Seminar on
In-network aggregation in sensor networks

Dhananjay S. Muli (05305906)
Sandeep Satpal (05329004)

March 7, 2006

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

{Samuel Madden, Michael, Joseph, Wei Hong}

Sensors : Overview

- Wireless, battery powered, small full fledged computers capable of measuring real world phenomena and filtering, sharing and combining these measurements
- Example : motes under development by UC Berkley
- Operating system : TinyOS
- Uses : monitor building integrity during earthquakes (civil engineering), monitoring temperature and power usage in the data centers (computer administration)

Ad-hoc networks : Overview

- Sensor nodes communicate through radio broadcast channel
- Links between nodes are symmetric
- Nodes communicate to each other in a hierarchy
- Communication is half-duplex
- Single bit transmission in motes is equivalent to 800 instructions
- Two constrainst
  - Must be able to deliver query results to all the nodes in a network
  - Must be able to provide one or more route to root

Ad-hoc Routing algorithm

- Routing tree approach
  - One node designated as root through which user interfaces to network
  - Root broadcast a message with its own id and level
  - Any node without an assigned level, hears this message and assigns its own level received value plus 1
  - It then chooses sender as parent, through which it will route its message to root
  - Repeat the same process till every node in the network becomes child of a node
- Periodically re-run the above algorithm for topology maintainence
  - If parent fails, select new parent
  - If topology changes, will adapt easily

Centralized approach of query execution

- Only base station (root node) computes the aggregate values
- All other nodes in the network collect and send values to base station
- Disadvantage : Higher network latency, communication overhead, power consumption

TAG (Tiny AGgregation) approach of query execution

- In-network aggregation
  - Leaf nodes get the values from sensors, apply aggregate function, clauses and sends to parent nodes
  - Internal nodes combine values from child nodes with its local value and apply aggregates
- Two phases
  - Distribution phase - queries are pushed down in the network
  - Collection phase - aggreagate values are routed up in the network
- Distribution and execution of aggregate query in power efficient manner
- Simple, declarative interface for querying

Query Model

- Assume queries are fired on single table - sensors
  select {*agg(expr)* , *attrs*} **from** *sensors*
  where {*selPred*}
  group by {*attrs*}
  having {*havingPreds*}
  epoch duration *i*
- Output of a standard SQL query is single value while TAG query gives stream of values
- Epoch duration specifies the amount of time sensors wait before acquiring and transmitting values
- Epoch duration should be atleast as large as time taken from a sensor to process value and to transmit it over a radio channel
- Values in stream are of the form (group id,aggregate value) appended with timestamp of epoch duration

Distribution phase

- Node receives query request from node/user
- It forwards the request down the network
- Sets the delivery interval for children slightly before the time its parent expect value
- How to choose interval
  - Power/Correctness tradeoff
  - One approach : (EPOCH Duration / d) d: depth
  - Other approach : learn from topology, environment specific

Collection phase

- During EPOCH every node listens to its children
- Computes partial state
- Finally it transmits partial state up the network

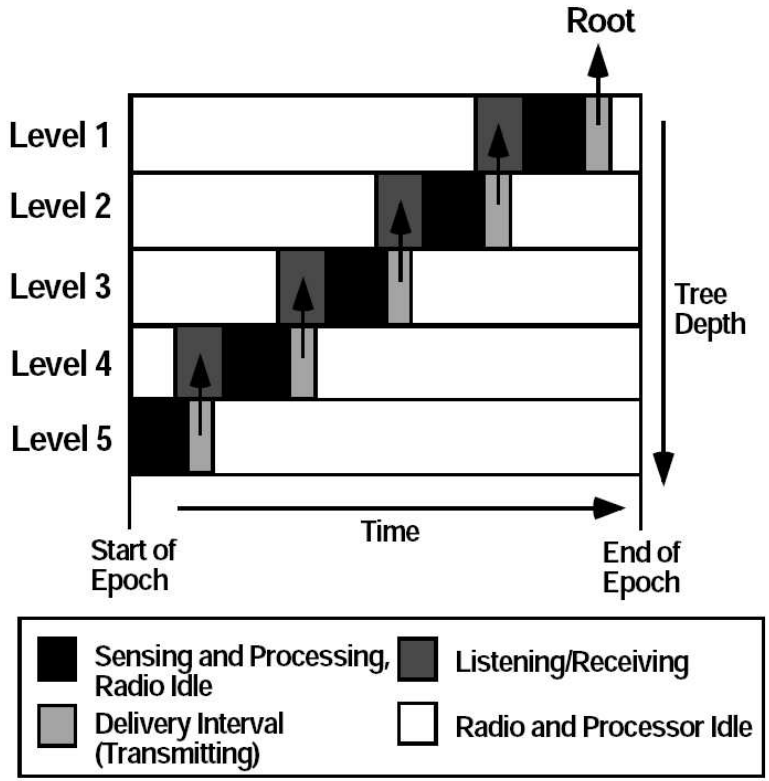
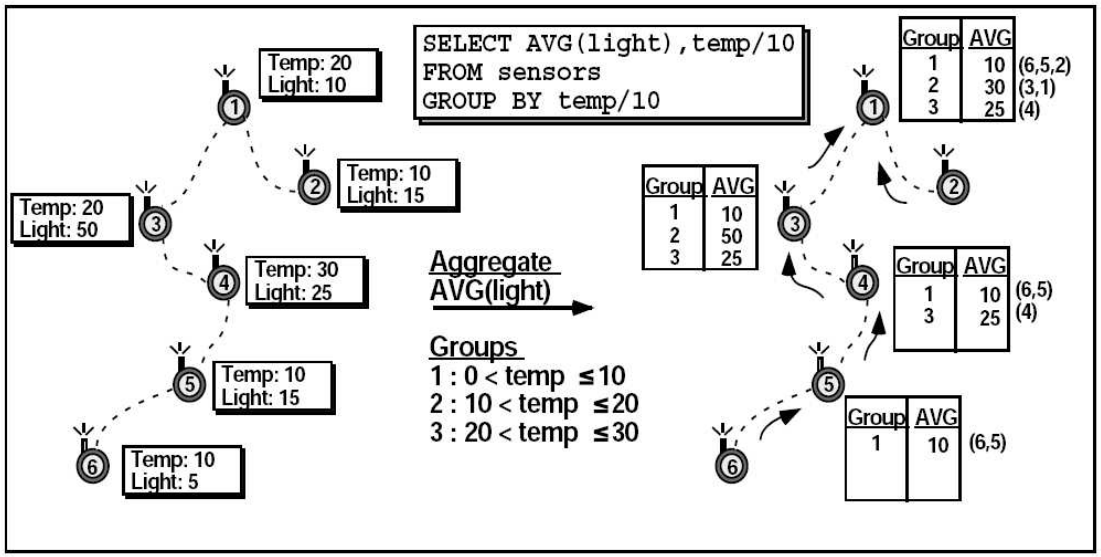


Figure 1: *Partial state records flowing up the tree during an epoch.*

Figure: Partial state record flow

Collection phase (contd...)

