Developing and Deploying Sensor Network Applications with AnduIN

Daniel Klan
Katja Hose
Kai-Uwe Sattler

Databases & Information Systems Group

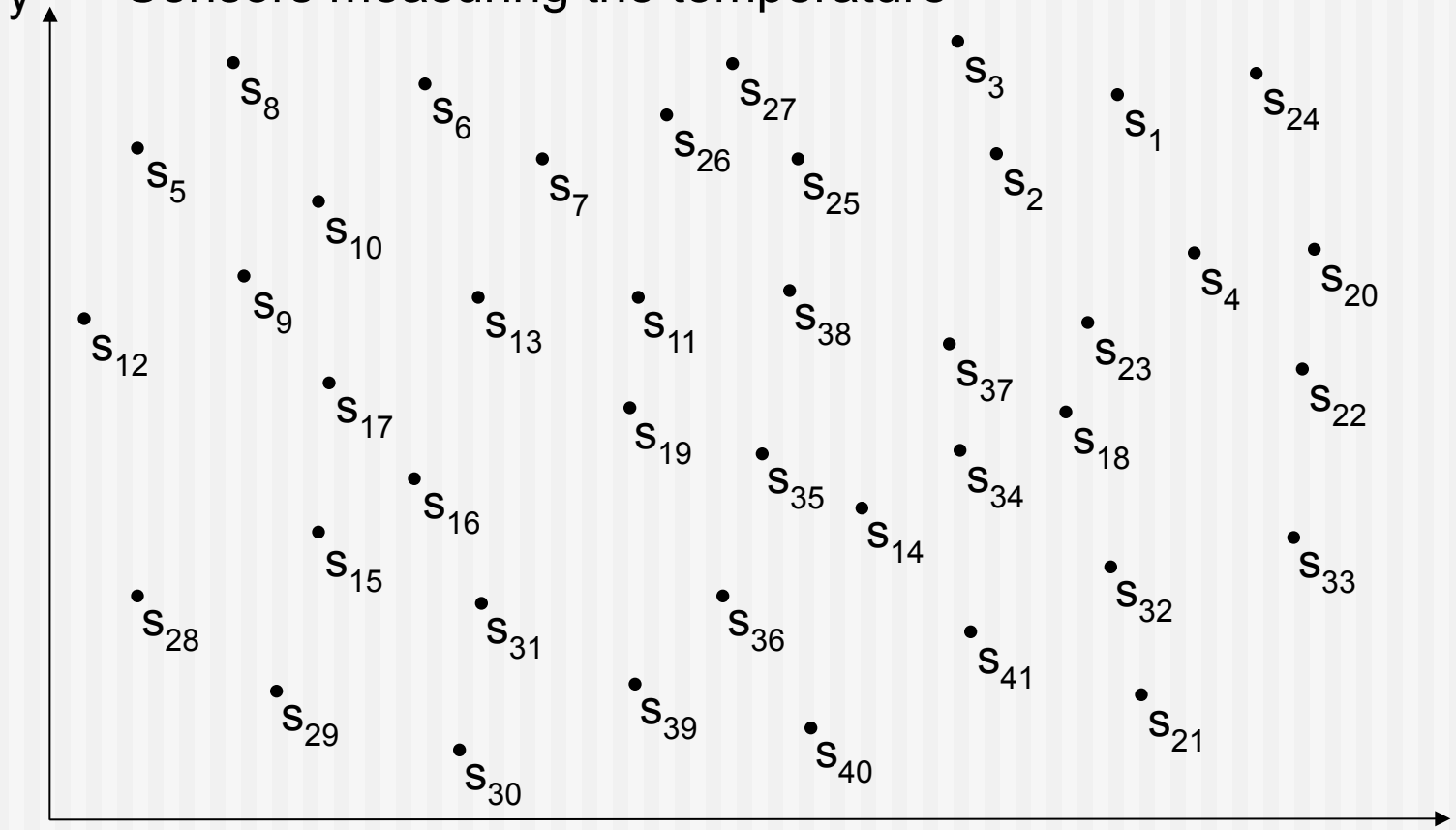
Ilmenau University of Technology, Germany

