Fuzzy-based Sensor Search in the Web of Things

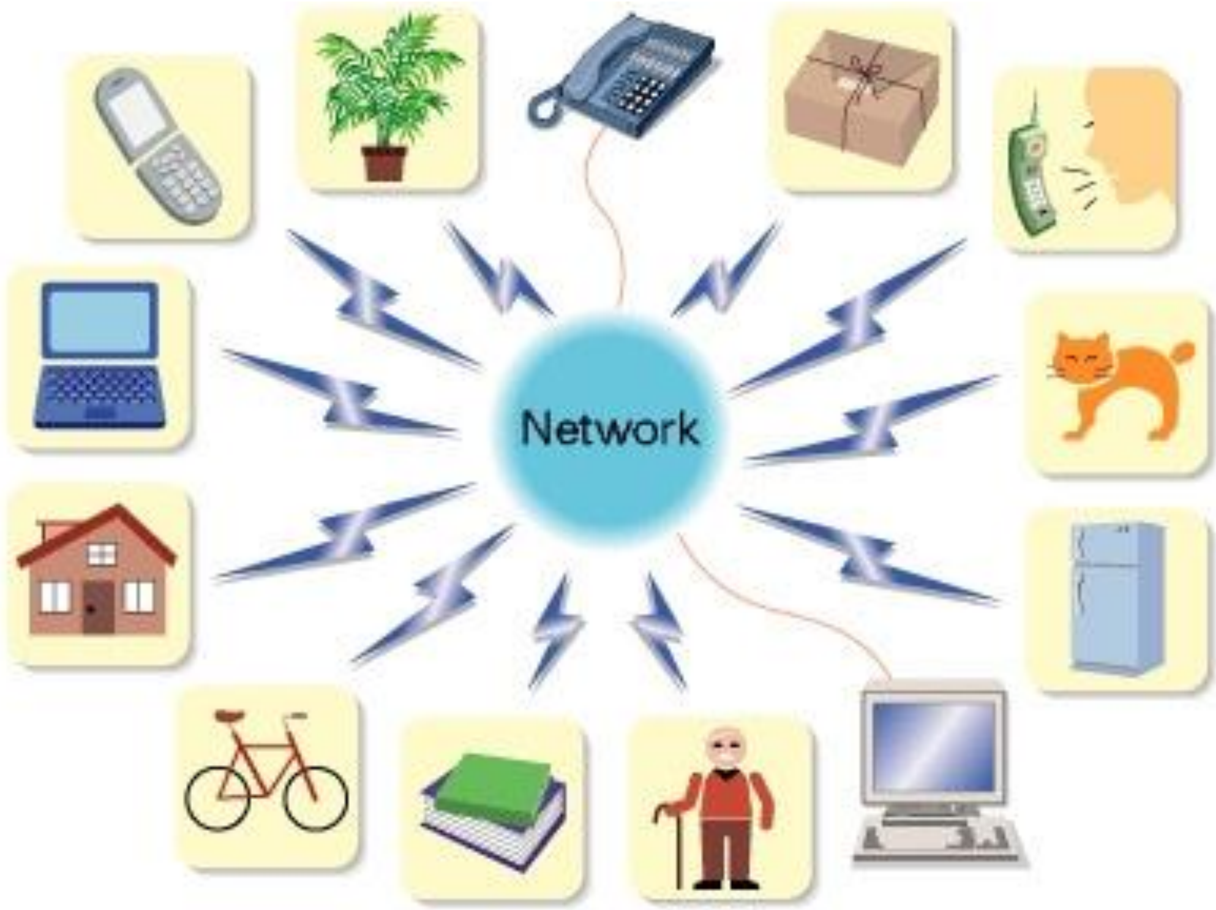
by

Cuong Truong

University of Lübeck, Germany

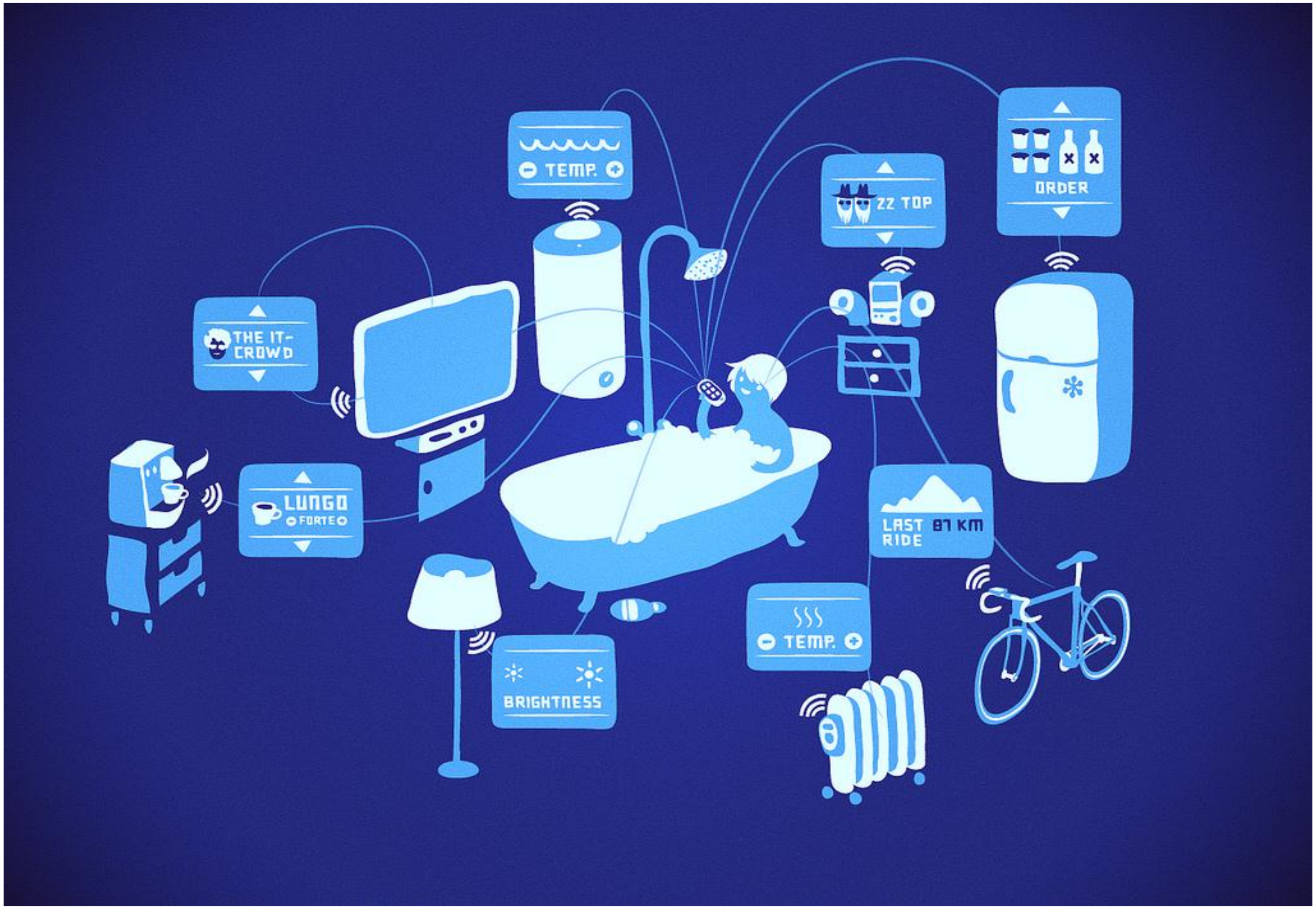
truong@iti.uni-luebeck.de

The Vision of the Internet of Things



real world objects will be uniquely identifiable and connected to the Internet

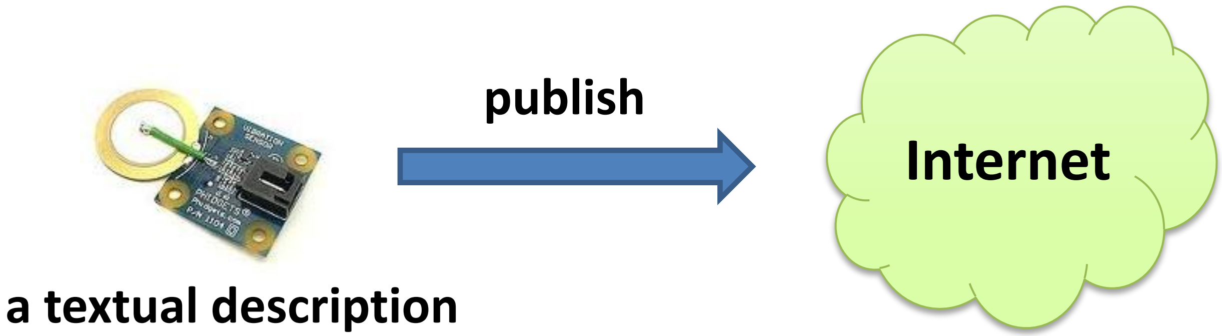
The Vision of the Web of Things



mashing up sensors and actuators with services and data available on the Web

Sensor Search in WoT: Start-of-the-art

pachube GSN SensorMap



