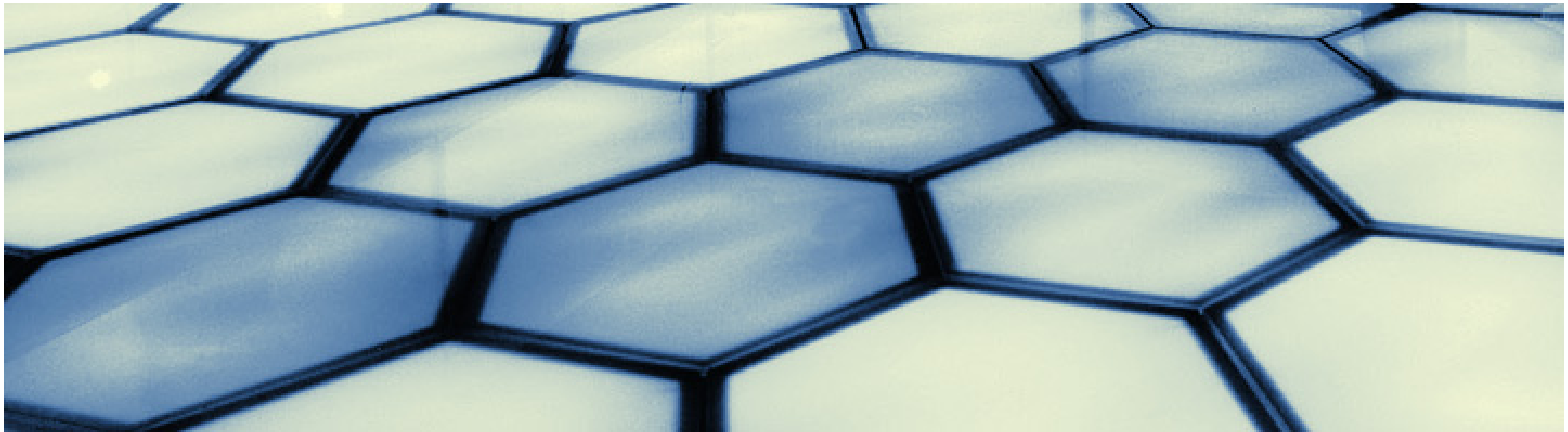
Self-Organizing Search in the Web of Things

Kay Römer, University of Lübeck



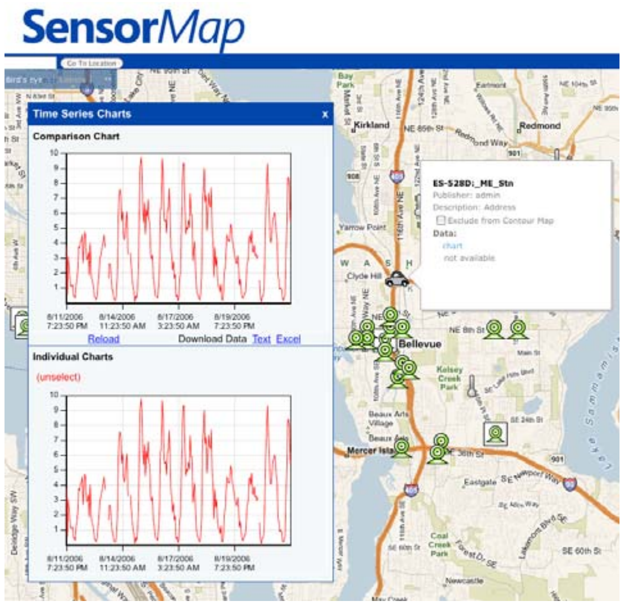
Networked Embedded Sensing Research

- Protocols
  - Adaptation to interference
- Programming Models
  - Role assignment
- Services
  - Content-based Sensor Search
  - Minimally-Invasive Management
- Systems
  - Body sensor networks

