Advanced Topics in Computer Systems, CS262B
Prof Eric A. Brewer

TAG

April 29, 2004

I. Basics

Program the ensemble

SQL-like query language

In network aggregation

II. Query Language

SELECT AVG(volume), room FROM sensors

WHERE floor = 6

GROUP BY room

HAVING AVG(volume) > t

EPOCH DURATION 30s

Output is one row for normal queries (columns are AVG(volume), room); or a table for groups with one row per group.

HAVING filters the groups (not the readings)

EPOCH is how time between query executions -- output is thus a stream of values

Queries run until cancelled

III. Aggregation

Aggregation functions:

- messages contain state records (same for all motes); e.g. average = <sum, count>
- define three functions i, f, e
- i(x) is the initializer; e.g. i(x) -> <x, 1> for average
- z = f(x, y) is the merge procedure; where x, y, z are state records
  - <sz = sx + sy, cz = cx + cy>
- e is the evaluator; computes the final version from a state record

- $e(z) = sz/cz$

- queries use names for values; a local catalog maps names to local ids

Taxonomy:

- duplicate sensitive; MAX no, AVG yes
- Exemplary vs. Summary; MAX is E, AVG is S
- Monotonic: MAX yes, AVG no
- Partial state
  - distributive: constant size for each group and same as final value ($e(z) = z$); e.g. MAX
  - algebraic: constant size, but need an evalutor to get final result; e.g. AVG
  - holistic: state records proportional to total network size; e.g. MEDIAN
  - unique: holistic, but only keep unique values, not all values
  - content-sensitive: size depends on data attributes; e.g. histogram

Basics:

- distribute the query
- collect output for each epoch
- tree based routing
- sub-intervals based on depth in the tree; epoch duration = d * interval, where d is total depth
- aggregate state records for each group, but can batch messages
- can push down the HAVING predicates if the aggregation is monotonic
- may need to evict group values for space reasons (too many groups)
  - evicted groups can be pushed up the tree and fully aggregated later (trading bandwidth and power for temporary space)

IV. Other Issues

Deal with disconnections and loss

- use potential parents to recover more quickly
- error due to losses can be very large for COUNT, but is generally small (e.g. 10% for AVG)
- cache old values and reuse them if needed; more effective use of space than retransmission
- use multiple paths (via broadcast), can reduce the variance (but faults are not really independent)

Even power usage (assuming aggregation)

Long idle times to save power

Overall, up 20x savings in messages via aggregation (for max), 10x for AVG

Optimizations:

- snooping can reduce traffic, e.g. MAX
- hypothesis testing