Outline

- Introduction
- AnduIN
- Query Specification
- Query Optimization
- Case Study
- Conclusion

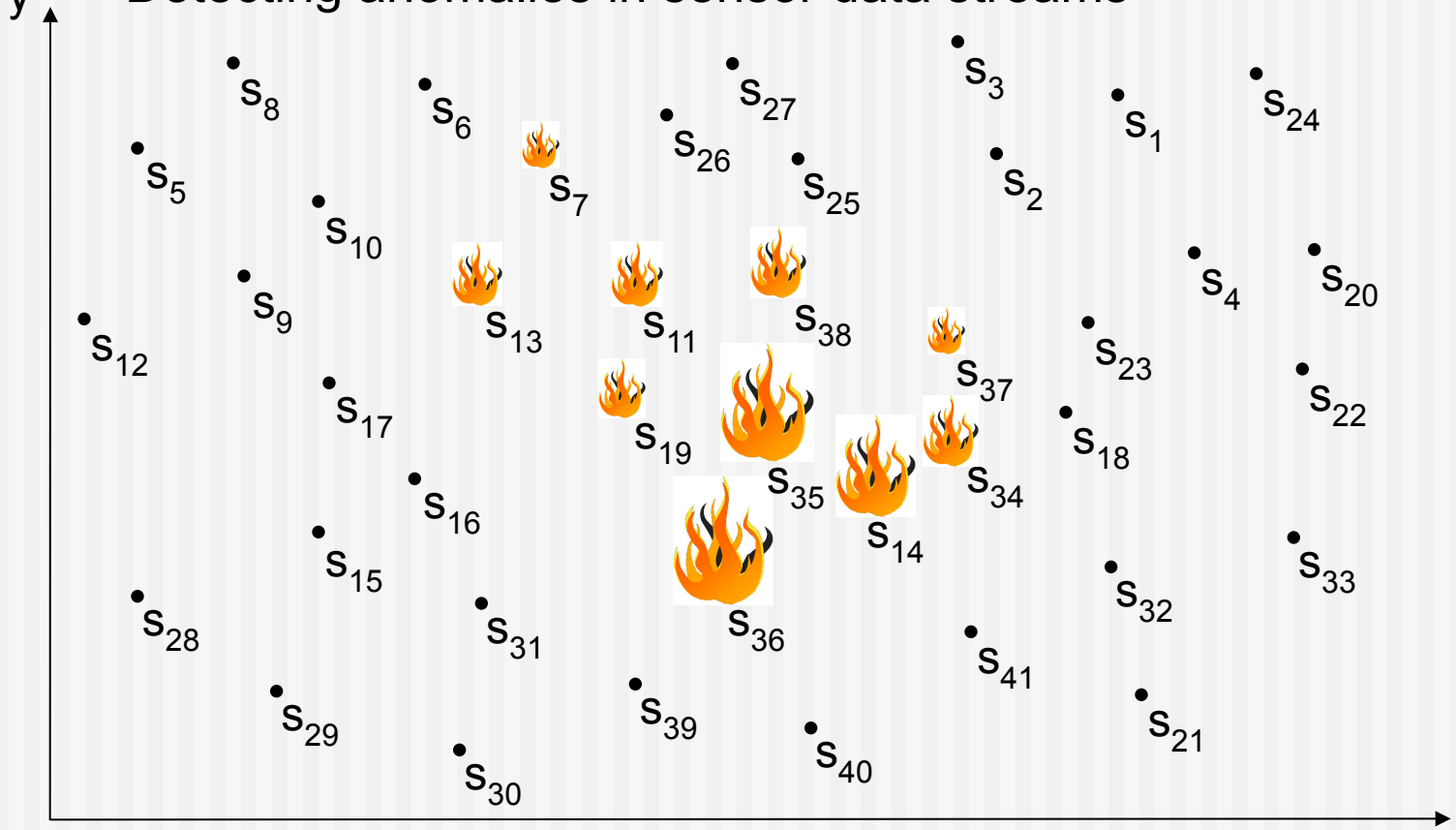
Introduction

- Anomalies in sensor networks
  - Sensors measuring the temperature



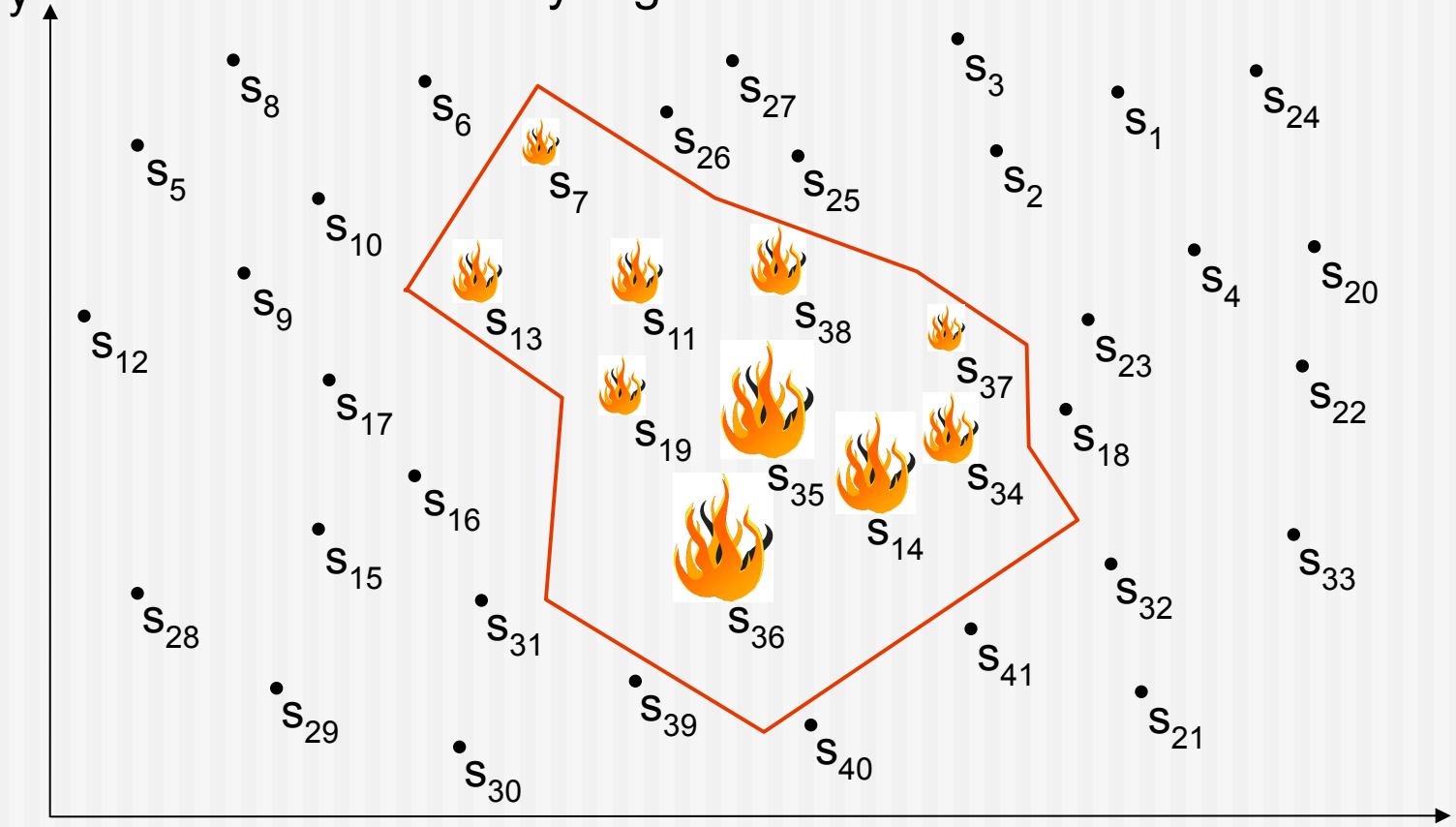