Motivation

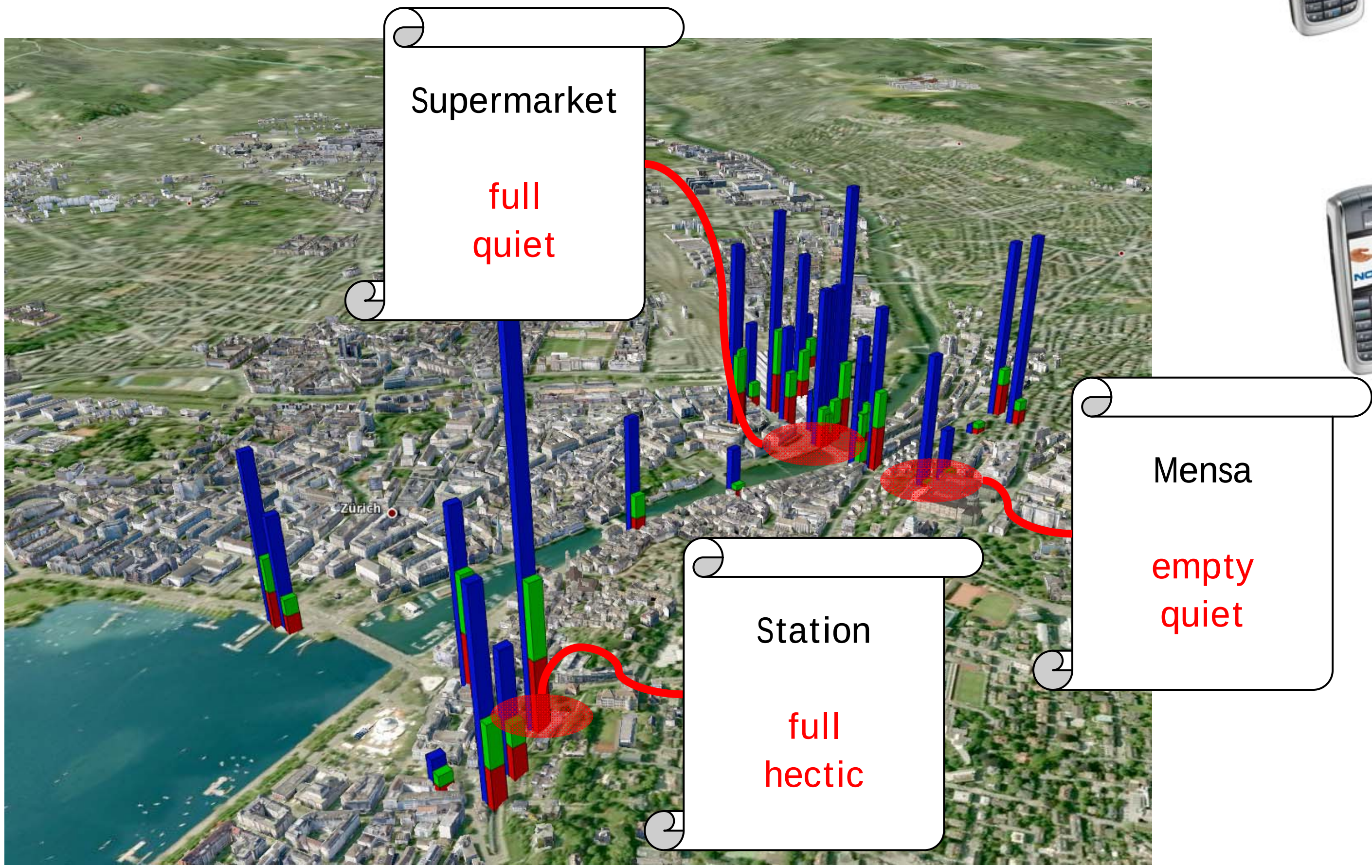
- Mobile phones equipped with sensors and connected to the Internet

- Sensors published on the Web: state of the real world available in real-time



➔**Search** the real world by its **current state**!

A Web of Real-World Places



Web of Things

- Web presence of things, people, and places with real-time state information
  - Web of real-world entities, not Web of sensors
  - High-level states, not raw sensor data
- Searching the Web of Things
  - Search for real-world entities: places, people, things, …
  - by their current state: empty, hot, broken, …
  - in real-time

Searching the Real World: Examples

- *Quiet picnic places at waterfront?*
- *Route through city avoiding traffic jams?*
- *Which rental station has bicycles available?*
- *Where are many people who share my interests?*
- *Which trains from A to B are not crowded?*
- *Where to enter train to get free seat?*
- *Supermarkets with short waiting queues?*

Problem: Content-based Sensor Search

- Find sensors reading given state in real time
  - Potentially huge, distributed set of candidate sensors
  - More state updates than queries, push not a good idea!
- Sensor output is highly dynamic
  - Indexing sensor output not a good idea!
- We need only a limited number of results at a time
  - Heuristics to select good candidates!