- Grouping



- Equivalent to the GROUP BY clause in SQL
- Partial state record are tagged with group id
- Internal node update aggregate value in an appropriate row
- Problem
  - Number of groups may exceed available storage of a node
  - Solution : evict one or more groups to parent which either continue forwarding up in the network or hold
  - Problem : number and size of message increase if proper group is not selected

Collection phase (contd...)

- Having clause
  - Depending on type of aggregate, Having clause is propagated into network which reduce the number and size of messages
  - MAX(temperature) per floor$<$ x
  - Groups with MAX(temperature) $>$ x need not be transmitted up the tree
  - Also node notify other nodes to suppress information for that group

Aggregates

- Structure of aggregates
  - Aggregates evaluated via three functions
  - Initializer : instantiate a state record for single sensor value
  - Merger function : merge partial states from two sensors into single partial state, $< z >= f(< x >, < y >)$
  - Evaluator function : takes partial state record and computes actual aggregate value
  - Example : Merger function for average
    $f(< S_1, C_1 >, < S_2, C_2 >) =< S_1 + S_2, C_1 + C_2 >$
  - Evaluator function returns S/C
- Taxonomy of aggregates
  - Duplicate sensitivity
  - Exemplary / summary aggregates
  - Monotonic aggregates
  - Partial state of aggregates

Aggregates (contd...)

Partial state of aggregates

- Distributive aggregates
- Algebraic aggregates
- Holistic aggregates
- Unique aggregates
- Content sensitive aggregates

	MAX, MIN	COUNT, SUM	AVERAGE	MEDIAN	COUNT DISTINCT [4]	HISTOGRAM [5]
Duplicate Sensitive	No	Yes	Yes	Yes	No	Yes
Exemplary (E), Summary (S)	E	S	S	E	S	S
Monotonic	Yes	Yes	No	No	Yes	No
Partial State	Distributive	Distributive	Algebraic	Holistic	Unique	Content-Sensitive

Table 1: Classes of aggregates

Attribute catalog

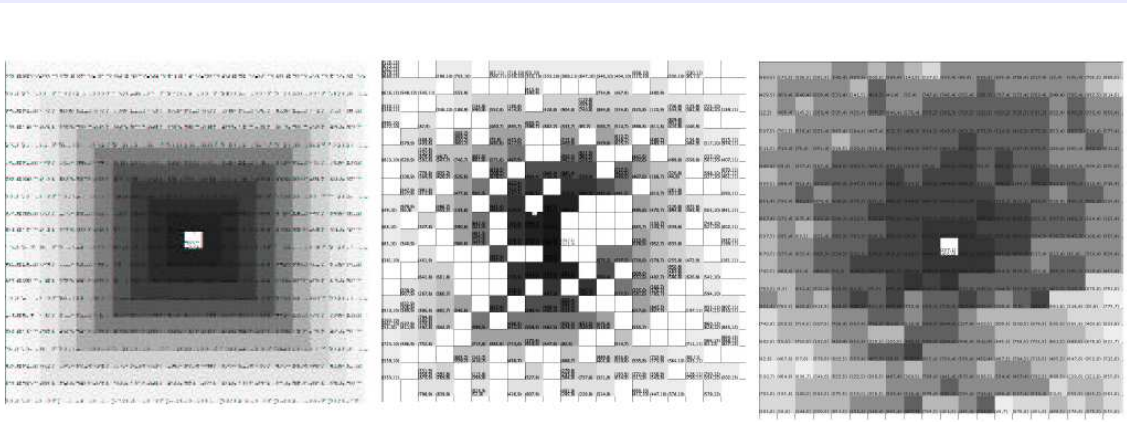
- Each mote contains catalog of attributes
- When mote receives a query, it converts named fields to catalog identifiers and tags missing attributes a NULL value in result
- All motes need not have similar catalogs, thus allowing heterogenous sensing capabilities
- Attributes in TAG can be direct representation of sensor values (light, temperature) or introspective (remaining energy, network neighbourhood information)

Additional advantages of TAG

- Ability to tolerate disconnections which are very likely in sensor networks
- Each mote is required to transmit one message per epoch, regardless of its depth in tree
- By explicit divisioning of epoch time, convinient mechanism for idling processors can be obtained

Simulation environment

- Simulator : time is divided into units of epoch, messages are encapsulated in Java objects
- No fine-grained model of time, hence radio contention not simulated



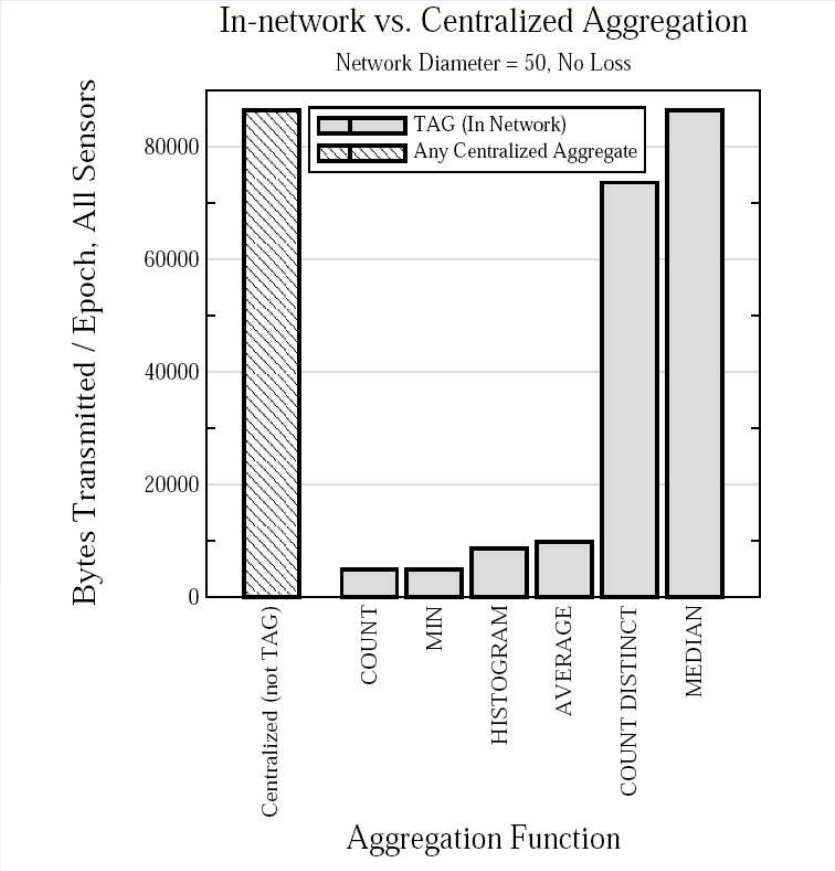
(a) Simple (b) Random (c) Realistic

- Experiments are run on three communication models
  - Simple model - nodes have perfect communication with their immediate neighbours
  - Random model - nodes are placed randomly
  - Realistic model - attempts to capture actual behaviour of the radio channel and link layer on TinyOs motes

Simulation environment (contd...)

