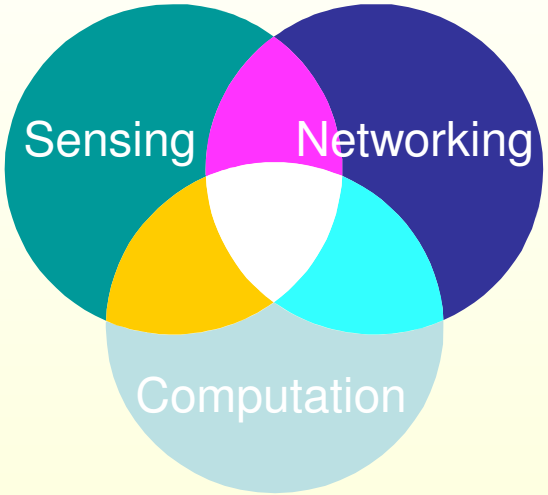
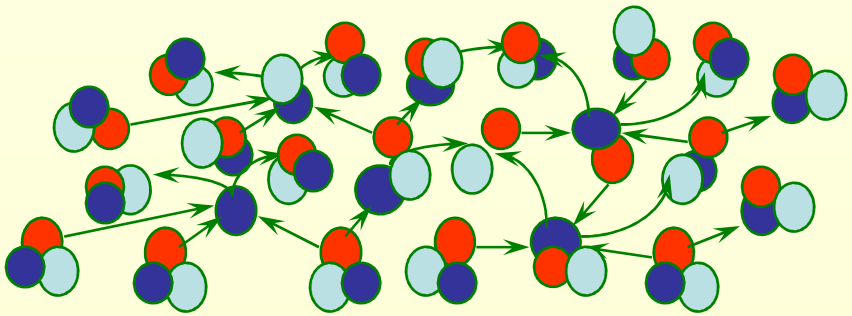
CS321: Distributed Probabilistic Reasoning, Regression

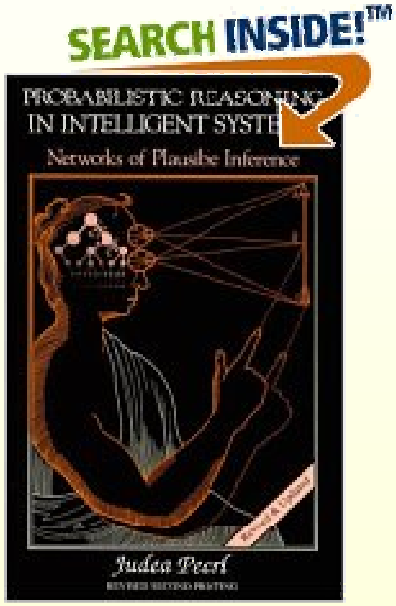


Leonidas Guibas
Computer Science Dept.
Stanford University



Beyond simple data extraction: Reasoning directly in the network

Yet Another Detour



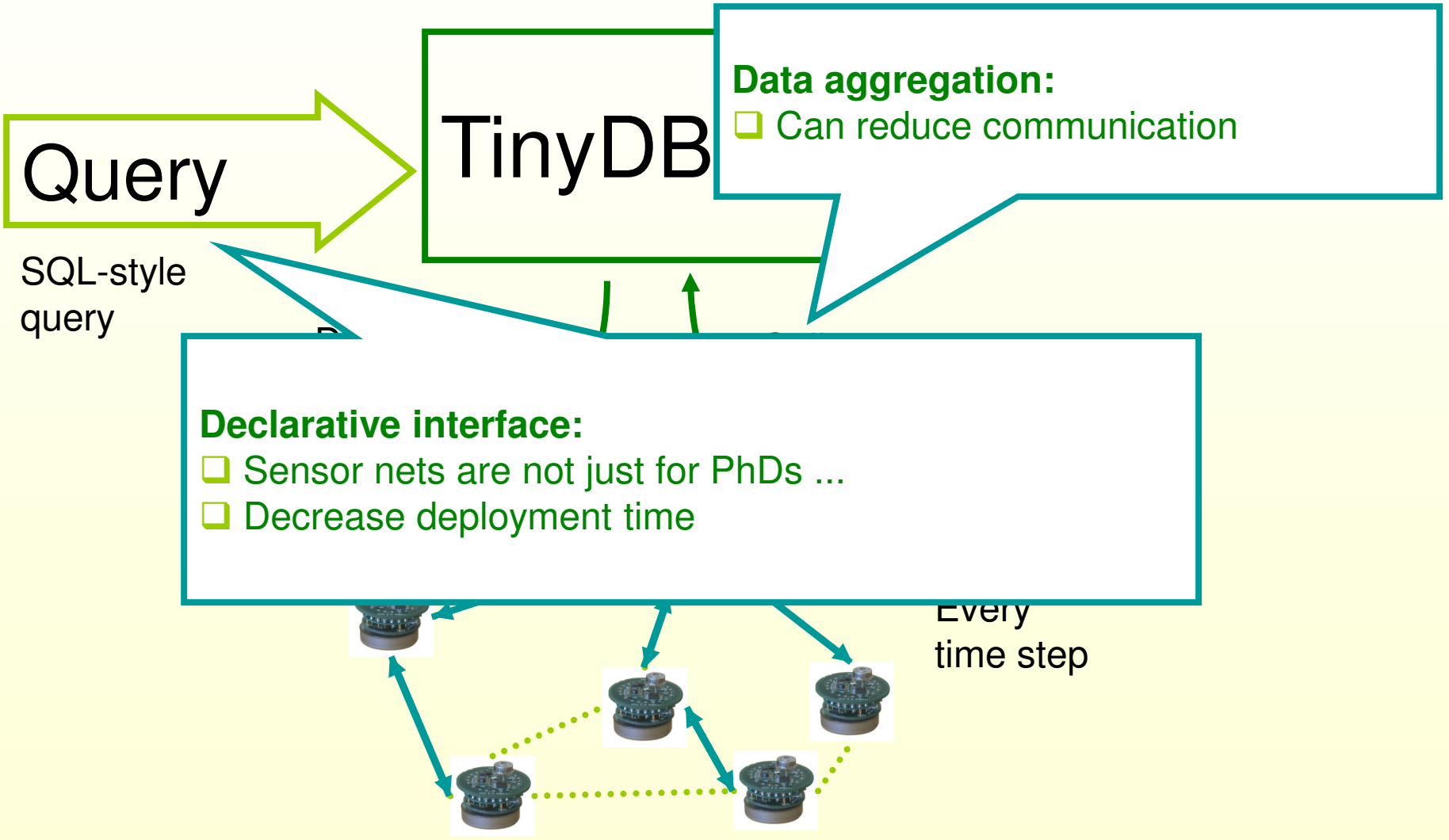
Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference
by Judea Pearl

Model-driven data acquisition, exploitation of correlations

Papers

- A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, W.Hong. "Model-Driven Data Acquisition in Sensor Networks," In the 30th International Conference on Very Large Data Bases (VLDB 2004), Toronto, Canada, August 2004.
- A. Deshpande, C. Guestrin, W. Hong, S. Madden. "Exploiting Correlated Attributes in Acquisitional Query Processing" In the 21st International Conference on Data Engineering (ICDE 2005), Tokyo, Japan, April 2005.

Analogy:
Sensor Net as a Database



Limitations of TinyDB Approach

Query distributio
- Every node mus

approximate answ

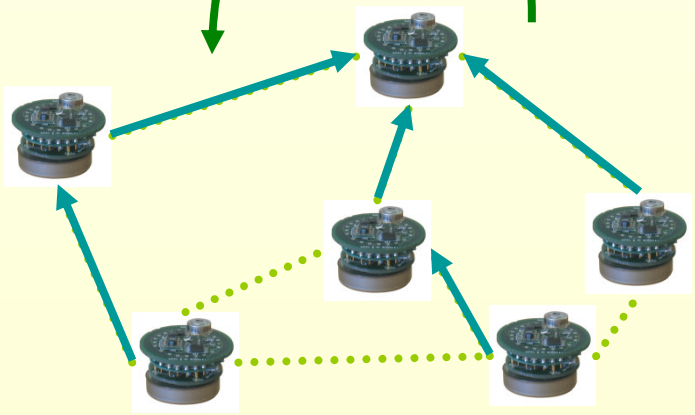
Data collection:
- Every node must wake up at every epoch
- Data loss is ignored
- No quality guarantees
- Inefficient data gathering – ignoring correlations

SQL-style query

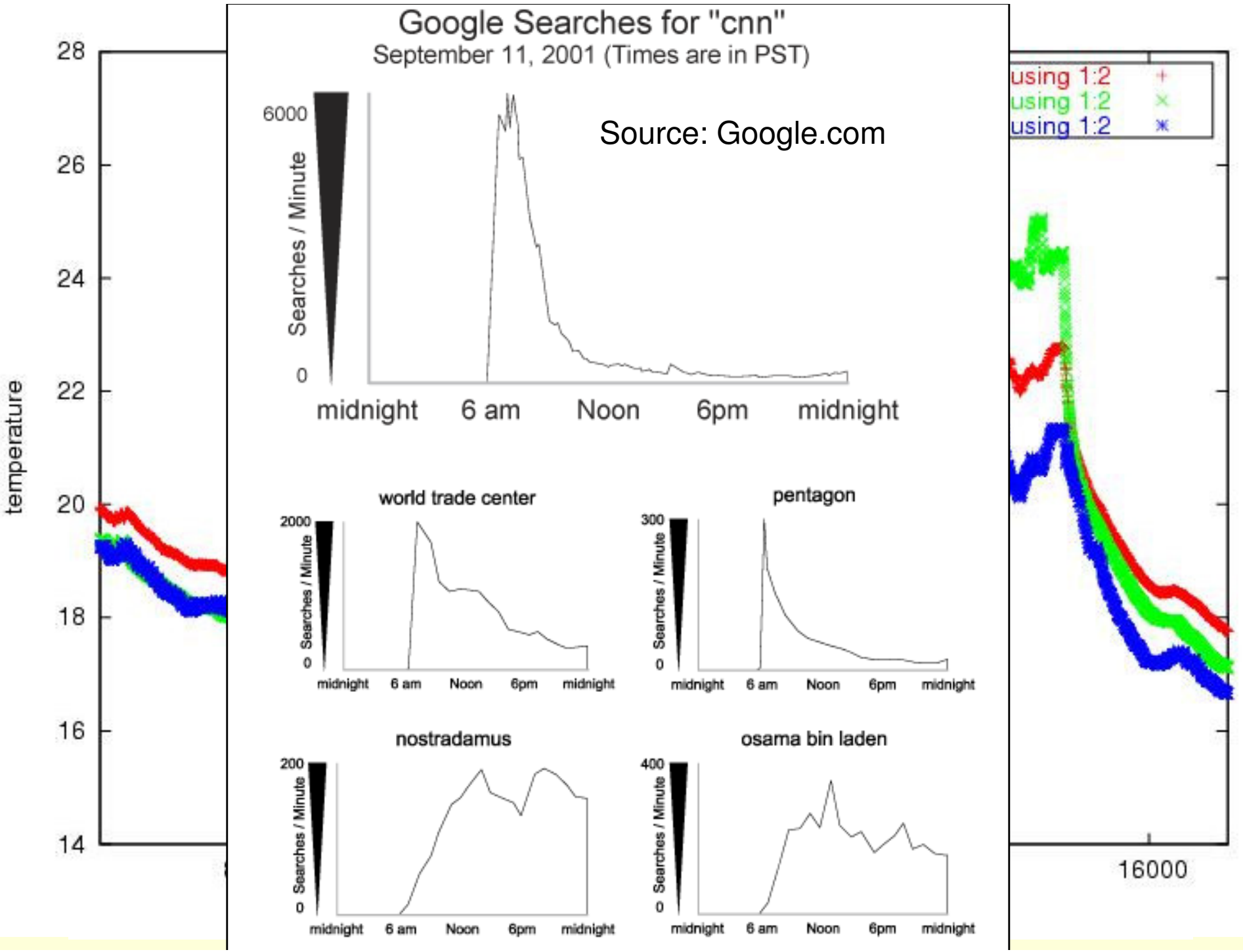
Distribute query

Collect data

every time query changes

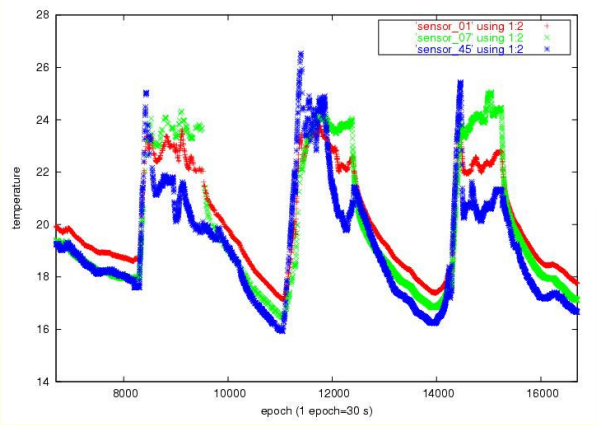


Every time step

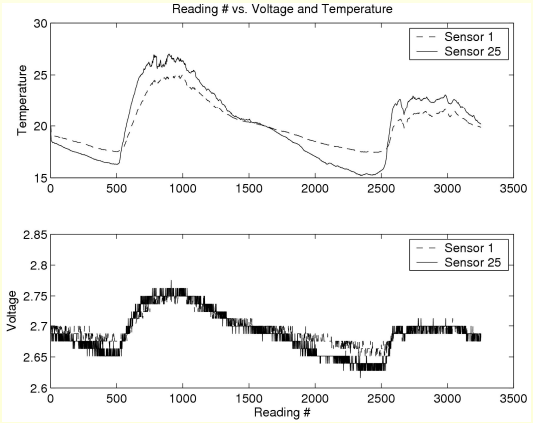

Google Searches for "cnn"
September 11, 2001 (Times are in PST)
Source: Google.com
6000
Searches / Minute
0
midnight
6 am
Noon
6pm
midnight
world trade center
2000
pentagon
300
nostradamus
200
osama bin laden
400
temperature
28
26
24
22
20
18
16
14
using 1:2
using 1:2
using 1:2
16000

Sensor Net Data is Correlated

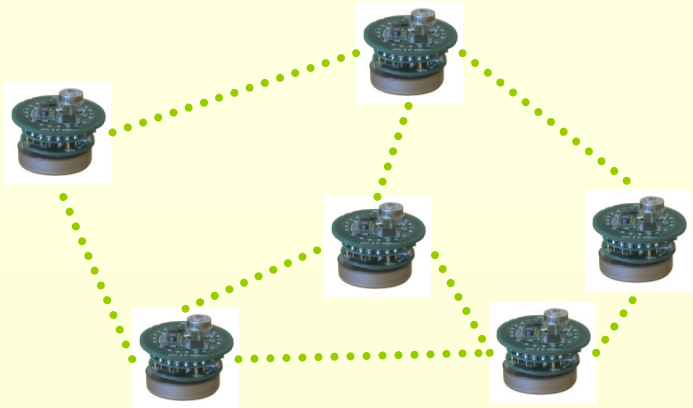
Spatial-temporal correlation



Inter-attributed correlation



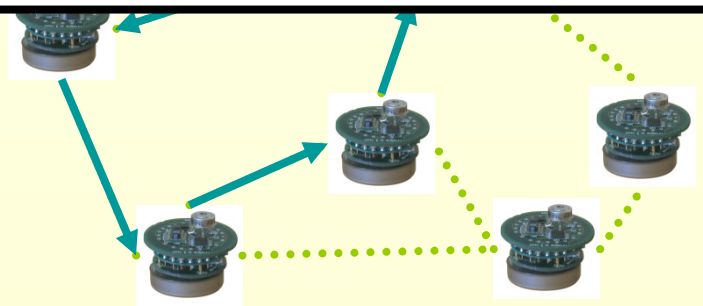
- Data is not i.i.d. ⇒ shouldn't ignore missing data
- Observing one sensor ⇒ information about other sensors (and future values)
- Observing one attribute ⇒ information about other attributes

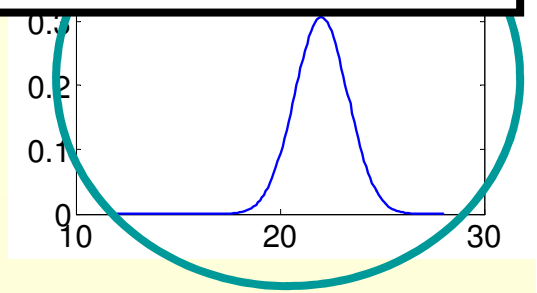


Model-Driven Data Acquisition: Overview

Strengths of model-based data acquisition

- Observe fewer attributes
- Exploit correlations
- Reuse information between queries
- Directly deal with missing data
- Answer more complex (probabilistic) queries





Benefits of Statistical Models

- More robust interpretation of sensor net readings
  - Account for biases in spatial sampling
  - Identify faulty sensors
  - Extrapolate missing values of sensors
- More efficient data acquisition
  - Fewer of attributes to observe
  - Reuse of information between queries
  - Exploit correlations – acquire data only when model not able to answer query with acceptable confidence
- More complex queries
  - Probabilistic queries

Issues Introduced by Models

- Optimization problem
  - Given query and model, choose data acquisition plan to best refine answer
  - Two factors:
    - statistical benefit of acquiring readings
    - sensing costs
- Any non-trivial statistical model can capture first factor
  - Improving model-driven estimates for nearby nodes
- Connectivity of wireless sensornet affects second factor

Probabilistic Models and Queries

User's perspective:

Query

SELECT nodeId, temp ± 0.5°C, conf(.95) FROM sensors
WHERE nodeId in {1..8}

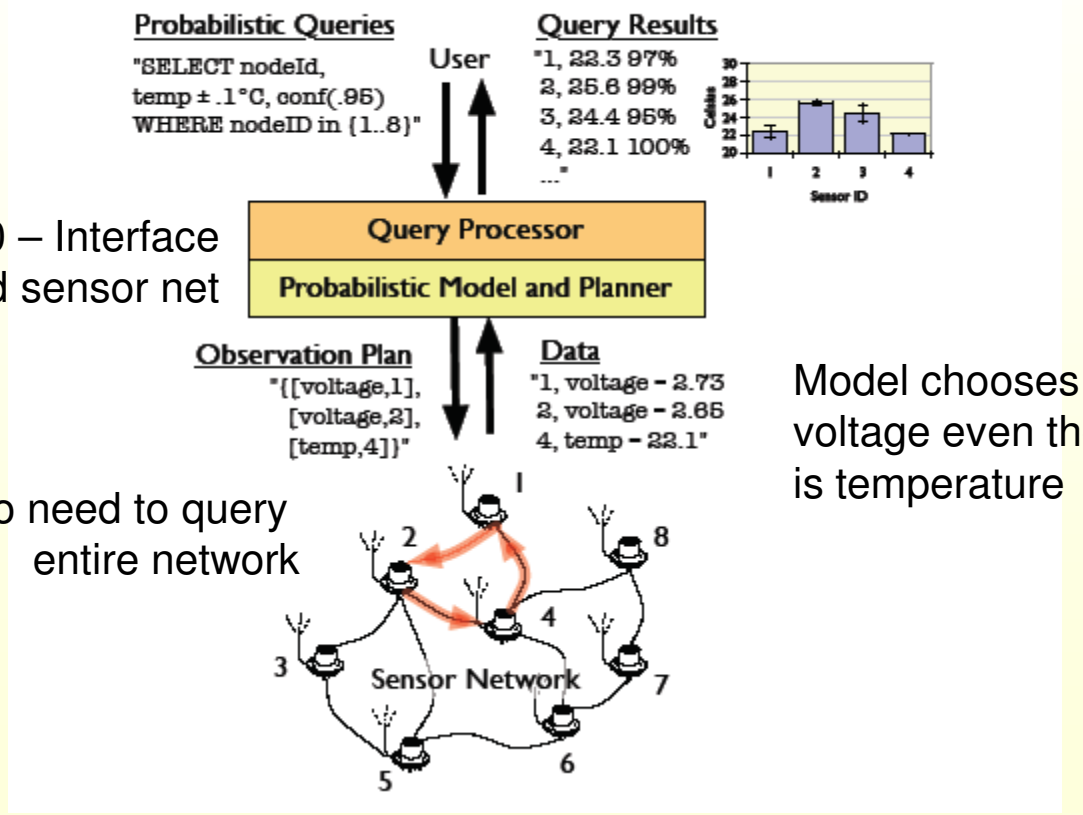
System selects and observes subset of nodes

Observed nodes: {3,6,8}

Query result

Node	1	2	3	4	5	6	7	8
Temp.	17.3	18.1	17.4	16.1	19.2	21.3	17.5	16.3
Conf.	98%	95%	100%	99%	95%	100%	98%	100%

Probabilistic Models - Illustration



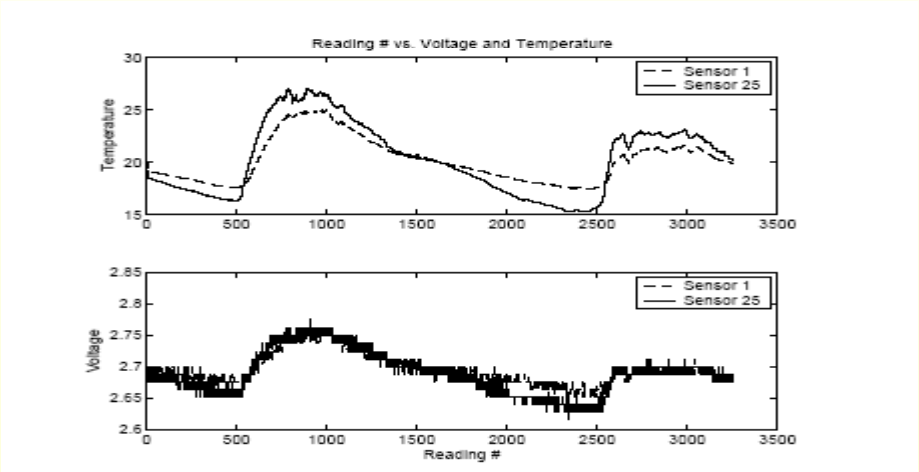
Node 0 – Interface between user and sensor net

No need to query entire network

Model chooses to observe voltage even though query is temperature

Probabilistic Models (Contd.)

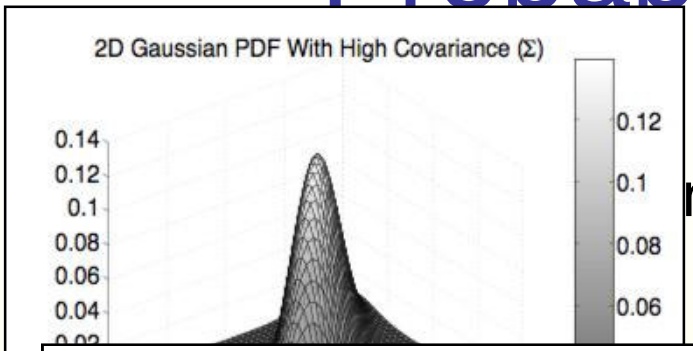
- Why did model choose to observe voltage instead of temperature?
  - Correlation in value