Approach: Sensor Ranking

Indexing Time

- Sensors create prediction model using past readings
- Prediction models are published on the Web
- Search engine periodically indexes prediction models

Query Time

- Prediction models are used to rank candidate sensors
- Highest ranking sensors are read first
- Goal: Minimize the number of read sensors

System Model

- **Sensor** maps discrete time to a finite discrete set of states:

$$s : T \mapsto V$$

- Sensor output time series: $s(t_i) = v_i$

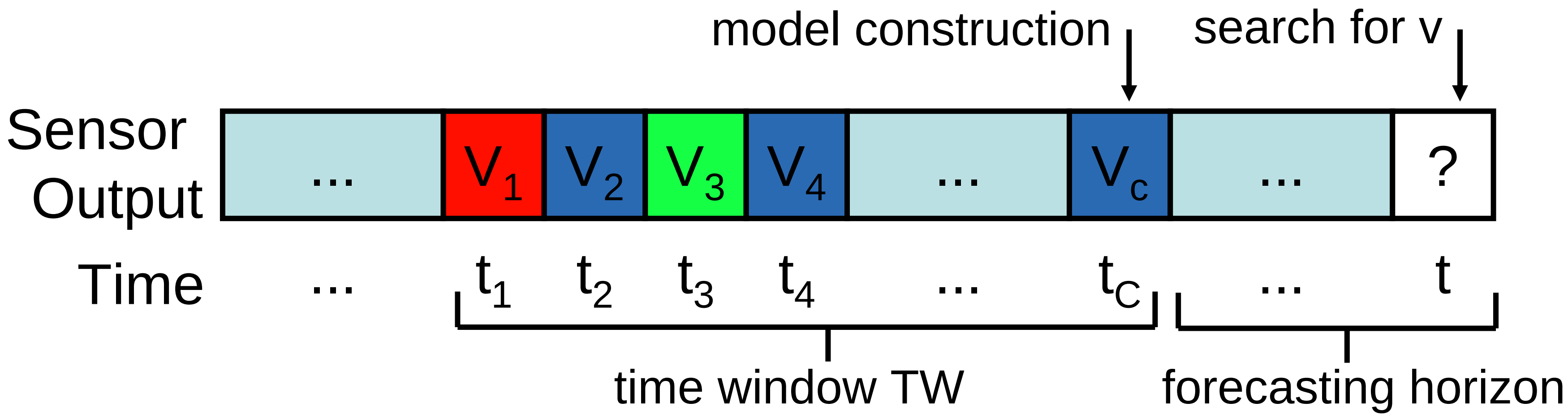
Sensor Output	...	V_1	V_2	V_3	V_4	...	$V_{present}$
Time	...	t_1	t_2	t_3	t_4	...	$t_{present}$

System Model (Continued)

- **Prediction model** maps query time and query value to a probability estimate:

$$P : T \times V \mapsto [0,1]$$

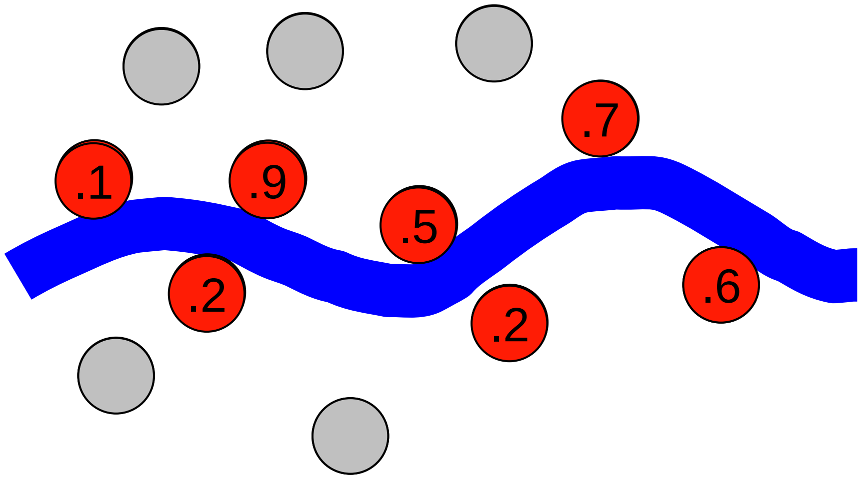
- P(t, v): Probability that s(t) = v

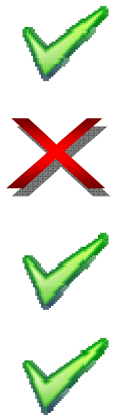


Query Resolution

Example: Quiet places at waterfront

1. Filter static (waterfront, occupancy)
2. Predict (quiet)
3. Rank
4. Read
5. Return





Ranking Metrics

- Normalized overhead for reading non-matching sensors
- Ranking error e(t,v)

$$\frac{\text{Number of non-matching sensors above last matching sensor}}{\text{Rank of last matching sensor}}$$