Sensor Search in WoT: Start-of-the-art

SensorML

SSN

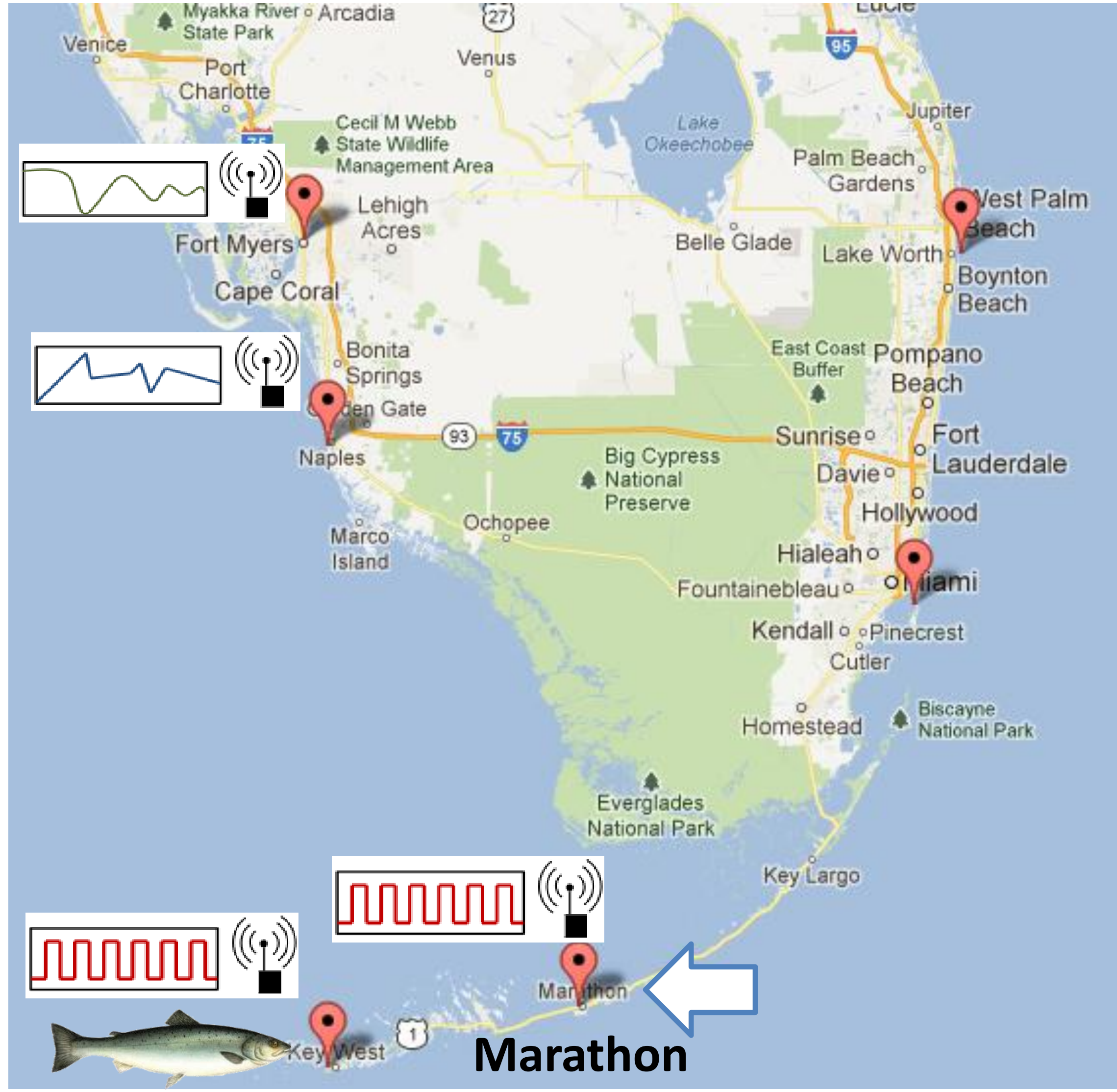
```
:SN-Node-TSB-ABC01
  a ssn:System ;
        ssn:hasOperatingProperty :TSBOperatingPowerRange
:TSBOperatingPowerRange a ssn:OperatingPowerRange ;
        ssn:hasValue :Current_Draw_Idle .
:Current_Draw_Idle a dul:Amount ;
        dul:hasDataValue "'21"' ;
        muo:measuredIn ucumunit:microAmpere .
:PhysicalPlace_UNiSTestBED-BA03A a dul:PhysicalPlace ;
        dul:isLocationOf :UNiS-TSBPlatform .
```

→ complex for end user!