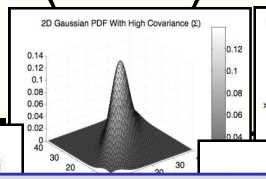


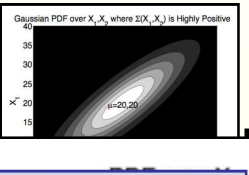
  - Cost differential

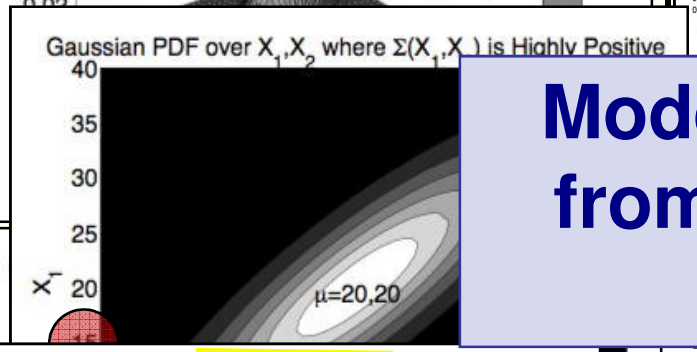
Probabilistic Models

n (PDF) to estimate current state

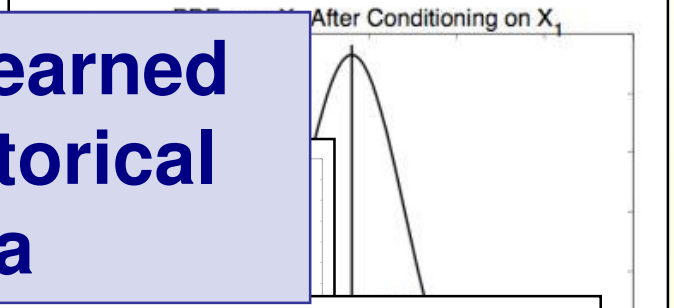


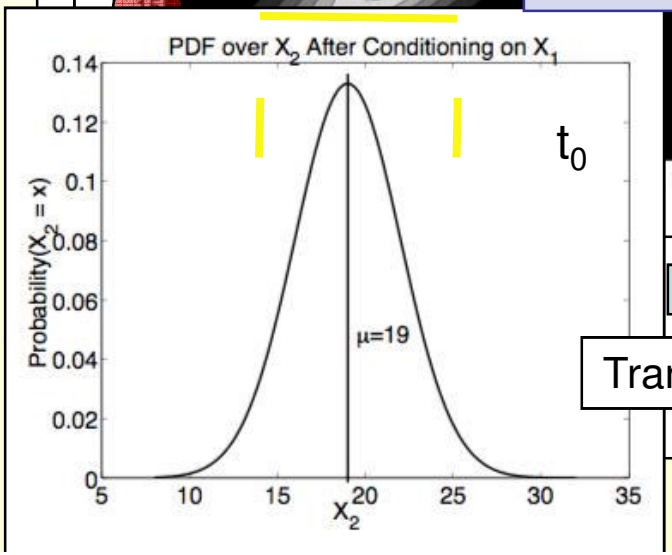




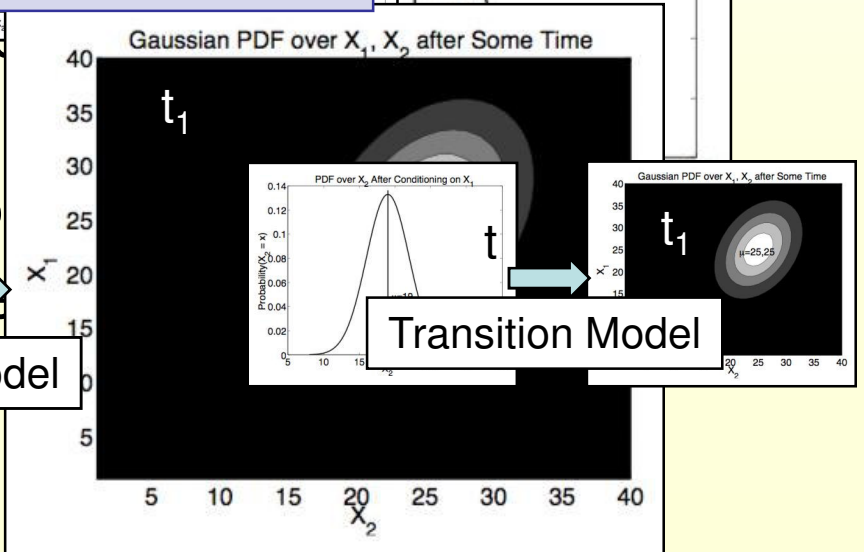


Models learned from historical data





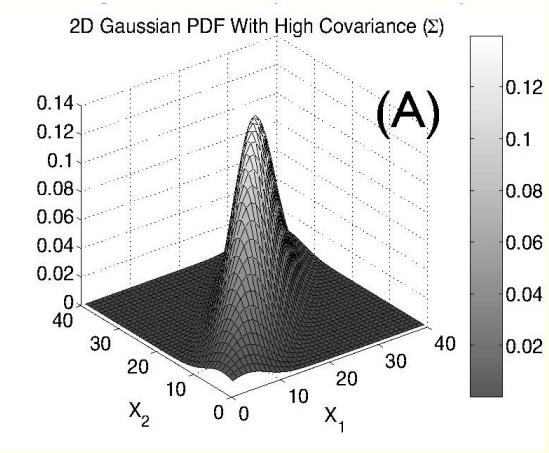
Transition Model



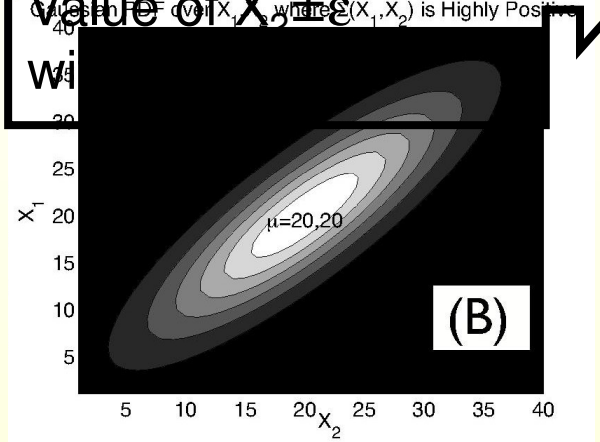
Transition Model

Probabilistic Models and Queries

Joint distribution
$P(X_1,\ldots,X_n)$



Probabilistic query
Example:
Value of $X_2 \pm \epsilon$
wi



$$P(X_2 \in [\mu_2 - \epsilon, \mu_2 + \epsilon]) = \int_{\mu_2-\epsilon}^{\mu_2+\epsilon} P(x_2)dx_2$$

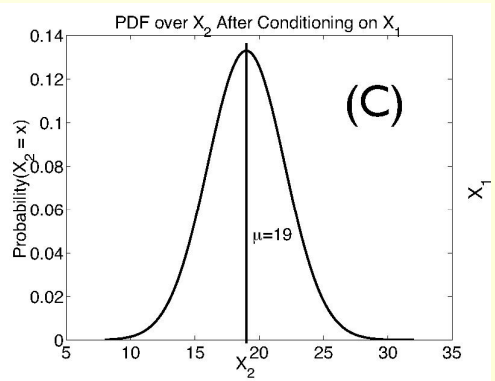
Prob. below 1-δ?

Observe attributes
Example: Observe X_1=18

$P(X_2|X_1=18)$

$$P(X_2 \in [\mu_2 - \epsilon, \mu_2 + \epsilon] \mid X_1 = 18) = \int^{\mu_2+\epsilon} P(x_2 \mid X_1 = 18)dx_2$$

- Learn from historical data



Dynamic Models: Filtering

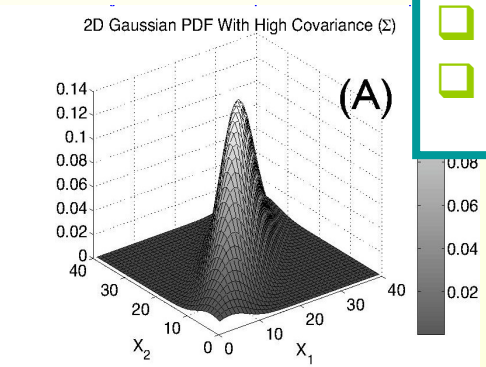
Joint distribution at time t

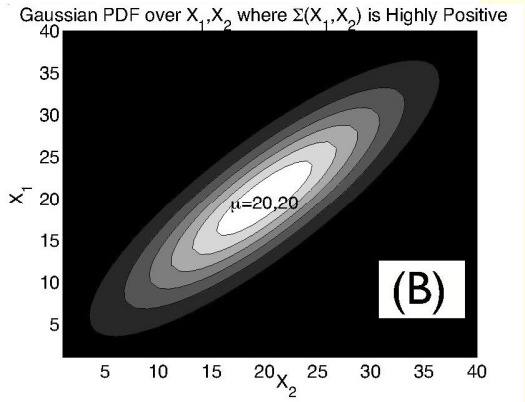
$$P(X_1^t, \ldots, X_n^t)$$

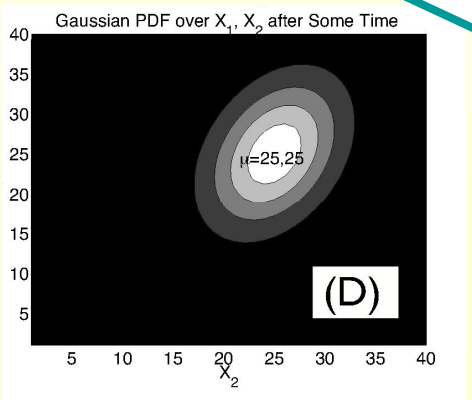
Condition on observations

$$P(X_1^t, \ldots, X_n^t \mid o^t)$$

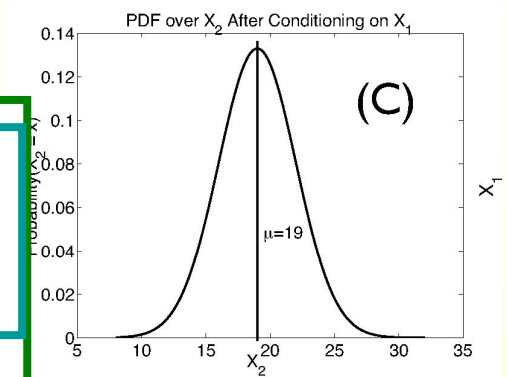
- Example: Kalman filter
- Learn from historical data







$$P(X_1^{t+1}, X_2^{t+1} \mid X_1^t = 18)$$



$$P(X_2^t \mid X_1^t = 18)$$

Δt

$$P(\mathbf{X}^{t+1} \mid \mathbf{X}^t)$$

Supported Queries

- Value query
  - $X_i \pm \varepsilon$ with prob. at least $1-\delta$
- SELECT and Range query
  - $X_i \in [a,b]$ with prob. at least $1-\delta$
  - which sensors have temperature greater than 25°C ?
- Aggregation
  - average $\pm \varepsilon$ of subset of attribs. with prob. $> 1-\delta$
  - combine aggregation and selection
  - probability > 10 sensors have temperature greater than 25°C ?

Queries require the evaluation of integrals

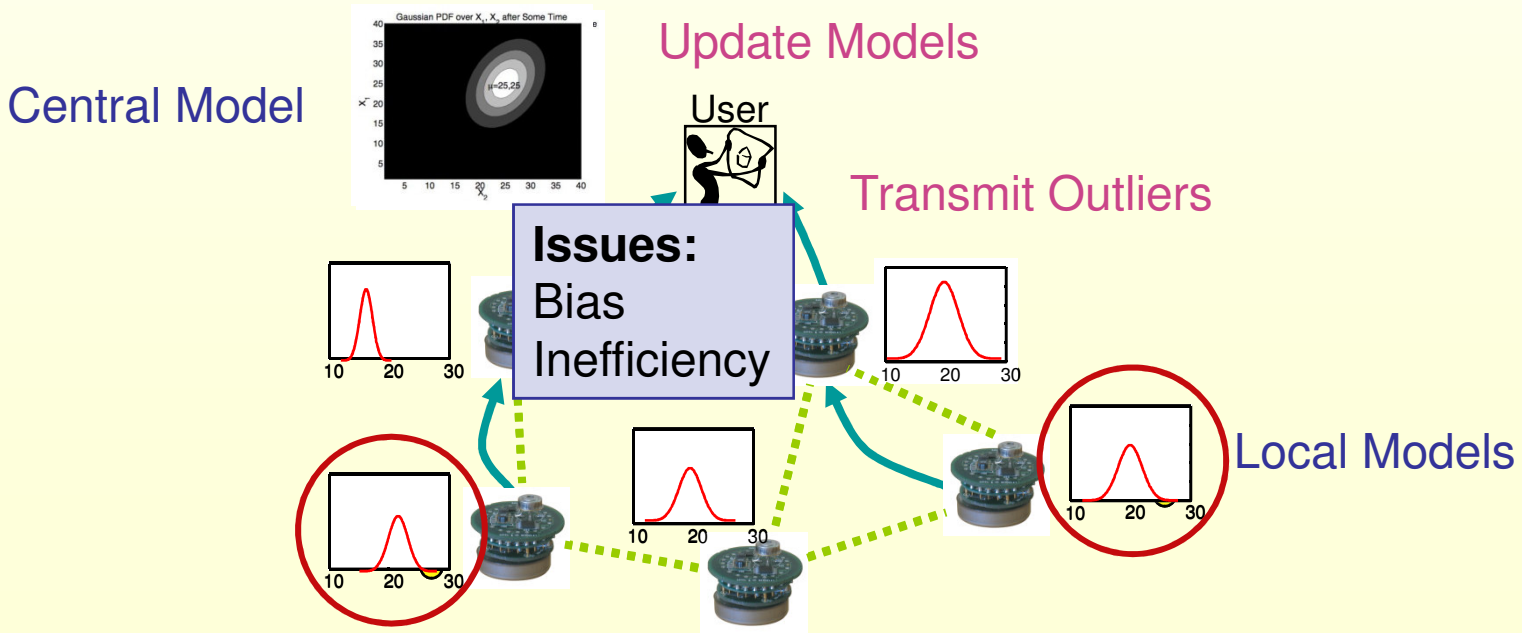
- Many queries can be computed in closed-form
- Some require numerical integration/sampling

Probabilistic Queries

- Range queries – $P(X_i \in [a_i, b_i])$
  - First marginalize multivariate gaussian – Done by dropping entries being marginalized
  - Compute confidence of query using error function
  - If confidence is less, make observation to improve confidence
  - Conditioning a gaussian on value of some attributes gives another gaussian

More New Queries

- Outlier queries
  - "Report temperature readings that have a 1% or less chance of occurring."
- Extend architecture with local filters:



Even More New Queries

- Prediction queries
  - "What is the expected temperature at 5PM today, given
- Influence qu
  - "What percentage of network traffic at site A is explained by traffic at sites B and C?"

Queries could not be answered without a model!

Model-Driven Data Acquisition: Overview

What sensors do we observe ?
How do we collect observations?

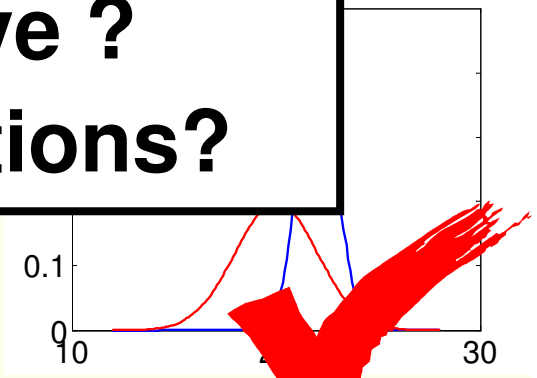
Que

SQL-style query with desired confidence

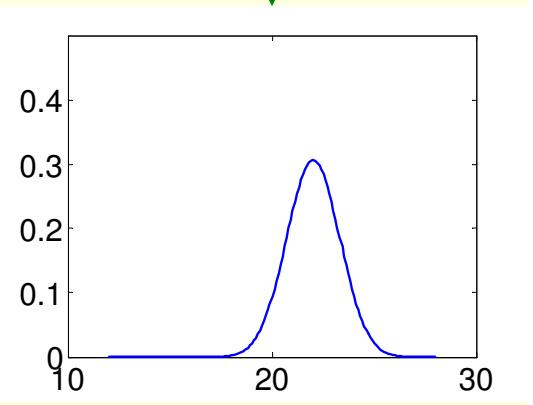
Data gathering plan

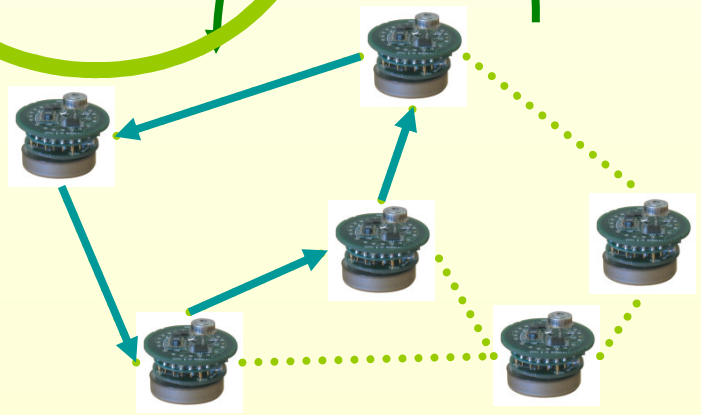
Condition on new observations

posterior belief



Δt

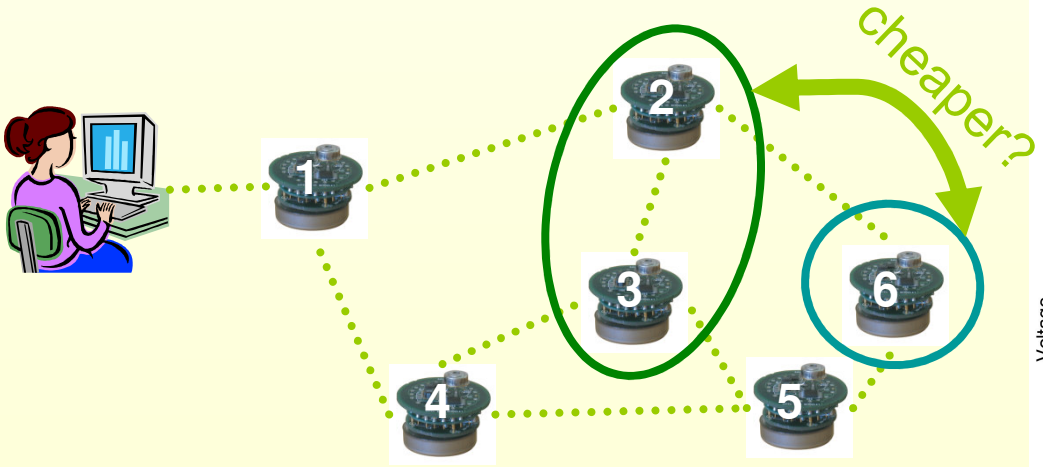


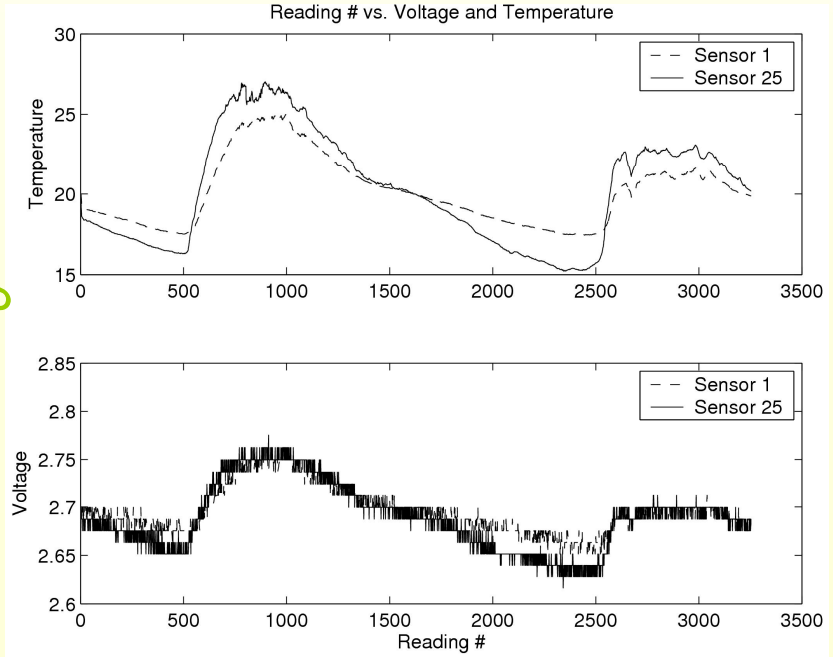


Acquisition Costs

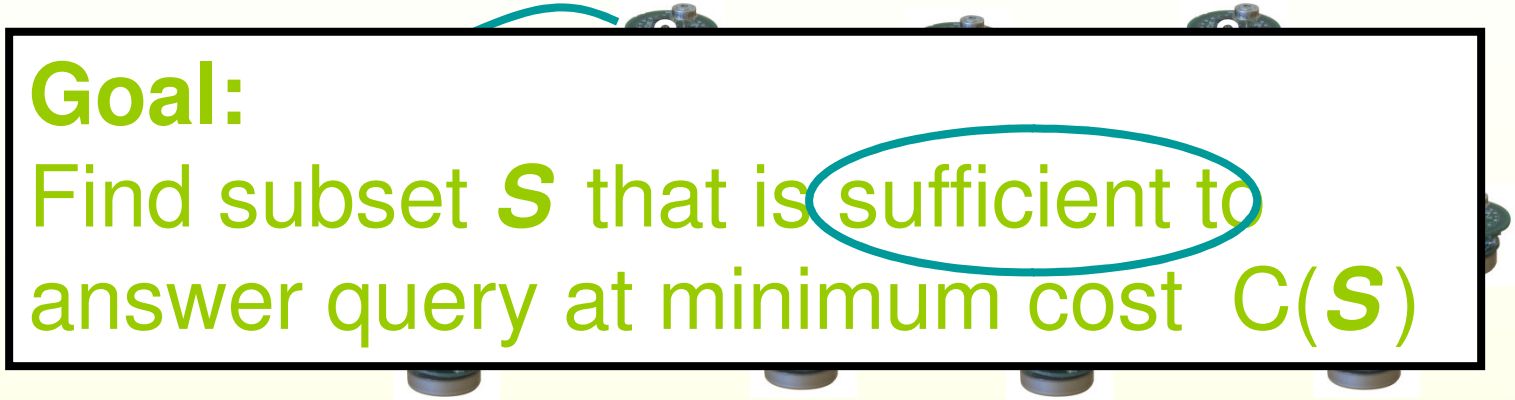
- Attributes have different acquisition costs
- Exploit correlation through probabilistic model
- Must consider networking costs

Sensor	Energy per sample (mJ)
Solar Radiation	.525
Barometric Pressure	0.003
Humidity and Temp.	0.5
Voltage	0.00009





Network Model and Plan Format



Cost of collecting subset ***S*** of sensor values:

$$C(\boldsymbol{S}) = C_a(\boldsymbol{S}) + C_t(\boldsymbol{S})$$

Sensor	Energy per sample (mJ)
Solar Radiation	.525
Barometric Pressure	0.003
Humidity and Temp.	0.5
Voltage	0.00009

- Assume known (quasi-static) network topology
- Define traversal using (1.5-approximate) TSP
- $C_t(\boldsymbol{S})$ is *expected* cost of TSP

Choosing Observation Plan

Is a subset $\boldsymbol{S}$ sufficient? $X_i \in [a,b]$ with prob. $> 1-\delta$

If we observe $\boldsymbol{S}=\boldsymbol{s}$:

$$R_i(\boldsymbol{s}) = \max\{ P(X_i \in [a,b] \mid \boldsymbol{s}),\ 1-P(X_i \in [a,b] \mid \boldsymbol{s})\}$$

Value of $\boldsymbol{S}$ is unknown:

$$R_i(\boldsymbol{S}) = P(\boldsymbol{s}) \int R_i(\boldsymbol{s})\, d\boldsymbol{s}$$

Optimization problem:

$$\text{minimize}_{\mathbf{S} \subseteq \{1,\ldots,n\}} \quad C(\mathbf{S}),$$
$$\text{such that} \quad R(\mathbf{S}) \geq 1 - \delta.$$

Observation Plan

- General optimization problem is NP-hard
- Two algorithms
  - Exhaustive search – exponential
  - Greedy search
    - Begin with empty observation plan
    - Compute benefit R and cost C for added attribute
    - If confidence reached, choose attribute with minimum cost
    - Else add the attribute which has maximum benefit/cost ratio
    - Repeat until you reach desired confidence

BBQ System

- Multivariate Gaussians
- Learn from historical data

- Exhaustive or greedy search
- Factor 1.5 TSP approximation

- Simple matrix operations

Query

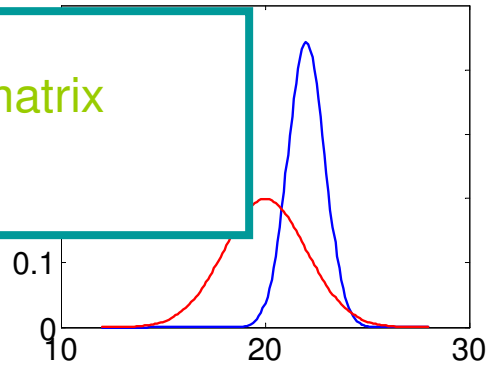
Model

SQL-style query with desired confidence

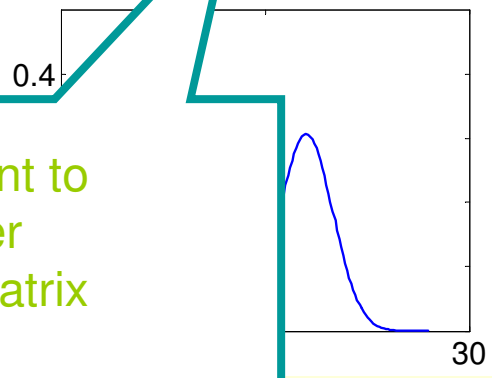
Data gathering plan

Condition on new observations

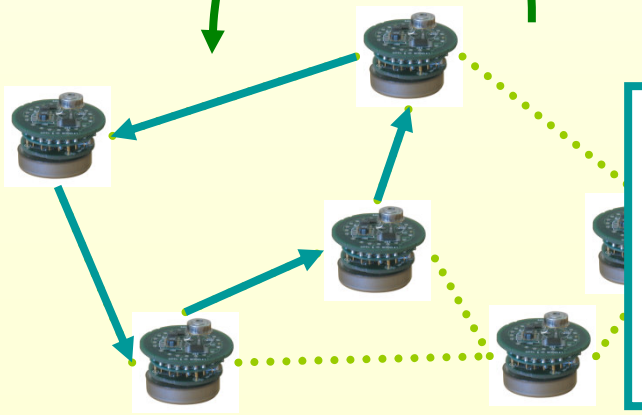
posterior belief



Δt



- Value
- Range
- Average



- Equivalent to Kalman filter
- Simple matrix operations

Exploiting Correlated Attributes

- Extension of single plan to conditional plan
  - Useful when cost of acquisition non-negligible
  - Correlations exist between one or more attributes
- Multi-predicate range queries
- Query evaluation can become cheaper by observing additional attributes
  - If additional attributes are low-cost

Conditional Plans

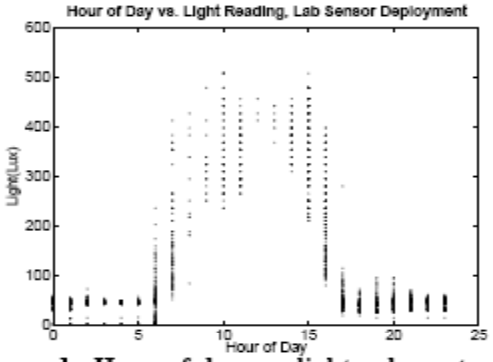


Figure 1: Hour of day vs. light values at a single sensor in authors' lab.