- Realistic model
  - This model uses results from real world experiments to approximate the actual loss in radio channel communication
  - It models cost of topology maintainance

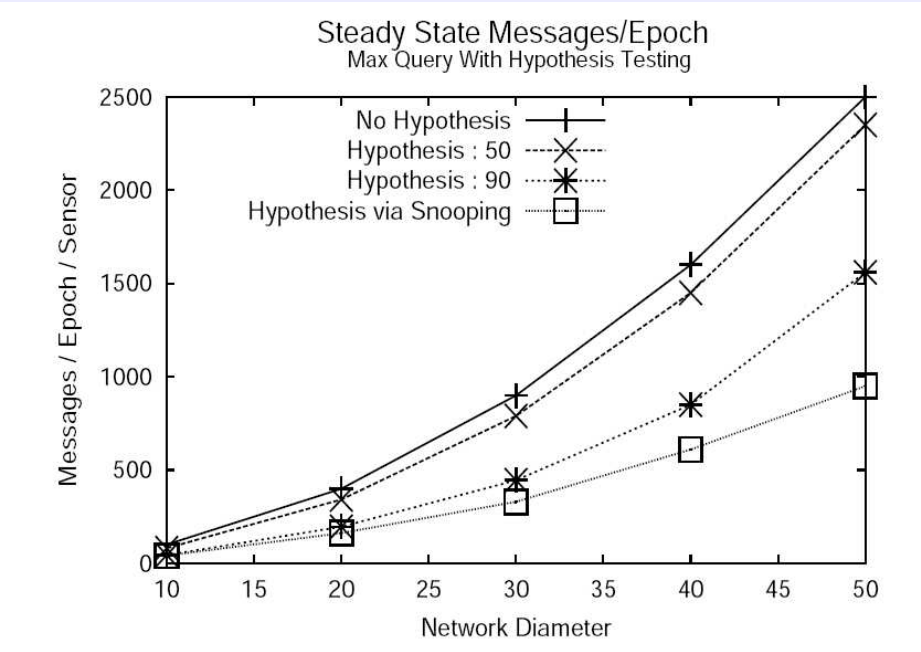
Performance of TAG



- Communication cost in centralized approach is independent of aggregate types
- Benefits of TAG are dependent on topology adapted

Optimizations

- Taking advantage of shared channel
  - Aggregation can be initiated even after missing the start request
  - Reduction in number of messages by snooping
- Hypothesis testing
  - Node can decide locally whether contributing its reading will affect the value of aggregate if it is provided with a guess of value of an aggregate



Improving tolerence to loss

Topology maintainance and recovery

Netorking faults are handled at two levels :

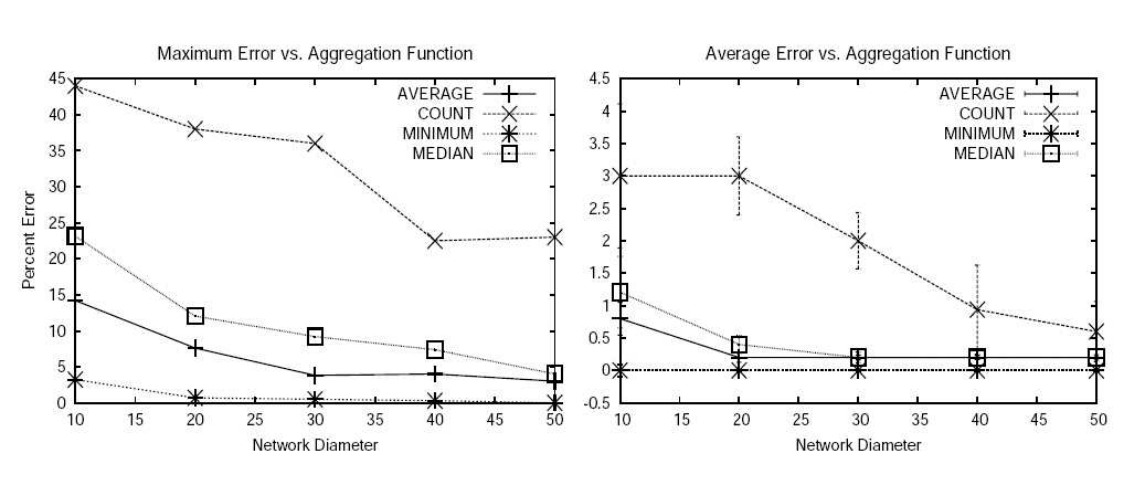
- Monitor quality of links
  - Each node maintains small, fixed-size list of neighbours
  - It monitors quality of link to each of the neighbours
  - If it observes that quality of its parent node is significantly worse than that of some other node p', it chooses p' as its new parent
- If parent is failed or moved away
  - If a node observes it had not heard from its parent for some fixed amount of time, it resets its level to ∞ and picks up a new parent from neighbours table

Improving tolerence to loss (contd...)

Effects of single loss:

- Because of hierarchical aggregation, single node going offline can cause the entire subtree rooted at the node to be disconnected
- Experimental setup : After running for several epochs, a random node is selected to disable
- The maximum and average temporary deviation from true value of aggregate is measured

Improving tolerence to loss (contd...)

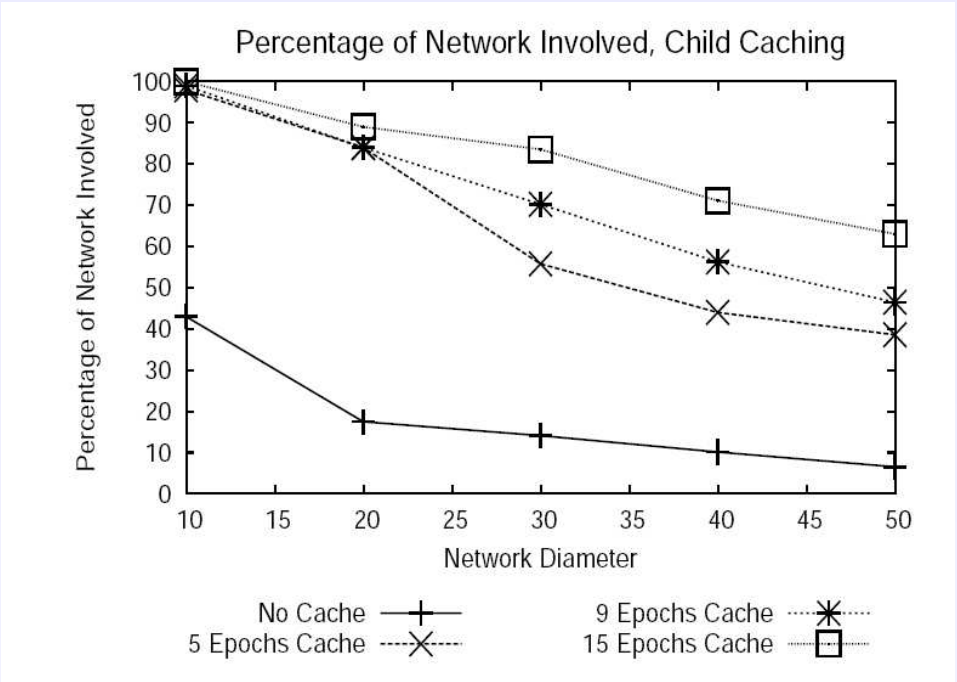


- As the network diameter increases, the percentage of error decreases because probability that a node in the upper part of the tree goes down is lesser

Improving tolerence to loss (contd...)