Sensor Similarity Search: An Illustration

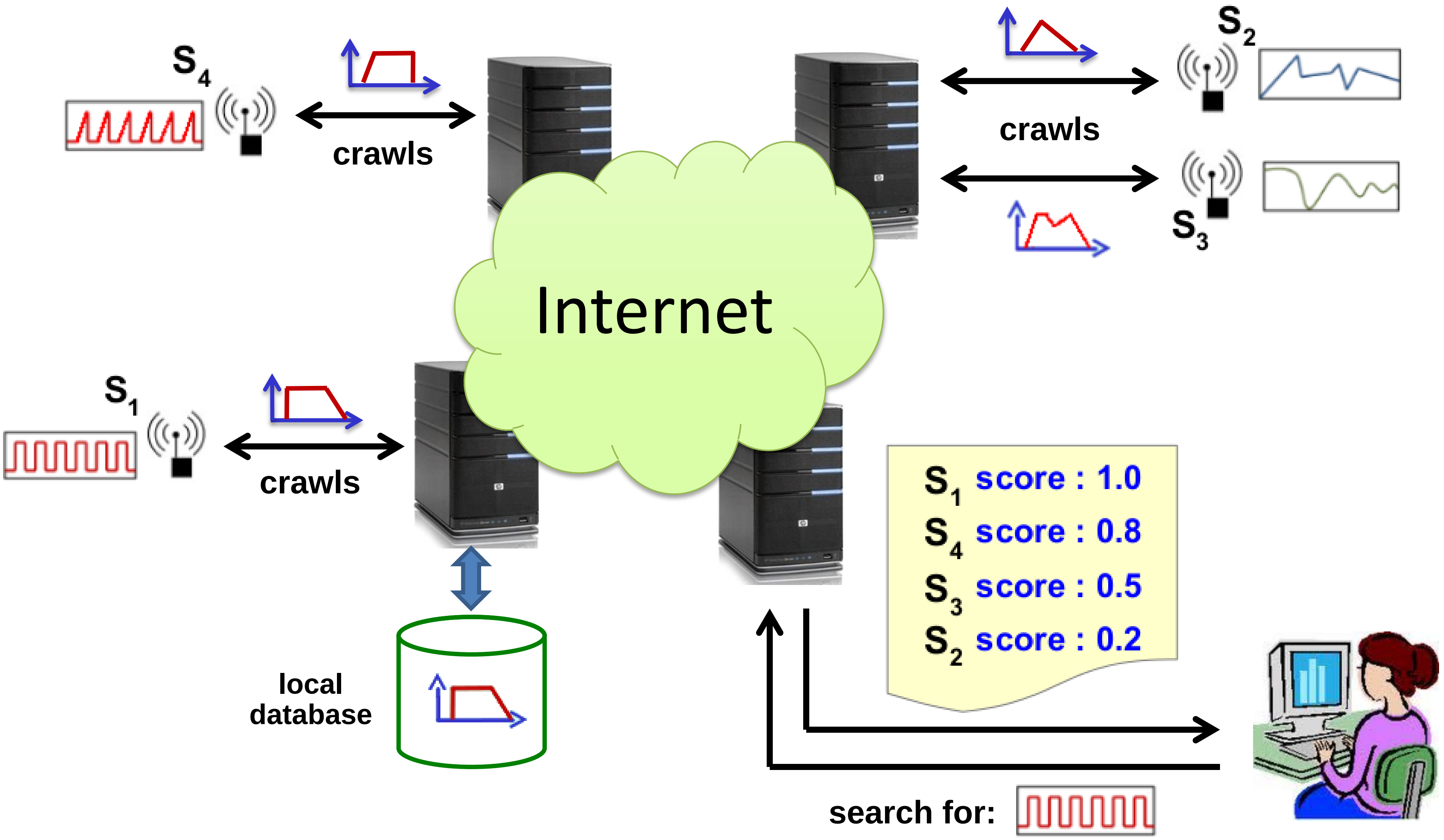
Pick a climate sensor in Key West, and search for similar sensors

Fishery owner



Key West

Sensor Similarity Search: Architecture



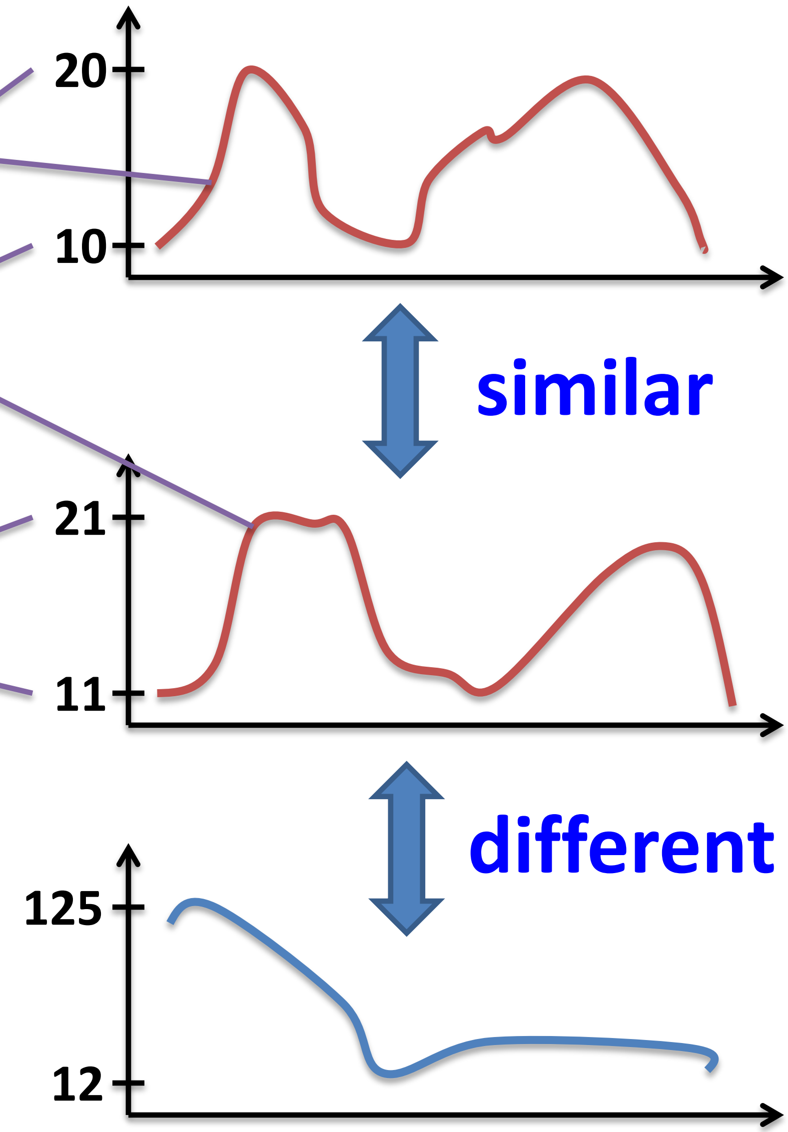
Questions to be addressed

I. How to define and compute similarity between two sensors?
II. How to construct a fuzzy set from historical sensor readings?
III. How to minimize the cost of storing such fuzzy sets?
IV. How to efficiently compute a similarity score between a pair of sensors?
V. How to objectively evaluate the approach?