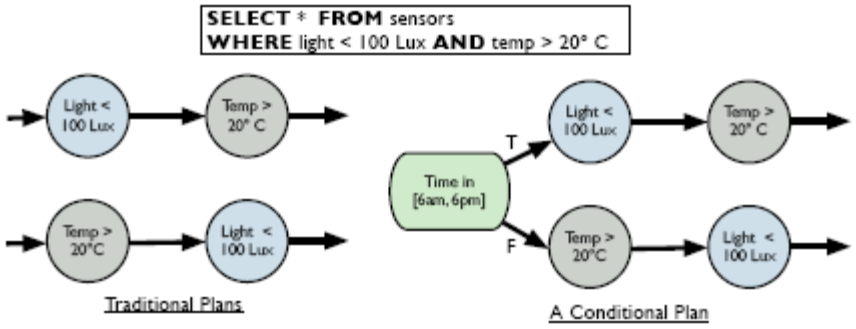
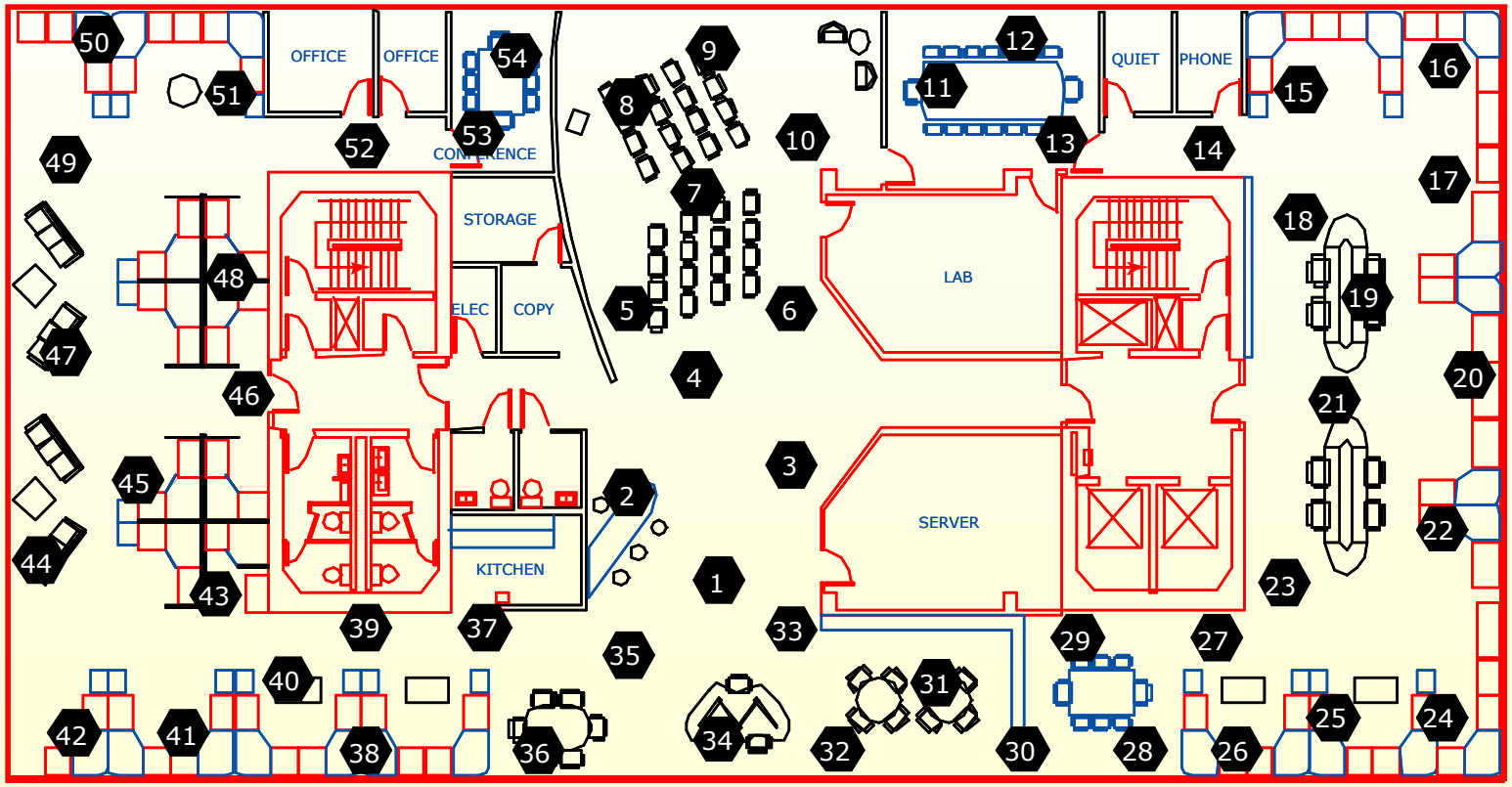
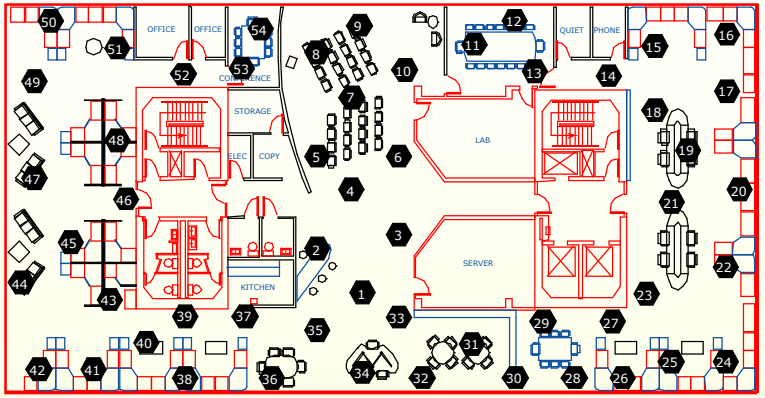


Figure 2: A conditional query plan that uses different ordering of query predicates depending on the time of day.

Example:
Intel Berkeley Lab Deployment

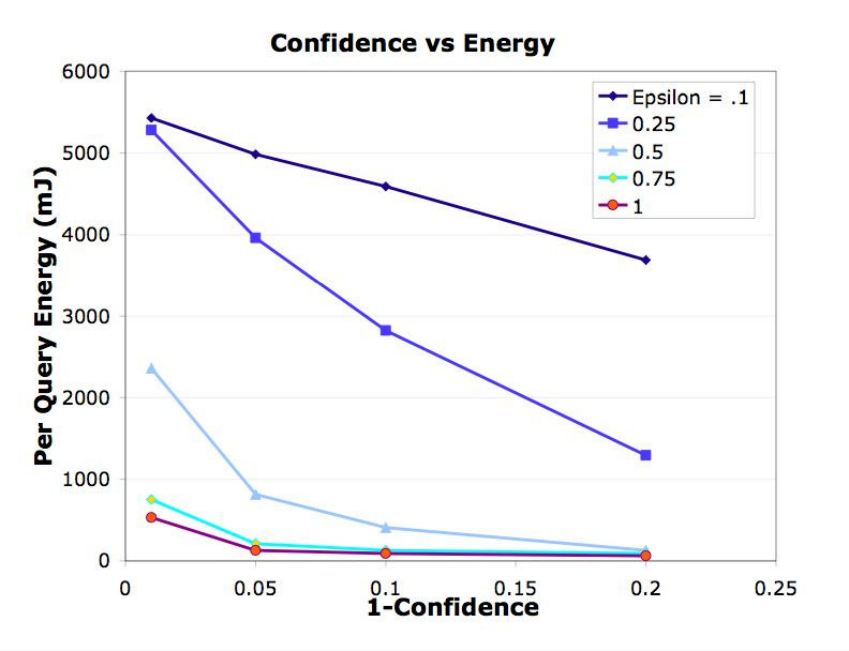


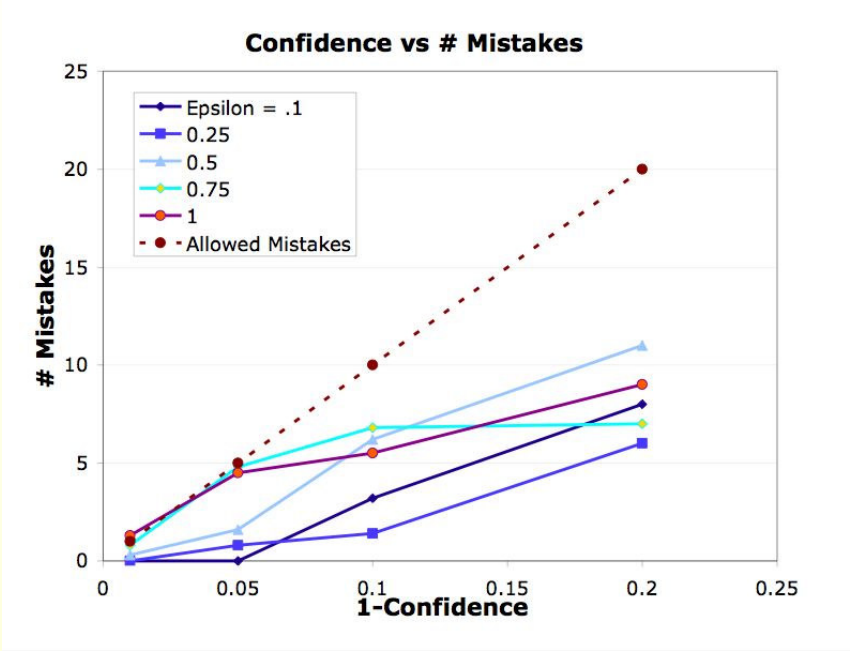
Experimental results



- Redwood trees and Intel Lab datasets
- Learned models from data
  - Static model
  - Dynamic model – Kalman filter, time-indexed transition probabilities
- Evaluated on a wide range of queries

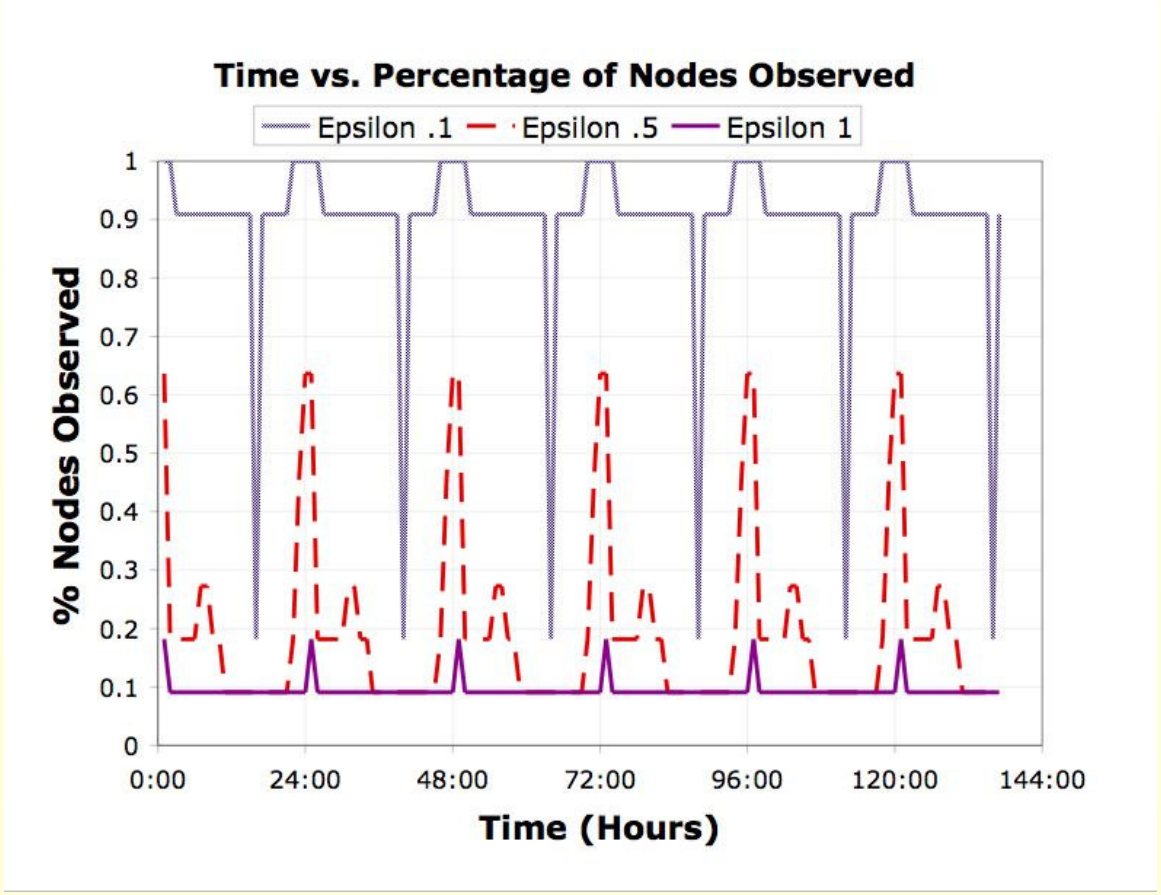
Cost vs. Confidence Level





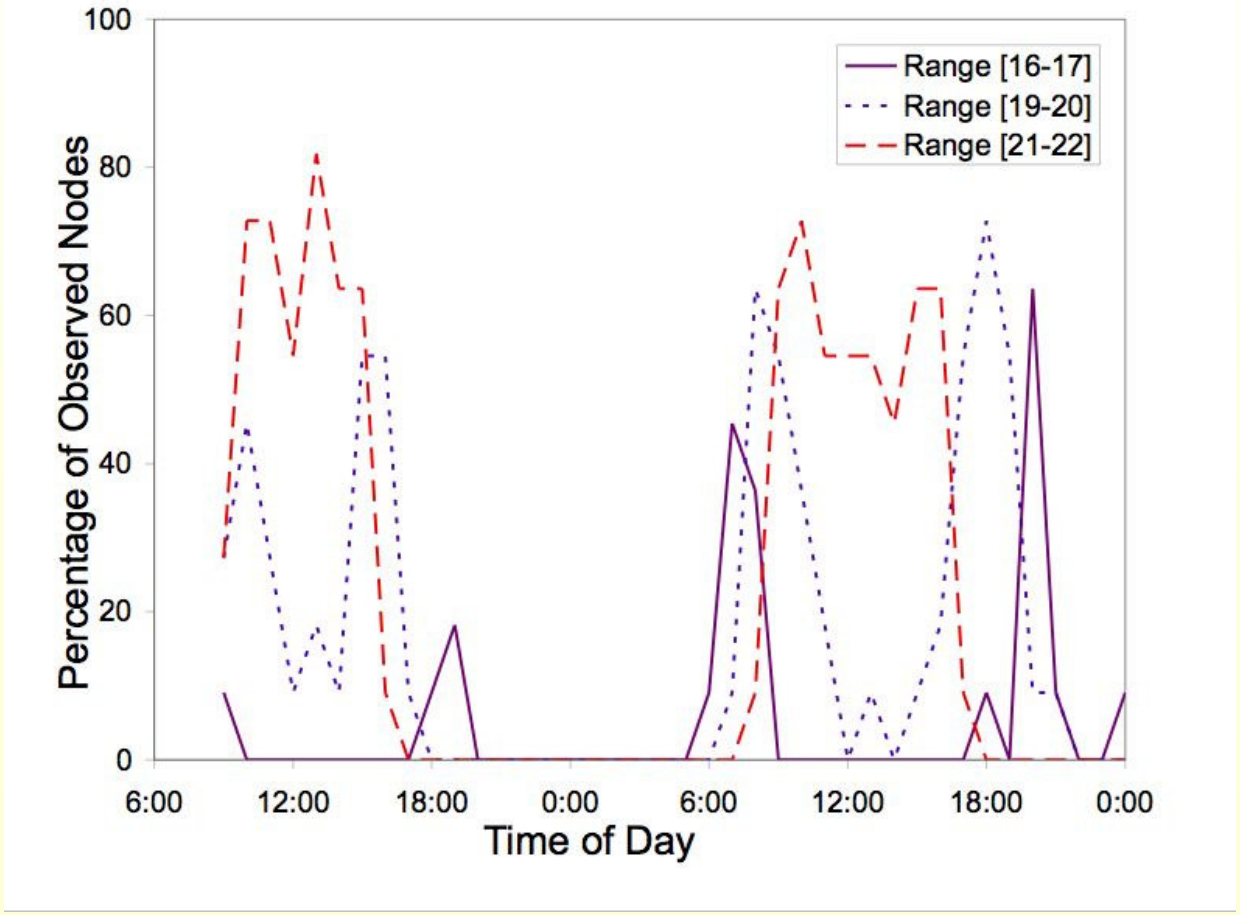
Obtaining Approximate Values

Query: True temperature value ± epsilon with confidence 95%

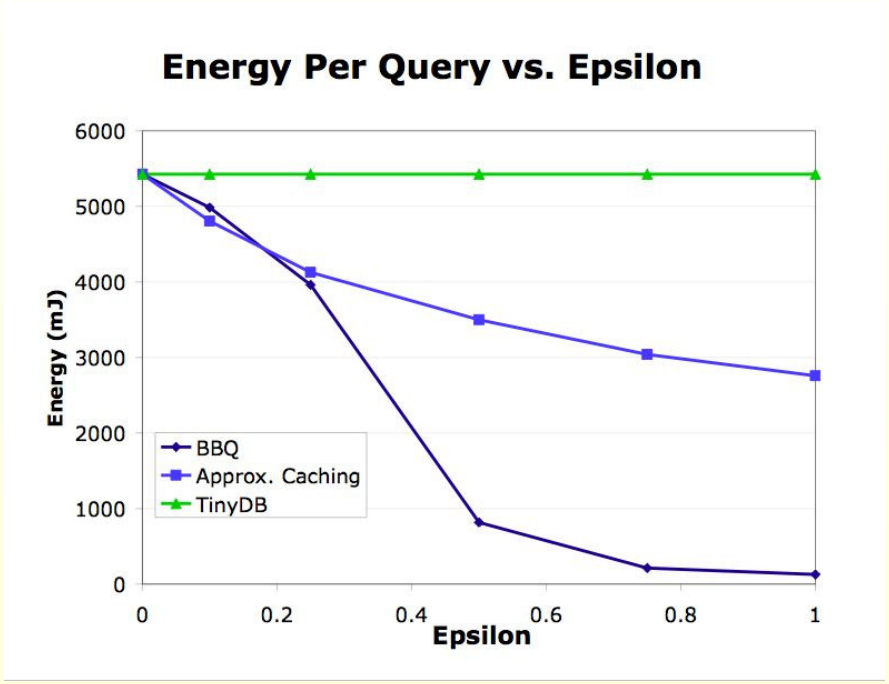


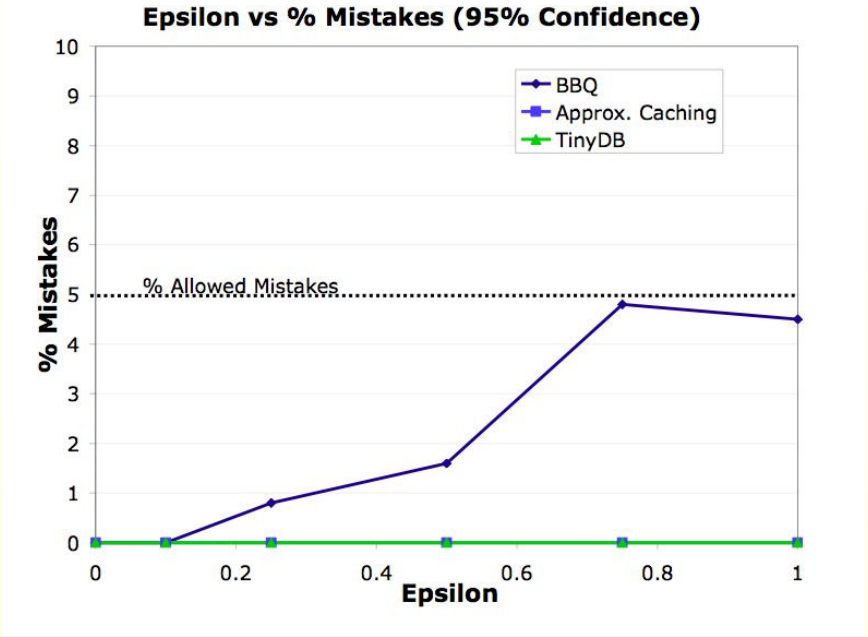
Approximate Range Queries

Query: Temperature in $[T_1,T_2]$ with confidence 95%



Comparison to Other Methods





Applications

- Sensor-based Building Monitoring
  - Often battery powered
  - 100s-1000s of nodes
- Example: HVAC Control
  - Tolerant of approximate answers
  - Reduction in energy significant

App: Distributed System Monitoring

- Goal: detect/predict overload, reprovision
- Many metrics that may indicate overload
  - Disk usage, CPU load, network load, network latency, active queries, etc.
  - Cost to observe
- Problem: What metrics foreshadow overload?
- Soln:
  - Train on data labeled w/ overload status
  - Choose obs. plan that predicts label

Model-Driven Summary

- Model-driven data acquisition
  - Observe fewer attributes
  - Exploit correlations
  - Reuse information between queries
  - Directly deal with missing data
  - Answer more complex (probabilistic) queries
- Basis for future sensor network systems

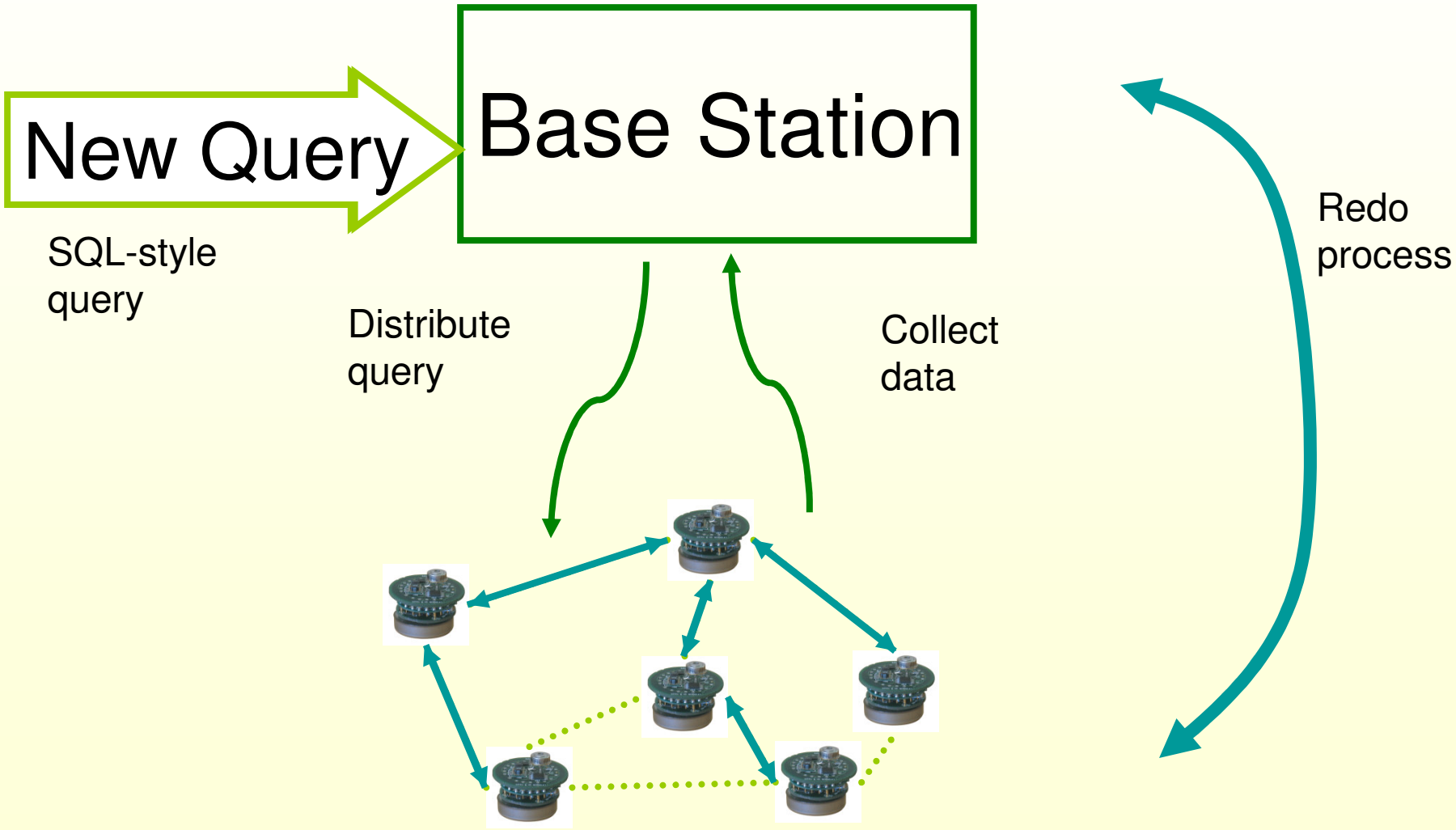
Discussion Questions

- What other models apart from multivariate Gaussian can be used? If other models are used, will their solution be in closed form?
- Model-driven techniques are suitable only if test data is same as training data. Will solution be adaptable if test region is different from training region?
- Optimization problem is hard and expensive to compute even with heuristics. Will it work for real-time data analysis?
- Outlier detection is not supported for model-driven acquisition. Is there any way to do it for model-based sensor networks?
- If in general your needed confidence on the query is low then some nodes may not be queried at all.

Distributed Regression:
an Efficient Framework for Modeling Sensor Network Data

C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, S. Madden
IPSN 2004

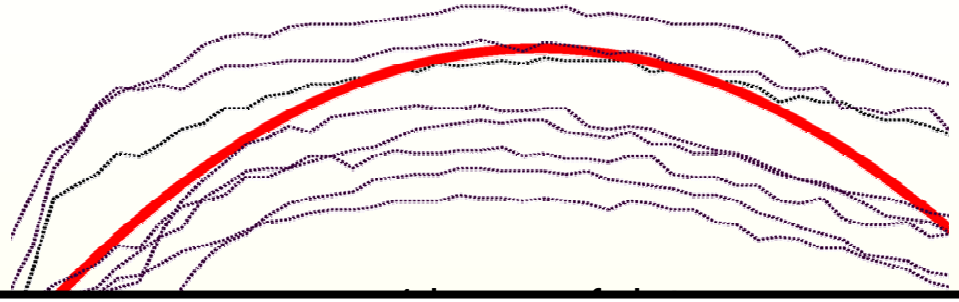
Data Collection Paradigm



Data is Highly Correlated

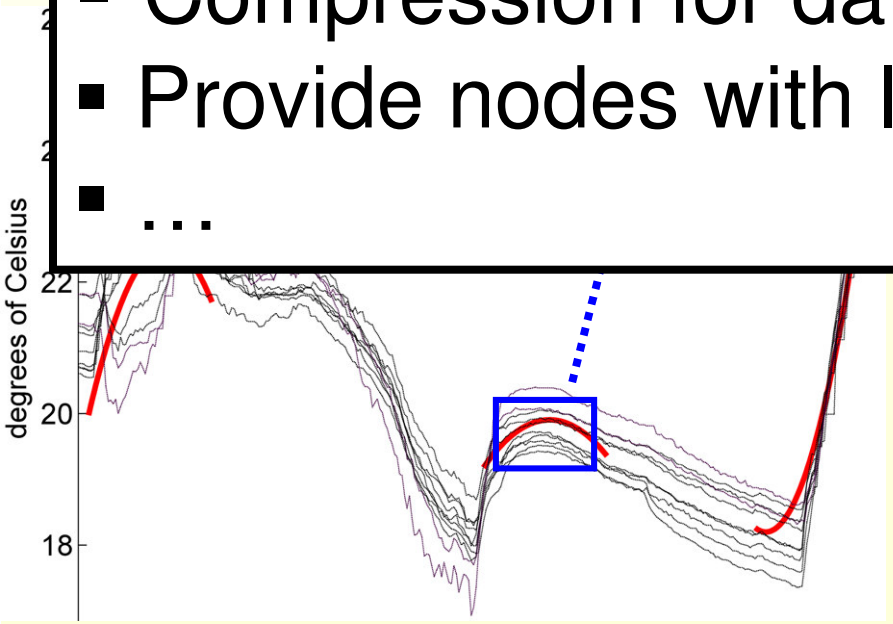
Example: temperature data from 10 nearby sensors:

- Slow changes over time
- Measurements correlated



Build lower dimensional representation

- Compression for data transmission
- Provide nodes with local view of global state
- …



Approximate measurements as

$$c_1 + c_2\mathbf{x} + c_3\mathbf{y} + c_4\mathbf{t} + c_5\mathbf{t}^2$$

using regression:

send 5 numbers!!