Introduction

- Anomalies in sensor networks
  - Detecting anomalies in sensor data streams



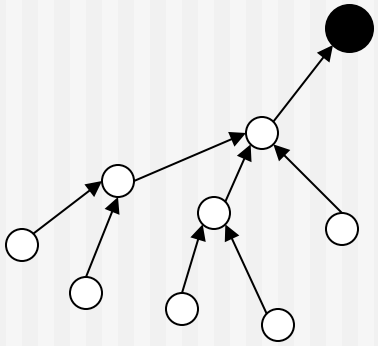
Introduction

- Anomalies in sensor networks
  - Determine anomaly regions

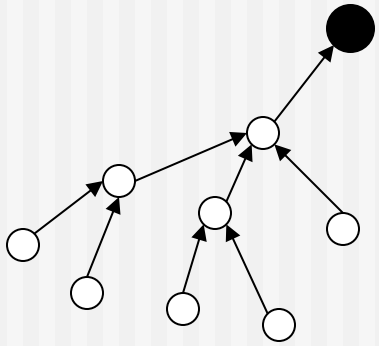


Introduction

- Wireless sensor networks
  - Continuous monitoring (environment, traffic, ...)
  - Limited capacities (computing power, battery lifetime, ...)
- Query processing
  - Centralized processing
  - In-network query processing
  - Hybrid processing



