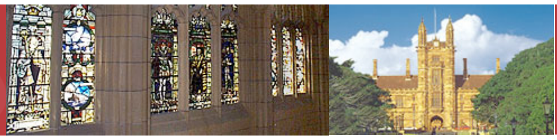


Adaptive In-Network Query Processing for Data-Intensive Sensor Networks

Uwe Röhm

University of Sydney
School of Information Technologies

Research Motivations

- Wireless Sensor Networks (WSNs) of growing importance
  - automatic and continuous monitoring of physical phenomena
    - e.g. structural or environmental monitoring
  - WSN can consist of hundreds of sensor nodes
- Data-Intensive Sensor Networks
  - focus on data acquisition
    (“How to get the data out of the field?”)
  - data-centric, high-level abstraction wanted

- Sensor hardware increasingly more powerful
  - more built-in memory
  - more powerful processors
  - IEEE802.15.4 radio standard

Improved WSN Hardware

	Berkeley Mote	BTnode	Spec	Intel iMote	Intel iMote2 (gateways)	Sun SPOT
Year	2001	2001	2003	2003	2005	2006/2007
Vendor	UC Berkeley; now CrossBow	ETH Zurich	UC Berkeley	Intel Research Berkeley	Intel Research Berkeley	Sun Microsystems
CPU	4 MHz 8 bit Amtel	7.3 MHz ATMEL Mega	4-8MHz AVR-like RISCcore	12-18 MHz ARM 7TDMI	13-416 MHz 32bit XScale	180 MHz 32bit ARM 9
RAM	4 K RAM 128 K Prog Flash 512 K Data Flash	4 K EEPROM 128 K Flash	3 K RAM	64 K RAM 512 K Flash	32 M SDRAM 32 M Flash	512 K RAM 4 M Flash
Radio	40 kB Radio	Bluetooth	FSK radio…	Bluetooth 1.1 ~30m range	IEEE 802.15.4 (ZigBee)	IEEE 802.15.4
OS	TinyOS	own	TinyOS	TinyOS	TinyOS / Linux	Squawk (J2ME)
Battery	AA battery pack			1 CR2 battery		750 mAh LiON
Energy	active: ~24 mW sleep: ~45 uW	active:285 mW idle: 50 mW	peak: 3 mW idle: 3 uW	active: ~120mW idle: ~ 1 mW		deep sleep:~32uA

Resource Constraints in Sensor Networks

- Processing Power: started very low, becoming better
- Memory: 512KB already, megabytes in the future
- Radio: limited bandwith and reliability
- **Battery**: very limited
  - typically running on AA batteries

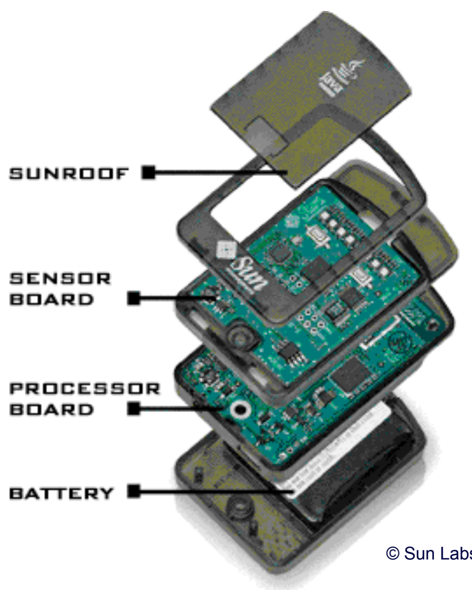
- Energy Efficiency of highest priority
  - Priority 1: Minimise Communication / Radio Usage
    - Power to transmit 1 bit = 100s of instructions
  - Priority 2: Minimise Sensor Usage
    - some sensors have very high activation costs
  - Priority 3: Minimise CPU usage
    - maximize sleep periods

General Research Goals

- Hardware progress allows for more advanced processing
  - => true **in-network** data processing
    such as distributed event detection or data clustering
    (going beyond simple 'sense - collect - deliver' tasks)
- Resources are scared and precious in WSNs
  - => need good resource utilization & adaptiveness to changes
  - => **resource awareness**
- To be real successful, WSNs need high-level interfaces
  - Declarative Query Interface
    - => **data abstraction layer**
  - Virtual Machine Technology
    - => small code sizes, rapid prototyping