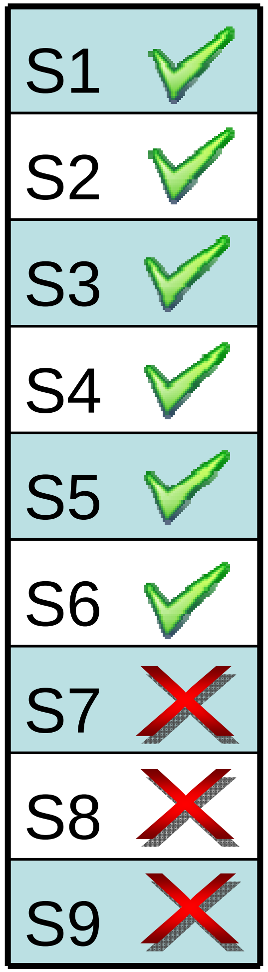
- Top-m ranking error $e_{top}(t,v,m)$
  - Dito, but only first m sensors considered

Ranking Metrics: Examples

Optimal ranking:
- $e = 0/6$
- $e_{top} = 0/5$

m

Last match



Sensors that match the query

Sensors that do not match the query

Prediction Models

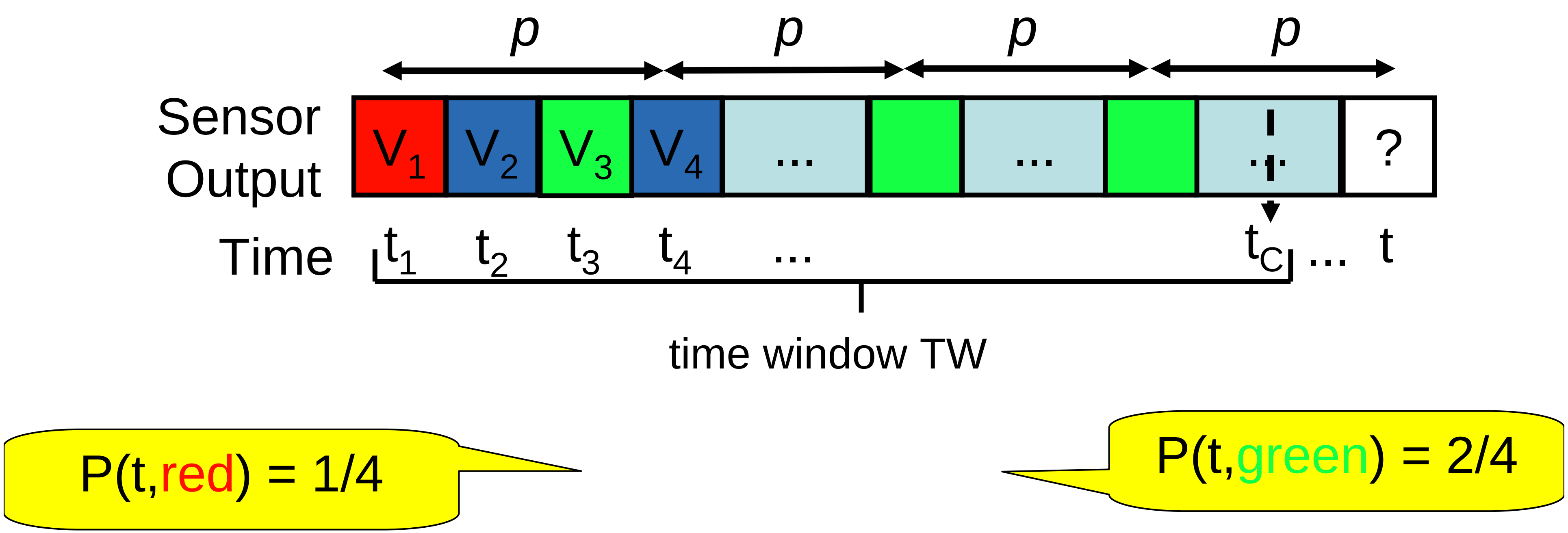
- Focus on people-centric sensors
  - Tend to show periodic behaviour
- Requirements
  - Accurate predictions for forecasting horizons that match indexing frequencies (days - weeks)
  - Deal with imperfect periodic behavior

Considered Prediction Models

- Single-period prediction model (SPPM)
  - Assumes single dominant period of known length (e.g., 1 week)
- Multi-period prediction model (MPPM)
  - Assumes multiple periodic processes of unknown length (e.g., 1 week, 4 weeks)
- Select appropriate models at runtime