Centralized processing

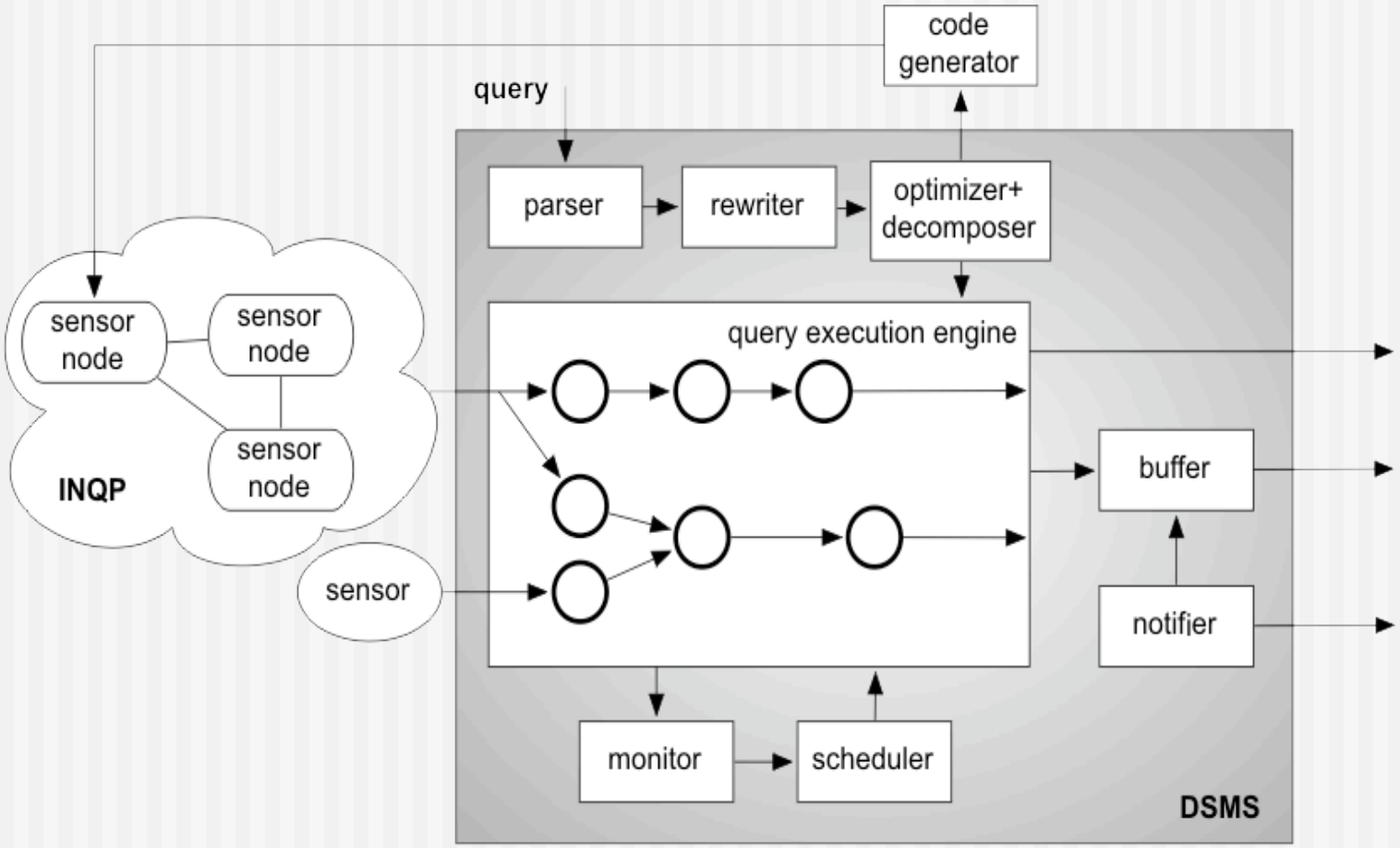


INQP

Introduction

- Problem:
  - Which parts of a query to evaluate within the network?
- AnduIN:
  - Combining in-network query processing with data stream processing
  - Query decomposition and optimization,
    objective: minimization of energy consumption
  - Comfortable way of specifying queries

AnduIN


code generator
query
parser
rewriter
optimizer+ decomposer
query execution engine
buffer
notifier
monitor
scheduler
DSMS
sensor node
sensor node
sensor node
INQP
sensor

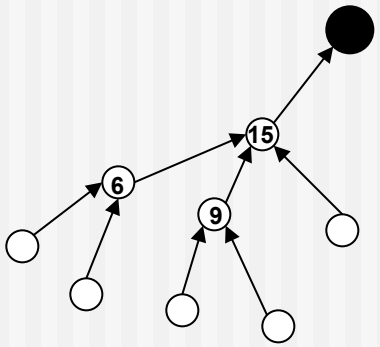
Logical Sensor Networks

- Registering a sensor

  ADD SENSOR 15 (temperature double, humidity double)
  LOCALIZATION [47° 25', 010° 59']

- Registering a logical sensor network

  CREATE STREAM net_stream (id int, temperatur double)
  NETWORK [15 (6, 9)]
  SAMPLING 30 SECONDS