Sun™ Small Programmable Object Technology

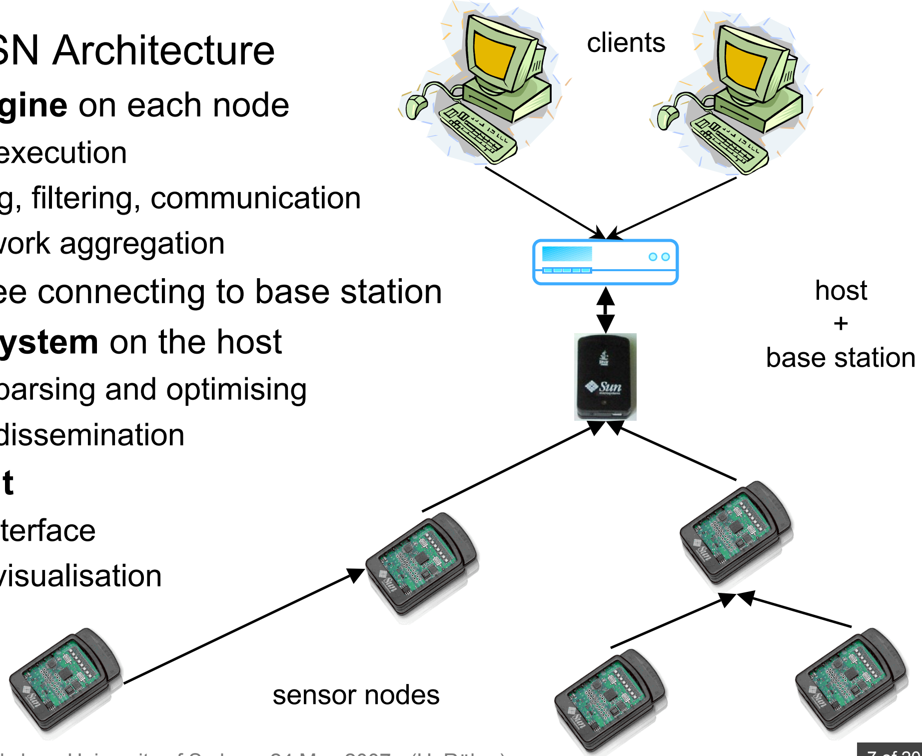
- Sun SPOT devices
  - battery board
    - 750mAh Li-Ion battery
  - processor board
    - 180 MHz ARM 9
    - 512 K RAM / 4M Flash
    - 802.15.4 radio
  - sensor board
    - temp, light, accelerometer, …
- Java VM on the bare metal
  - Squawk JVM (J2ME)
  - small program code
  - rapid prototyping



© Sun Labs

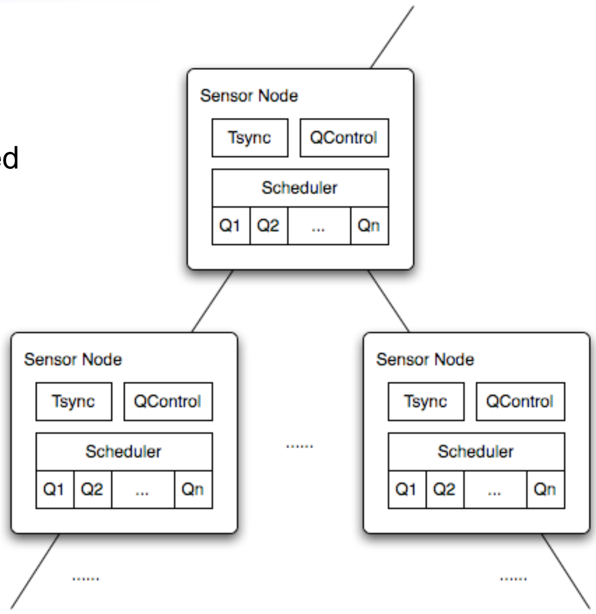
SSDQP - Sun SPOT Distributed Query Processor

- Classic WSN Architecture
  - **query engine** on each node
    - query execution
    - sensing, filtering, communication
    - in-network aggregation
  - routing tree connecting to base station
  - **control system** on the host
    - query parsing and optimising
    - query dissemination
  - **GUI client**
    - user interface
    - result visualisation



SSDQP Core Features

- Data-Program Independence
  - Data abstraction:
    virtual relation; horizontally partitioned
  - SQL-like queries
- Multi-Tasking/-Querying
  - WSN shared by several users
- Time Synchronization
  - query engine is time-triggered
- Adaptive In-Network Processing
- Optimised Messaging
- Graphical UI



