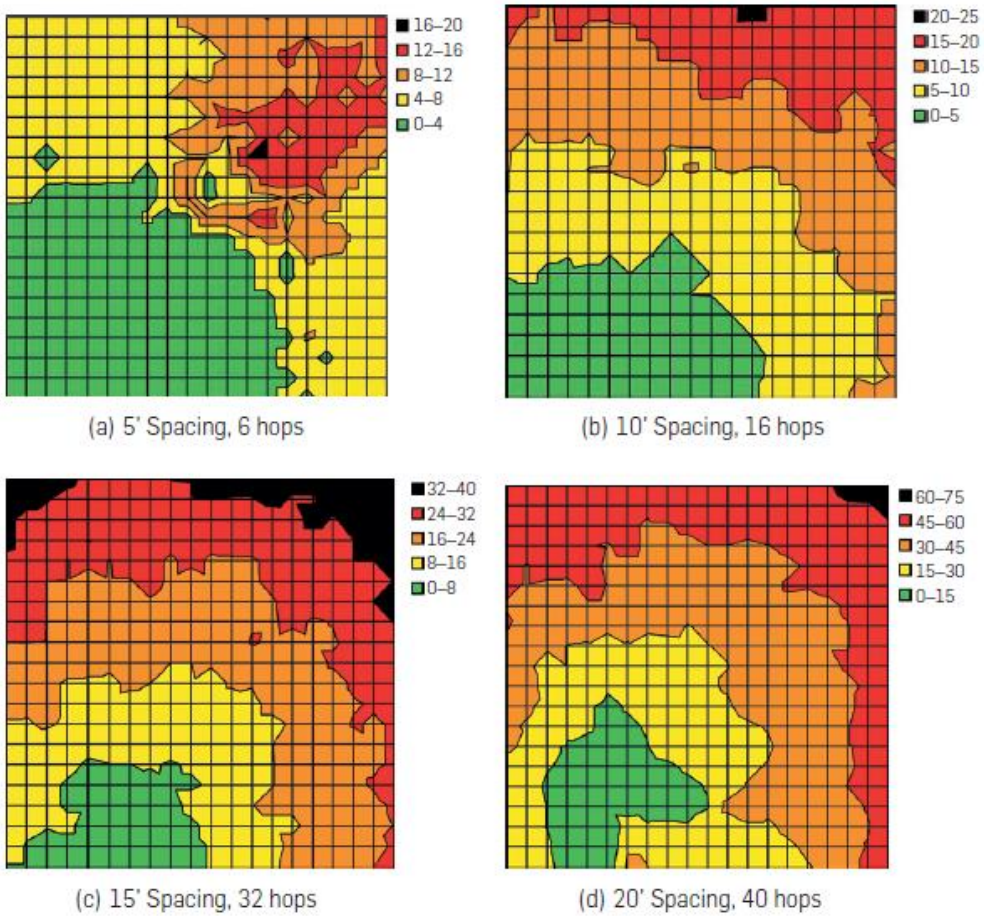
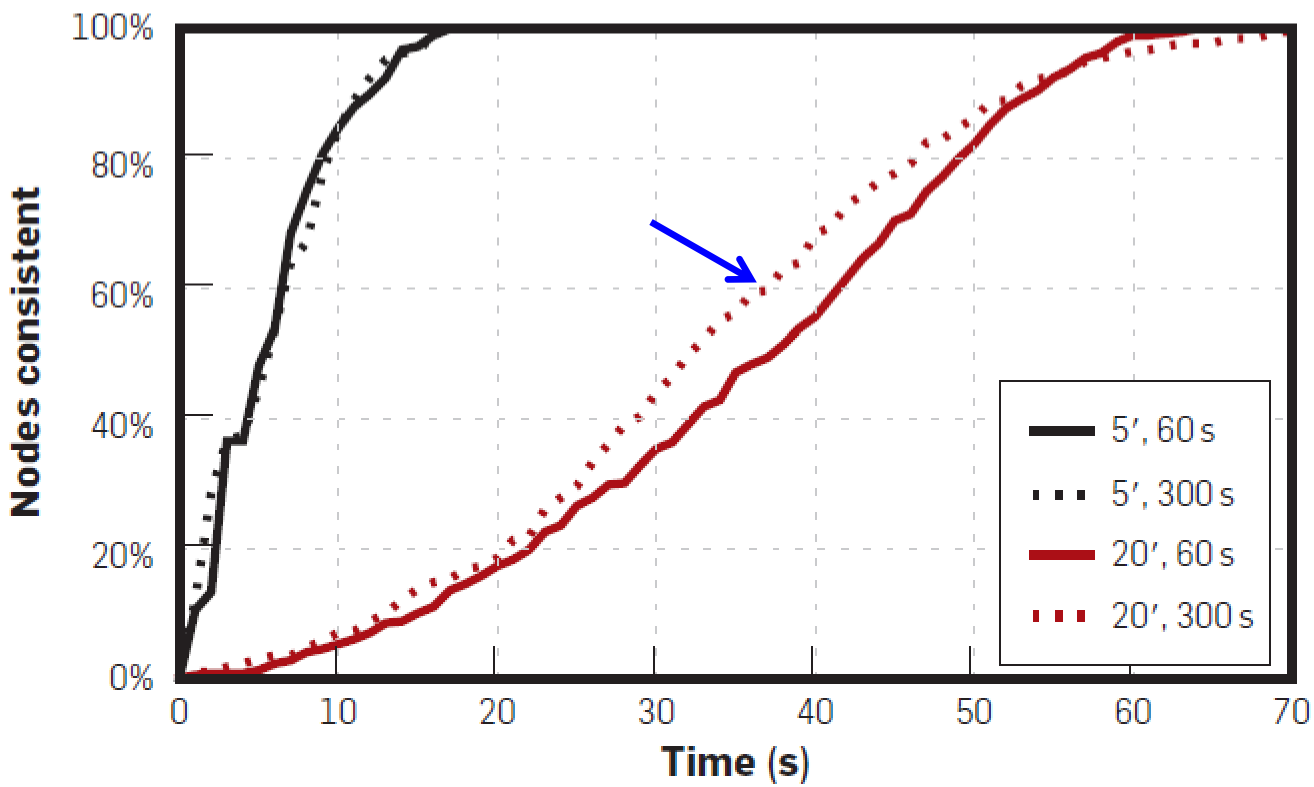