Effect of realistic communication:

- Without some technique to counteract loss, large number of partial state records are dropped and does not reach the root of the tree
- At network diameter about 10, only 40 % partial state records are reflected in the aggreagate result, whereas at network diameter 50, it falls down to 10 %



Improving tolerence to loss (contd...)

Child cache:

- Parents remember the partial state records of their childs for some rounds
- These values are used by parent if new values from children are not available due to loss
- Drawbacks of caching :
  - Memory usage which can be used for group storage
  - Sets minimum bound on time that devices must wait before determining their parent has gone offline
- Drawbacks are compensated by benefits in terms of accuracy

Improving tolerence to loss (contd...)

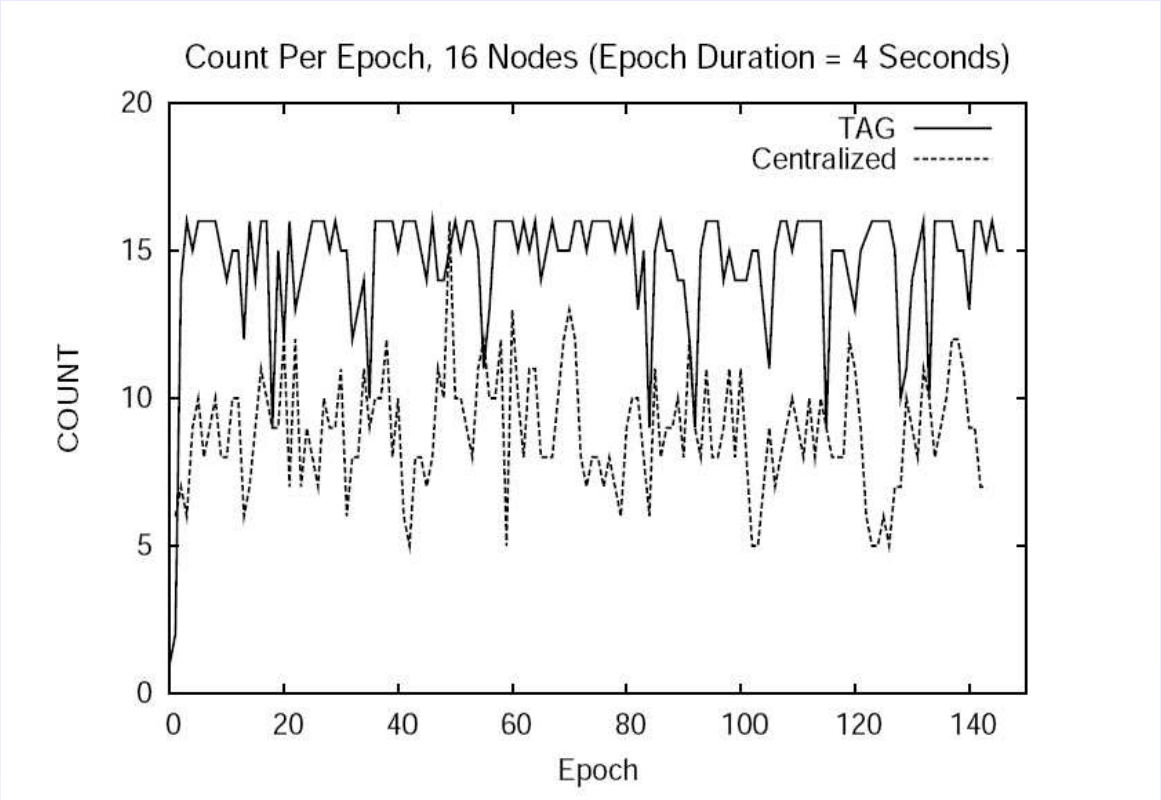
Using available redundancy:

- An alternate choice to child cache, since child cache might not be always desirable
- Cosider a node with two possible choices of parent. Instead of sending data to one parent, it can send it to both parents
- For duplicate sensitive aggregates, it can send part of aggregate to one parent and rest to the other

Prototype implementation

- The prototype does not include many of the mentioned optimizations, but contains the core TAG aggregation algorithm
- Simulations results mentioned are consistent with actual behaviour and substantial message reductions are possible over centralized approach
- No chlid caching nor snooping techniques used

Prototype implementation (contd...)



- Prototype setup : sixteen motes arranged in a tree(depth=4), computing a COUNT aggregate over 150 4-second epochs

Balancing Energy Efficiency and Quality of Aggregate Data in Sensor Networks

{Mohamed Sharaf, Jonathan Beaver, Alexandros, Panos}

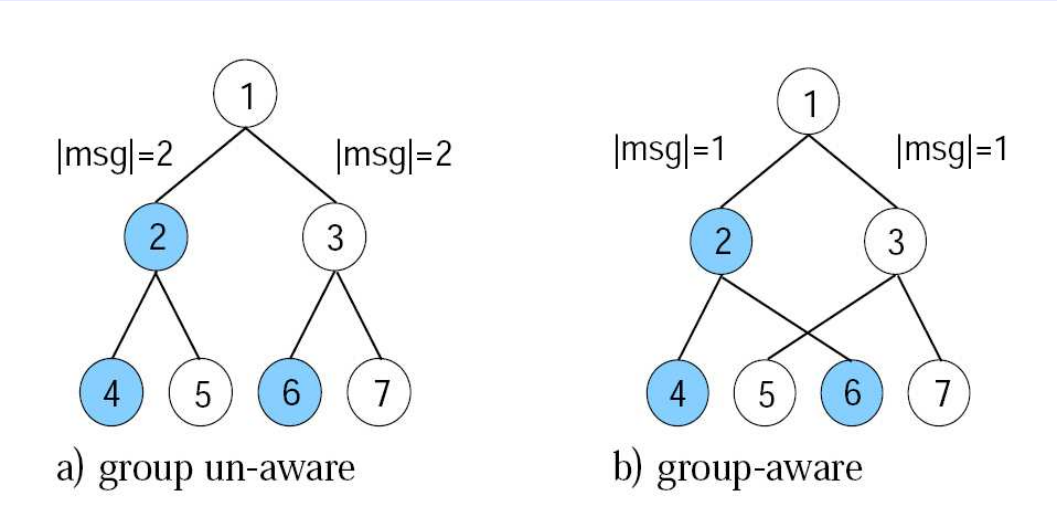
Improvements over TAG

- Influencing the construction of the routing trees for sensor networks with the goal of reducing the size of transmitted data
- Exploiting temporal correlation in streams of sensor reading to suppress insignificant readings which can be tolerated