Single-Period Prediction Model (SPPM)

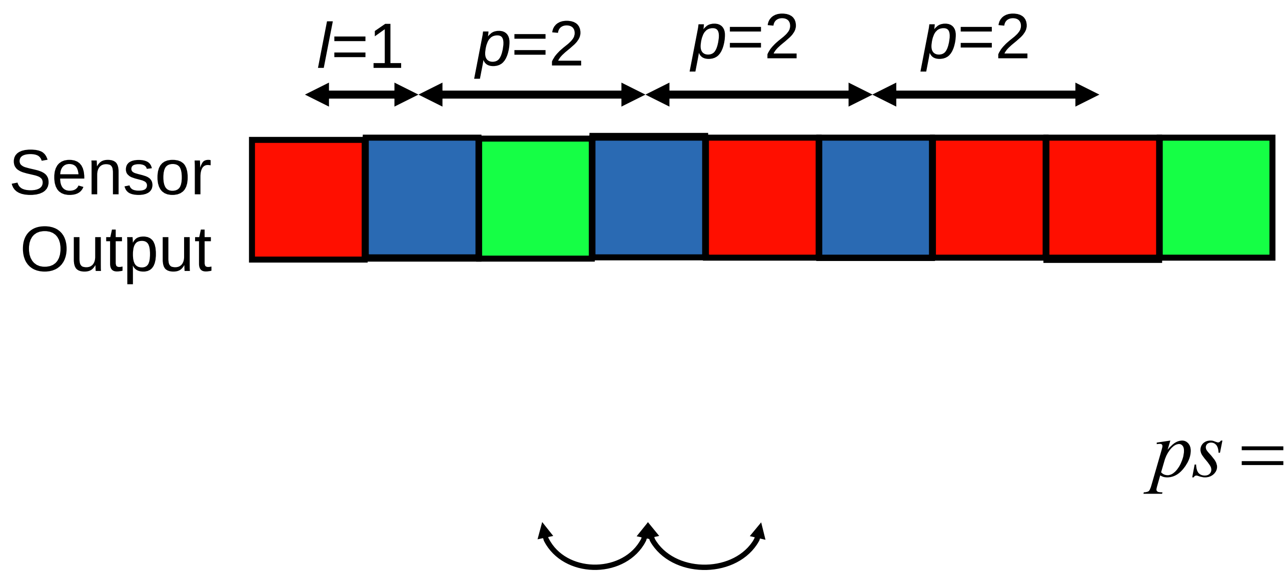
- **Assumption:** Single dominant period with length *p*



Multi-Period Prediction M

$$\frac{\text{Number of consecutive appearances of symbol } \alpha \text{ in period } p \text{ at offset } l}{\text{Max. possible occurences}}$$

- Periodic symbol (α, p, l, ϕ)
  - α: symbol; p: period; l: offset; ϕ: support
- Example: α=blue, *p*=2, *l*=1



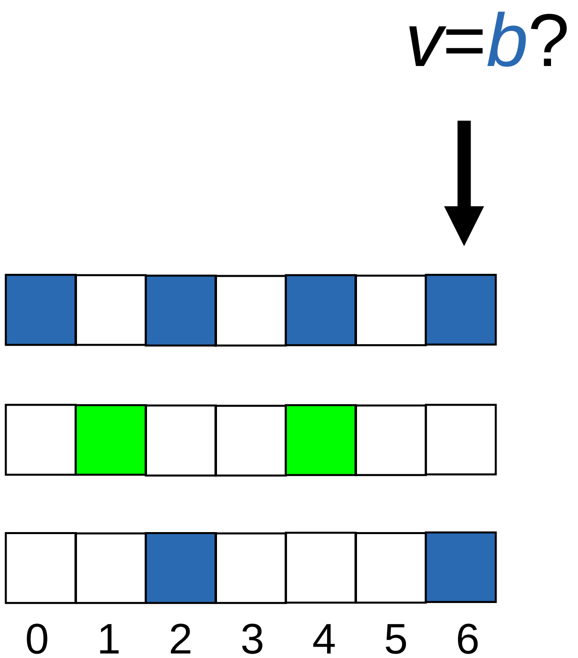
$$ps = \left(blue, 2, 1, \frac{2}{3} \right)$$