I. Similarity Definition

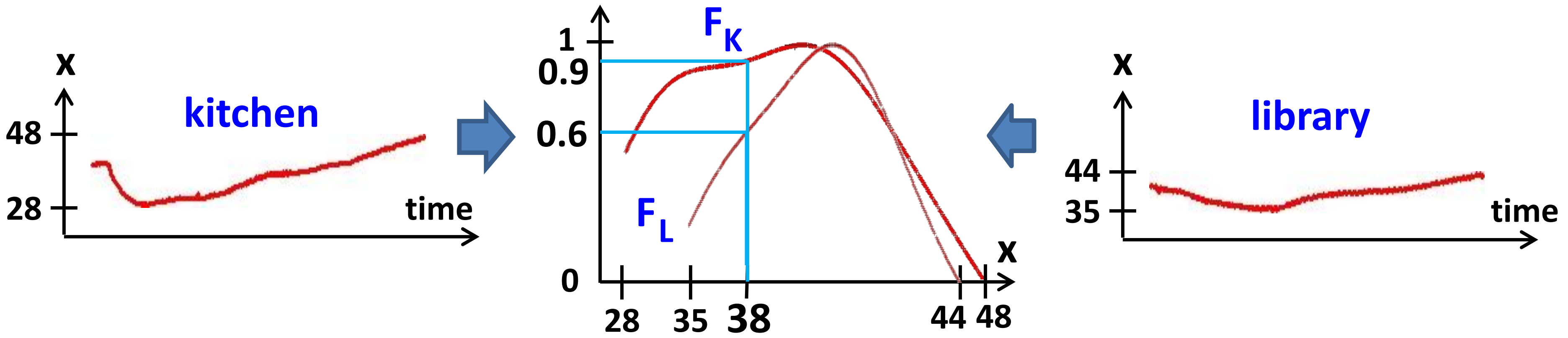
(1) similar reading curves

(2) similar reading ranges



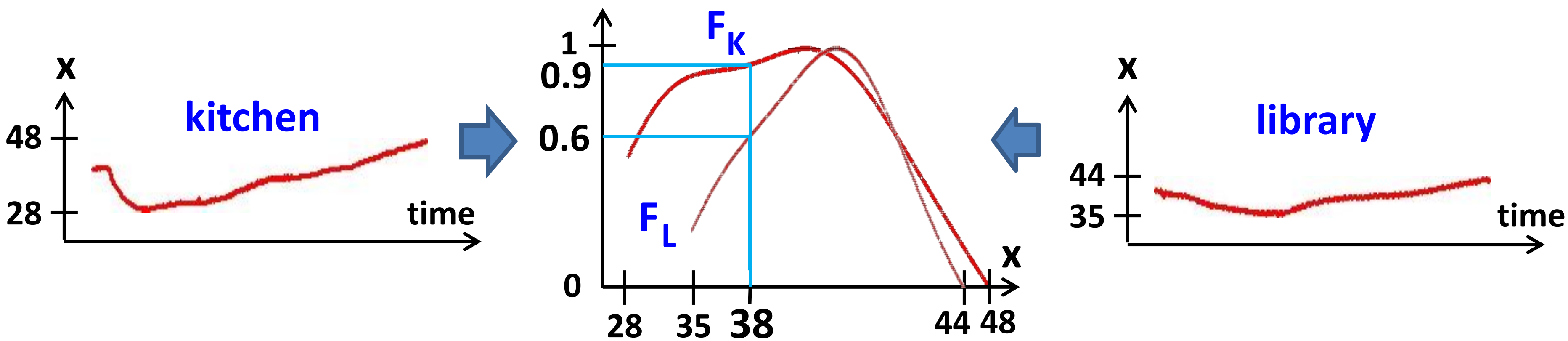
what about me?

I. Similar Reading Curves: Captured by Fuzzy Set



- Degree of membership of elements of fuzzy set
  - $F_K(38) = 0.9$
  - $F_L(38) = 0.6$
- Key idea: Same value, different degree of memberships in different fuzzy sets

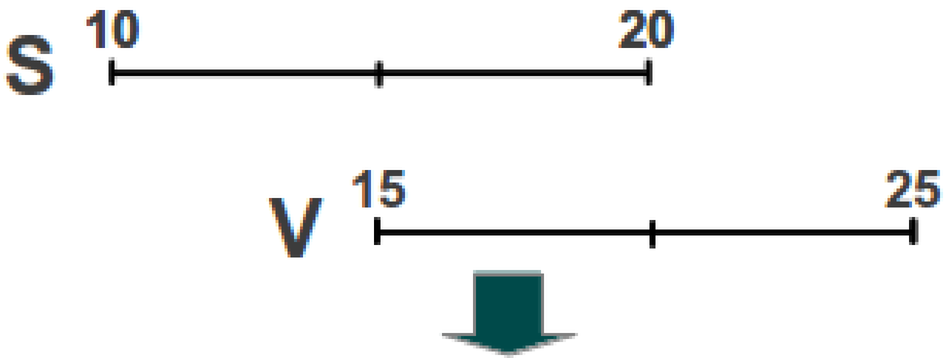
I. Similar Reading Curves: Captured by Fuzzy Set



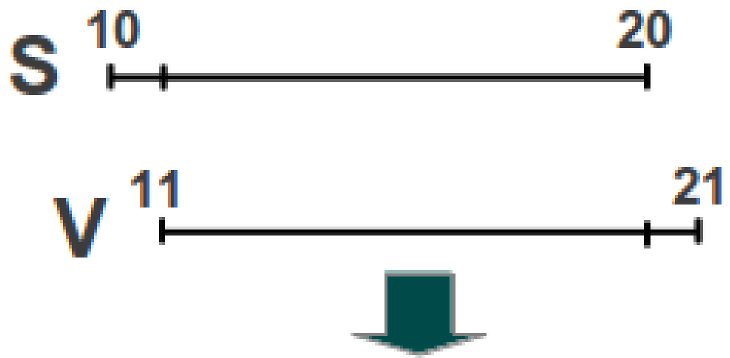
- The reading **38** is likely read by sensor in kitchen:
  - $F_K(38) = 0.9 > 0.6 = F_L(38)$
- Given a sensor **S** with set of readings X = {x}, **S** is likely located in kitchen if:

$$\sum_{x \in X} F_K(x) > \sum_{x \in X} F_L(x)$$

I. Similar Reading Ranges



$\delta(S,V) = (15-10) + (25-20) = 10$



$\delta(S,V) = (11-10) + (21-20) = 2$

captured by the reading range difference

I. Similarity Computation

- Given a sensor **V**, and a sensor **S** whose set of readings is X = {x}
- Combining the two above mentioned similarity conditions:
  - Similar reading curves (defined by fuzzy set)
  - Similar reading ranges (defined by reading range difference)

$$SimilarityScore_S(V) = \frac{1}{\delta(S,V)} \frac{1}{|X|} \sum_{x \in X} F_V(x)$$

Questions to be addressed

I. How to define and compute similarity between two sensors?

II. How to construct a fuzzy set from historical sensor readings?

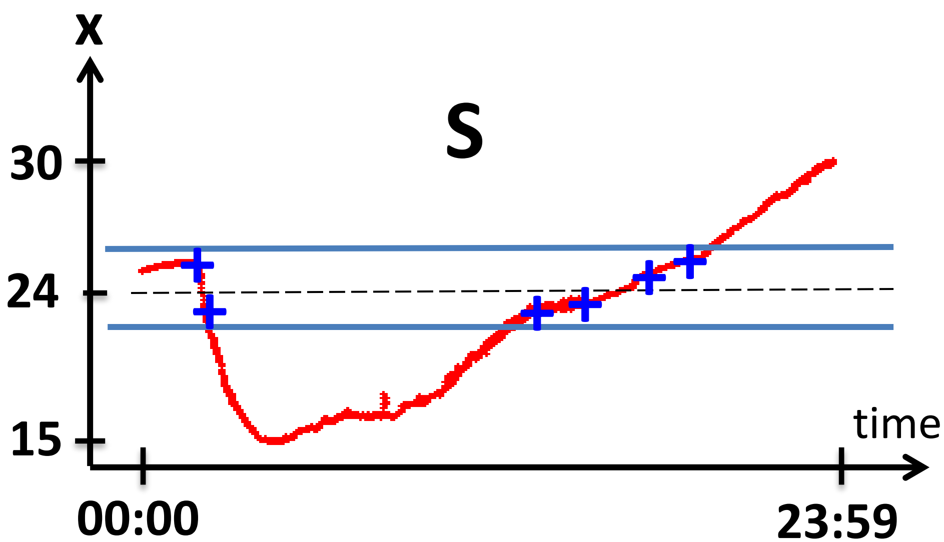
III. How to minimize the cost of storing such fuzzy sets?