Group-aware Network Configuration (GaNC)

- First-Heard-From (FHF) protocol of tree construction (TAG)
- List of aggregates in Group-By clause subdivides the query result into a set of groups
- Two readings from two different sensors can be aggregated only if they belong to the same group
- Hence, creating a routing tree that keeps members of the same group in the same path of routing tree helps decrease the energy usage, which is done by GaNC
- The GaNC works the same way as FHF, with difference that during tree construction, child can switch to a better parent
- Tie-breaker conditions
  - group id
  - distance factor

GaNC (contd...)



- Nodes 2,4,6 belong to same group and nodes 3,5,7 belong to same group

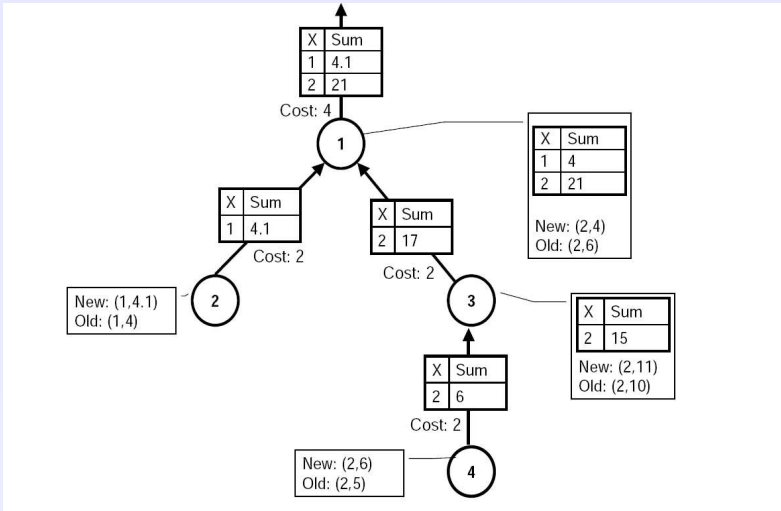
Temporal coherency-aware in-Network Aggregation (TiNA)

select {*agg(expr)* , *attrs*} *FROM sensors*
where {*selPred*}
group by {*attrs*}
having {*havingPreds*}
epoch duration i
tolerance tct *: Introduced by TiNA*

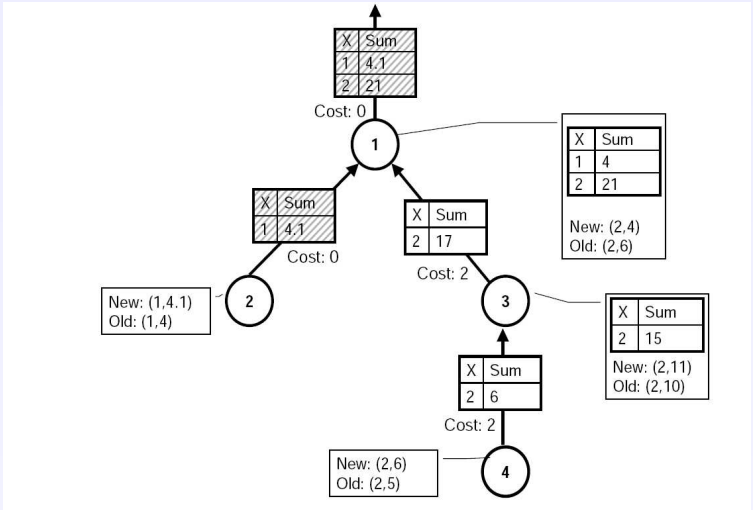
- The tct values act as an output filter, suppressing readings within the range specified by tct
- Sensor node must keep additional information to utilize temporal coherency tolerance
- Leaf node maintain its previous aggregate data, where as internal nodes maintain data received from each child along with its own data

TiNA (contd...)

- Query using TAG



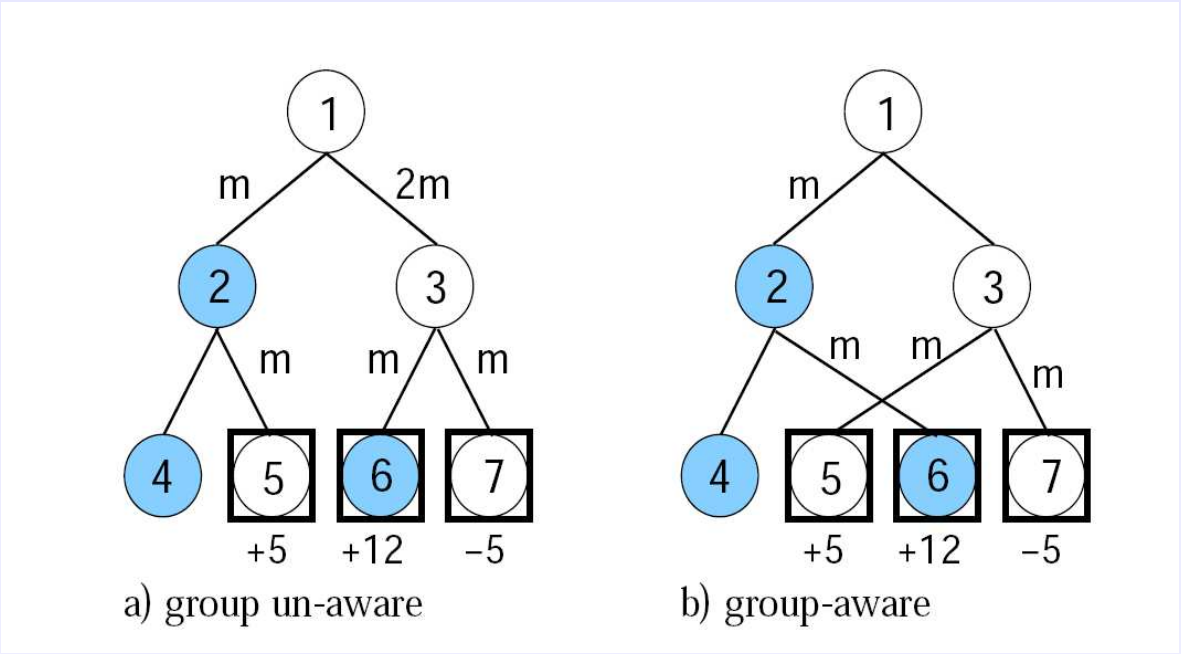
- Query using TiNA on top of TAG



TiNA (contd...)