(yet very good approximation)

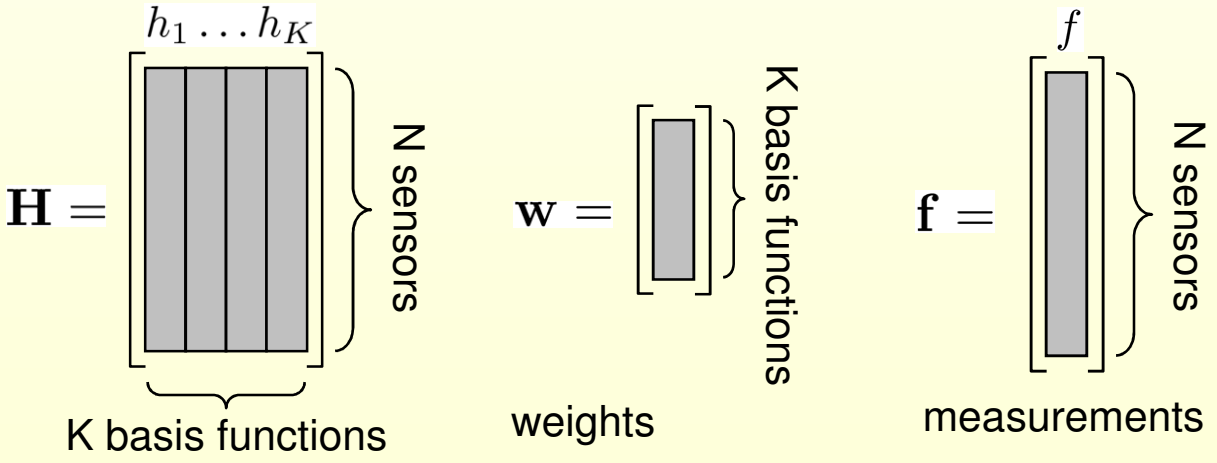
The Regression Problem

- Given, basis functions

$$H = \{h_1, \ldots, h_K\}$$

- Find coefficients **w**={w_1,…,w_k} such that

$$\underbrace{f(\mathbf{x}, t)}_{\text{data}} \approx \sum_i w_i h_i(\mathbf{x}, t)$$



The Regression Problem

- Given, basis functions

$$H = \{h_1, \ldots, h_K\}$$

- Find coefficients **w**={w_1,…,w_k} such that

$$\underbrace{f(\mathbf{x}, t)}_{\text{data}} \approx \sum_i w_i h_i(\mathbf{x}, t)$$

- More precisely, minimize the residual error:

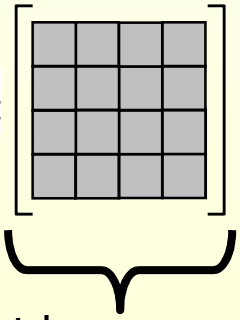
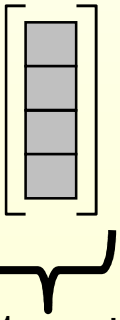
$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \sum_j \left(f(\mathbf{x}_j, t_j) - \sum_i w_i h_i(\mathbf{x}_j, t_j) \right)^2$$

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \underbrace{\|\mathbf{H}\mathbf{w} - \mathbf{f}\|}_{\text{residual error}}$$

Regression Solution

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \underbrace{\|\mathbf{H}\mathbf{w} - \mathbf{f}\|}_{\text{residual error}}$$

$$\text{solution: } \mathbf{w}^* = \underbrace{(\mathbf{H}^T\mathbf{H})^{-1}}_{\mathbf{A}^{-1}} \underbrace{\mathbf{H}^T\mathbf{f}}_{\mathbf{b}} = \mathbf{A}^{-1}\mathbf{b}$$

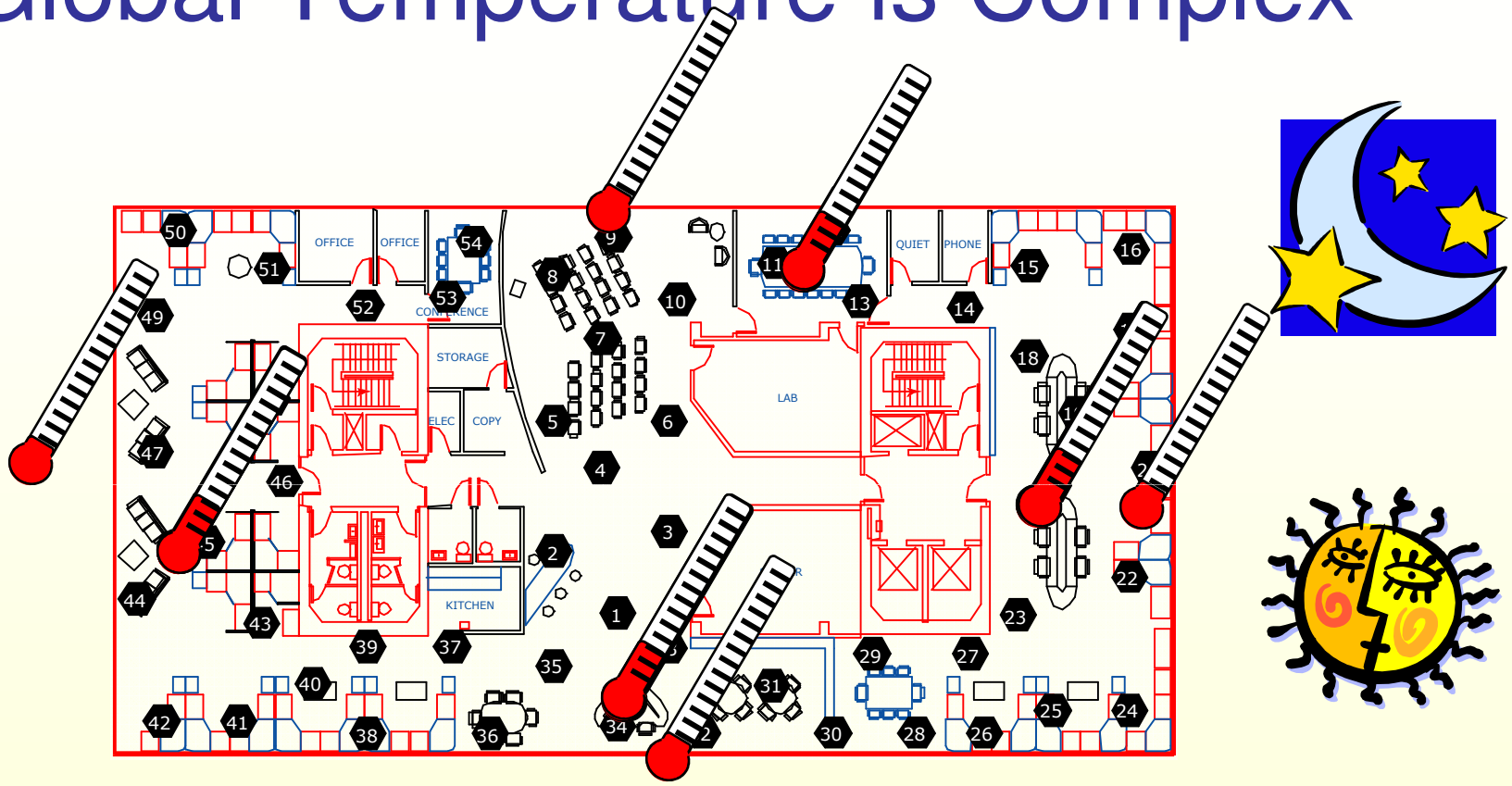
where $\mathbf{A} = \mathbf{H}^T\mathbf{H} =$  $\mathbf{b} = \mathbf{H}^T\mathbf{f} =$ 

k×k matrix
for k basis functions

k×1 vector

Classical least squares ...

Global Temperature is Complex



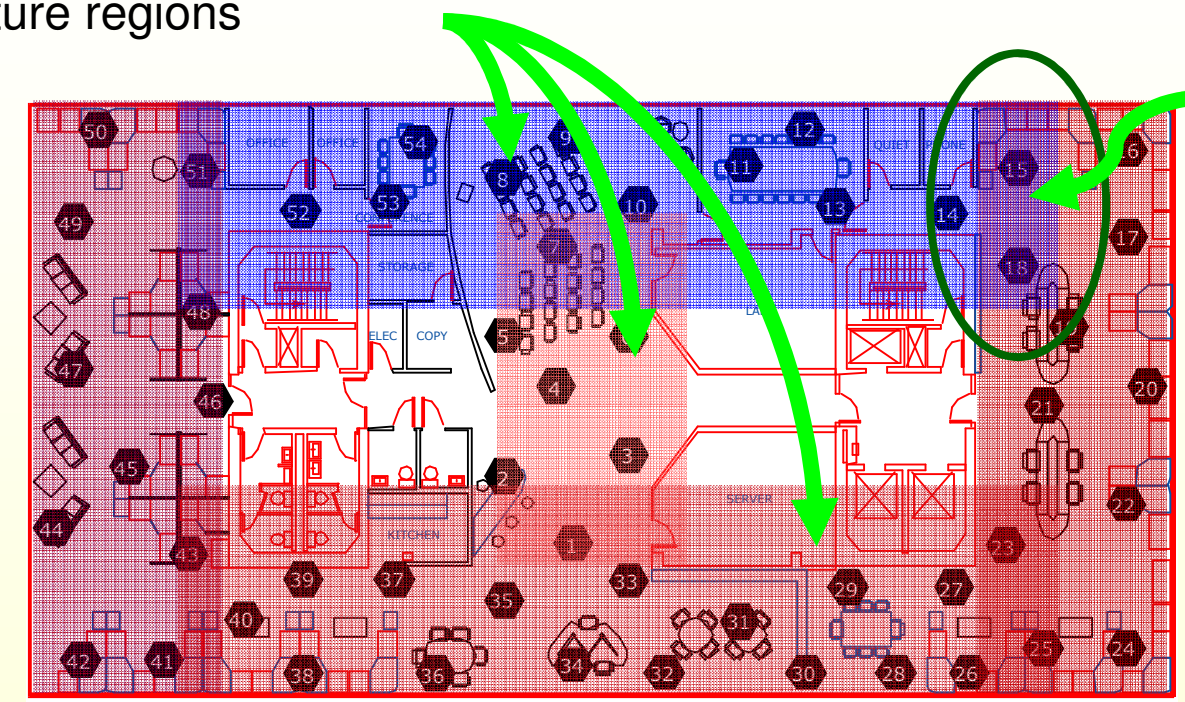
Temperature surface is complex

⇓

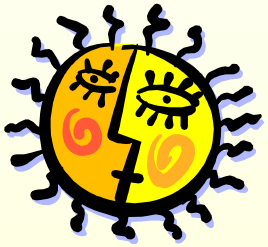
Need complex, globally supported basis functions?
Lots of communication?

What are we Missing?

Local temperature regions



Do the right thing in the overlaps

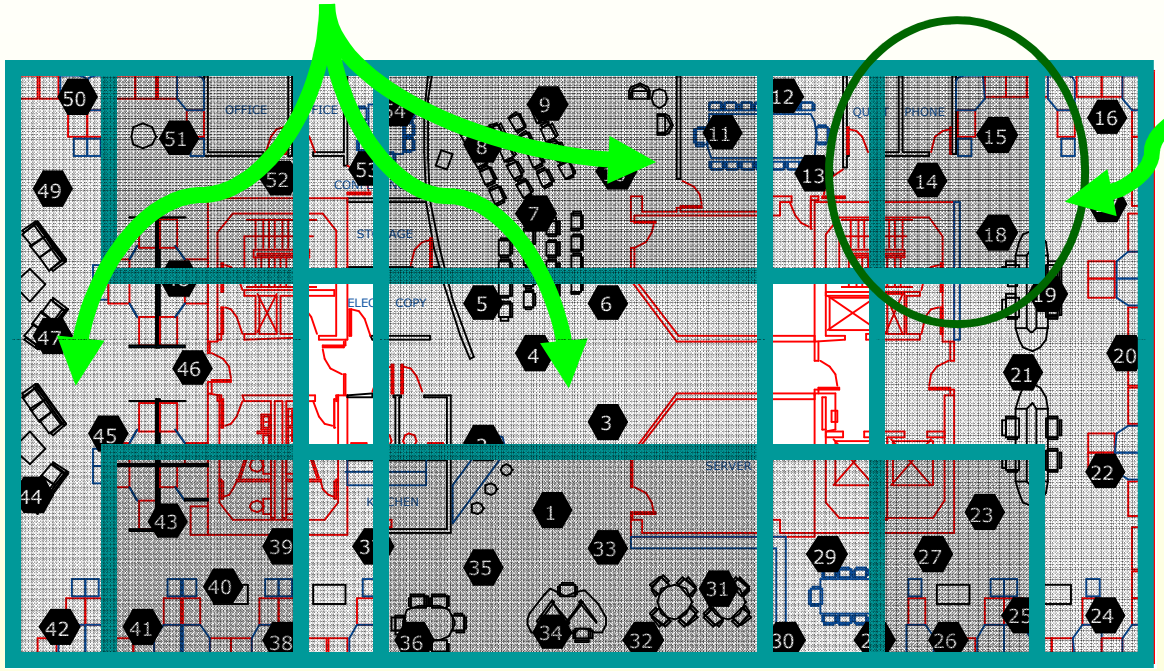


Temperature surface is complex, but there is

lots of local structure!

Kernel Regression

Local basis functions for each region

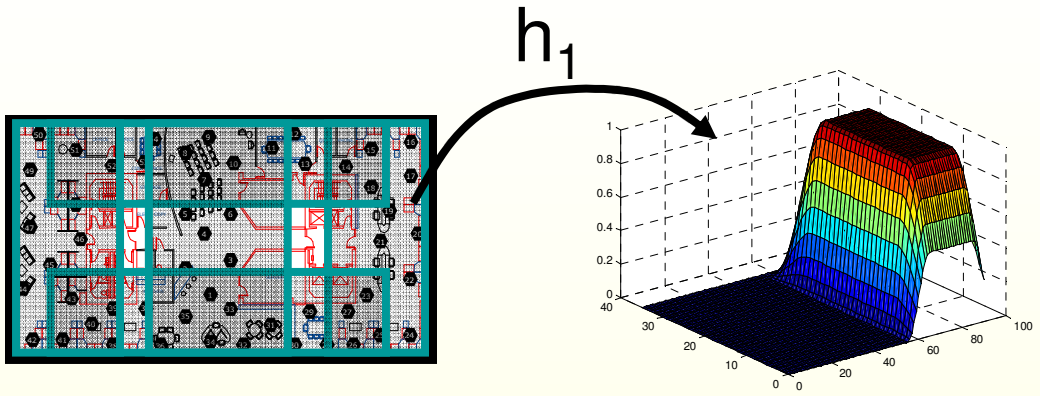


Kernels average between regions

- Distributed algorithm for obtaining coefficients
- Sin g tree
- Ro

Need global optimization to find optimal coefficients

Kernel Regression ⇒ Sparse Matrices



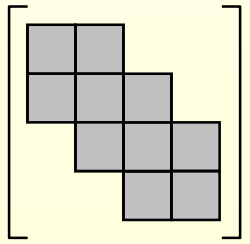
⇒ Sparse basis

Kernel basis functions have local support

$$\mathbf{H} = \begin{bmatrix} & 0 \\ 0 & \end{bmatrix} \} \text{ sensors}$$

basis functions

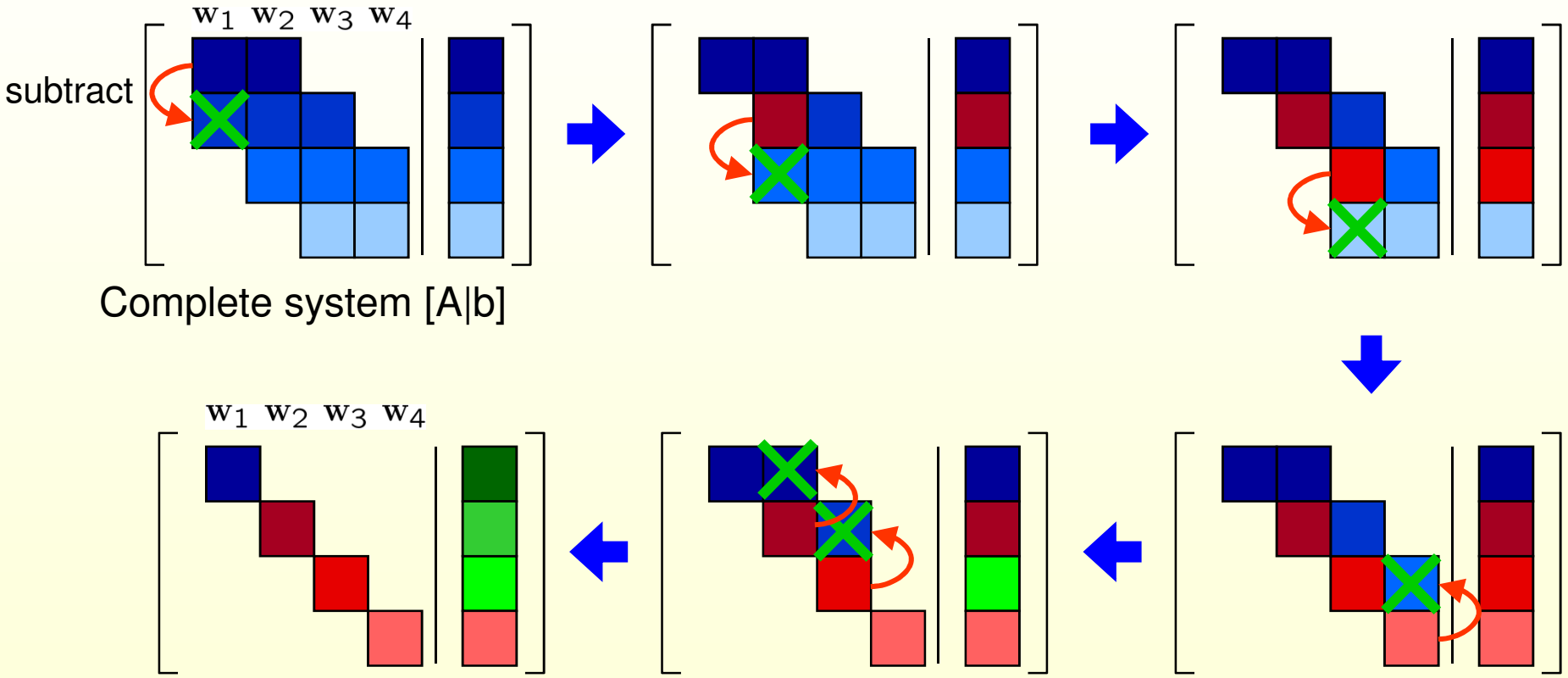
$$\Rightarrow \mathbf{A} = \mathbf{H}^\top \mathbf{H} =$$



(sparse)

Gaussian Elimination

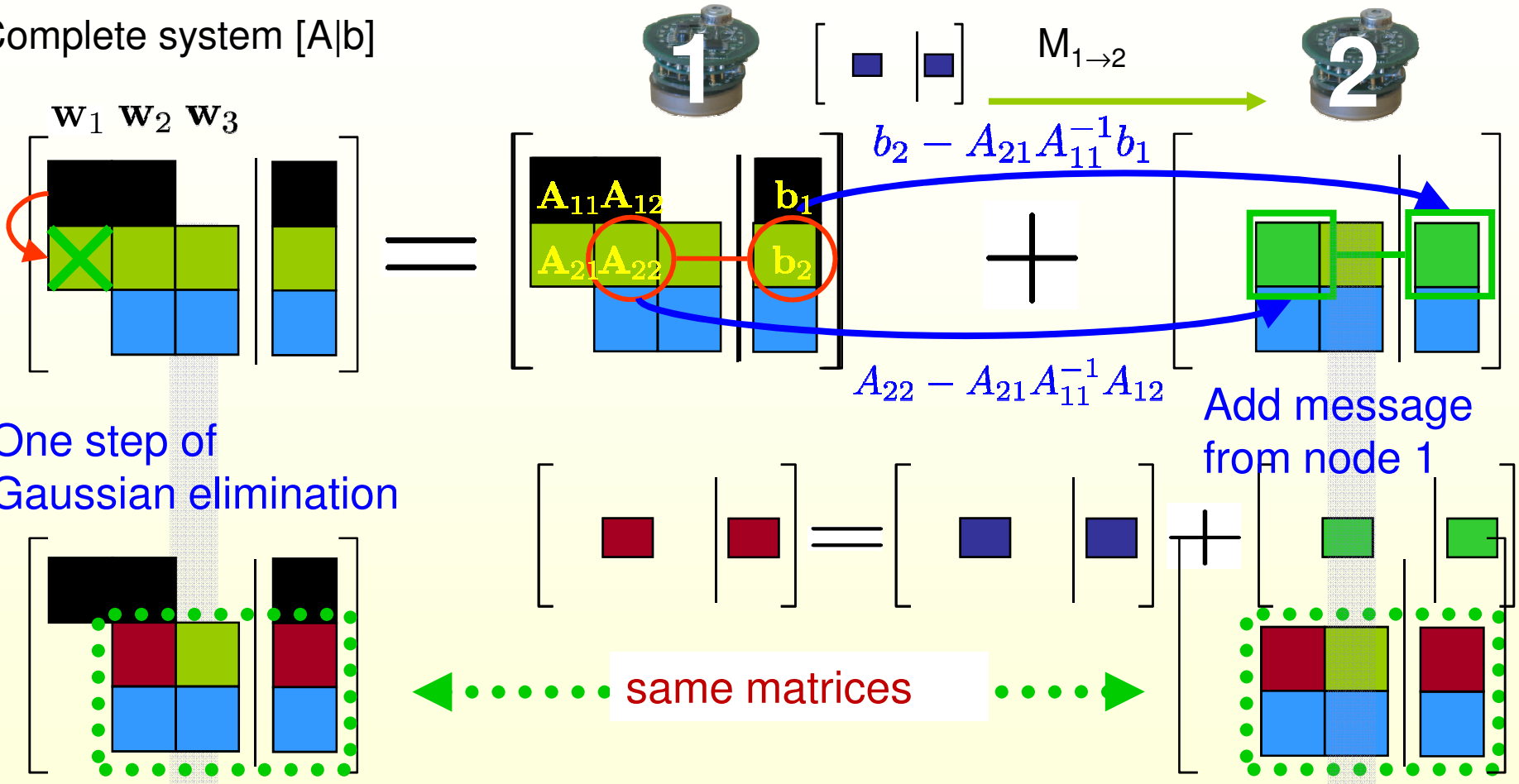
If A is sparse then do efficient Gaussian elimination:



After Gaussian elimination,
solve linear system by k simple divisions

Distributed Regression

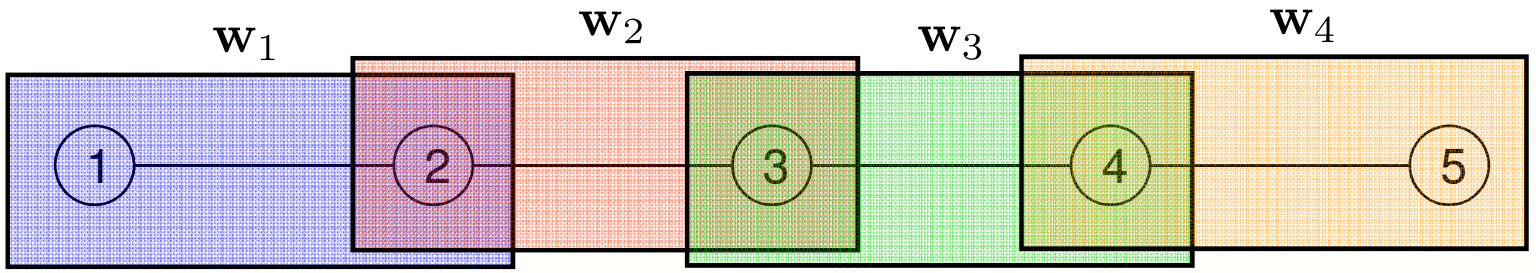
Complete system [A|b]



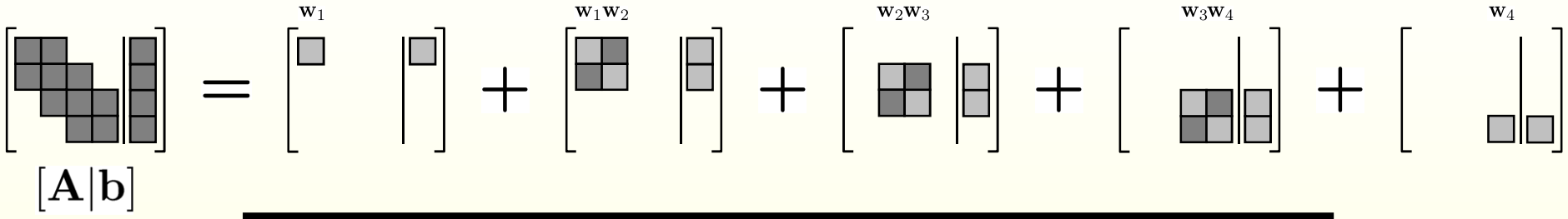
This subsystem is enough to compute w_2, w_3