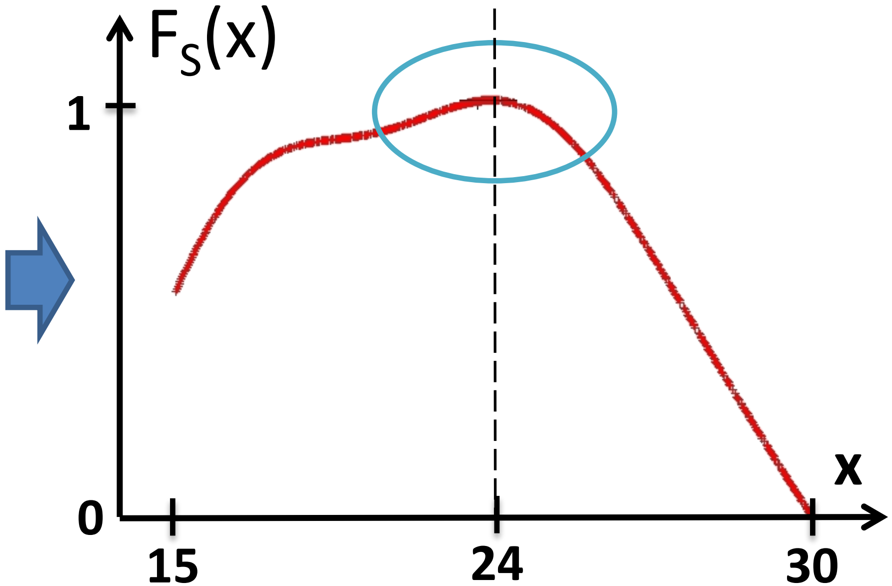
IV. How to efficiently compute a similarity score between a pair of sensors?

V. How to objectively evaluate the approach?

II. Fuzzy Set Construction

- Temperature sensor **S** has been monitoring a room for 24 hours from 00:00 -> 23:59





$$F_S(x) = \sum_{y \in X} e^{-\left[\frac{2d_E(x,y)}{r}\right]^2}$$

Questions to be addressed

I. How to define and compute similarity between two sensors?

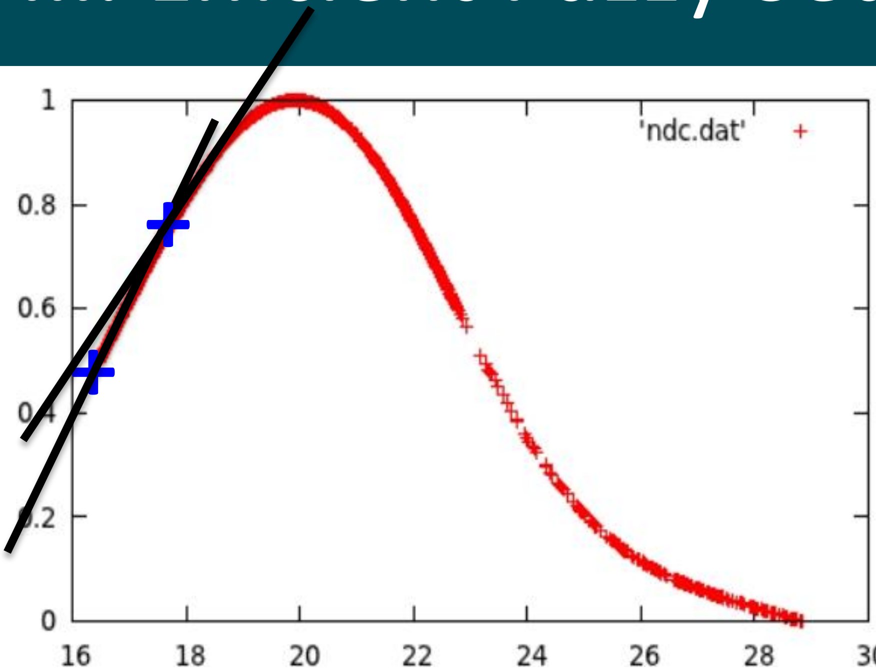
II. How to construct a fuzzy set from historical sensor readings?

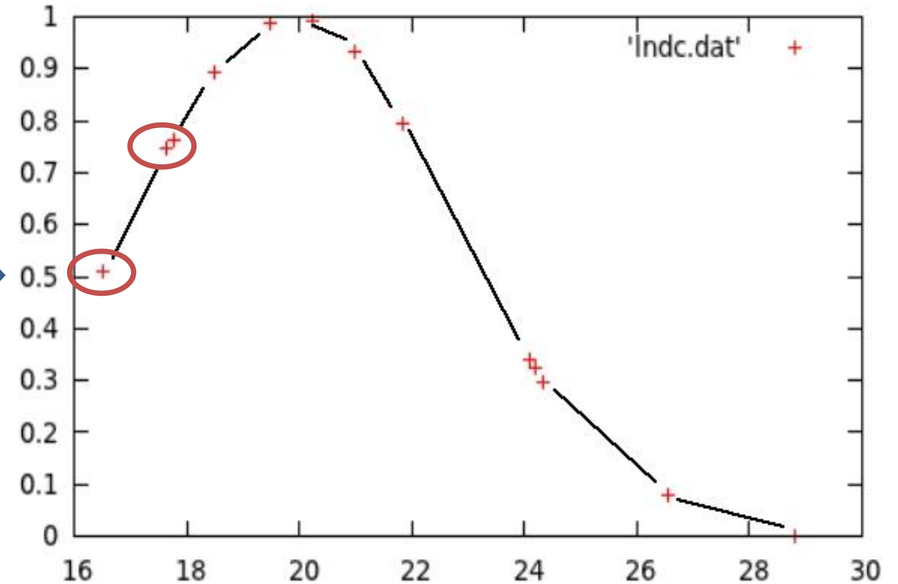
III. How to minimize the cost of storing such fuzzy sets?