- Distinction between the cases when data is not received from child node
  - value not meeting the condition of WHERE clause
  - value inside the tolerance range
  - child node is dying
- Solution : Notification message

TiNA with GaNC

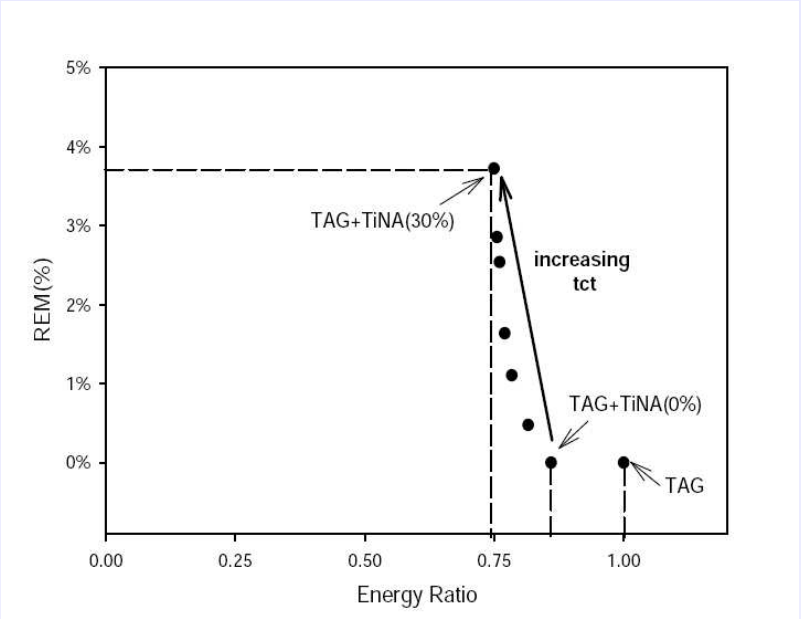


- Further improve its energy efficiency
- Example: Keeping track of the number of people in the building group by floor number

Experimental results

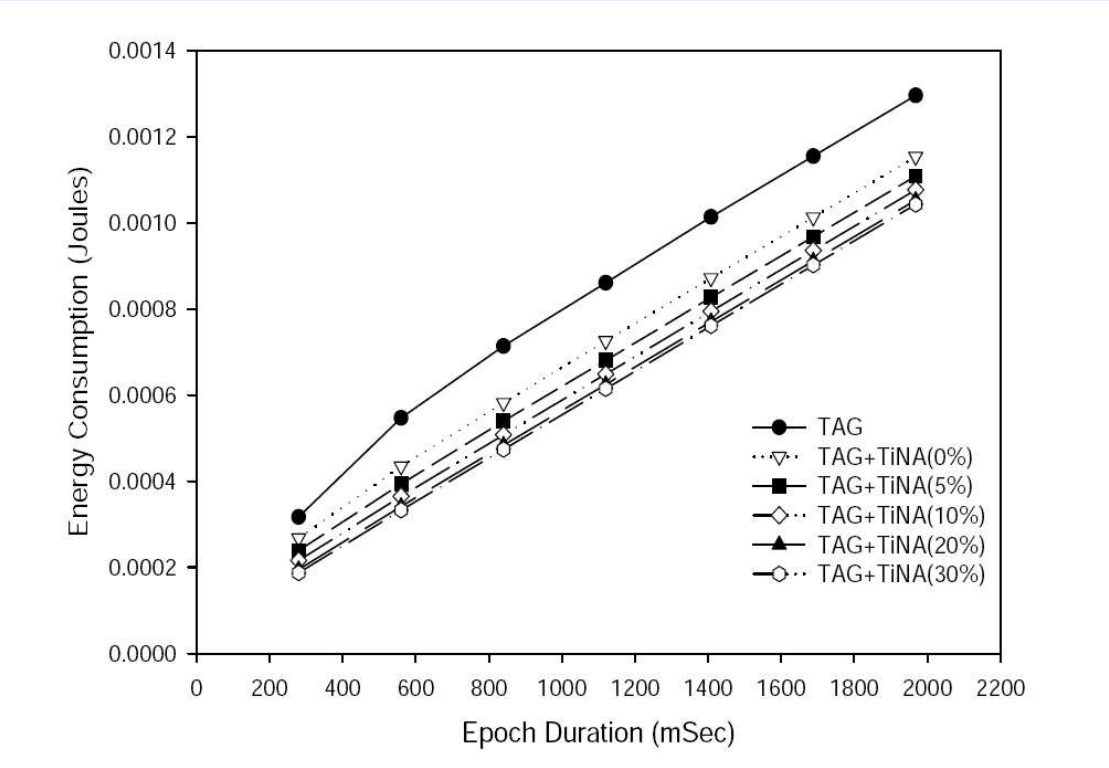
- Performance metric
  - Energy : consumed in transmitting, listening, processing, and sampling.
  - Relative Error Metric : how close the exact and approximate answer are

Sensitivity to temporal coherency tolerance



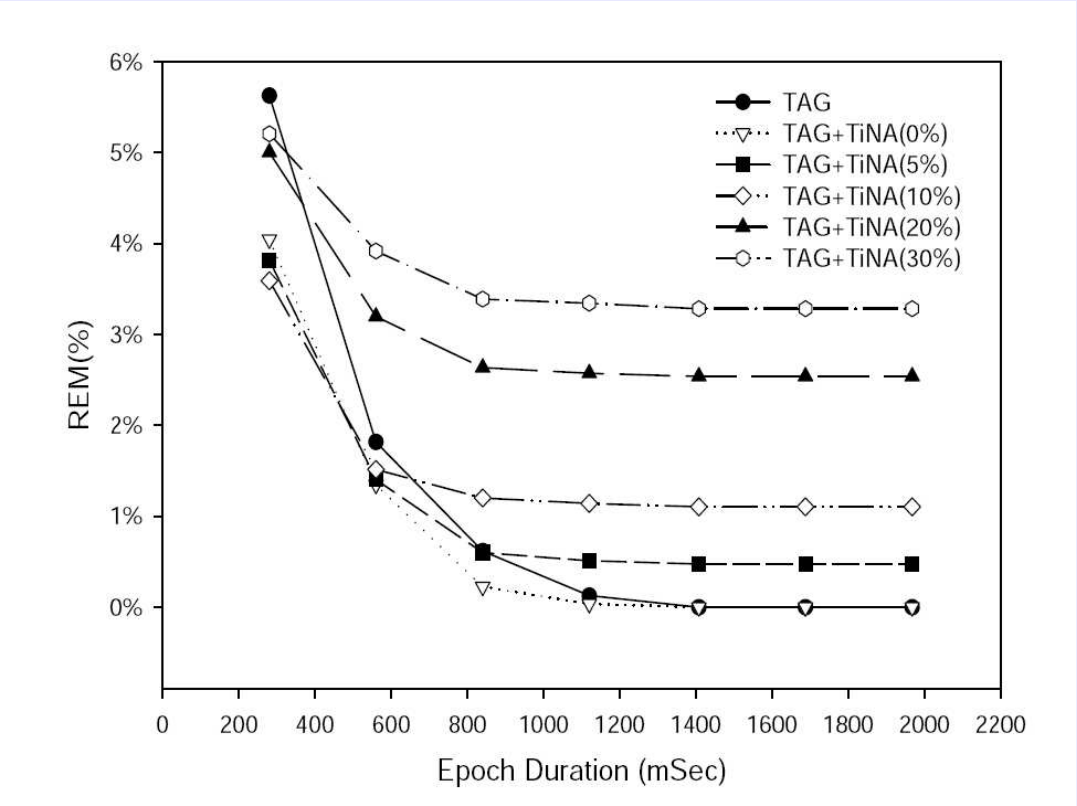
- Vary tct values and measures REM and energy
- As tct increases, REM increase and energy reduces