Sensor 2 can locally compute w_2, w_3

① Specify regions



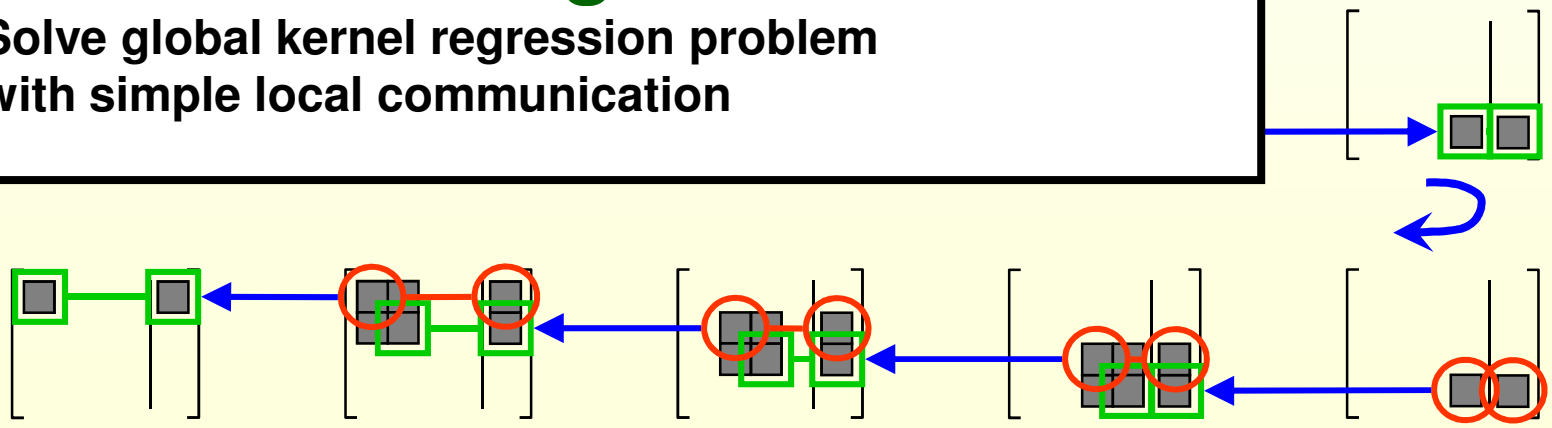
② Sensors compute small matrices that add up to [**A**|**b**]:



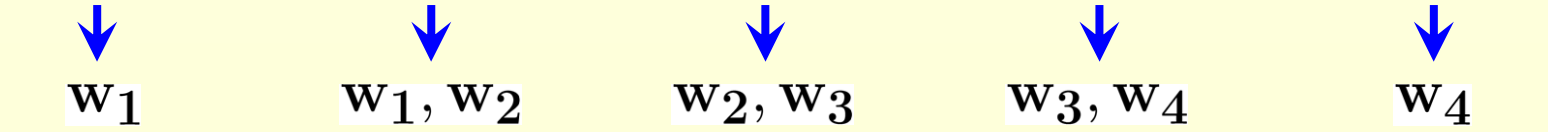
③ Message Passing

Distributed Regression:

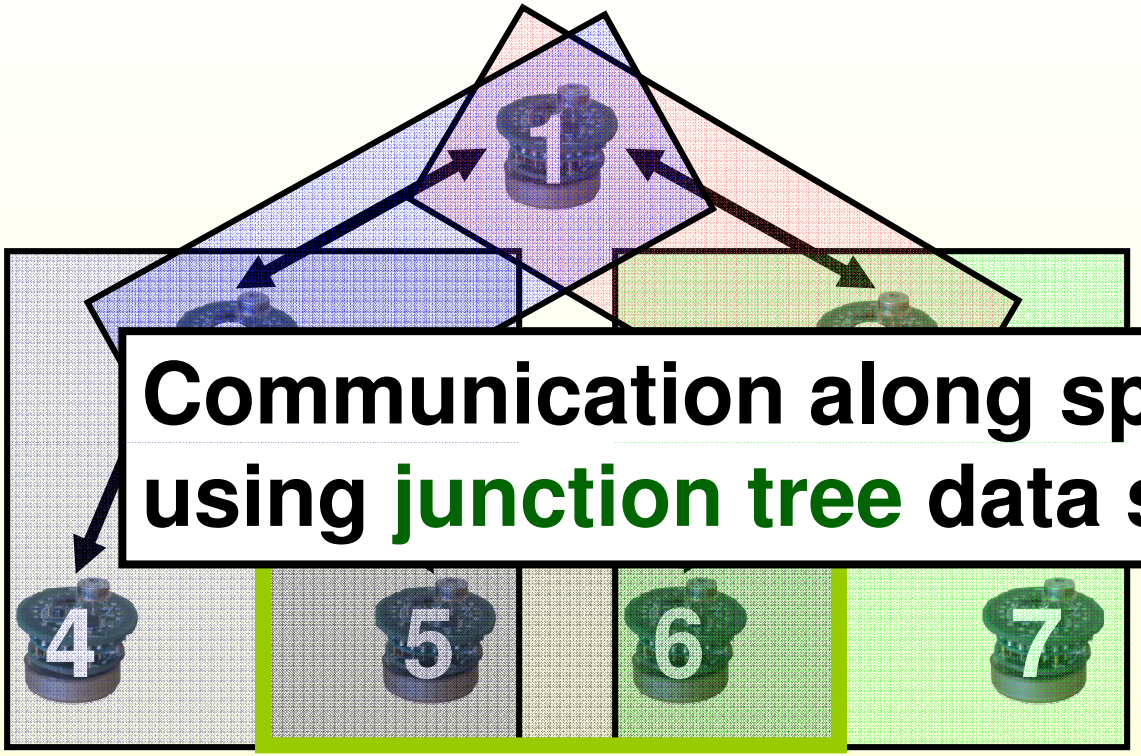
Solve global kernel regression problem with simple local communication



④ Solve local Systems



Communication Pattern

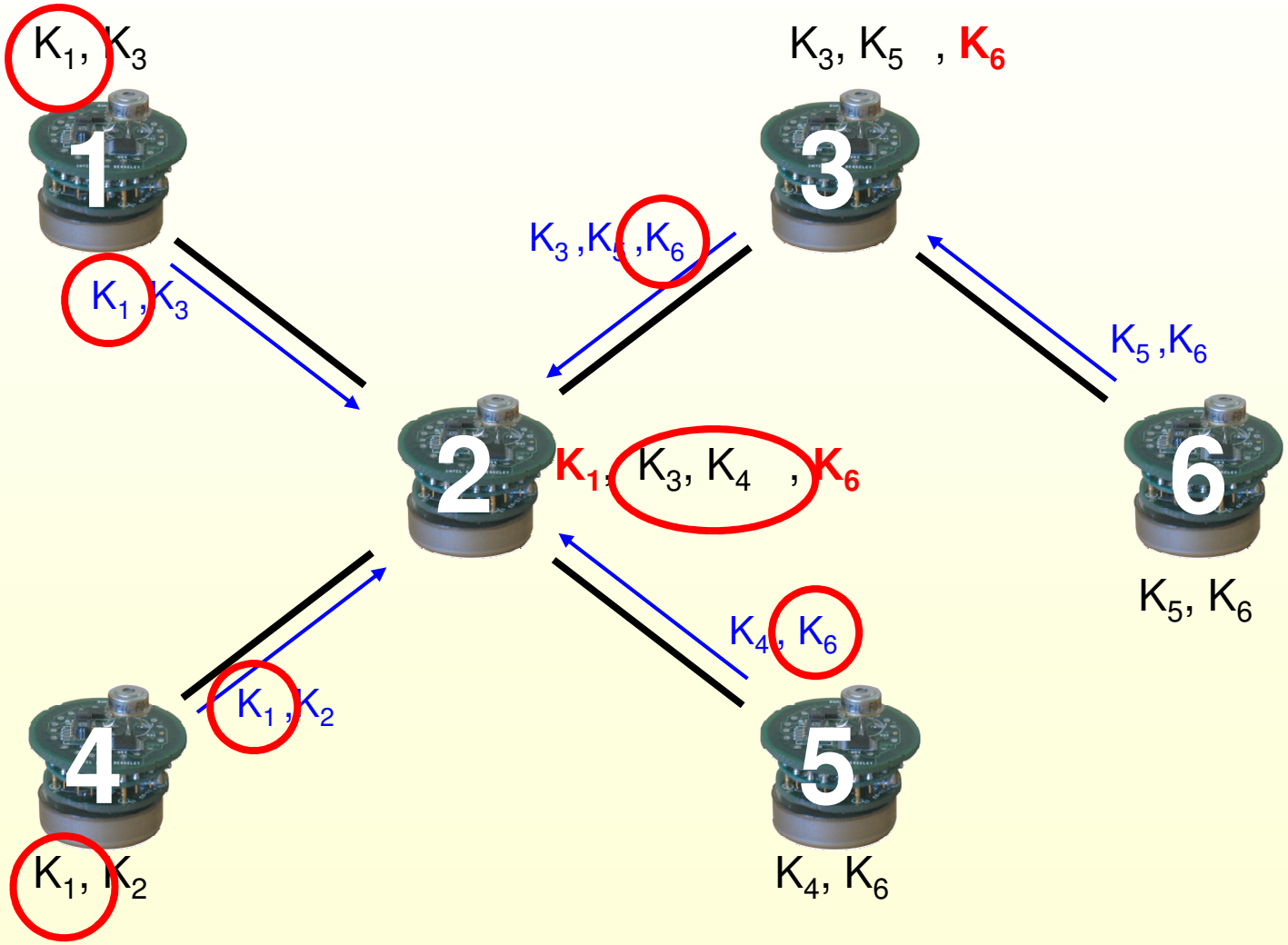


Kernels form
a tree structure
⇓

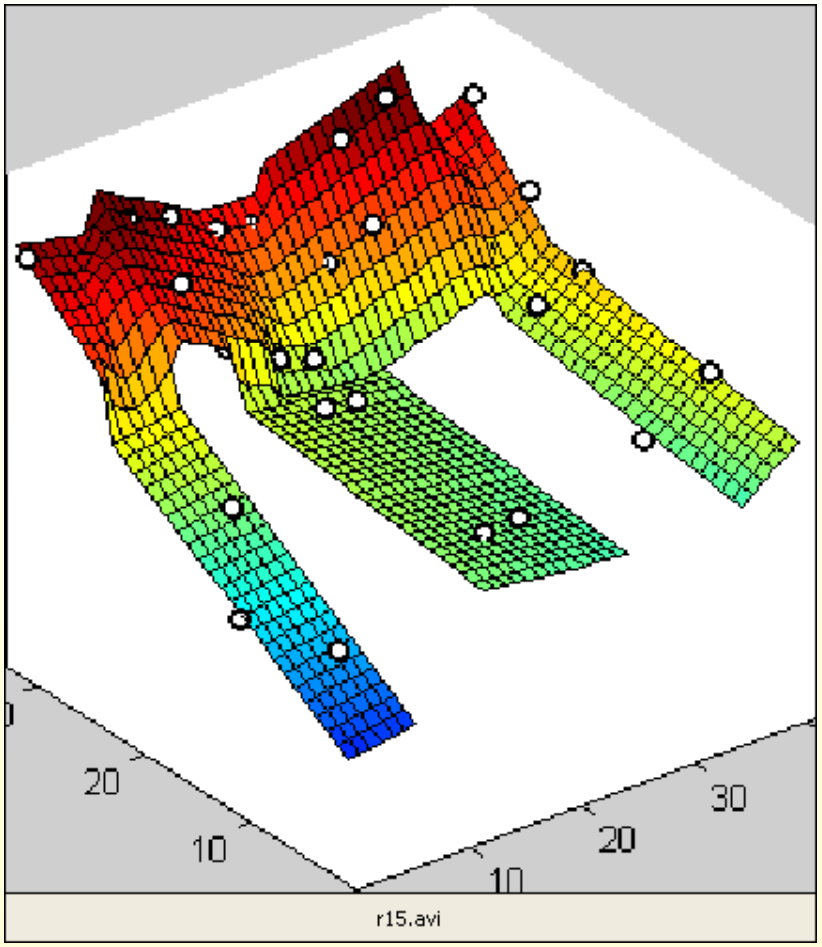
Communication along spanning tree, using junction tree data structure

- High quality links may not align with kernel topology
- Kernels may not form a tree structure

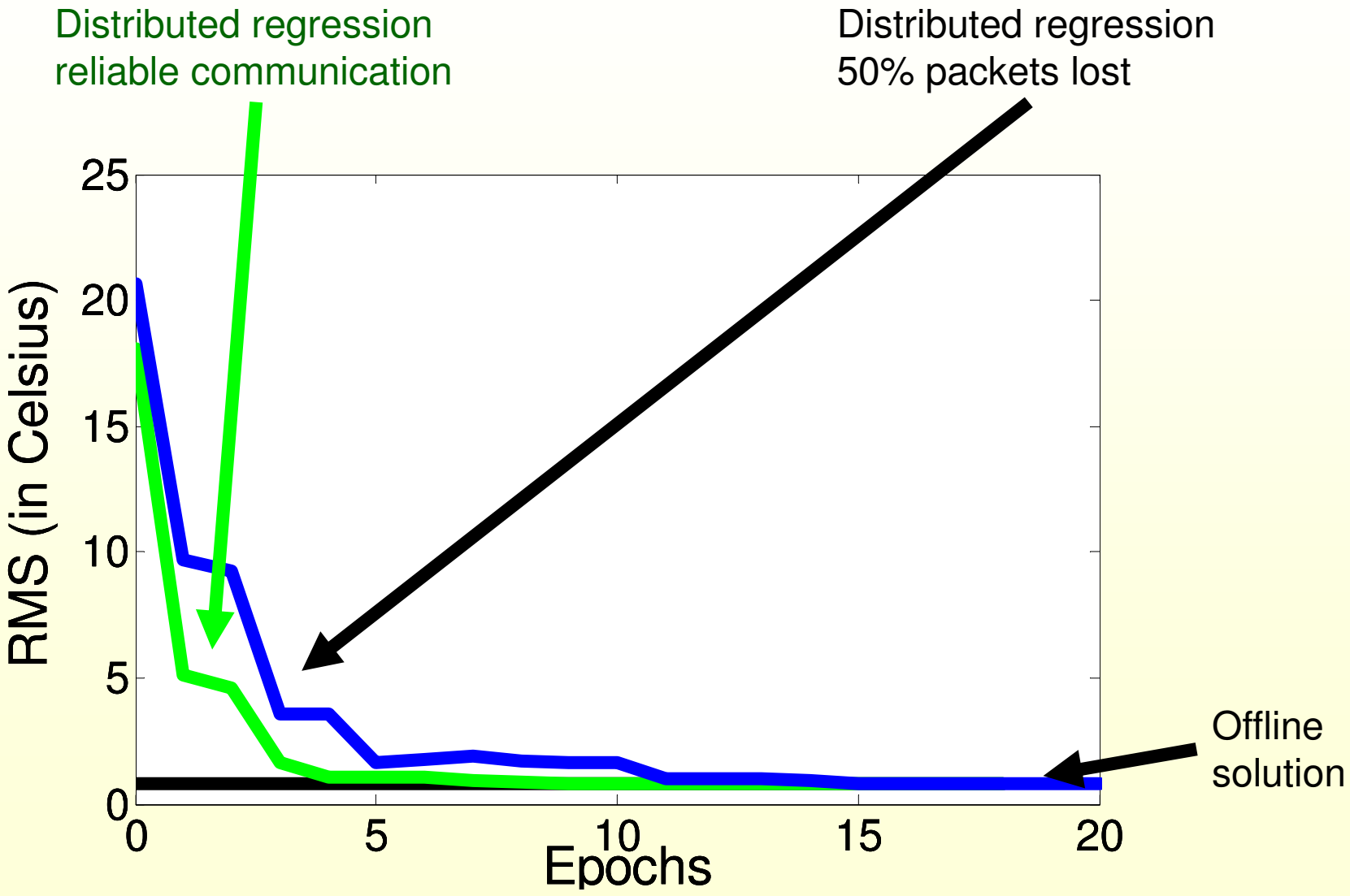
Distributed Junction Trees



Temperature Model for Lab Data

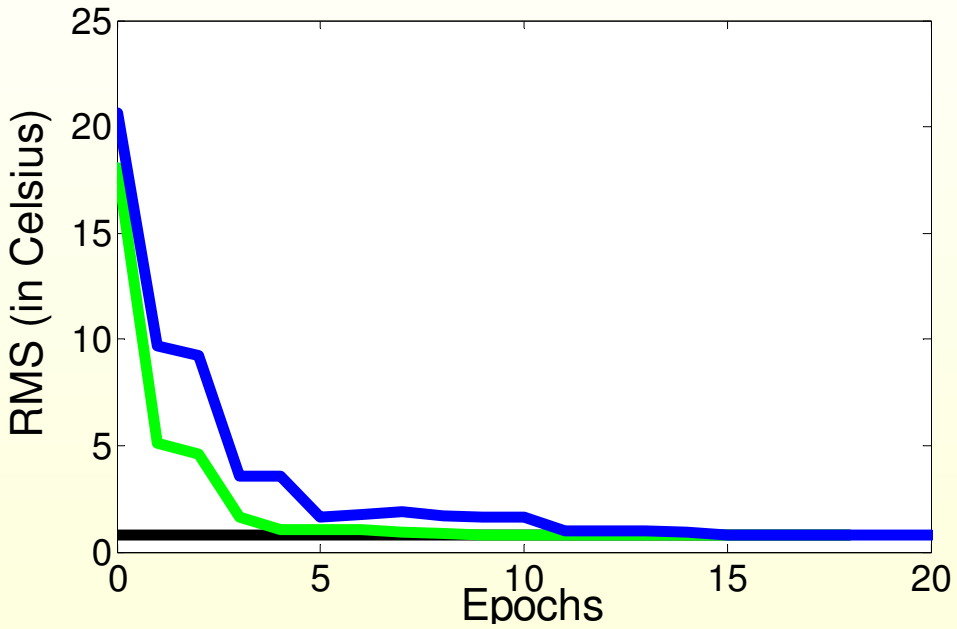


Convergence and Robustness

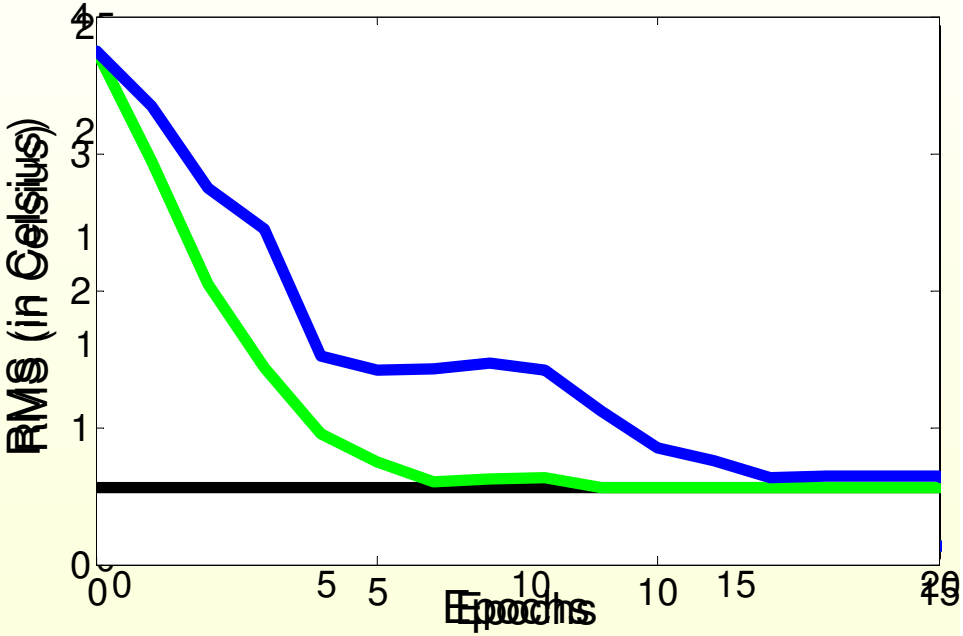


Incremental Changes

Initializing with
noon temperatures



At 6pm, initializing from
noon results



Offline solution

Distributed regression w. reliable communication

Distributed regression w. 50% packets lost

Residual Error Varies over Time

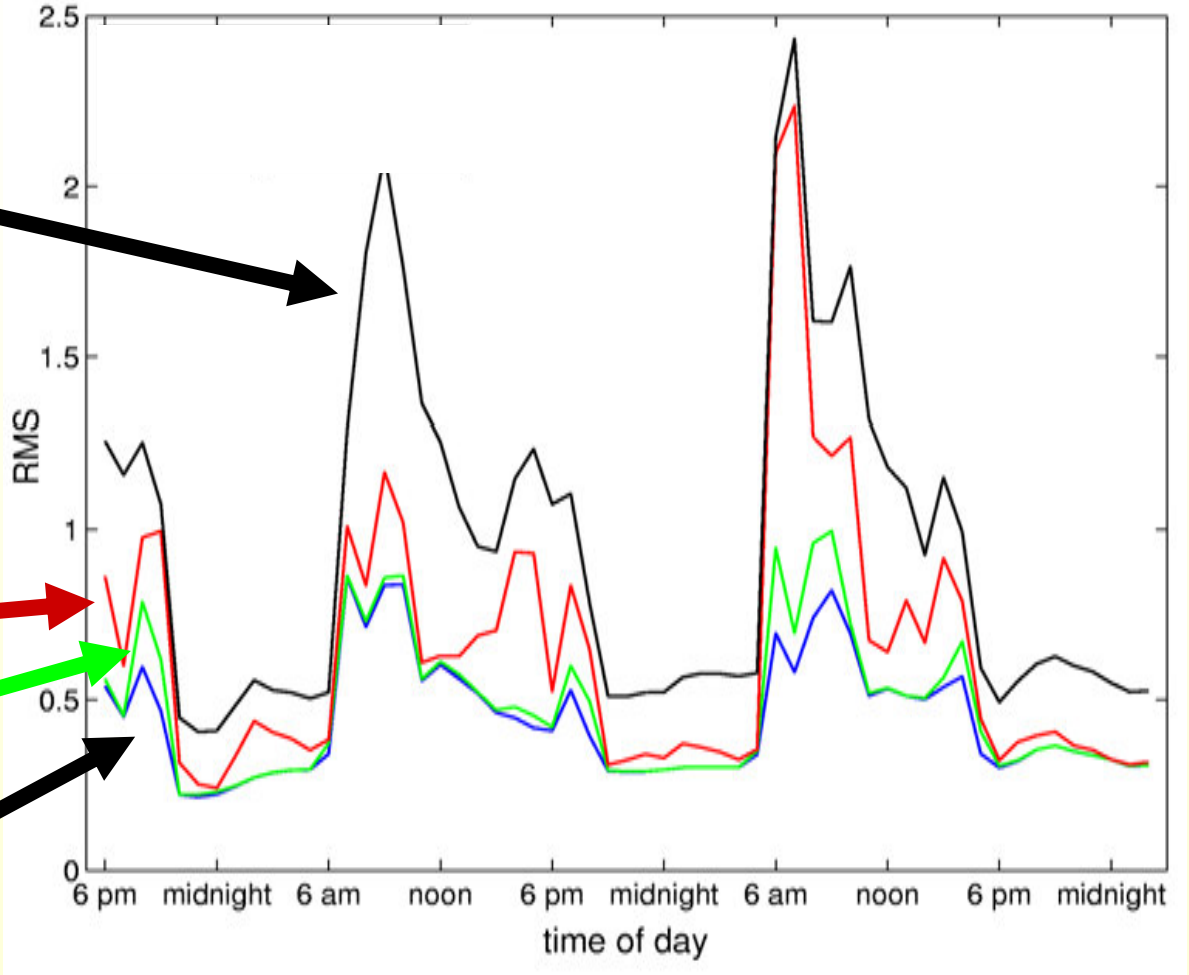
Average over regions

Regression with
linear spatial components:

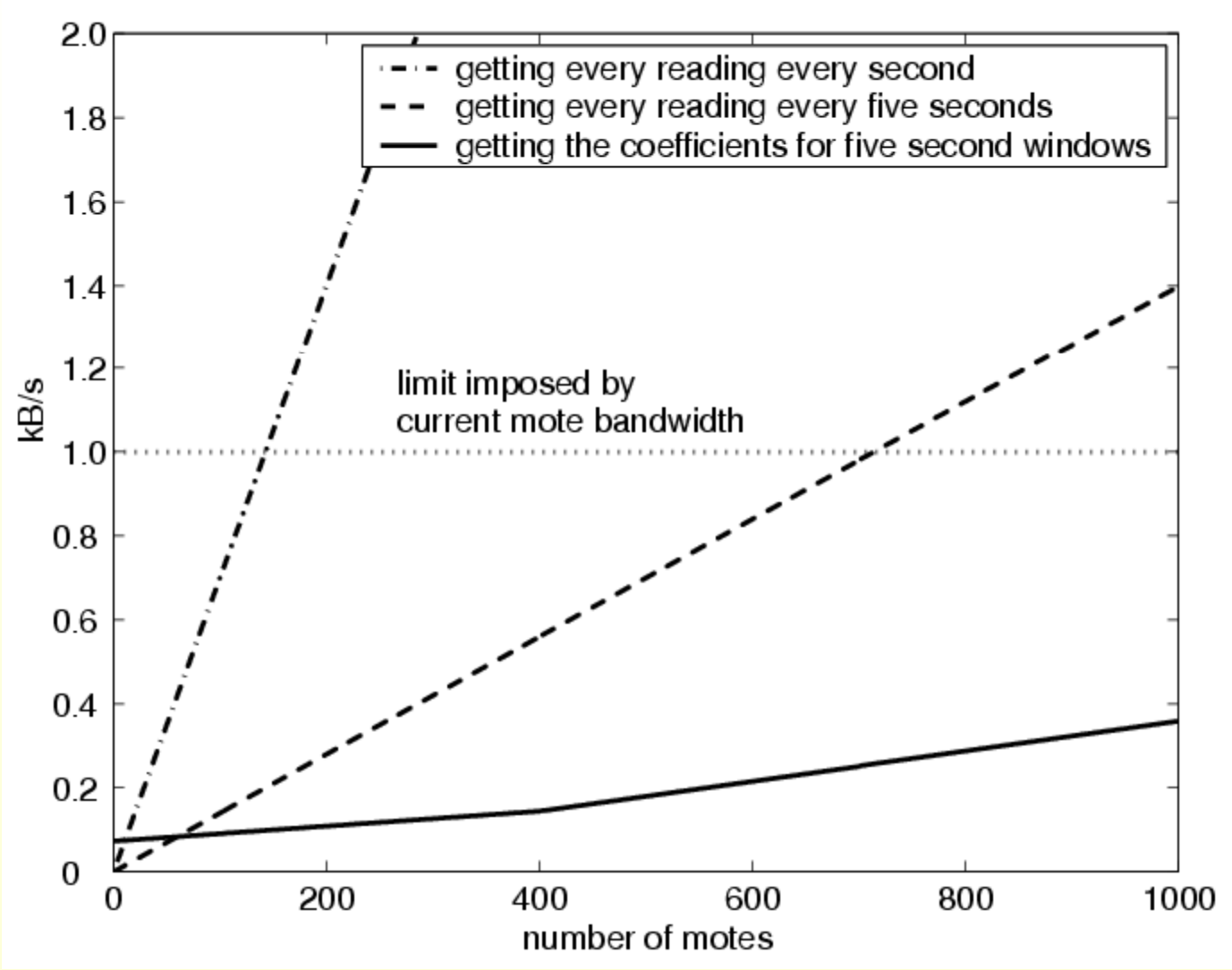
Constant in time

Linear in time

Quadratic in time



Communication Complexity



Extensions and Applications

- Adaptive sampling
- Outlier and faulty sensor detection
- Contour finding
- Adaptive data modeling
  - Basis function selection