Data Independence via SSDQP

- SQL data abstraction of sensor network
  - Virtual relation that is horizontally partitioned over all nodes
    - Meta-attributes
      (ID, time, parent, …)
    - Resource attributes
      (battery level, free memory, CPU load)
    - Sensors
      (light, temperature, x/y/z accelerometer, …)
    - Actuators *(currently read-only)*
      (LEDs, buttons)
- Acquisitional-SQL query language

```
SELECT attributes
  FROM sensors [, buffer]
 WHERE condition
 START AT    timestamp
 PERIOD      duration
 RUNCOUNT    count
```


Distributed Query Execution

- Client
  - **Query** entered via GUI
- Host
  - compiles query into **query execution plan**
  - disseminates query plan into network
- Nodes
  - instantiate new **query task** ; scheduled for specified start time
  - periodically execute according to query sample interval specification
  - sensing, processing and communication are separate tasks
    - Sharing of sensor readings between multiple queries
    - Minimising sensor activation and communication

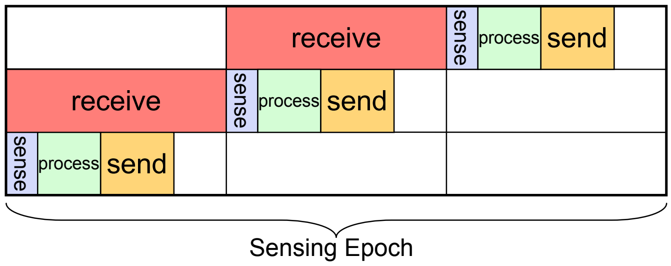
forward
|
merge
|
filter
|
sense

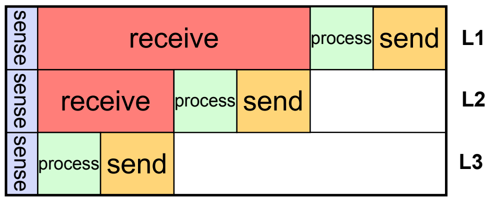
Waking Window Optimisation

[ICESS2007]

- **TinyDB**
  - Waking window length := EPOCH length / *d*
  - *d* : depth of the routing tree
  - All nodes have the same waking window length
  - Length typically overestimated, i.e. it is longer than needed
- **SSDQP**
  - Waking window length := waiting for all children to answer
  - Synchronised sense on all nodes
  - Upper nodes have longer waiking period that depends on subtree
  - Overall shorter communication length

Example: *d* = 3





In-Network Data Clustering

[CIDM2007]

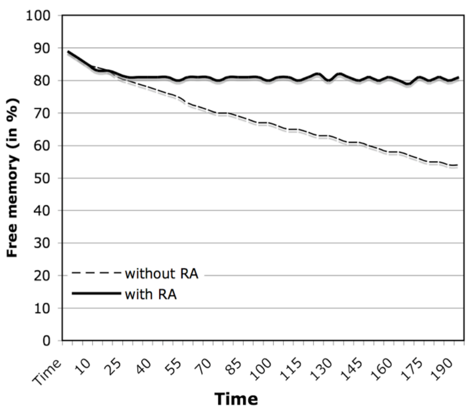
- Goal: true **in-network** data processing
  - instead of collecting all raw data from sensors & then process at host
- ERA-Cluster Algorithm
  - own resource-aware online data clustering algorithm for WSNs
  - provides distance-based clustering of sensor readings
    - assign sensor measurement to nearest existing cluster
      - updated with weighted average of new value and existing cluster
    - if no cluster within given distance threshold, start new cluster
    - result is { (centroid,weight) } of clustered sensor readings
  - Example:
    - SELECT *
      FROM CLUSTER('temperature', *threshold*)
    - keeps internal state

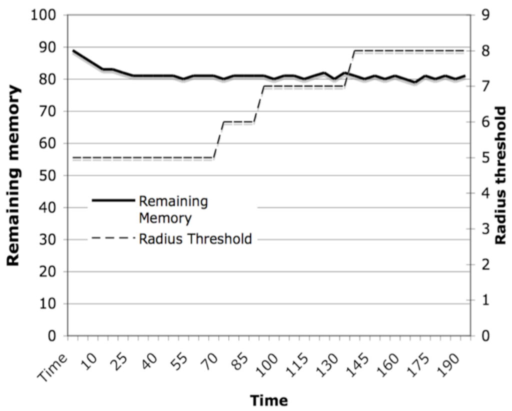
In-Network Resource-Awareness

- Goal: adaptive, resource-aware processing
- Approach: resource monitoring framework
  - Self-reflective resource attributes (memory, battery, load)
  - resource monitor task on each node
- Adaptive clustering algorithm:
  - if battery level exceeds a preset threshold:
    - frequency of sending/receiving data is reduced -> possible because of dynamic scheduler
  - if free-memory exceeds a preset threshold:
    - discourage of new clusters formation is applied through a distance threshold approach
    - in addition continuous release of inactive clusters
  - If CPU load is high:
    - randomization of the result space is applied