Inferring Prediction Estimates

Query for value v=*b* at time t=*6*

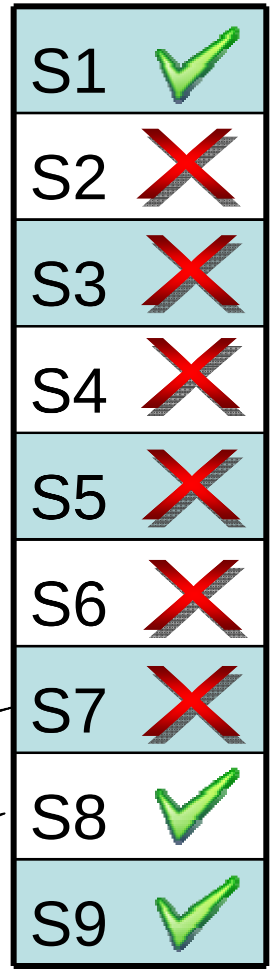
1. Filter periodic symbols
   - Same value: $\alpha = v$
   - Same phase: $l \equiv t \bmod p$
2. P(v,t) = max ϕ

α	p	l	ϕ
b	2	0	0.7
g	3	1	0.1
b	4	2	0.9

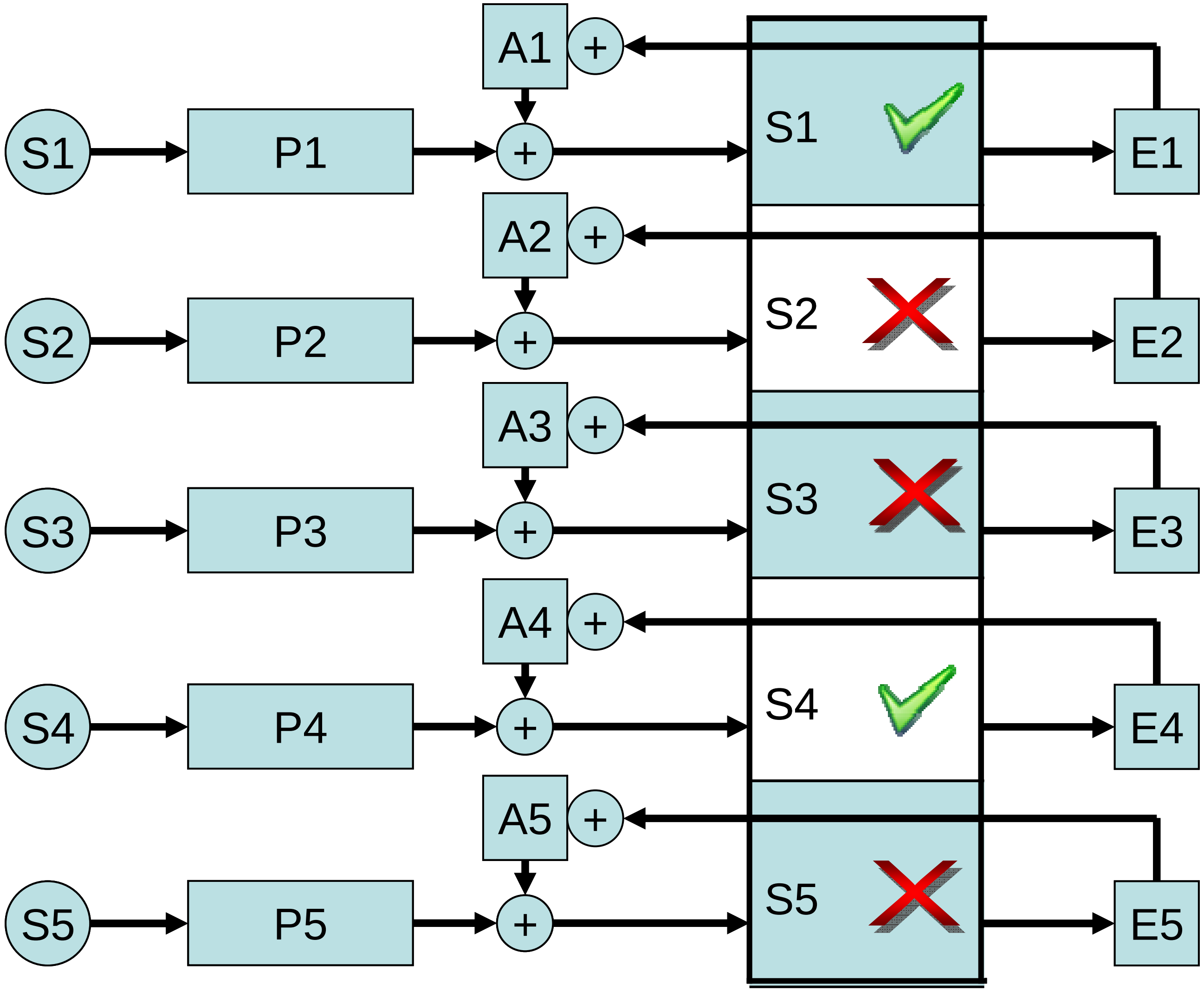


Adjustment Process

- Faulty/malicious sensors, inaccurate predictions may result in persistent misranking
- Individual ranking error for each sensor
  - S8 ranked to low: increase prediction value
  - S7 ranked to high: decrease prediction value
- Idea: adjustment term for each sensor
  - Updated after each query using ranking error