Regression Summary

- General distributed regression algorithm for sensor networks
- Robust to node and message losses
- Kernel regression is an effective model for wide range of sensor network data

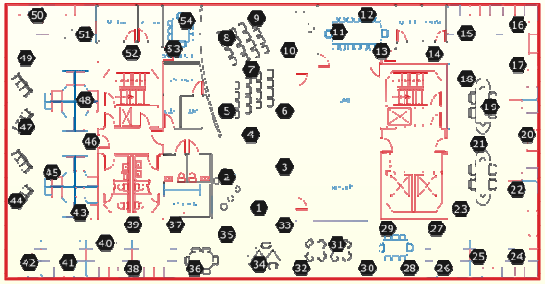
Discussion Questions

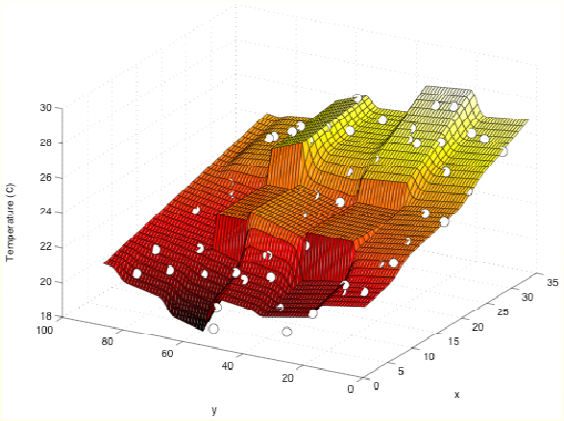
- Why regression?
- Temperature data somewhat easy
  - Taking every eighth datapoint gives same results
  - Other types of data that could be used instead
- High spatial/temporal correlation
  - Why not fewer sensors/readings? Redundancy?
  - Why adaptivity would be useful

A Robust Architecture for Distributed Inference in Sensor Networks

Mark Paskin, Carlos Guestrin and Jim McFadden
IPSN 2005

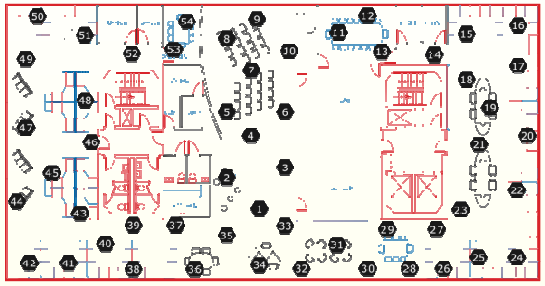
Sensor Network Tasks

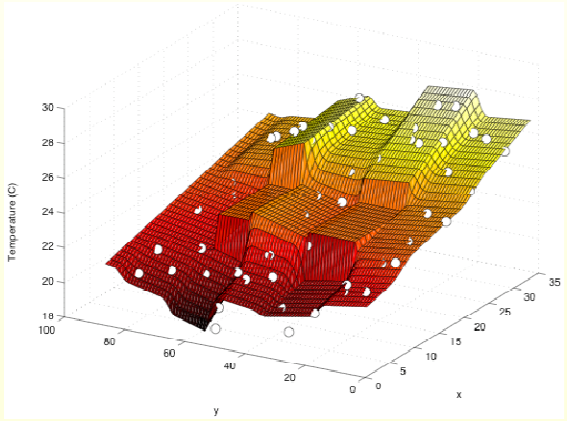


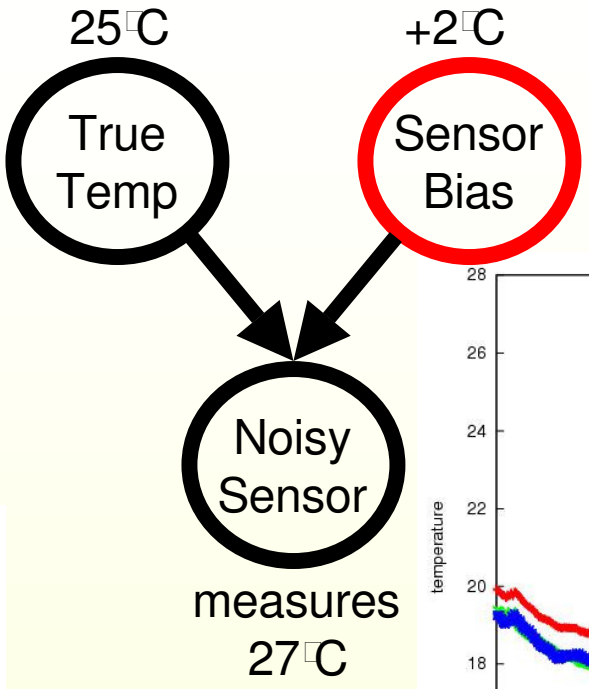


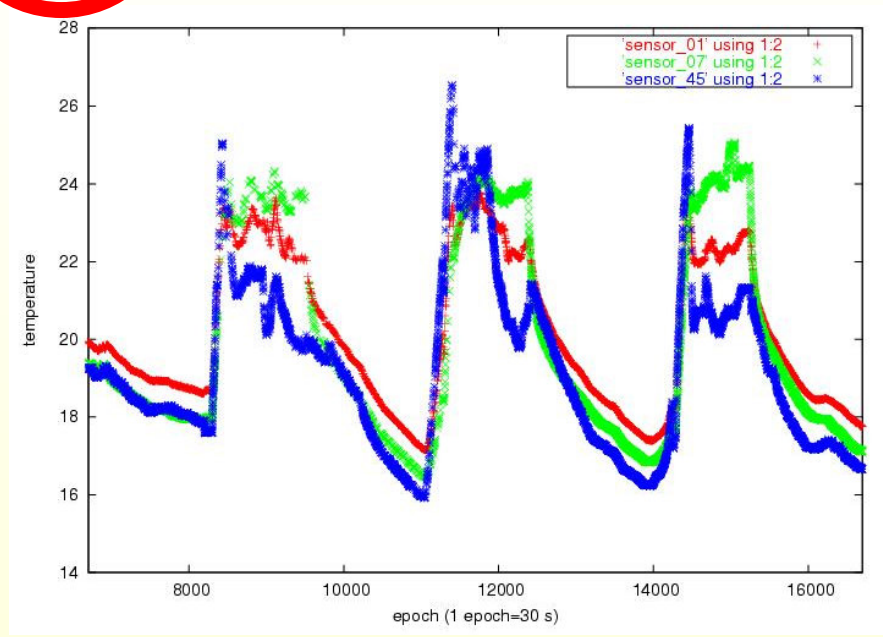
sensor field
monitoring

Sensor Network Tasks







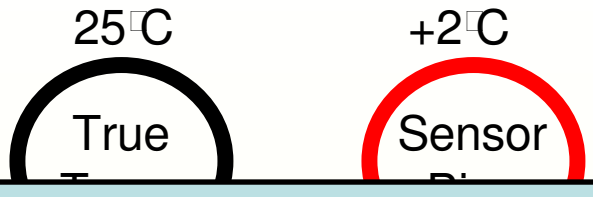


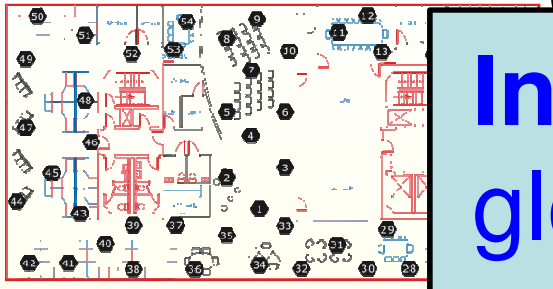
spatial correlation

sensor field monitoring

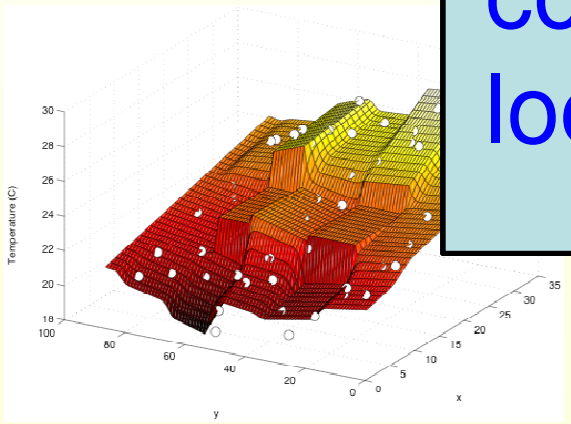
automatic sensor calibration

Sensor Network Tasks





Inference:compute global conclusions from local information



spatial correlation

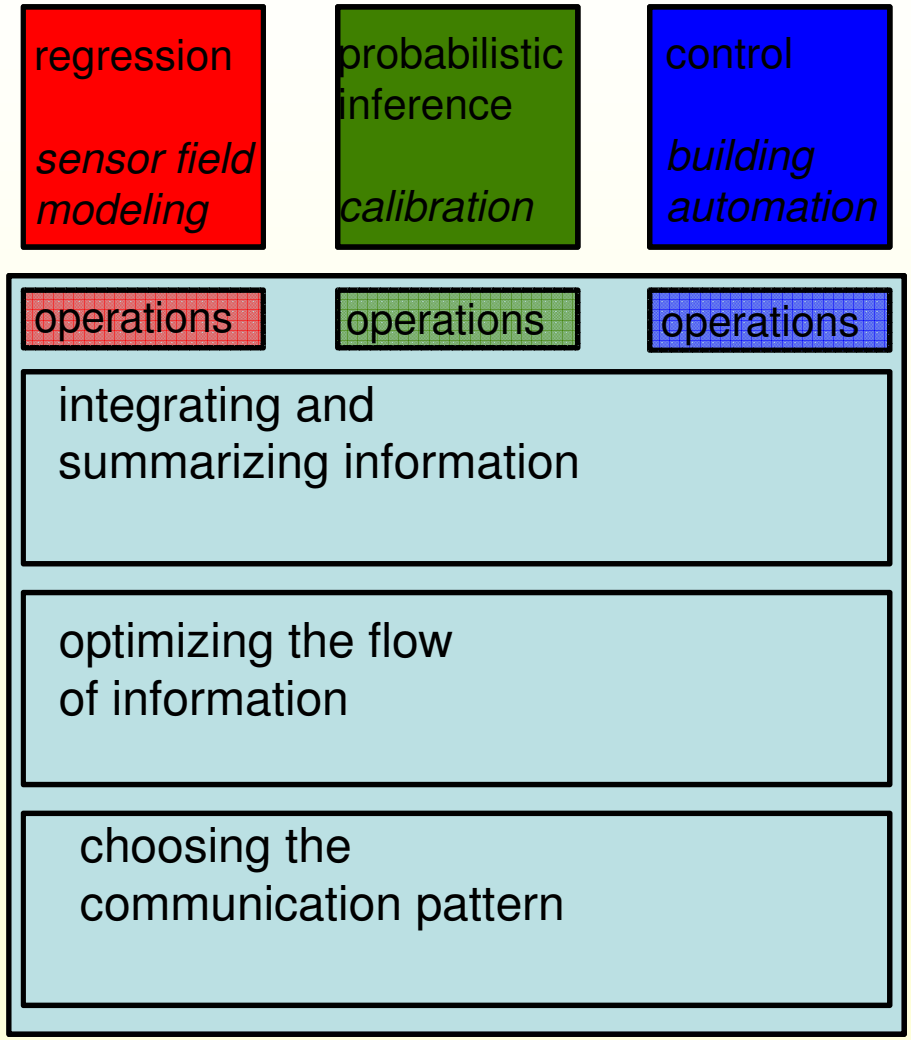
sensor field monitoring	automatic sensor calibration	building automation

Inference Motivation

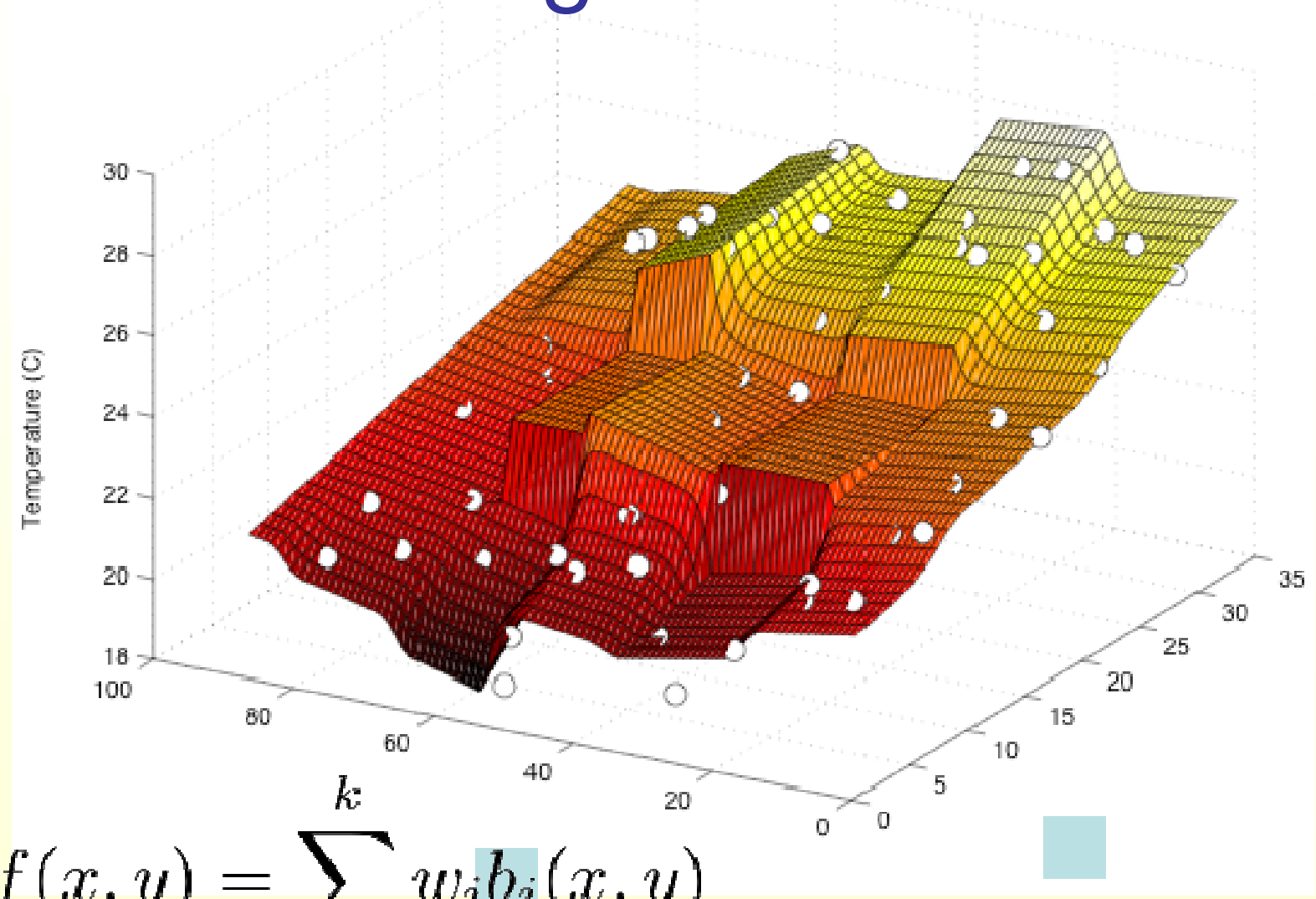
- Inference tasks in sensor networks are challenging because they require:
  - **combining information** that is distributed across heterogeneous units
  - **robustness** to communication and node failures
  - **resource awareness** to cope with limited computation and communication
- Solving a sensor network task requires a substantial effort and resources
- Our goal: address these issues *once*; then solve many sensor network tasks easily

The Contribution

- An architecture for inference in sensor networks that is:
  - **general**
  - **efficient**
  - **robust** to communication and node failures



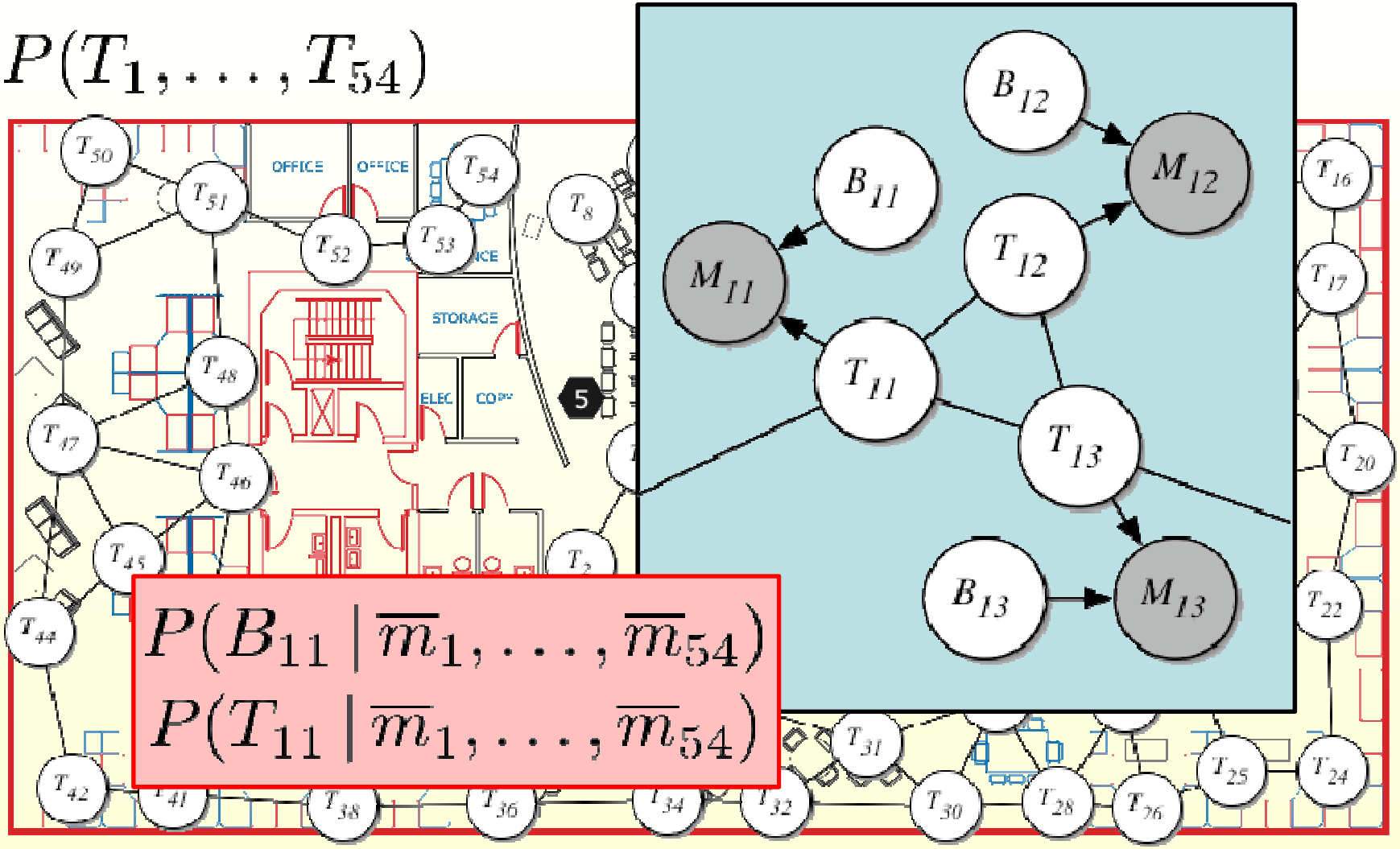
Regression



$$f(x,y) = \sum_{i=1}^{k} w_i b_i(x,y)$$

Probabilistic Inference

$$P(T_1, \ldots, T_{54})$$



$$P(B_{11} \mid \overline{m}_1, \ldots, \overline{m}_{54})$$

$$P(T_{11} \mid \overline{m}_1, \ldots, \overline{m}_{54})$$

Control

- Sensors that control their environment must balance conflicting goals
- Each sensor has a reward function:
- Goal: maximize sum of rewards:

$$Q_i(\mathbf{a}_i, \mathbf{x})$$

$$\mathbf{a}^* = \arg\max_{\mathbf{a}} \sum_i Q_i(\mathbf{a}_i, \mathbf{x})$$

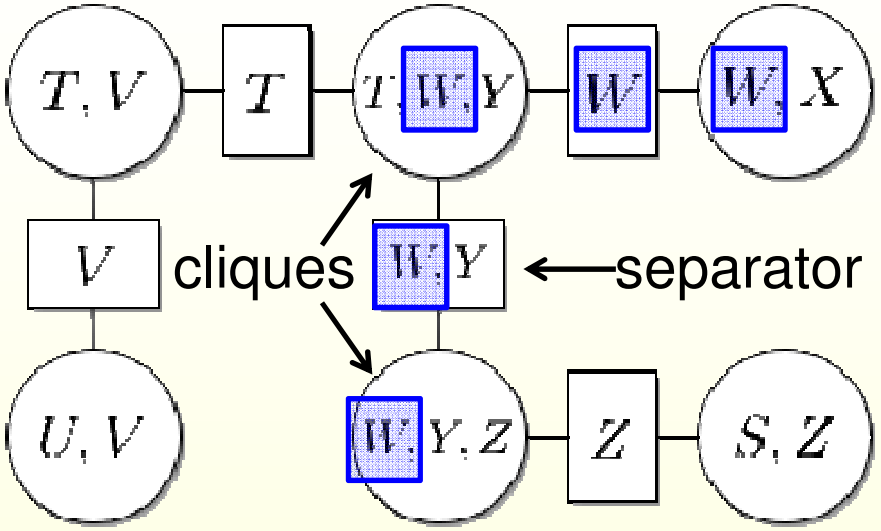
Inference: from Local Information to Global Conclusions

- These problems have a common structure
  - variables, the unknown quantities
  - factors, which represent information about subsets of variables
  - operations that combine factors and summarize factors

	variables	*factors*
linear regression	optimal weights	matrix/vector pair
probabilistic inference	random variables	potential functions
optimal control	action variables	reward functions

Inference by Message Passing

Problems with this structure can be solved by message passing on a junction tree: a tree of sets of variables



with the *running intersection property* *the cliques containing any variable form a subtree*

By passing a single message (factor over separator) between each pair of nodes, every node obtains the solution for its set of variables.

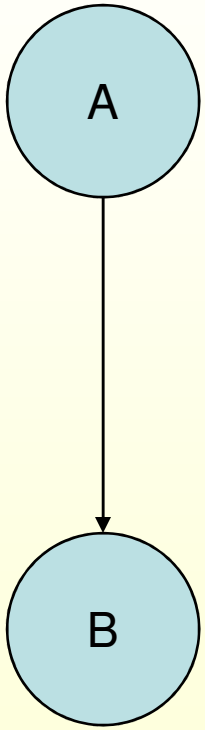
What are Junction Trees?

Junction Trees

- Message-passing on a junction tree to solve
  - Regression
  - Optimal control
  - Probabilistic inference

Graphical Models

Compact graphical representation of joint probability distributions



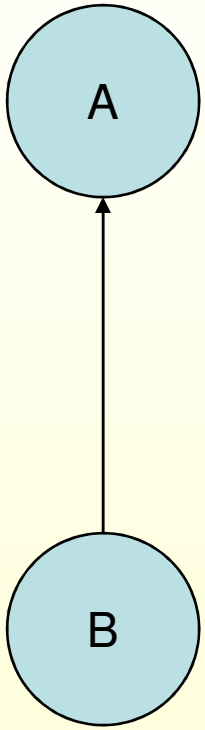
A ‘causes’ B

Bayes Net

$$P(A,B) = P(A)P(B|A)$$

Graphical Models

Compact graphical representation of joint probability distributions

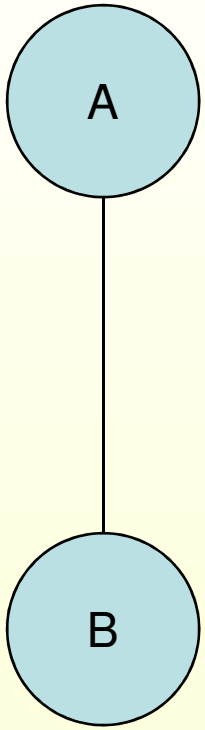


B ‘causes’ A

$$P(A,B) = P(B)P(A|B)$$

Graphical Models

Compact graphical representation of joint probability



P(A,B)

A Simple Example

$$P(A,B,C) = P(A)P(B,C \mid A)$$

$$= P(A)\ \ P(B|A)\ \ P(C|B,A)$$

$$= P(A)\ \ P(B|A)\ \ P(C|B)$$

C is conditionally independent of A given B

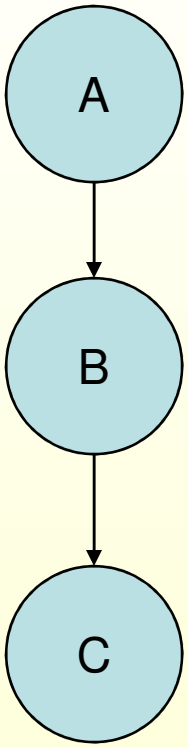
Graphical Representation ???