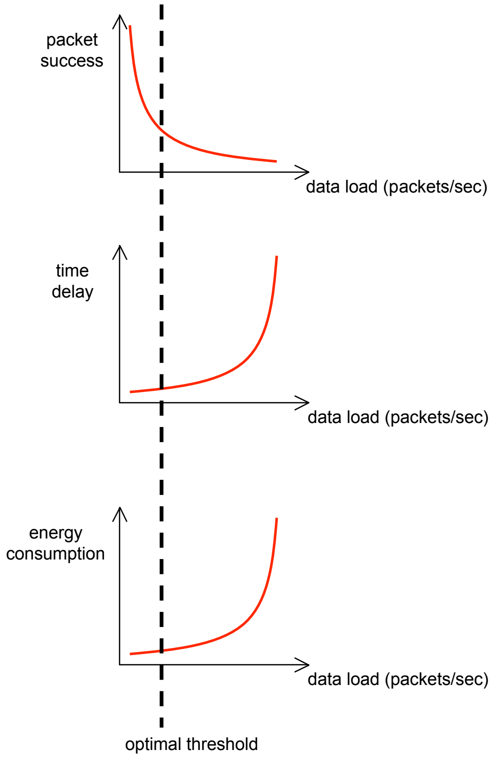
Some Evaluation Results

- Evaluation of *validity* of the approach in terms of *resource-awareness* and *accuracy* of the adaptive mining algorithm
  - ERA-Cluster can effectively adapt to resource availability while maintaining acceptable level of accuracy [CIDM2007]





Performance Effect of High Transmission Rates



- Problem:
  The higher the packet rate
  - the lower the packet success rate (e.g. because of network collisions)
  - the higher the time delay
  - the higher the energy consumption
- Hence overall WSN goal:
  find appropriate threshold of good packet rate that balances errors, delay and energy consumption.

Solution: Asynchronous Data Acquisition [DISN2007]

- Multi-tasking allows for de-coupled processing and communication:
  - E.g. clustering and querying can be scheduled with different frequencies
- **Clustering task**
  - clusters current sensor readings
  - state kept locally on node
  - into **local buffer**

```
  SELECT *
    INTO TempClusters
    FROM CLUSTER(temp,t)
  PERIOD 1s
RUNCOUNT 10000
```

- **Query task**
  - Retrieves data clustering results from storage point either periodically or on-demand

```
  SELECT *
    FROM TempClusters
  PERIOD 60 s
RUNCOUNT 60
```

Comparison to Related Work

- TinyDB
  - The 'golden standard' of acquisitional query processors
  - Compiled down to machine code running on TinyOS
  - Very limited multiple-queries, no dynamic adaptiveness
    - LIFETIME clause statically pre-computed at host
- SwissQM
  - Own virtual machine, merging query- and sensor-specific opcodes with JVM opcodes
  - Queries compiled into VM code and then disseminated and executed
  - Multiple queries (apparently?)
  - No declarative adaptiveness, not time triggered
- SSDQP
  - Running on JVM, but using standard ACQP approach for queries
  - Multiple queries, time triggered, dynamic adaptiveness

Conclusions and Outlook

- Next generation of more powerful sensor nodes allows for complex in-network data processing in WSNs
- SSDQP: a powerful platform for distributed data acquisition
  - data abstraction layer
  - sharing of network via multiple queries / tasks
  - resource-awareness / dynamic adaptivity of tasks
  - allows for decoupling sensing and data acquisition
- Ongoing Work:
  - Next slide

WSN Research @SIT

- Researchers:
  - Selvakennedy Selvadurai, Bernhard Scholz, Uwe Roehm, Mohamed Gaber
- Many special thanks to all our project students who developed the prototype systems
  - Tim Dawborn
  - Raymes Khoury
  - Edmund Tse
  - Quincy Tse
  - Duc Nhan Phung
- This work is supported by
  - Australian Research Council (ARC) under grant ARC DP0664782
  - Sun™ Microsystems Laboratories

Overview of WSN Projects @SIT

- T-Ant dynamic Network Clustering Protocol
- ERA-Cluster: In-network resource-aware data clustering
- SSDQP: Sun SPOT Distributed Query Processor
- In-network Data Stream Processing in WSNs
- Event Processing Middleware:
  - Event Boundary Detection
  - Higher-Level Processing
- Applications:
  - Emergency Evacuation System
  - Building monitoring system

Ongoing work