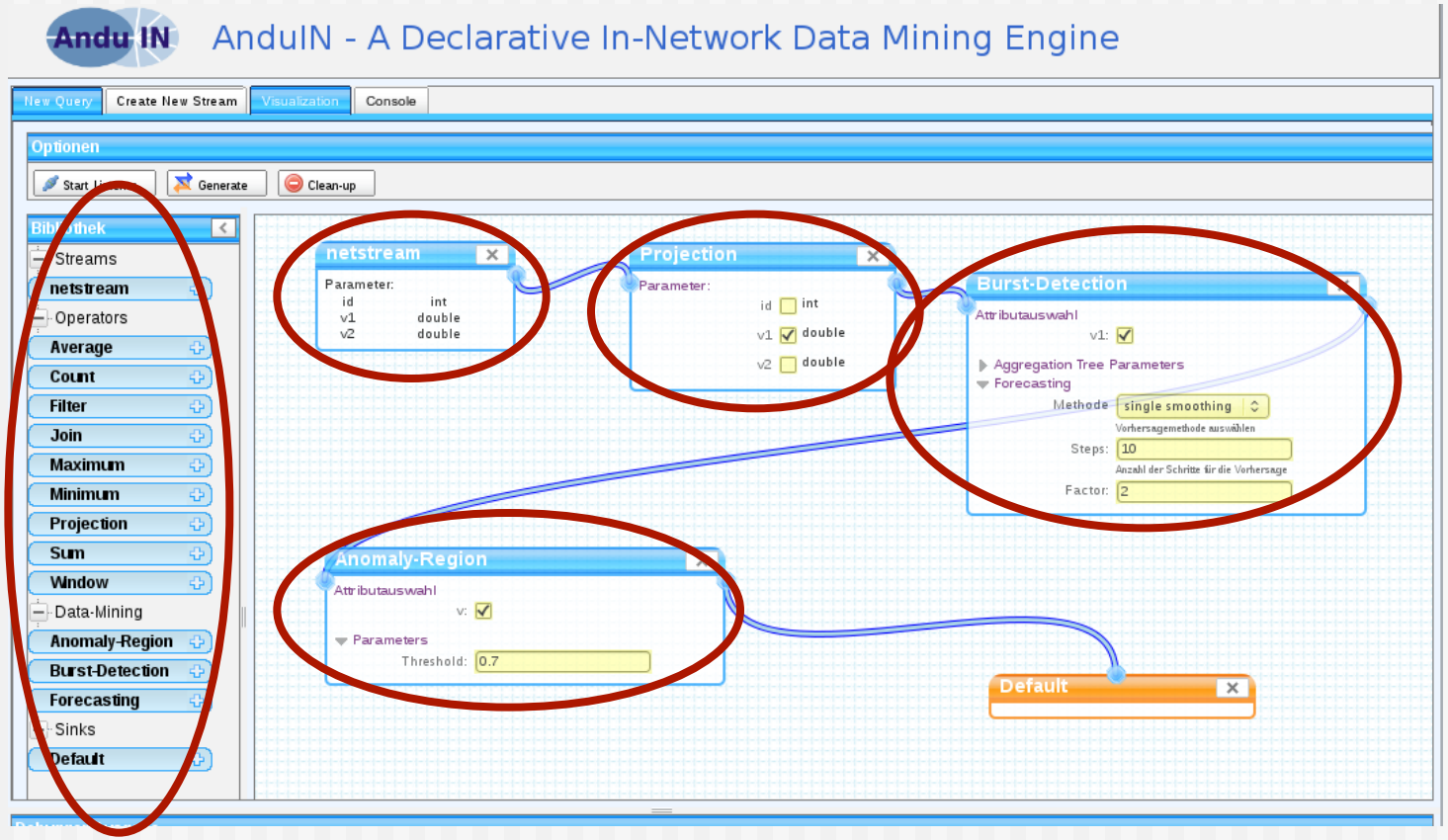
Web-based Box-and-Arrow Frontend

- Example query:
  - Detecting anomaly regions based on bursts

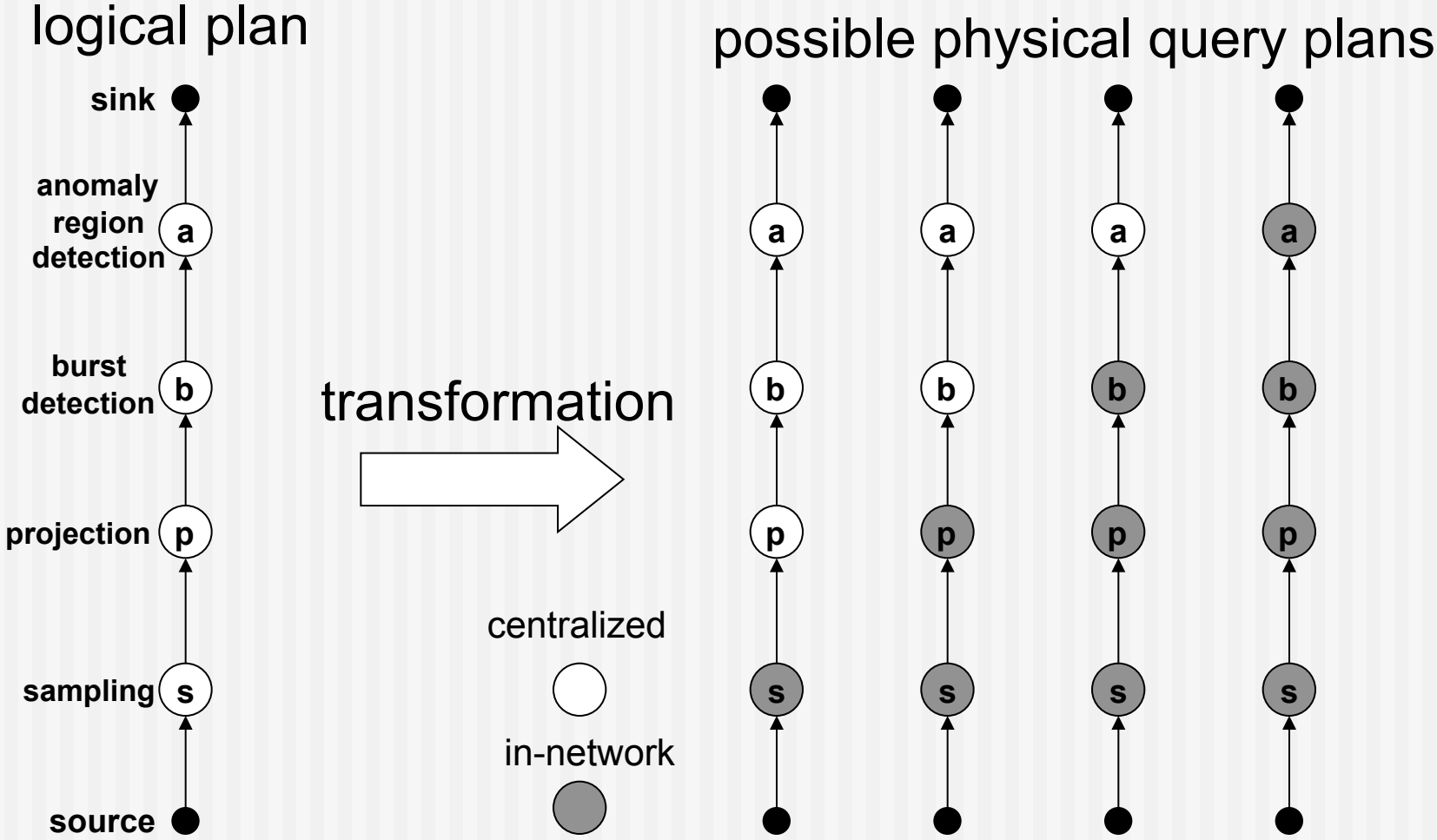
```
CREATE STREAM s_burst AS
SELECT timestamp, temperature
FROM net_stream [ burst-detection(w => 1000,
        threshold =>‘forecast‘) ];

SELECT timestamp, temperature
FROM s_burst [anomaly-region (t => 0.5)]
```

Web-based Box-and-Arrow Frontend

- Example query:
  - Detecting anomaly regions based on bursts



Query Decomposition and Optimization



Cost Model

- Costs for in-network query processing
- Factors influencing costs:

m	total number of sensor nodes
m_l	number of leader nodes
r_s	sampling rate in $\frac{1}{s}$
c_{wake}	wake up costs
c_{sample}	sampling costs
c_{msg}, $\hat{c}_{msg}$	message costs
h	average tree height
c_{loc}^{cpu}	local computation costs at a sensor
c_{neigh}^{cpu}	costs for processing data from a neighborhood
σ^j	selectivity of operator j, $\sigma_0 = 1$

