SmartCIS: Integrating Digital and Physical Environments

Mengmeng Liu, Svilen Mihaylov
Zhuowei Bao, Marie Jacob,
Rahul Iyer, Easwaran Subbaraman,
Zack Ives
Motivation

• Green and Smart Computing
  – Intelligent power grids
  – Smart hospitals
  – Home health monitors
  – Energy efficient data centers
  – building visitor guides.

• Integrating sensor, stream, web and database data
  – **Sensor**: Temperature, Light, Energy level, Seat-occupied, RFID, ...
  – **Stream**: CPU-usage, TCP packets, ...
  – **Web**: Calendars, ...
  – **Database**: Labs, Machines, Softwares, People,...
Challenges

• Developing new query execution and optimization algorithms suitable for integrating highly distributed data sources (low-power sensors + PCs).

• Developing query optimization techniques across a federation of stream processors specialized for sensor, WAN and LAN settings.

• Developing new datatypes, query extensions, and data description language abstractions for environmental monitoring and for routing information to users.

• A unified declarative query and integration substrate!
Goals

• **High-level goal:**
  – Multi-purpose data acquisition and integration
  – Extensible for new device types, new network types, and new datatypes

• **Specific goal in this project:**
  – Room monitoring
  – Machine-state monitoring
  – Workstation monitoring
  – Detection of occupants
  – Databases and Web sources
Stream SQL Parser (Zhuowei)

- Eclipse DataTools
- Parse queries into our internal abstract syntax tree presentation
  - Relations, Head
  - EquivClass \( (a = b) \), Predicate\( (a < b) \)
  - GroupBy, Aggregate, Function
- Parse window specifications in Stream SQL
  - Window specs:
    \[
    \text{SIZE size ADVANCE increment} \{\text{TUPLES | TIME}\}
    \text{[PARTITION BY field_id[, ...]]}
    \]
  - E.g.:
    \[
    \text{SELECT * FROM A [SIZE 1 ADVANCE 1 TUPLES PARTITION BY A.a]}
    \]
Catalog (Marie)

- BerkeleyDB
- What information needs to be stored in catalog?
  - Stream/relation names and their schemas
  - Indexes on each relation, including their types and parameters
  - Number of sensors and machines in the network, node IDs/IP-ports etc
  - Lists of static and dynamic attributes
  - Initial selectivities for dynamic streams/relations
  - Statistics (cardinalities of static relations, histograms, min/max values,... )
Federated Optimizer (Mengmeng)

• Partition a single global query into segments that can be pushed to either stream or sensor engine
• Validate a query based on capabilities of subsystems
  – E.g.: flexible enough to different number of sensor joins (1? 2? 3? ...)
• Heuristically determine whether or not to execute a query segment in the sensor network
  – Always push selections and aggregates on sensor relations to sensor engine
  – Push joins if it is capable of doing so
  – Long-term goal: cost-based federated optimizer based on statistics from wrappers of subsystems
• Determine layers between sensor/stream engine
Stream Optimizer & Execution
(Mengmeng & Marie)

• Distributed stream engine
  – Support a stream of insertion/deletion tuples as input data
  – Built-in distributed operators: selection, projection, join, aggregate, group by, union, fixpoint, scan, ship
  – w/o provenance annotations on tuples

• Routing
  – Pastry DHT Networks (Mengmeng)
  – Mesh Networks (Marie)

• Optimizer
  – Volcano-style cost-based optimizer based on statistics such as cardinality/selectivity, and calibration information (cpu, bandwidth, etc)
  – How to estimate cardinality/selectivity in streams? Sketches rather than histograms!

• Stream Wrappers
  – Monitoring running tasks
  – Monitoring network usage
  – PDU Power usage
  – Calendars, Maps
Sensor Optimizer & Execution (Svilen)

- **Supported queries**
  - Selections and Projections
  - In-network sensor joins (single join)
  - Aggregations and Functions
  - Support Stream SQL windows
- **Indexes**
  - Bloom Filter
  - Histograms
- **Optimizer**
  - Optimizing number of messages being transmitted
- **Base Station Server Protocol**
  - Federated optimizer sends sensor query segment to the base station server
  - Base station server returns a series of tuple results to stream engine
  - It also provides statistics info e.g.: min/max value
Sensor Devices (Rahul)

• Sensor device types
  – IRIS
  – Imote2

• Sensor measurements
  – Accelerometers, light, pressure, temperature, humidity, GPS

• NesC language
  – Users write NesC programs, tools are setup to execute them on IRIS or IMote2 platforms

• TinyDB
  – A query processing system deployed on these motes to extract information from the network
SmartCIS GUI (Easwaran, Zack)
Progress Report

• Done
  – SQL parser (Zhuowei)
  – Catalog (Marie)
  – Federated optimizer (Mengmeng)
  – Stream execution engine (Mengmeng)
  – Sensor optimizer & execution engine (Svilen)
  – NesC apps IRIS/iMote2 (Rahul)
  – GUI (Easwaran, Zack)
  – A demo paper submitted to SIGMOD (all)

• In Progress
  – Stream SQL features
  – Mesh Networks
  – Stream wrappers
  – Stream engine talking to sensor engine
  – TinyDB deployment on motes
  – More GUI functionalities

• To-Do
  – Enhancing federated optimizer
  – Translating user preferences to SQL queries
  – Stream optimizer
  – Displaying results on the GUI
Thanks!