IV. How to efficiently compute a similarity score between a pair of sensors?

V. How to objectively evaluate the approach?

III. Efficient Fuzzy Set Storage: Approximation





- Fuzzy set's storage overhead
- Membership function is smooth

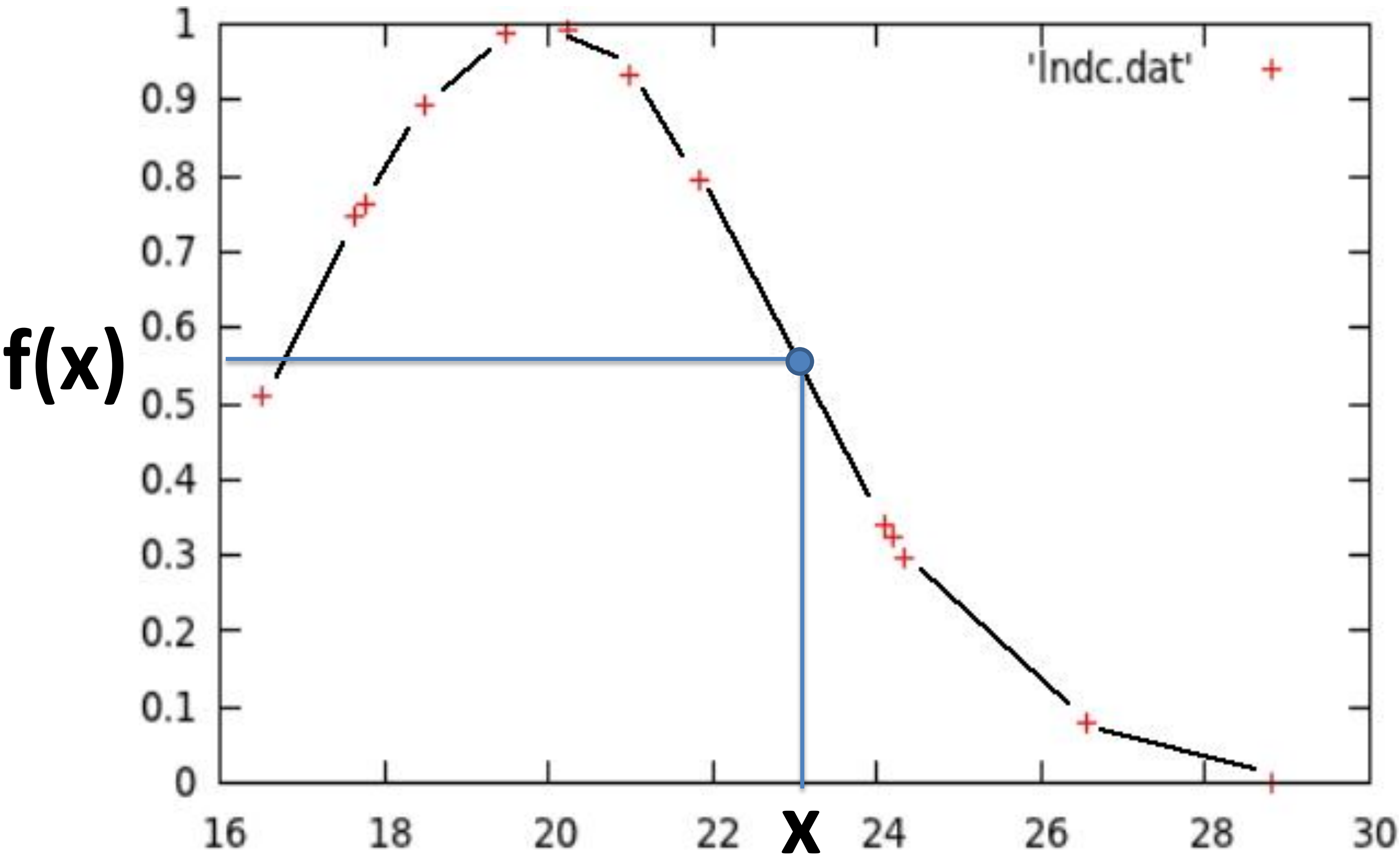
Approximation using set of line segments

Only few tens of bytes are required!

Questions to be addressed

I. How to define and compute similarity between two sensors?
II. How to construct a fuzzy set from historical sensor readings?
III. How to minimize the cost of storing such fuzzy sets?
IV. How to efficiently compute a similarity score between a pair of sensors?
V. How to objectively evaluate the approach?

III. Efficient Similarity Score Computation



Questions to be addressed

I. How to define and compute similarity between two sensors?

II. How to construct a fuzzy set from historical sensor readings?

III. How to minimize the cost of storing such fuzzy sets?

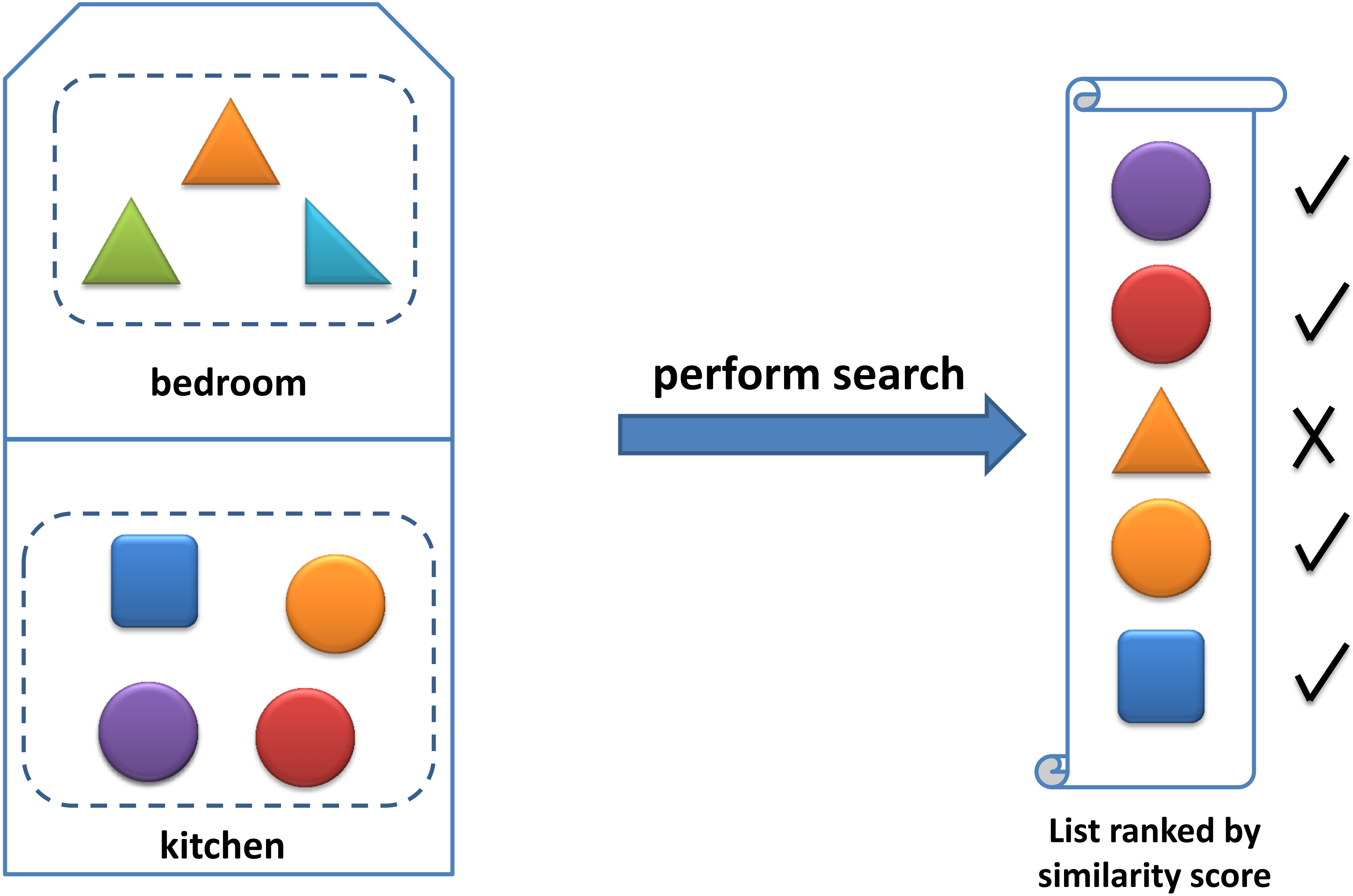
IV. How to efficiently compute a similarity score between a pair of sensors?