Cost Model

- Costs for in-network query processing
- No leader nodes

$$C_{simple} = \left\lceil c_{wake} \right\rceil$$

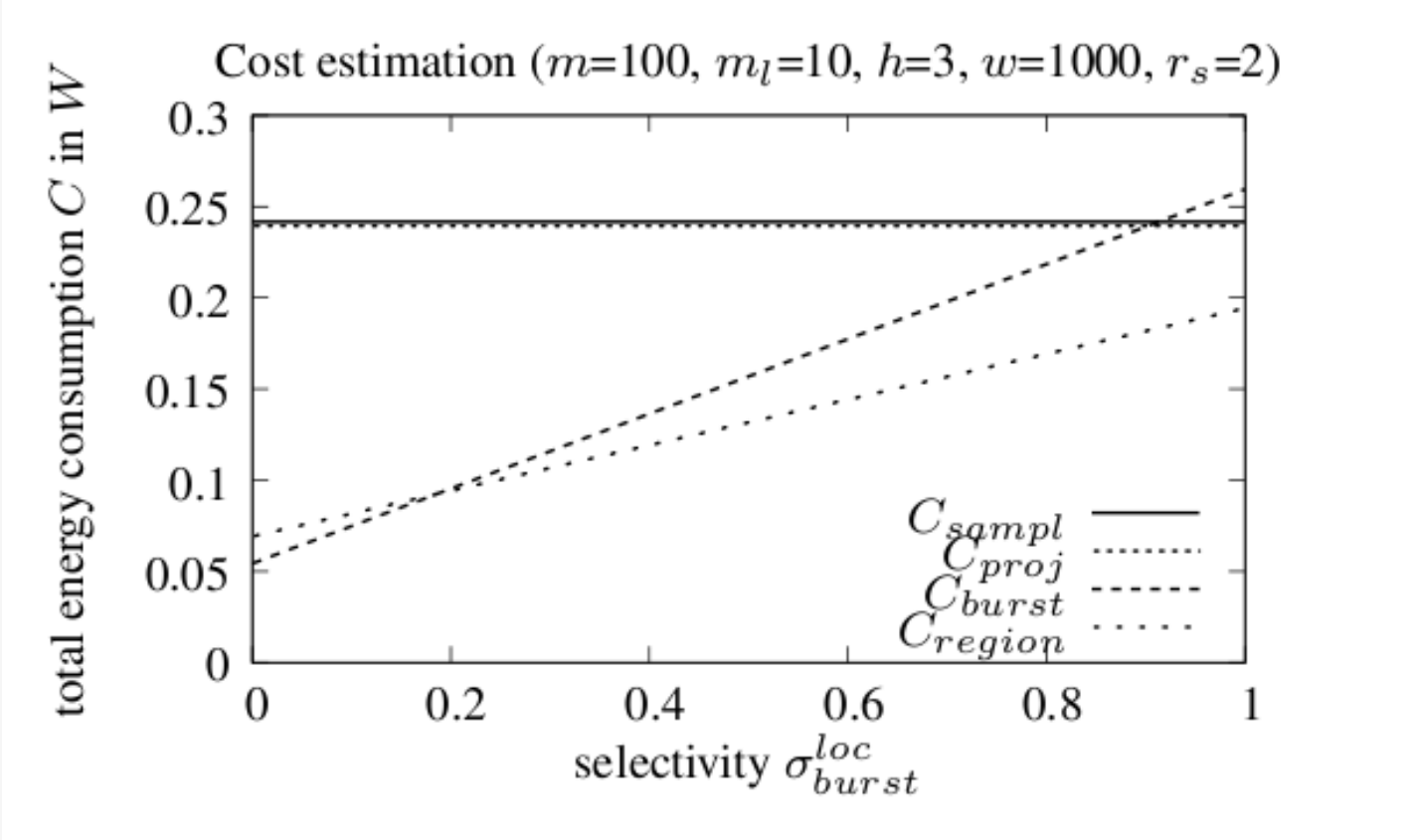
Cost Model

- Costs for in-network query processing
- Considering leader nodes

$$C_{innet} = \left\lceil m \cdot (\quad) \right\rceil$$

Case Study

- Detect anomaly regions based on bursts
- 100 nodes



Summary

- AnduIN
  - Declaritive query formulation
  - Comprehensive optimization of complex queries
  - Cost model
- Future Work
  - Improving the implementation
  - Multi-query optimization
  - Online query optimization

Thank you...

Case Study

- Tmote Sky sensor nodes with
  - 16 bit MCU MSP430F1611,
  - 4 MHz clock rate,
  - IEEE 802.15.4 compatible CC2420 transceiver with 250kBit/s

measurement	time	energy
compute average of 10 values	$52.3\mu s$	$0.272\mu J$
compute average of 100 values	$245\mu s$	$1.274\mu J$
single addition	$2\mu s$	$0.010\mu J$
single division	$27\mu s$	$0.140\mu J$
single multiplication	$16.2\mu s$	$0.08\mu J$
sending 1 byte	$4.85ms(2.33 - 6.95ms)$	$240.19\mu J(121 - 361\mu J)$
sending 10 bytes	$4.9ms(2.8 - 7.4ms)$	$252.93\mu J(146 - 385\mu J)$