Adjustment Process: Feedback Loop



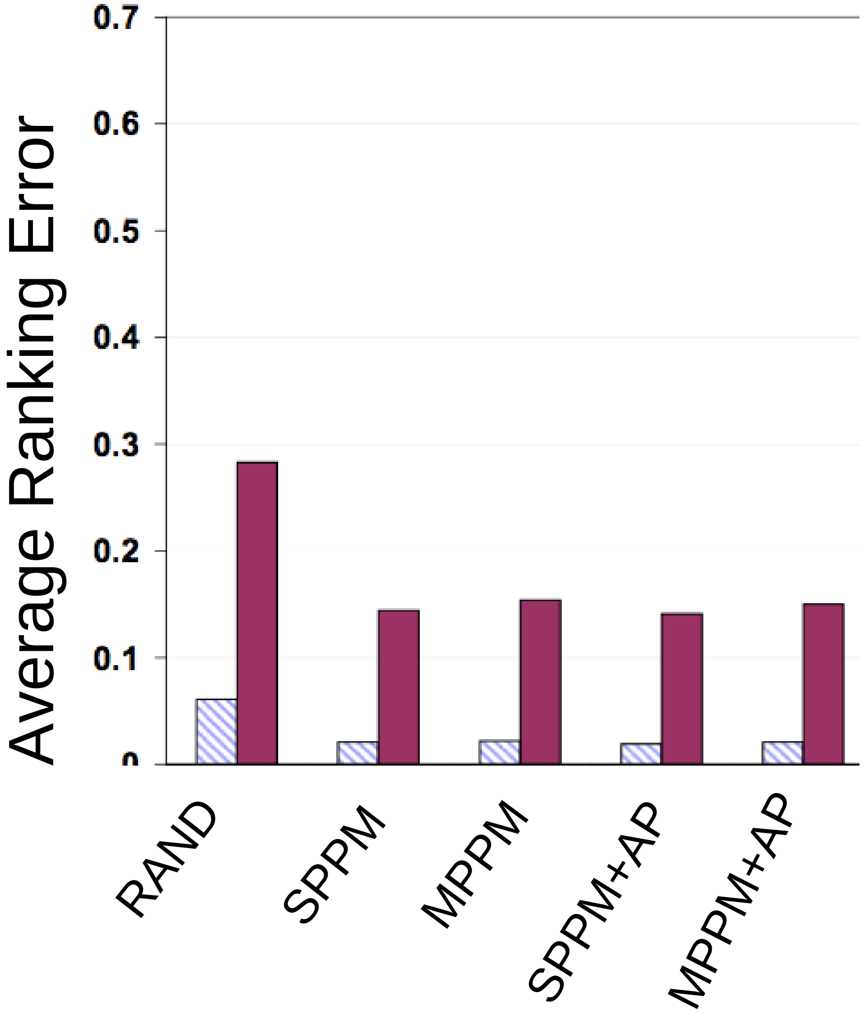
Evaluation

- Simulation of a realistic search engine
  - Periodic rebuild and indexing of models (1 week)
  - Periodic queries for possible values
  - Measure average ranking error
- Prediction models: Random, SPPM, MPPM
  - With / without adjustment