V. How to objectively evaluate sensor similarity search?

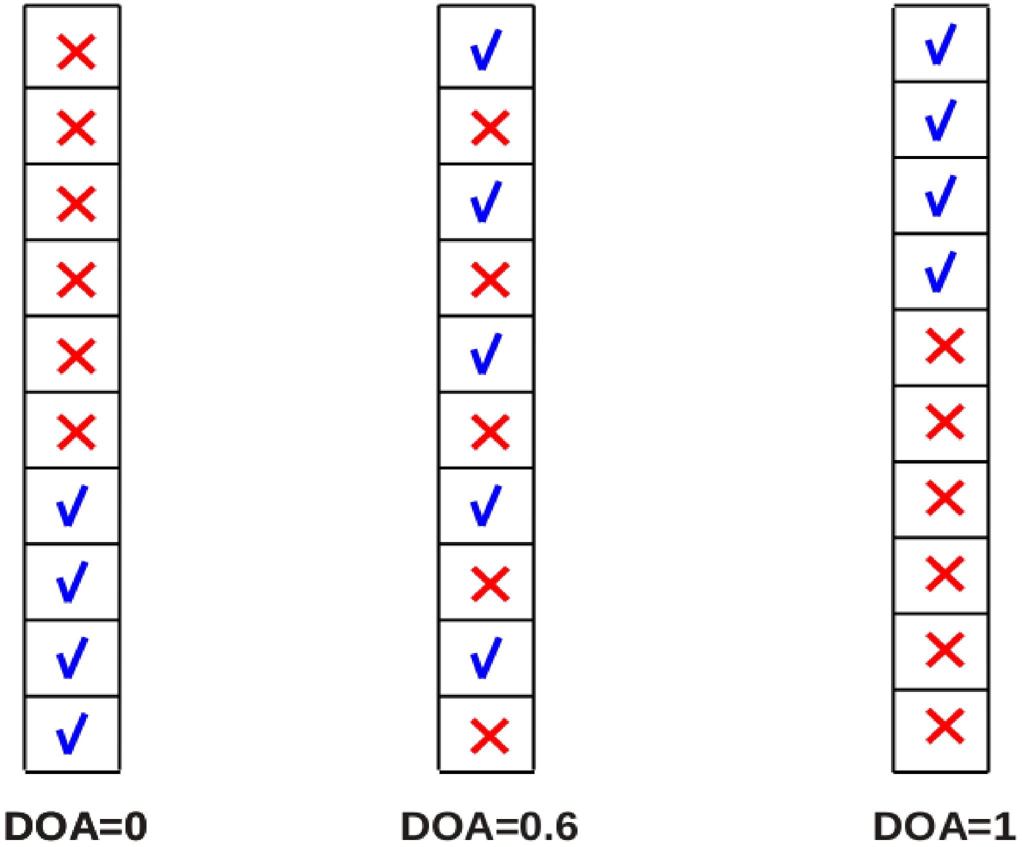
V. Evaluation: Approach

- For a search, a list of sensors is returned
  - Ranked by decreasing similarity score
  - Similar sensors are ranked on top
- Issue: „**Similarity**“ is highly subjective! → no ground truth
- Fact: Sensors close to each other have similar readings
- Approach: Group sensors based on location and annotated group with its location

V. Evaluation: Approach



V. Ranked List: Degree of Accuracy



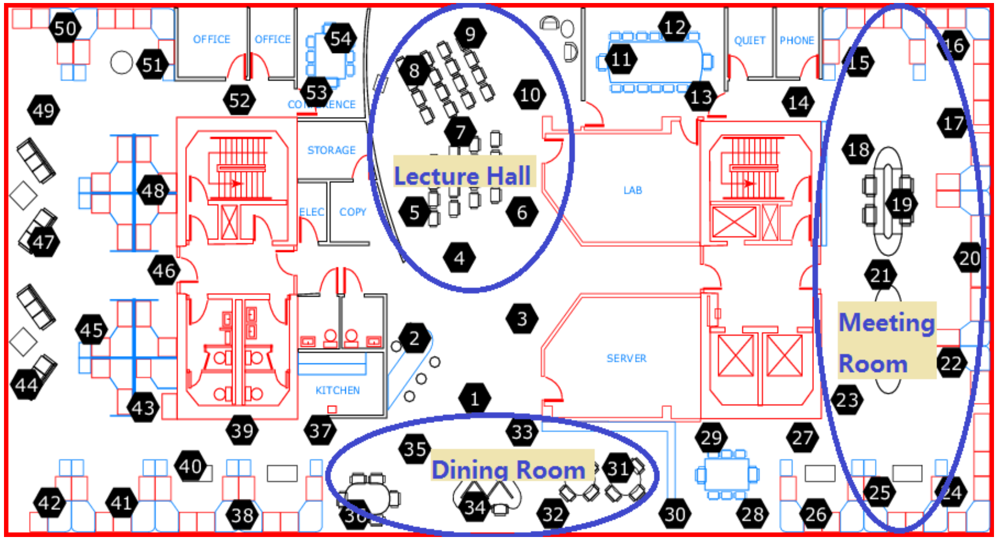
$$DOA(L) = 1 - \frac{1}{C_L(N_L - C_L)} \times \sum_{i=1}^{N_L} e_L(i)$$

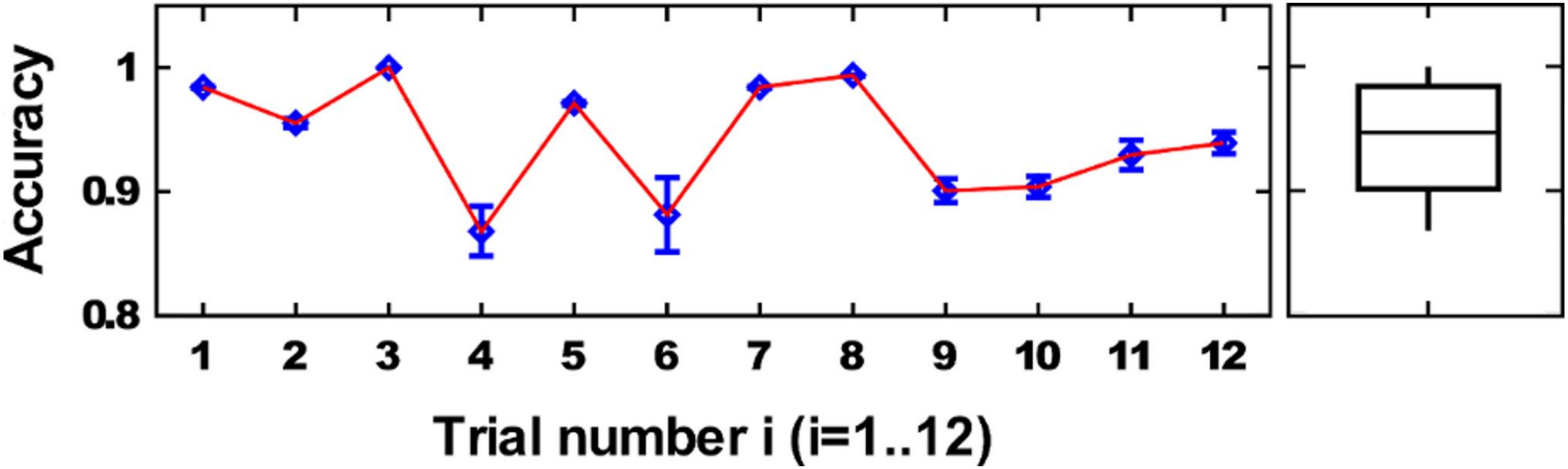
V. Evaluation: Multiple Real Data Sets

- For each data set, group sensors based on location, and define a **search trial** as
  - Picking a sensor and perform search
  - Compute DOA value of the obtained ranked list
- For each sensor
  - Last 24 hours of readings are used
- Evaluation is done on a PC
  - Java VM
  - Intel Core i5 CPU at 2.4 Ghz clock rate