Bayesian Network

Directed Graphical Model

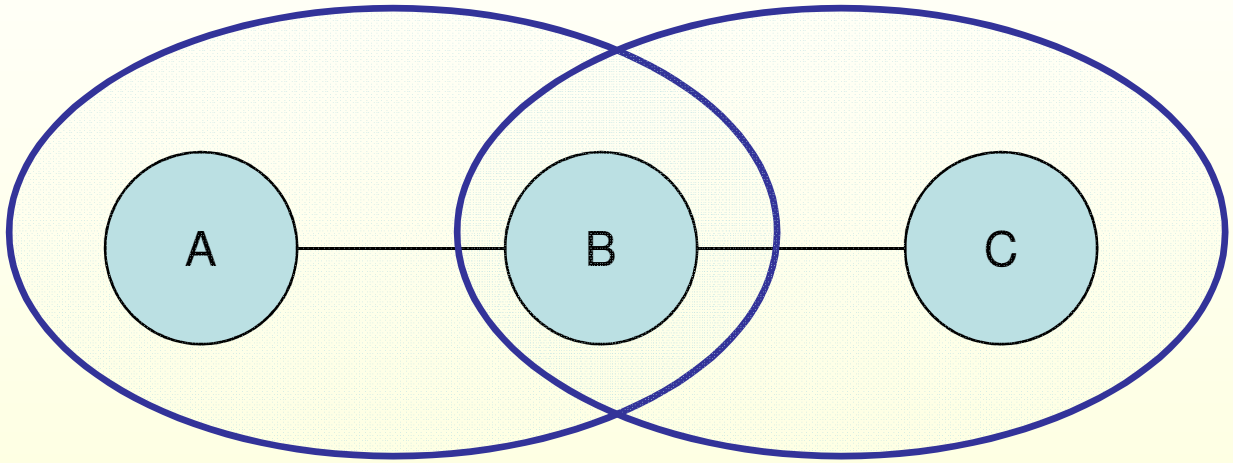
$$P(U) = \prod P(V_i \mid Pa(V_i))$$

$$P(A,B,C) = P(A)\, P(B \mid A)\, P(C \mid B)$$



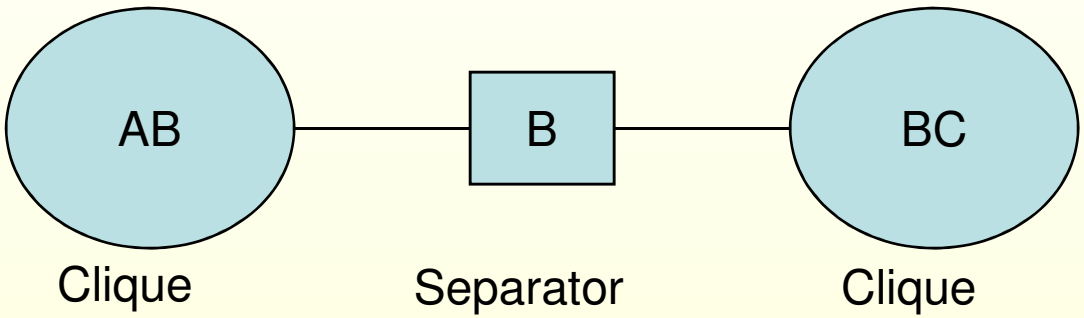
Markov Random Fields

Undirected Graphical Model



Markov Random Fields

Undirected Graphical Model



$$P(U) = \prod P(\text{Clique}) / \prod P(\text{Separator})$$

$$P(A,B,C) = P(A,B)\, P(B,C) / P(B)$$

Notation

p = probabilistic inference
r = regression

- Variables
  - Unobservable random variables (p)
  - Weights (r)

$$\Pr\{t, b \mid m\} = \Pr\{t\}\Pr\{b\}\{m \mid t, b\}$$

- Factors
  - Terms in posterior joint prob. distribution function (p)
  - Matrix $A = A^{(1)} + \ldots + A^{(n)}$ (r)
  - Vector $b = b^{(1)} + \ldots + b^{(n)}$ (r)

Notation

p = probabilistic inference
r = regression

- Combining factors
  - Multiplication of factors (p)
  - Adding matrices or vectors (r)
- Summarizing factors
  - Marginalization (p)

$$\Pr\{t \mid m\} = \sum_b \Pr\{t, b \mid m\}$$

  - Gaussian elimination (r)

Notation

p = probabilistic inference
r = regression

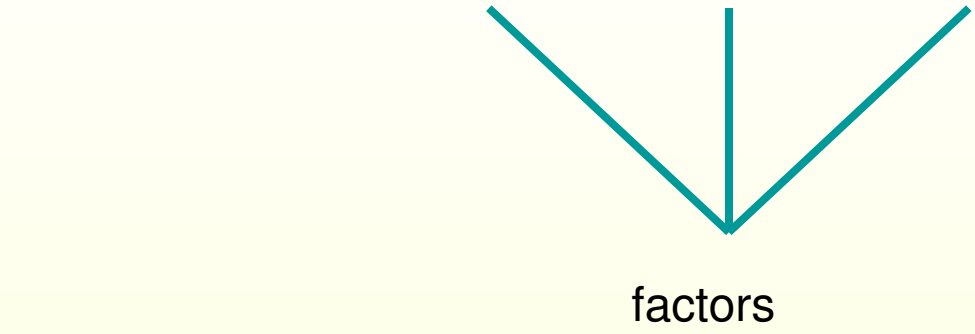
- Inference
  - Posterior marginal for variables given evidence (p)
  - Optimal weights (r)
- Brute-force inference
  - Idea: Combine then summarize
  - Compute joint, marginalize out unneeded variables (p)
  - Solve Aw=b in a centralized way (r)
- Junction tree inference
  - Idea: Summarize then combine
  - Marginalize as much as possible before combining (p)
  - Solve Aw=b in a distributed way (r)

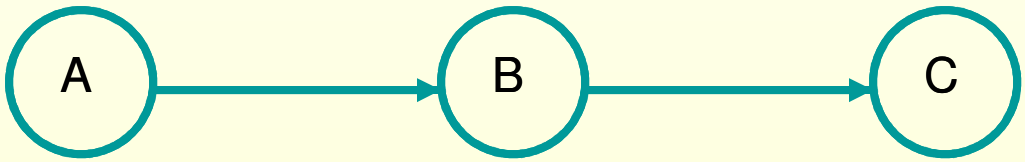
Junction Tree

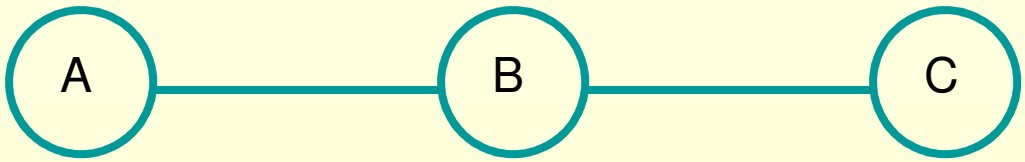
- Given variables + factors, construct junction tree
- Method of construction depends if (p) or (r)
- Junction tree
  - An undirected tree
  - Each node *i* has
    - Factor ψ_i with domain D_i
    - Set of variables C_i such that C_i contains D_i
  - Running intersection property holds
    - C_k for any node *k* on path between nodes *i* and *j* contains C_i intersect C_j
    - This ensures that message passing for local consistency gives global consistency

Probabilistic Inference

- P(A,B,C) = Pr(A) Pr(B | A) Pr(C | B)

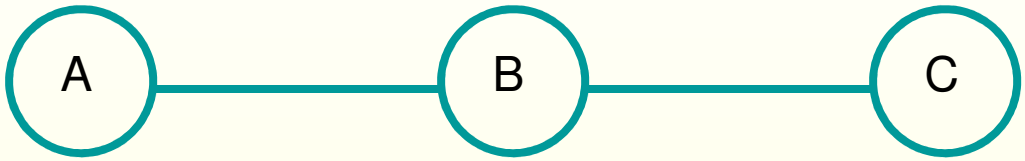






Probabilistic Inference

- P(A,B,C) = Pr(A) Pr(B | A) Pr(C | B)



Cliques: AB, BC

Junction Tree

	Factor	Want
AB :	Pr(A) Pr(B|A)	Pr(A,B)
BC :	Pr(C|B)	Pr(B,C)

Message-Passing

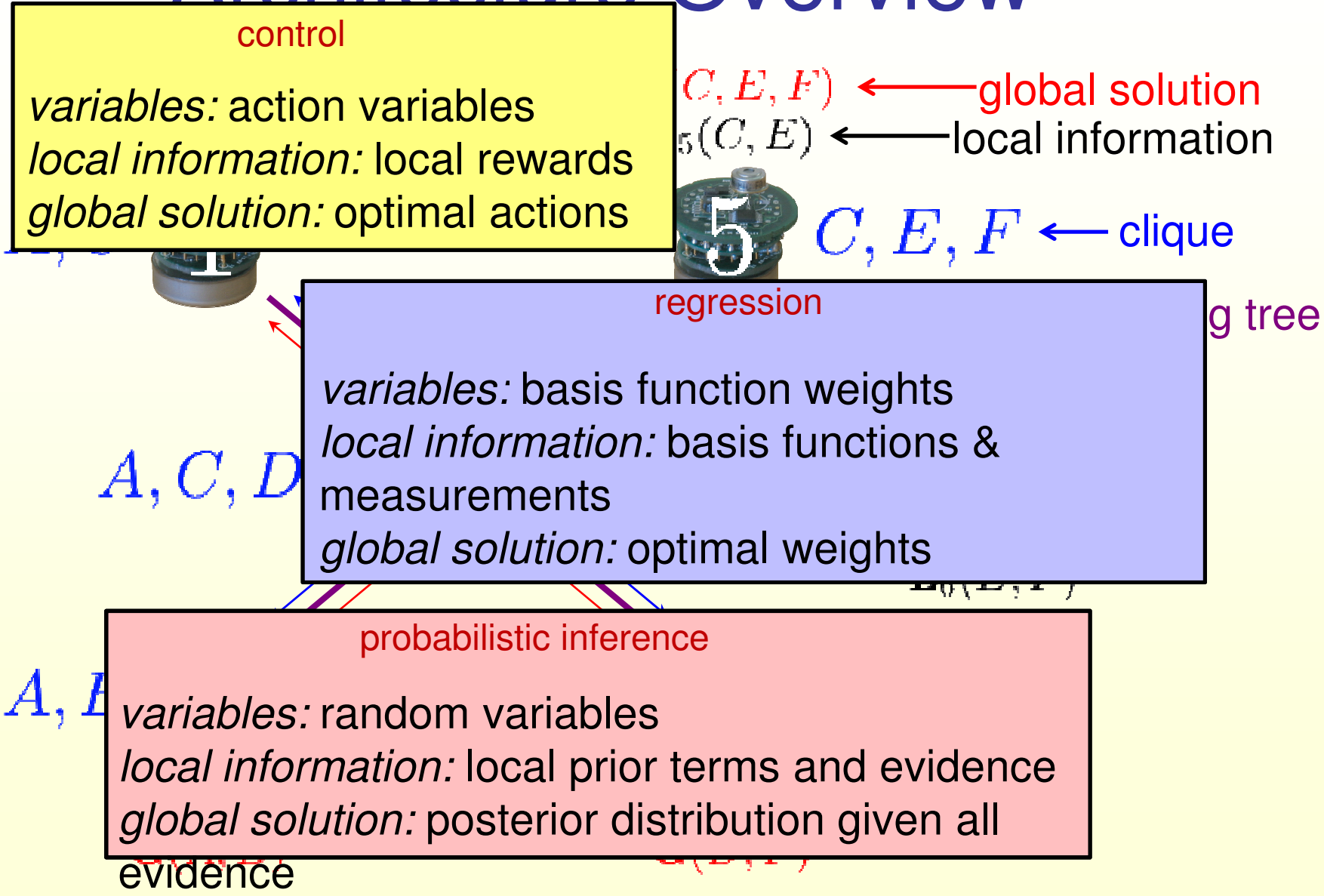
- Message from node *j* to node *k*
  - $\mu_{jk} = \Sigma_{s_{jk}} \psi_j \prod_{i \text{ in } n(j) \setminus k} \mu_{ij}$
  - Product of all messages received at node j excluding any message from node k, times the factor for node j and summarizing out any variables not in both node j and node k

Probabilistic Inference

- Messsage passing from AB to BC
  - $\mu_{AB,BC} = \Sigma_A \ \psi_{AB} = \Sigma_A$ P(A,B) $=$ P(B)
- Message passing from BC to AB
  - $\mu_{BC,AB} = \Sigma_C \ \psi_{BC} = \Sigma_C$ P(C|B) $= 1$
- New factors
  - AB : P(A,B)
  - BC : P(B) P(C|B) = P(B,C)

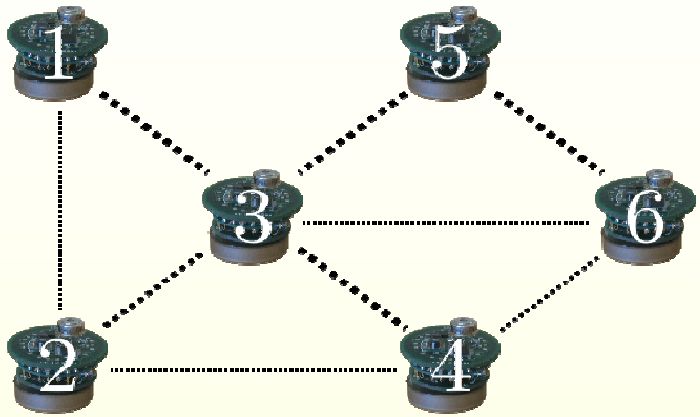
End of Detour

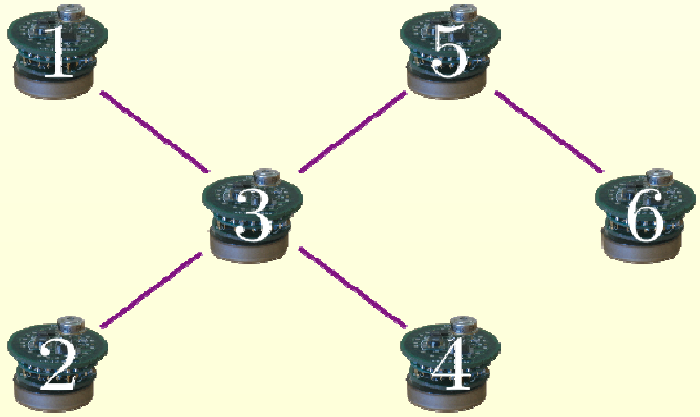
Architecture Overview



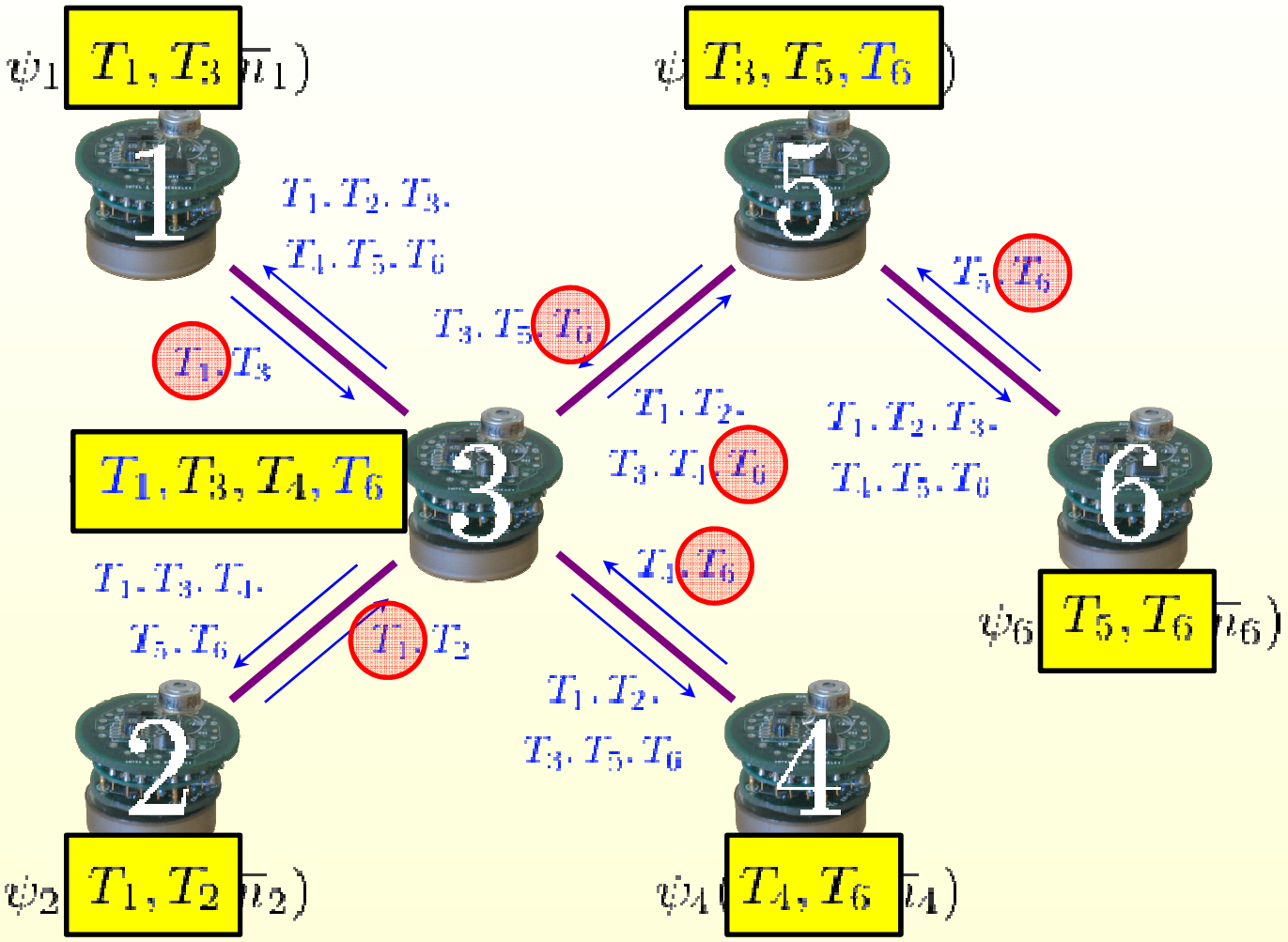
Spanning Tree Formation

- Each node must *choose neighbors* so that the connectivity graph is *undirected*, *acyclic*, and *connected*.
- Nodes observe only local information
- Our spanning trees must be
  - **stable**, so the inference algorithm can make progress
  - **flexible**, to optimize the communication pattern



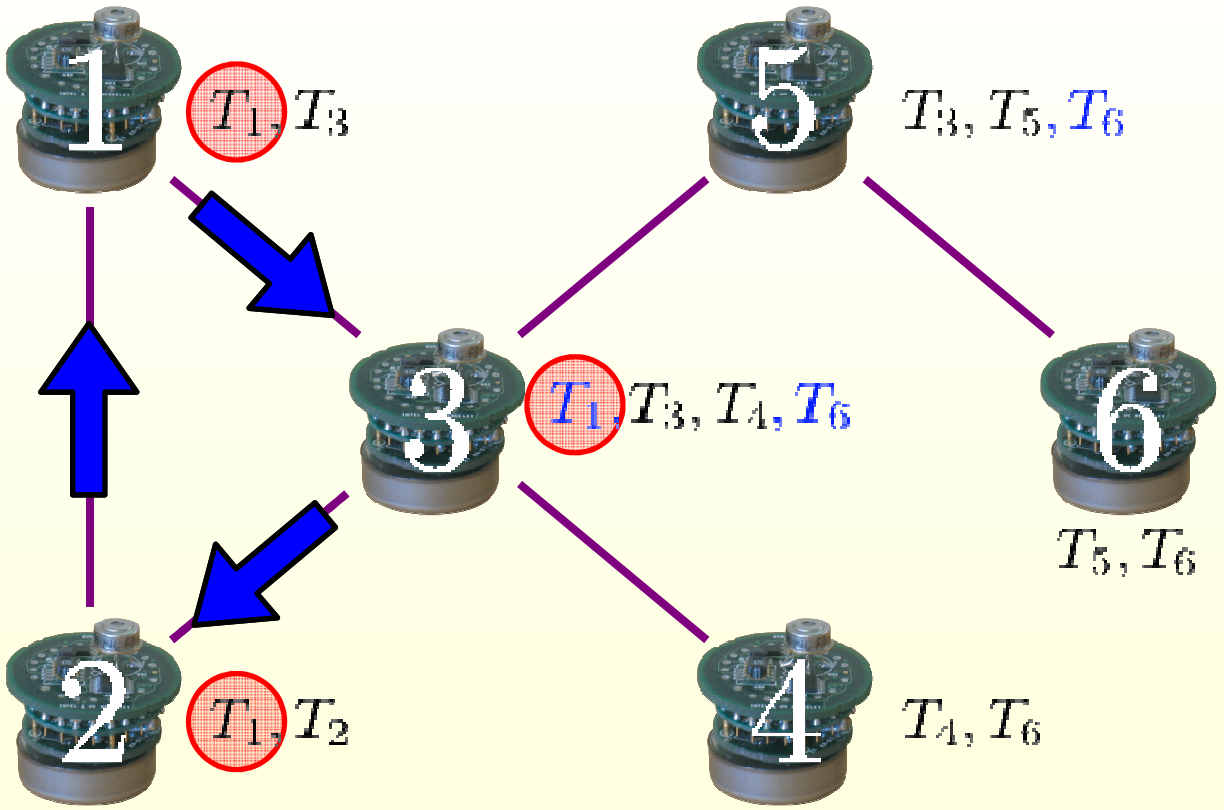


Junction Tree Formation



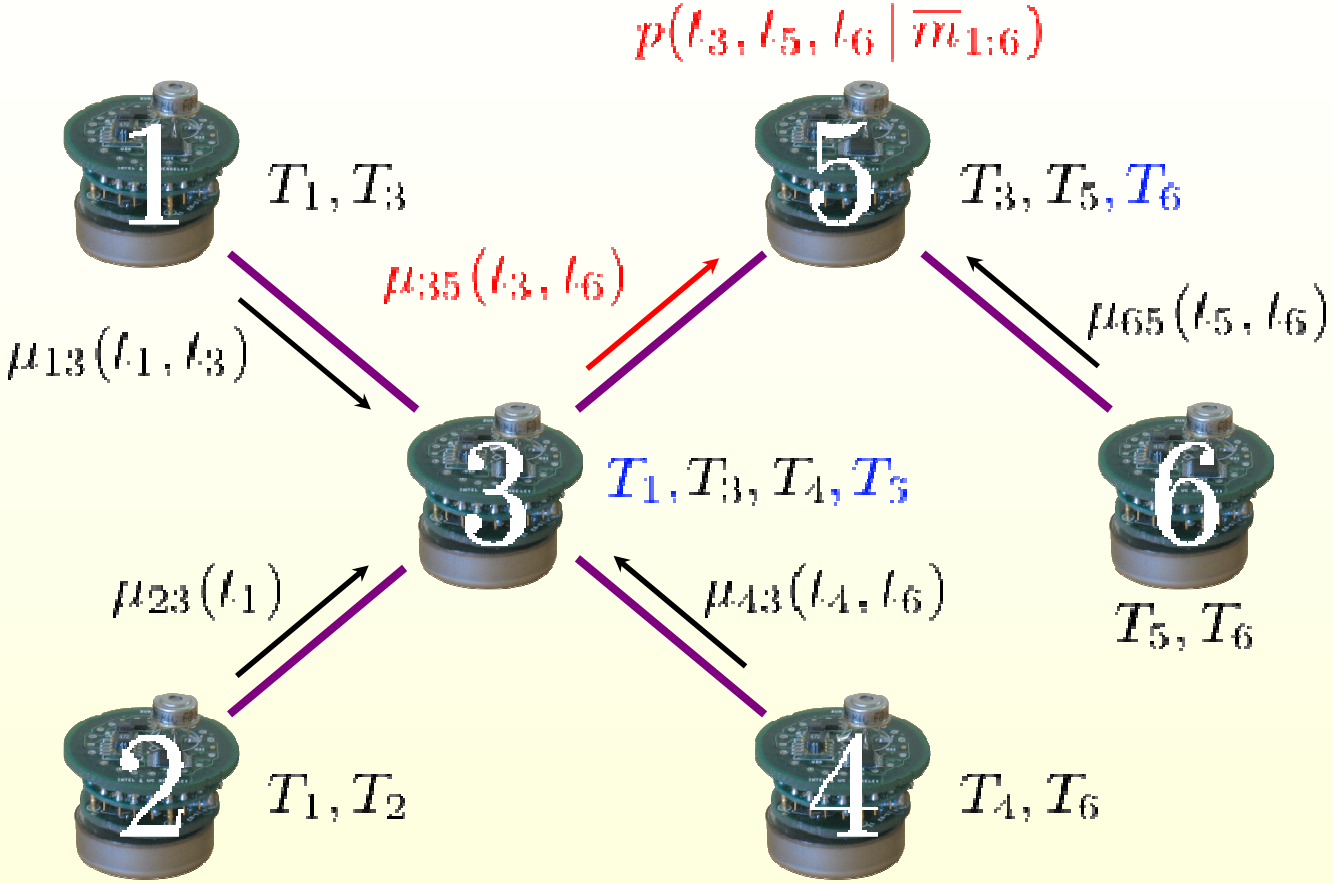
Tree Optimization

Different spanning trees induce different cliques:



A distributed tree optimization algorithm uses dynamic programming to greedily minimize the cliques.