Sensitivity to Epoch duration



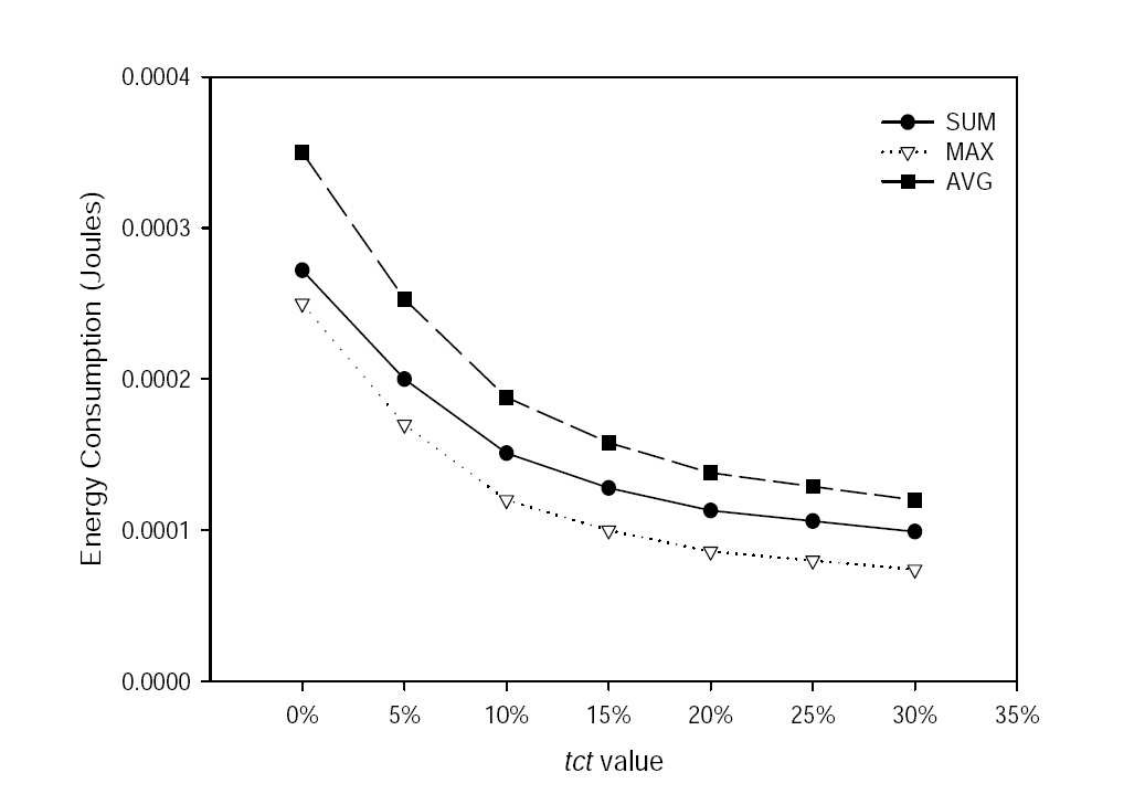
- As epoch duration increases, energy consumption increase but it give more correct result
- In short duration, parent may not able to listen to all child nodes

Sensitivity to Epoch duration (contd...)



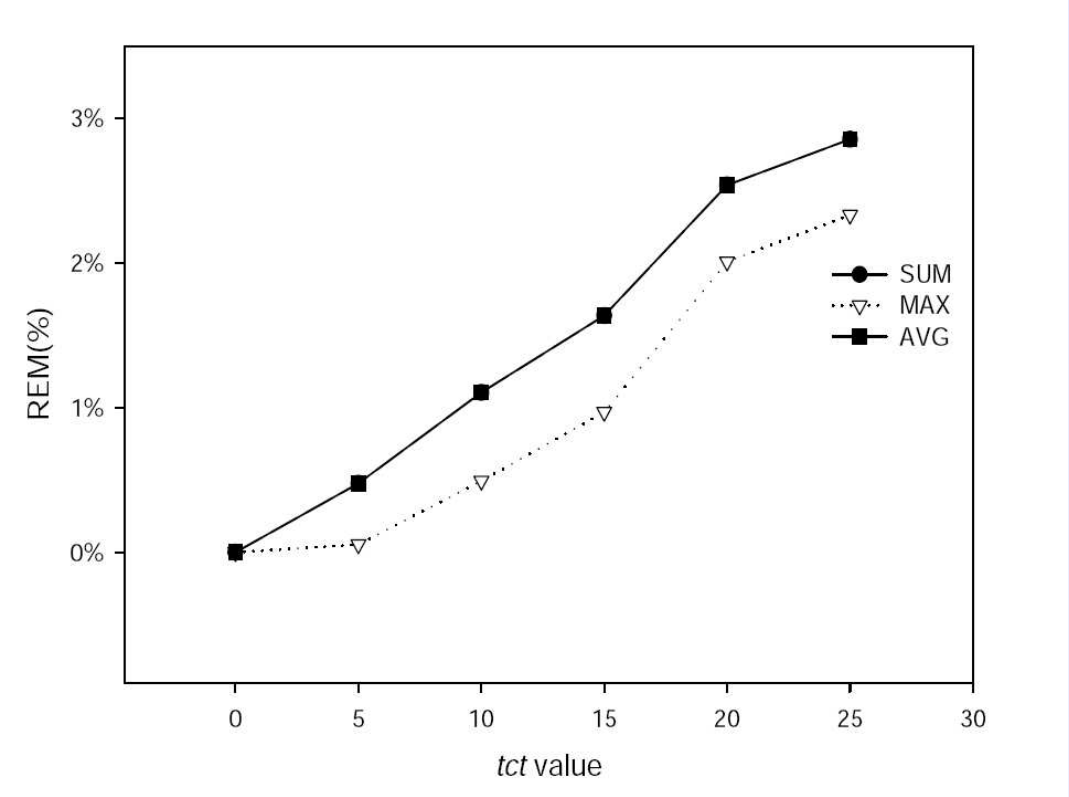
- As epoch duration increase, REM decrease
- After some duration, it stabalises when parent nodes get values from all the required child nodes

