TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks

Samuel Madden
UC Berkeley
with
Michael Franklin, Joseph Hellerstein, and Wei Hong

Slides based on Sam Madden’s talk

TAG Introduction

- What is a sensor network?
- Programming sensor nets is hard!
- Declarative queries are easy
  - Tiny Aggregation (TAG): In-network processing via declarative queries
- Example:
  - Vehicle tracking application: 2 weeks for 2 students
  - Vehicle tracking query: took 2 minutes to write, worked just as well!

Overview

- Sensor Networks
- Queries in Sensor Nets
- Tiny Aggregation
  - Overview
  - Simulation & Results

Device Capabilities

- “Mica Motes”
  - 8bit, 4Mhz processor
  - Roughly a PC AT
  - 40kb CSMA radio
  - 4kB RAM, 128kB flash, 512kB EEPROM
  - TinyOS based
- Variety of other, similar platforms exist
  - UCLA WINS, Medusa, Princeton ZebraNet, SmartIts
Sensor Net Sample Apps

- **Habitat Monitoring**: Storm petrels on great duck island, microclimates on James Reserve.
- **Earthquake monitoring**: in shake-test sites.
- **Vehicle detection**: sensors along a road, collect data about passing vehicles.

Traditional monitoring apparatus.

Metric: Communication

- Lifetime from one pair of AA batteries
  - 2-3 days at full power
  - 6 months at 2% duty cycle
- Communication dominates cost
  - < few ms to compute
  - 30ms to send message
- Our metric: communication

Communication In Sensor Nets

- Radio communication has high link-level losses
  - typically about 20% @ 5m
- Ad-hoc neighbor discovery
- Tree-based routing

Overview

- Sensor Networks
- Queries in Sensor Nets
- Tiny Aggregation
  - Overview
  - Optimizations & Results

Declarative Queries for Sensor Networks

<table>
<thead>
<tr>
<th>Examples:</th>
</tr>
</thead>
</table>

- `SELECT nodeid, light FROM sensors WHERE light > 400 EPOCH DURATION 1s`

<table>
<thead>
<tr>
<th>Sensors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

Aggregation Queries

- `SELECT AVG(sound) FROM sensors EPOCH DURATION 10s`
- `SELECT roomNo, AVG(sound) FROM sensors GROUP BY roomNo HAVING AVG(sound) > 200 EPOCH DURATION 10s`

Rooms w/ sound > 200
**Overview**

- Sensor Networks
- Queries in Sensor Nets
- Tiny Aggregation
  - Overview
  - Optimizations & Results

**TAG**

- In-network processing of aggregates
  - Common data analysis operation
    - Aka gather operation or reduction in || programming
  - Communication reducing
    - Operator dependent benefit
    - Across nodes during same epoch
  - Exploit semantics improve efficiency!

**Query Propagation**

```
SELECT COUNT(*) FROM sensors
```

**Basic Aggregation**

- In each epoch:
  - Each node samples local sensors once
  - Generates partial state record (PSR)
    - local readings
    - readings from children
  - Outputs PSR during its comm. slot.
- At end of epoch, PSR for whole network output at root
- (In paper: pipelining, grouping)

**Illustration: Aggregation**

```
SELECT COUNT(*) FROM sensors
```
**Aggregation Framework**

- As in extensible databases, we support any aggregation function conforming to:
  - $f_{init} \{a_0\} \rightarrow \langle a_0 \rangle$
  - $f_{merge} \{\langle a_1 \rangle, \langle a_2 \rangle\} \rightarrow \langle a_{12} \rangle$
  - $f_{evaluate} \{\langle a_1 \rangle\} \rightarrow \text{aggregate value}$

(Merge associative, commutative!)

**Example: Average**

- $\text{AVG}_{init} \{v\} \rightarrow \langle v, 1 \rangle$
- $\text{AVG}_{merge} \{\langle S_1, C_1 \rangle, \langle S_2, C_2 \rangle\} \rightarrow \langle S_1 + S_2, C_1 + C_2 \rangle$
- $\text{AVG}_{evaluate} \{\langle S, C \rangle\} \rightarrow \frac{S}{C}$

**Types of Aggregates**

- SQL supports MIN, MAX, SUM, COUNT, AVERAGE
- Any function can be computed via TAG
- In network benefit for many operations
  - E.g. Standard deviation, top/bottom N, spatial union/intersection, histograms, etc.
  - Compactness of PSR

**Taxonomy of Aggregates**

- TAG insight: classify aggregates according to various functional properties
  - Yields a general set of optimizations that can automatically be applied

<table>
<thead>
<tr>
<th>Property</th>
<th>Examples</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total State</td>
<td>UNBOUNDED</td>
<td>Unboundedness of TAG</td>
</tr>
<tr>
<td>Arithmetic Sensitivity</td>
<td>ADD, SUB, DIV</td>
<td>Accuracy</td>
</tr>
<tr>
<td>Boolean Sensitivity</td>
<td>AND, OR, NOT</td>
<td>Logic</td>
</tr>
<tr>
<td>Spatial Sensitivity</td>
<td>UNION, INTERSECT</td>
<td>Spatial Algebra</td>
</tr>
<tr>
<td>Geometric Properties</td>
<td>DISTANCE, AREA</td>
<td>Spatial Algebra</td>
</tr>
<tr>
<td>Aggregate</td>
<td>AVG, MEDIAN, STD</td>
<td>Accumulated Sum, Max, Min, Mean, Variance</td>
</tr>
</tbody>
</table>
TAG Advantages

- Communication Reduction
  - Important for power and contention
- Continuous stream of results
  - Smooth transient faults across epochs
- Lots of optimizations
  - Via operator semantics

Simulation Environment

- Evaluated via simulation
- Coarse grained event based simulator
  - Sensors arranged on a grid
  - Two communication models
    - Lossless: All neighbors hear all messages
    - Lossy: Messages lost with probability that increases with distance

Benefit of In-Network Processing

<table>
<thead>
<tr>
<th>Simulation Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500 Nodes</td>
</tr>
<tr>
<td>50x50 Grid</td>
</tr>
<tr>
<td>Depth = ~10</td>
</tr>
<tr>
<td>Neighbors = ~20</td>
</tr>
</tbody>
</table>

Optimization: Channel Sharing ("Snooping")

- Insight: Shared channel enables optimizations
- Suppress messages that won’t affect aggregate
  - E.g., MAX
  - Applies to all exemplary, monotonic aggregates

Optimization: Hypothesis Testing

- Insight: Guess from root can be used for suppression
  - E.g., ‘MIN < 50’
  - Works for monotonic & exemplary aggregates
  - Also summary, if imprecision allowed

- How is hypothesis computed?
  - Blind or statistically informed guess
  - Observation over network subset

Experiment: Hypothesis Testing

Messages/Epoch vs. Network Diameter
(SELECT MAX(attr), R(attr) = [0,100])

- Uniform Value Distribution
- Dense Packing
- Ideal Communication
- Uniform Value Distribution
- Dense Packing
- Ideal Communication
Optimization: Use Multiple Parents

- For duplicate insensitive aggregates
- Or aggregates that can be expressed as a linear combination of parts
  - Send (part of) aggregate to all parents
  - In just one message, via broadcast
  - Decreases variance

Multiple Parents Results

- Better than previous analysis expected!
- Losses aren't independent!
- Insight: spreads data over many links

Summary

- TAG enables in-network declarative query processing
  - State dependent communication benefit
  - Transparent optimization via taxonomy
  - Hypothesis Testing, Parent Sharing
- Declarative queries are the right interface for data collection in sensor nets!
  - Easier to program and more efficient for vast majority of users