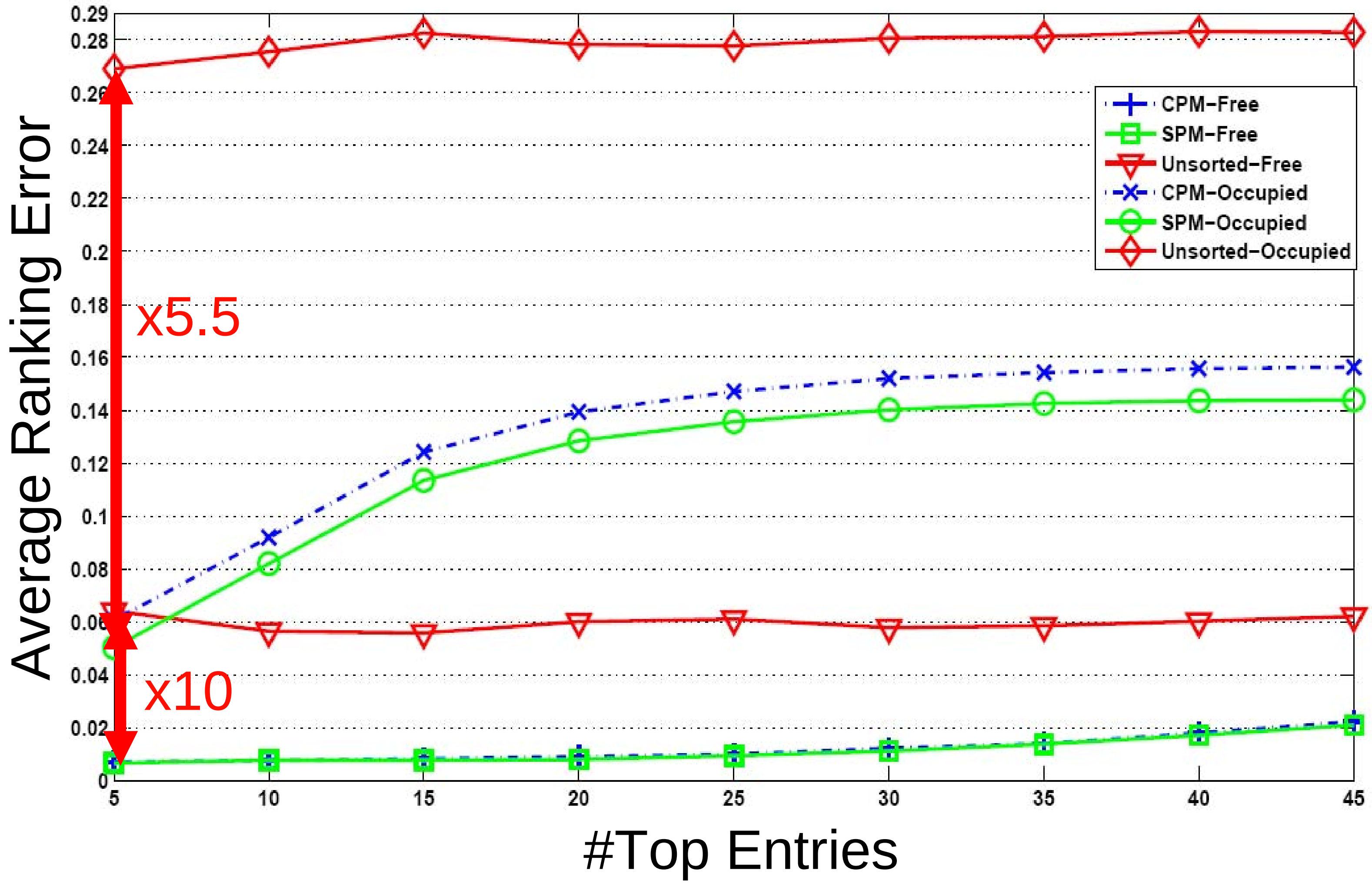
Evaluation: Data Sets

- MERL motion detector dataset
  - 50 PIR sensors in office building
  - PIR output mapped to “*free”* and “*occupied”*
  - With and without a “faulty” sensor
- ETH room reservation system
  - 7 “sensors”
  - Room occupancy: “*free*” or “*occupied*”
  - With and without “synthethic” multiperiod sensor
- Bicing data set (in progress)
  - 350 bicycle rental stations in Barcelona
  - Number of available bicycles: *“no”, “few”, “many”*

Average Ranking Error: MERL



Average Ranking Error vs. Top m



Summary

- Ubiquitous sensors connected to Internet
- Search for real-world entities by current state
- Sensor Ranking, a primitive for content-based sensor search utilizing prediction models
- Adjustment process to alleviate persistent inaccurate rankings
- Promising results on real-world data sets
- Ongoing work
  - Improved ranking based on correlations
  - Building a search engine

Ads

- Act-Control-Move: Beyond networked Sensors
  - Summer School, Schloss Dagstuhl, August 15-21, 2010
  - www.cooperating-objects.eu/school
- IEEE SUTC (Sensor Networks, Ubiquitous & Trustworthy Computing)
  - Conference, Newport Beach, California, June 7-9, 2010
  - sutc2010.eecs.uci.eu
- SESENA (Software Engineering for Sensor Nets)
  - ICSE Workshop, CapeTown, South Africa, May 3, 2010
  - www.sesena.info