Figure 8: Consistency Rate

Figure 8: Rate nodes reach consistency for different τ_hs in TOSSIM. A larger τ_h does not slow reaching consistency.



Uses and Improvements

- Trickle used in many dissemination protocols today (e.g., Deluge, MNP, Drip and Tenet).
- More efficient collection protocols also using Trickle for consistency (e.g., TinyOS, 6LoWPAN IPv6 routing tables and ICMP neighbor lists).
- Drawback – Trickle maintenance costs grow O(n) with number of data items.

Trickle Discussion

- WSNs do not know interconnection topology apriori. This topology is not static (even when nodes are NOT mobile).
- Redundancy both helps and hurts!
- Trickle's design comes from two areas:
  - Controlled, density-aware flooding algorithms from wireless and multicast networks
  - Epidemic and gossip algorithms for maintaining data consistency in distributed systems.

Trickle Discussion

- Trickle adapts to local network density like controlled flooding, but continually maintains consistency in a manner similar to epidemic algorithms.
- Trickle uses broadcast nature of wireless channel to conserve energy.
- Trickle's exponential times work in reverse of standard backoff. Namely, it defaults to largest time window and decreases only for inconsistency.

Trickle Discussion (cont)

- Trickle leads to energy-efficient, density-aware dissemination by avoiding collisions and suppressing unnecessary retransmissions.
- Trickle suppresses implicitly through nearby nodes that hear a broadcast.

Conclusions

- Two authors also authored original Trickle RFC (author list is impressive).
- Paper puts Trickle into a WSN routing context and does not just define Trickle.
- Trickle algorithm explanation is not concise.
- Discussed well the tension in performance associated with run time choices.
- Paper shows basic performance trends and Maté case study.

References

[Djamaa 2015] Djamaa, B. and Richardson, M. *The Trickle Algorithm: Issues and Solutions.* Cranfield Univesity Reseach Report. January 2015.