IntelLab Data Set

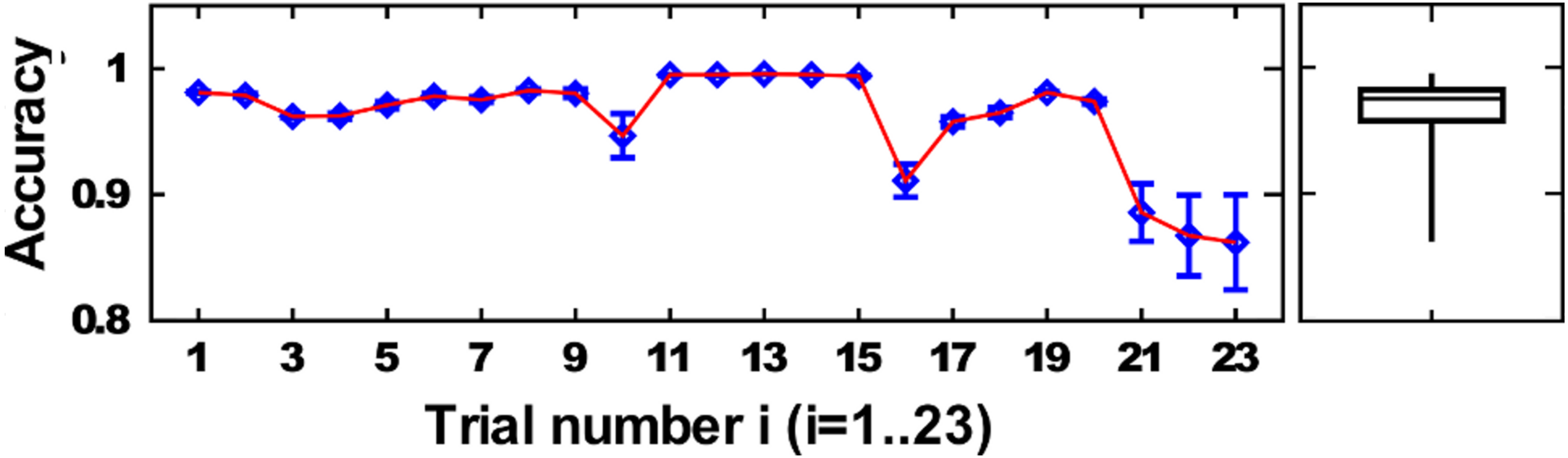
- http://db.csail.mit.edu/labdata/labdata.html
- 12 sensors in 3 groups
- 1500 data points/24 hours
- Performance: 222 μs / pair → 4505 sensors / second (brute force)





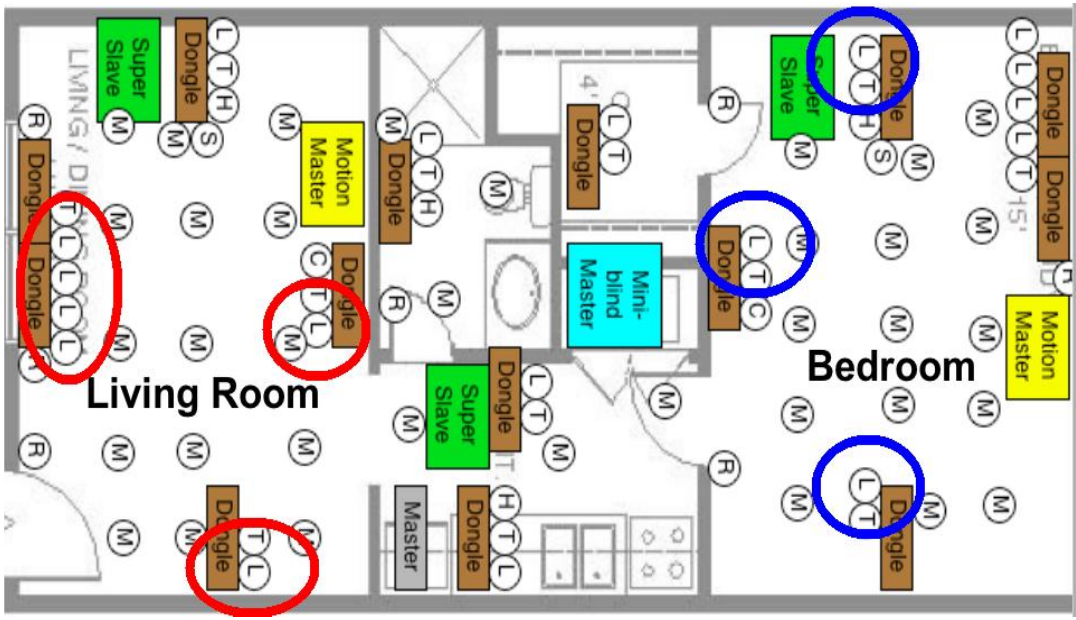
NOAA Data Set

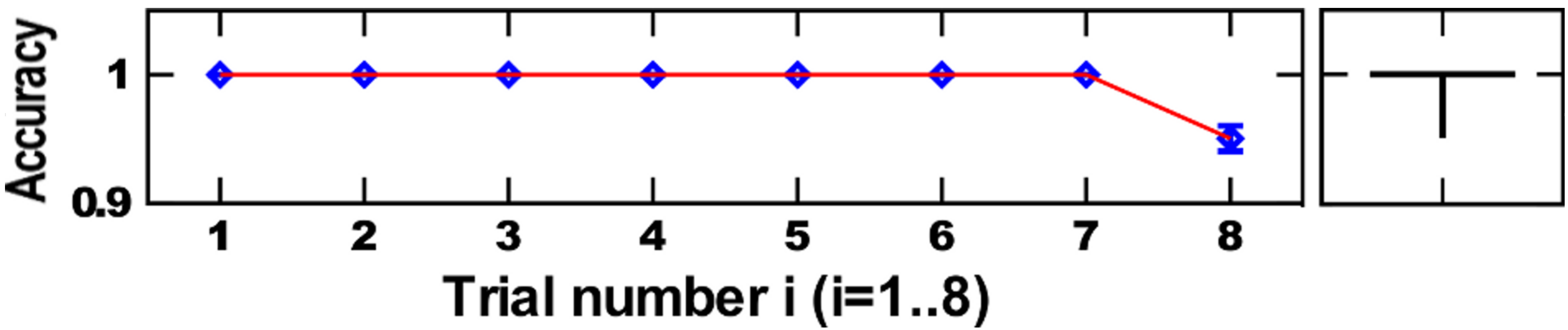
- http://tidesandcurrents.noaa.gov/gmap3
- 23 sensors in 5 groups
- 200 data points/24 hours
- Performance: 28 μs / pair → 35741 sensors / second (brute force)



MavPad Data Set

- http://ailab.wsu.edu/mavhome/index.html
- 8 sensors in 2 groups
- 500 data points / 24 hours
- Performance: 70 µs / pair → 14285 sensors / second (brute force)





Summary

- Sensor similarity search and distributed architecture to realize it
- Fuzzy-based approach to efficiently compute similarity score
- Evaluation metric for ranked list
- Accurate results of evaluation
- <u>Outlook</u>: Scalability
  - Paralellize search
  - More efficient similarity computation
  - Index and lookup of fuzzy sets at server side
  - Incremental search accuracy

THANK YOU!