PathDCS Overview

Practical Data-Centric Storage, NSDI 2006

By Cheng Tien Ee, Sylvia Ratnasamy, Scott Shenker

(slide material from http://www.eecs.berkeley.edu/~ct-ee/pathDCS/)

CIS700 presentation by Joe Kopena

Previously in CIS700

- Major theme in class: Data Centric Storage
- Key based get/put storage and query interface
- Challenging to apply to sensor networks
  - Limited CPU, memory, network resources
- Not worthwhile in some scenarios
  - E.g, all data required by gateway node
- But worthwhile in some scenarios
  - Summarized, periodically queried data
  - Disconnected gateways
  - Foundation for complex query support

DCS Requirements

- A system for consistently locating destination
  - Requires a common frame of reference
  - Mapping from key into that reference
- Routing mechanism to forward to destination

Distributed Hash Tables

- CAN
  - Frame of reference is virtual coordinate space
  - Mapping is hash function into coordinate space, node managing zone in which it falls is destination
  - Routing is greedy forwarding through coordinate space, with underlying point to point support
- Chord
  - Frame of reference is ring identifier
  - Mapping is hash function into identifier space, closest node chosen as destination
  - Routing is forwarding along finger & successor tables, with underlying point to point support

DHTs on Sensor Networks

- Have to handle network layer routing & forwarding in addition to overlay routing
  - No underlying point to point routing support
  - Have to do so in very constrained setting

Previous Reading: GHT

- Geographic Hash Table (GHT)
  - Data-centric storage for sensor nets
  - One of several systems utilizing physical coordinates
- Frame of reference is geographic region encompassing sensor network
- Mapping is hash onto coordinate in region, closest node chosen as destination
- Routing is greedy forwarding toward geographically closer nodes, or walk around network if necessary

Limitations of GHT & Others

- Must know boundaries of reference region
  - Requires preconfiguration and tuning, limits mobility
- Assumes unit-disc connectivity
  - Needed for correctness of perimeter walk
  - Addressed in later work (CLDP) but complex & costly
- Must know node position
  - Not available on all devices
  - Virtual coordinates (GEM, NoGeo) complex & costly
- In general, point to point routing on sensor nets very difficult (power, bandwidth, CPU limits)

7/19

PathDCS Goals

- Provide data centric storage for sensor nets
- Make no assumptions about availability of geographic information, or underlying routing
- Use simple, deployable techniques appropriate to limited nodes used in sensor networks

8/19

Basics

- Frame of reference is set of known, landmark beacon nodes in network
- Mapping is onto a walk on the network, anchored by those beacons
- Routing is along trees rooted at beacons

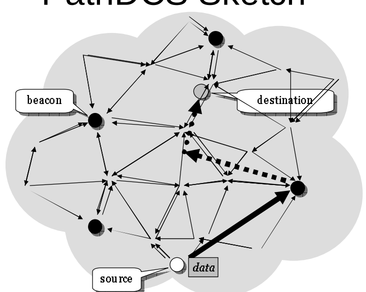
9/19

Algorithm

- Randomly elect or configure beacons
- Each beacon has logical numeric identifier
- To store data:
  - Hash key into identifier space, choose closest beacon
  - Forward to beacon along that beacon's routing tree
  - For each i <= p (a preconfigured parameter)
    - Hash key and i into identifier space, choose closest beacon
    - Forward toward beacon along its routing tree for the number of hops determined by the hash modulus distance to beacon
  - Final node is destination
- To query data, follow same procedure

10/19

PathDCS Sketch



Segment	Beacon	Hops
1	id closest to $h(key_{data},1)$	All the way to beacon
2	id closest to $h(key_{data},2)$	$[h\ (key_{data},\ 2)\ \%\ max\_hops\_2]+1$
3	id closest to $h(key_{data},3)$	$[h\ (key_{data},\ 3)\ \%\ max\_hops\_3]+1$

11/19

Beacon Election

- Each node is assigned a random identifier
- Identifier space divided by preconfigured # of beacons into equal-sized partitions
- Node with greatest id in partition becomes beacon for that partition
  - Based on announcement delay proportional to distance from upper partition edge
  - Snooping for announcements, closeby beacons
- Beacon ids for each partition advertised in distance-vector packets

12/19

Tree Routing

- PathDCS built entirely on tree routing
  - A common, deployed capability for sensor networks
  - Does not rely on complex, costly point to point routing
- A routing tree is constructed for each beacon
  - Nodes recursively pick parent closest to beacon among all their neighbors
  - Can apply metrics such as ETX (estimated throughput) or MT (minimum transmission)

Beacon Handoff/Takeover

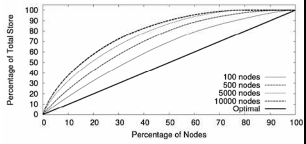
- Beacons can be overloaded or fail over time
- Can explicitly handoff beacon responsibility
  - Might handoff on dwindling power, nearby beacon
  - Beacon picks a one-hop neighbor & switches IDs
- On timeout, 1-hop neighbors conduct election
  - Very similar process to beacon initialization
- Proximity of new beacons minimizes change in paths, hopefully increase chance of finding data
  - Queries local flood at destination if data not present

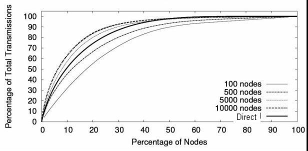
Data Preservation

- Nodes storing data periodically push it out
  - Refreshes data, making it accessible if topology has changed and node no longer owns that data
- Data also replicated within a preconfigured number of hops from storage node
  - Neighbor nodes will have data in even of failure
  - May help recover after topology change

High-Level Simulation

- Evaluate load-balancing
  - Storage & transmission
  - Simple network model
- 5000 nodes, ~14.5 neighbors, 20 beacons
- Storage: Seems okay
- Transmission: Not clear from reported graphs
  - Percentage close to optimal, direct routing
  - Stretch proportional to p, about 2.4 when p = 2

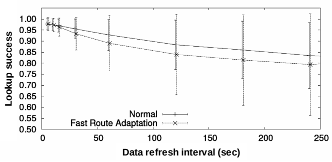


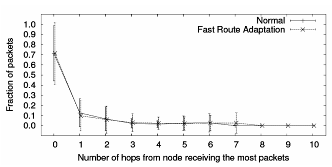


This graph: Not the most informative ever!

TOSSIM (Packet-Level) Simulation

- 500 node, 18 hop diameter network, ~10.4 neighbors, 5 beacons
  - Models loss, queues, etc
  - Node and link failure parameters not clear in paper





Other Simulations

- Includes demonstration on live sensor testbed
  - Not very dynamic, doesn't provide much insight beyond "It works." Results similar to TOSSIM graphs.
- Some key metrics not emphasized, e.g. stretch and total # transmissions rather than percentage
- Node mobility not addressed or evaluated

Conclusion

- PathDCS presents a novel approach to in-network sensor data storage
  - Combines distributed hash tables with tree routing
- Needs a strong comparison between the combined cost of maintaining trees and routing stretch, versus cost of maintaining point to point routing and other DHT overlay mechanism
  - Reported stretch not terrible, but may have considerable effect on sensor lifespan
- Probably geared toward refreshed data, certain usage scenarios/rates, static nodes

19/19