Sensitivity to aggregate function



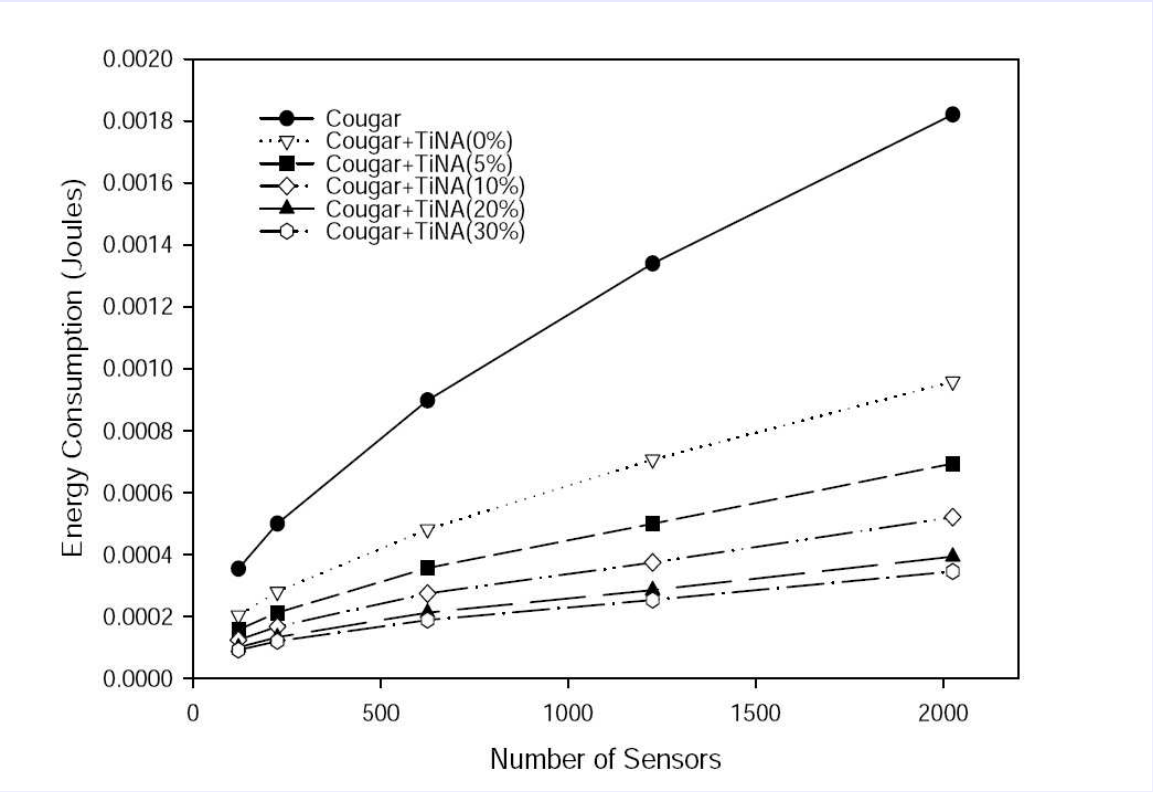
- As tct value increase, energy consumption decrease
- MAX has less energy consumption value change with less probability

Sensitivity to aggregate function(contd...)



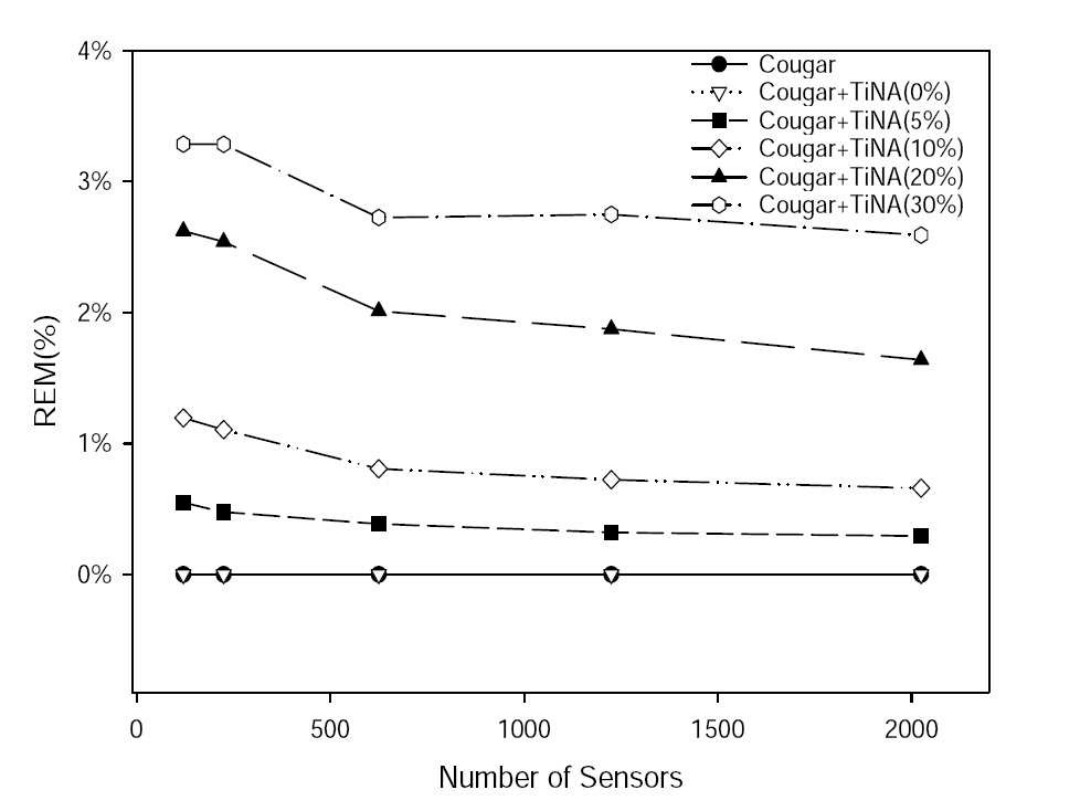
- MAX has less REM as compared to aggregate

Scalability



- Energy consumption is directly propotonal to number of sensors
- TiNA reduces energy consumption to large extent for large number of sensors

Scalability (contd...)



- As the network size increase, there is a better chance that parent's group is sent up already

Conclusion

- TAG
  - Declarative query interface
  - Reduced number of messages hence less power consumption
- GaNC: New network configuration: reduce number of messages
- TiNA: Reduce energy consumption by reducing number and size of messages