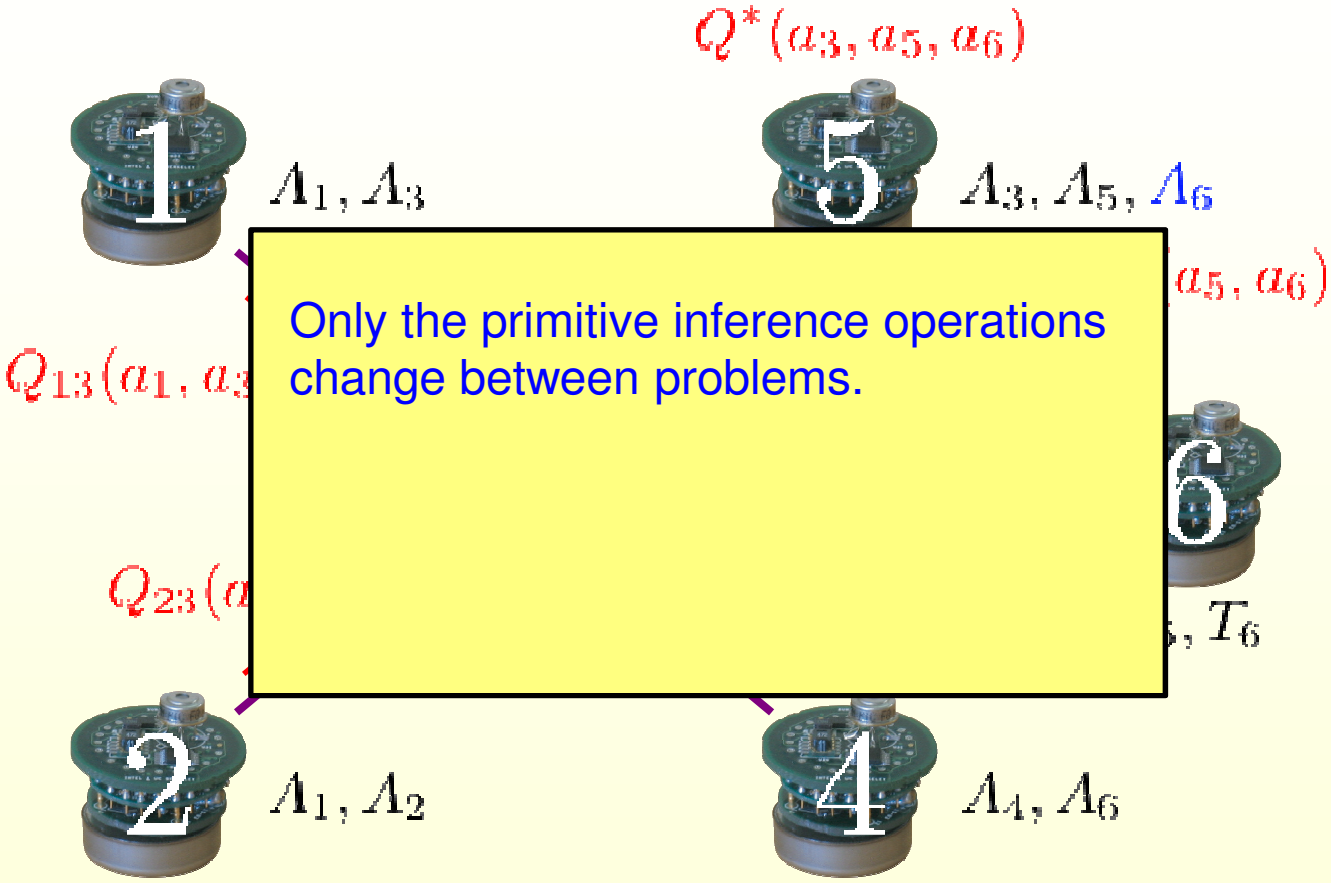
Message Passing (Calibration)



$$\mu_{35}(l_3, l_6) = \sum_{l_1, l_4} \psi_3(l_3, l_4; \overline{m}_3) \times \mu_{13}(l_1, l_3) \times \mu_{23}(l_1) \times \mu_{43}(l_4, l_6)$$

$$p(l_3, l_5, l_6 \mid \overline{m}_{1:6}) = \psi_5(l_3, l_5; \overline{m}_5) \times \mu_{35}(l_3, l_6) \times \mu_{65}(l_5, l_6)$$

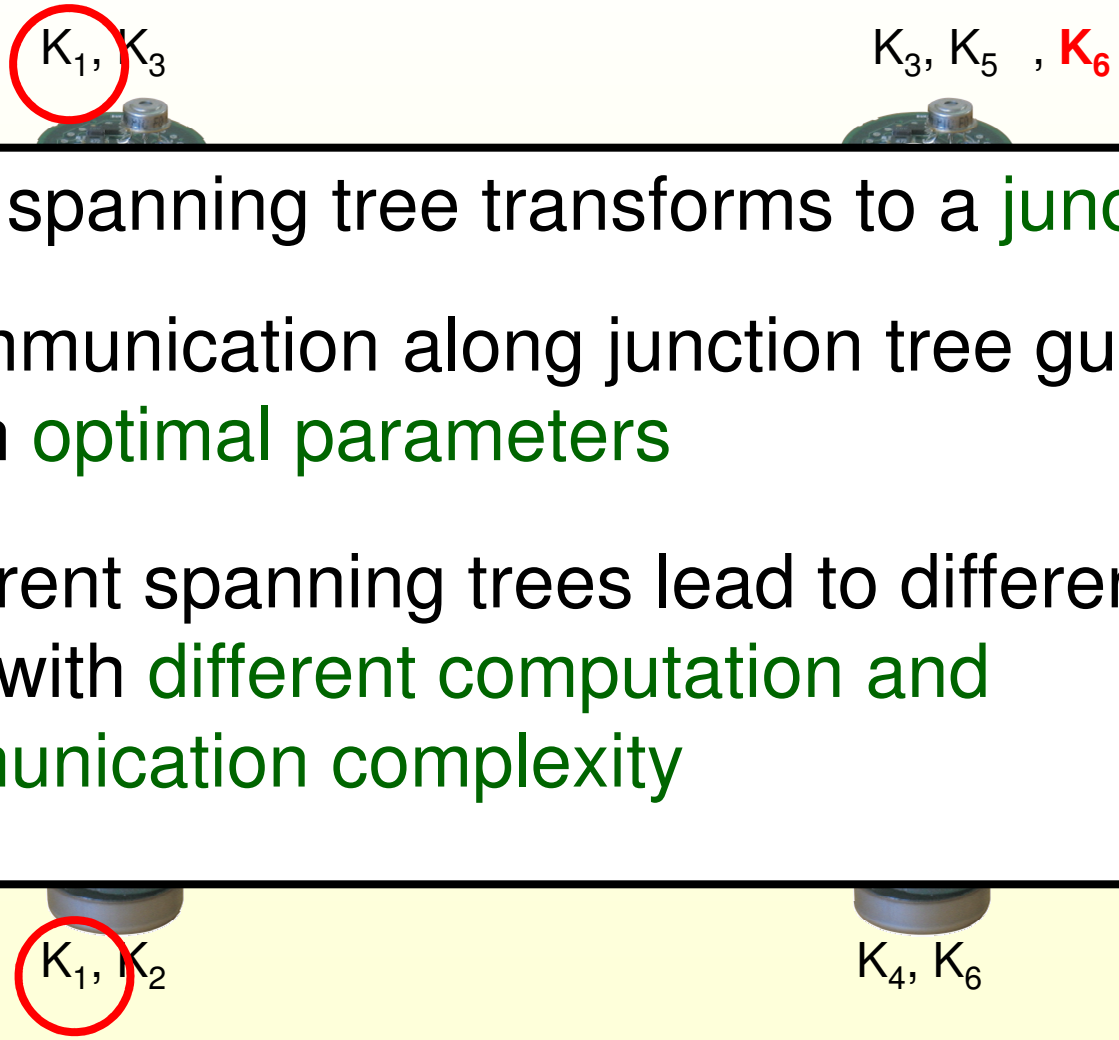
Message Passing (Control)



$$Q_{35}(a_3, a_6) = \max_{a_1, a_4} Q_3(a_3, a_4) + Q_{13}(a_1, a_3) + Q_{23}(a_1) + Q_{43}(a_4, a_6)$$

$$Q^*(a_3, a_5, a_6) = Q_5(a_3, a_5) + Q_{35}(a_3, a_6) + Q_{65}(a_5, a_6)$$

Distributed Junction Trees

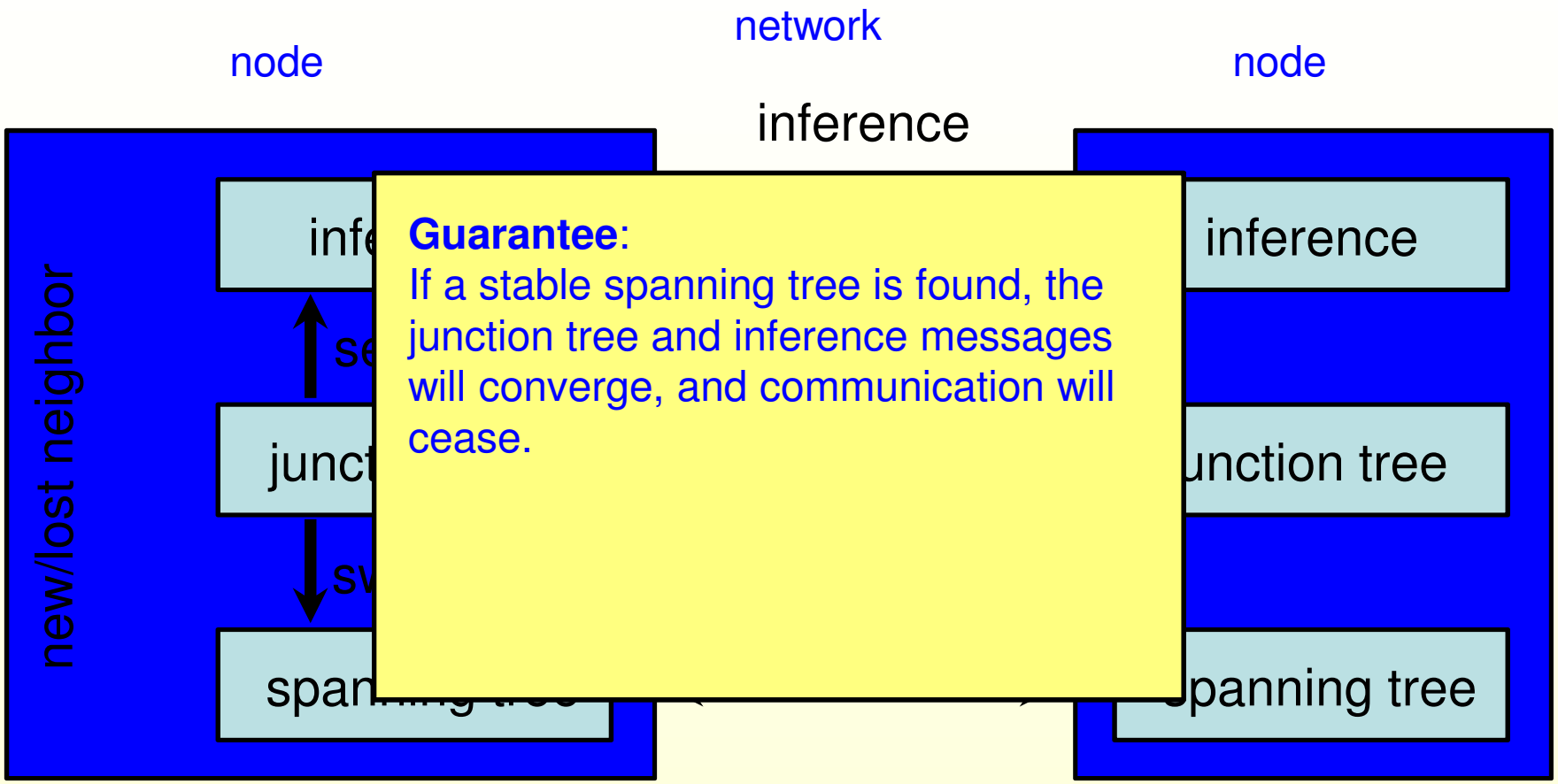


- Any spanning tree transforms to a junction tree
- Communication along junction tree guaranteed to obtain optimal parameters
- Different spanning trees lead to different junction trees with different computation and communication complexity

Robustness

- Robustness is key in sensor networks
  - Nodes may be added to the network or fail
  - Communication is unreliable
  - Link qualities change over time
- Distributed regression messages are robust:
  - Lost messages correspond to lost measurements
- Must make spanning tree and junction tree algorithms robust

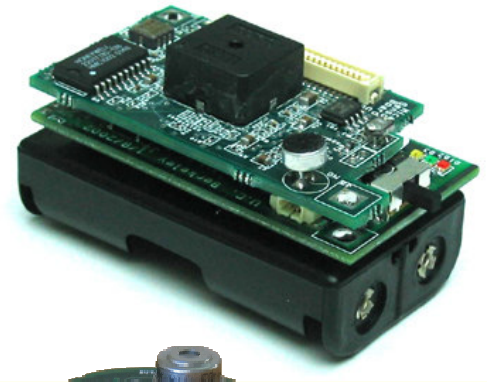
Robustness Through Interaction

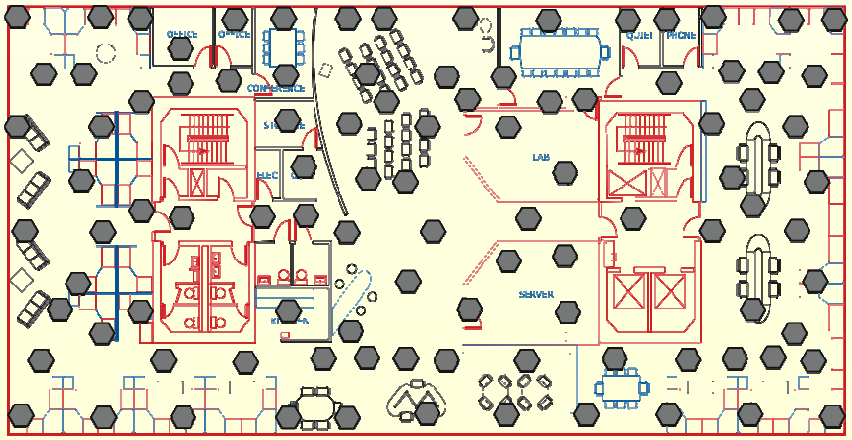


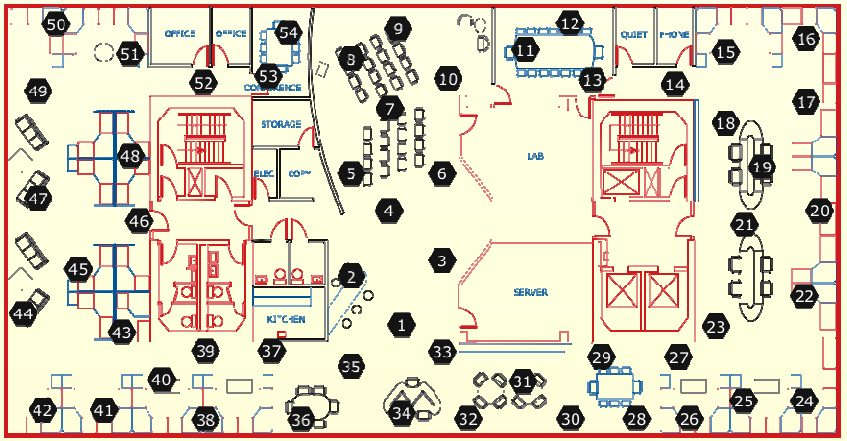
Programs on each node respond to changes in each other's states to adapt to network changes.

Experimental Results

- Testbed of 97 Mica2 motes at the Intel Berkeley Research Lab
- Simulation of 54 Mica2dot motes
  - based upon link quality statistics obtained from a real deployment

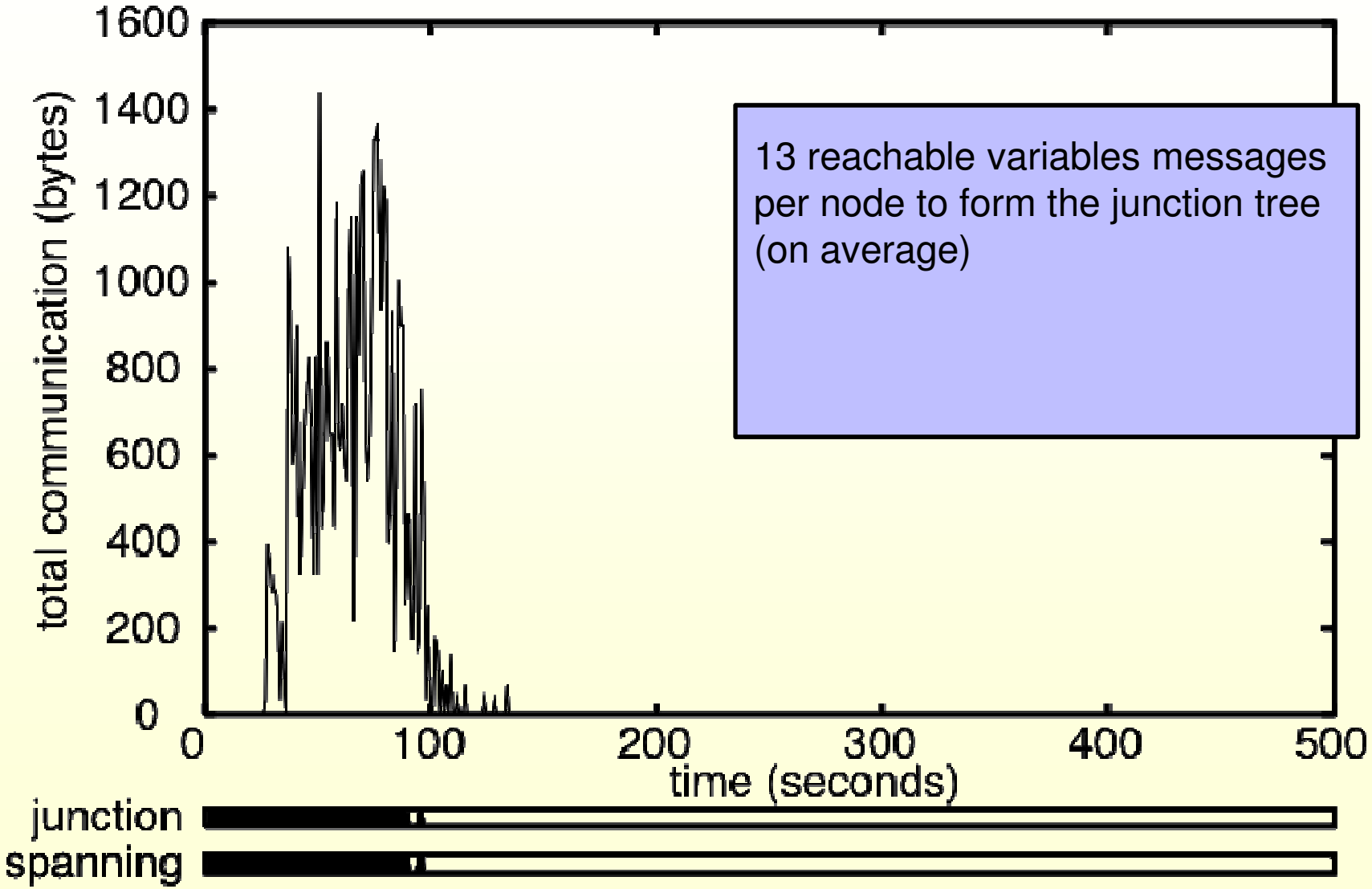






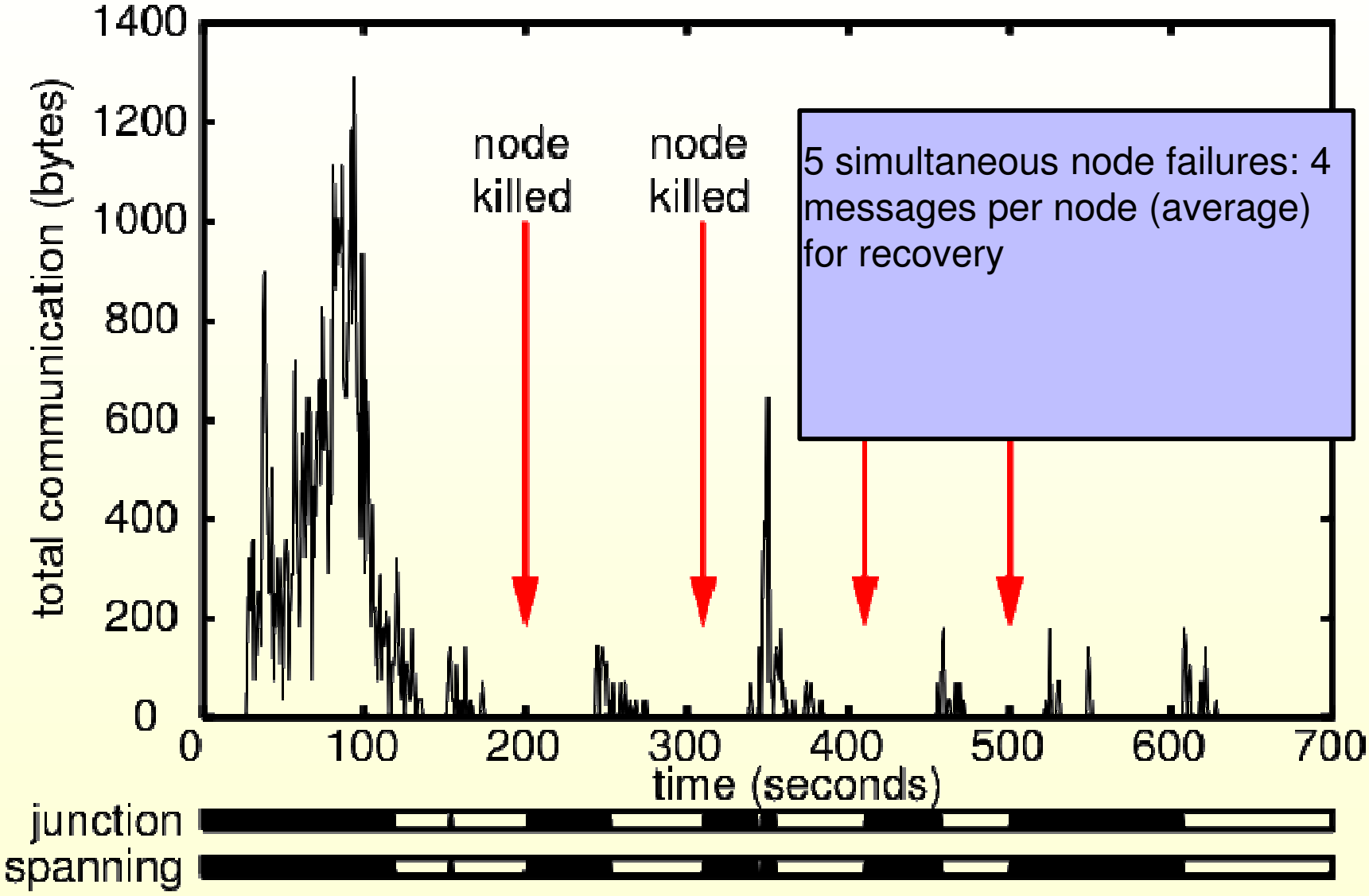
Communication Costs

84 mica2 motes



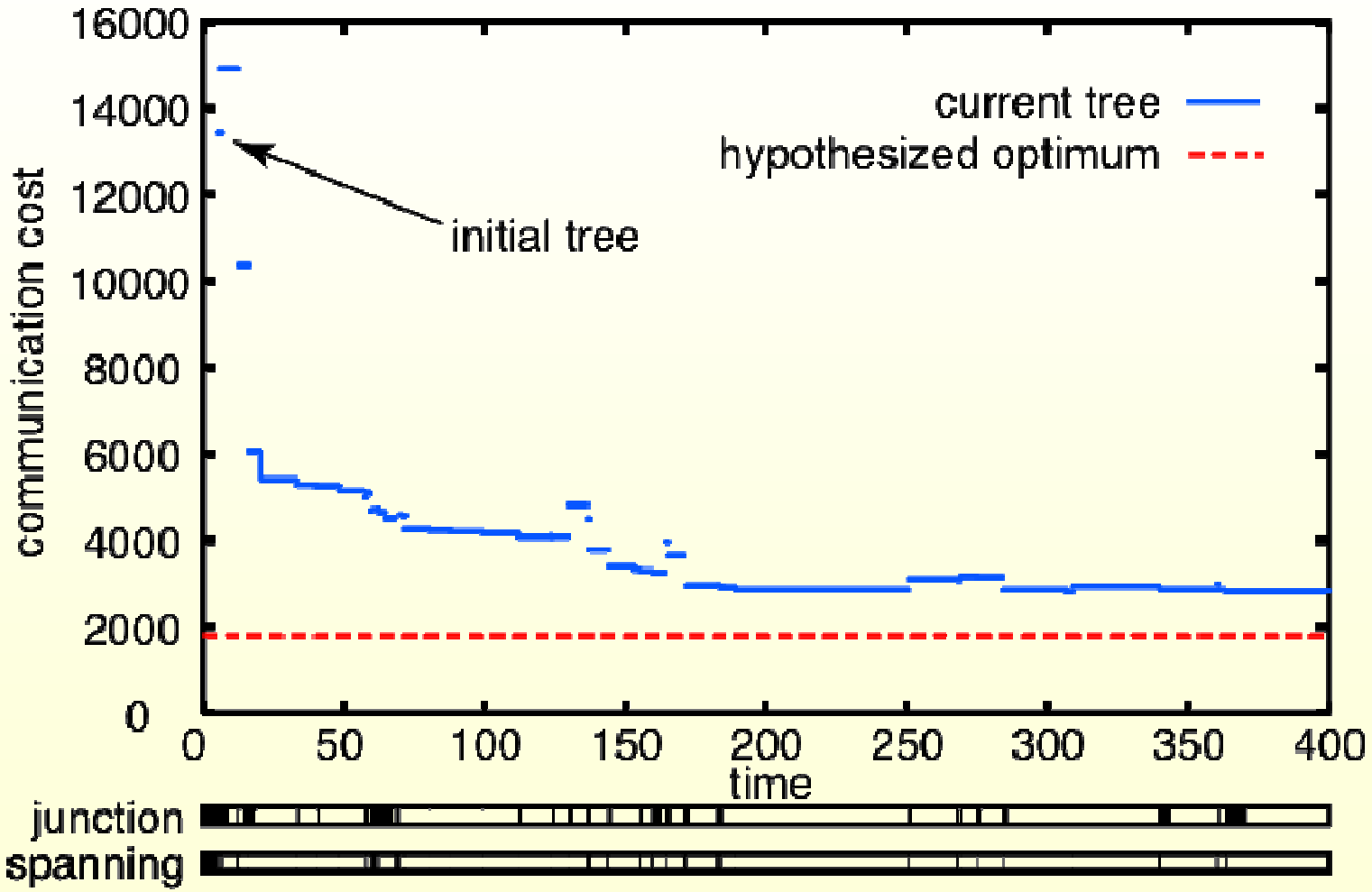
Robustness

84 mica2 motes

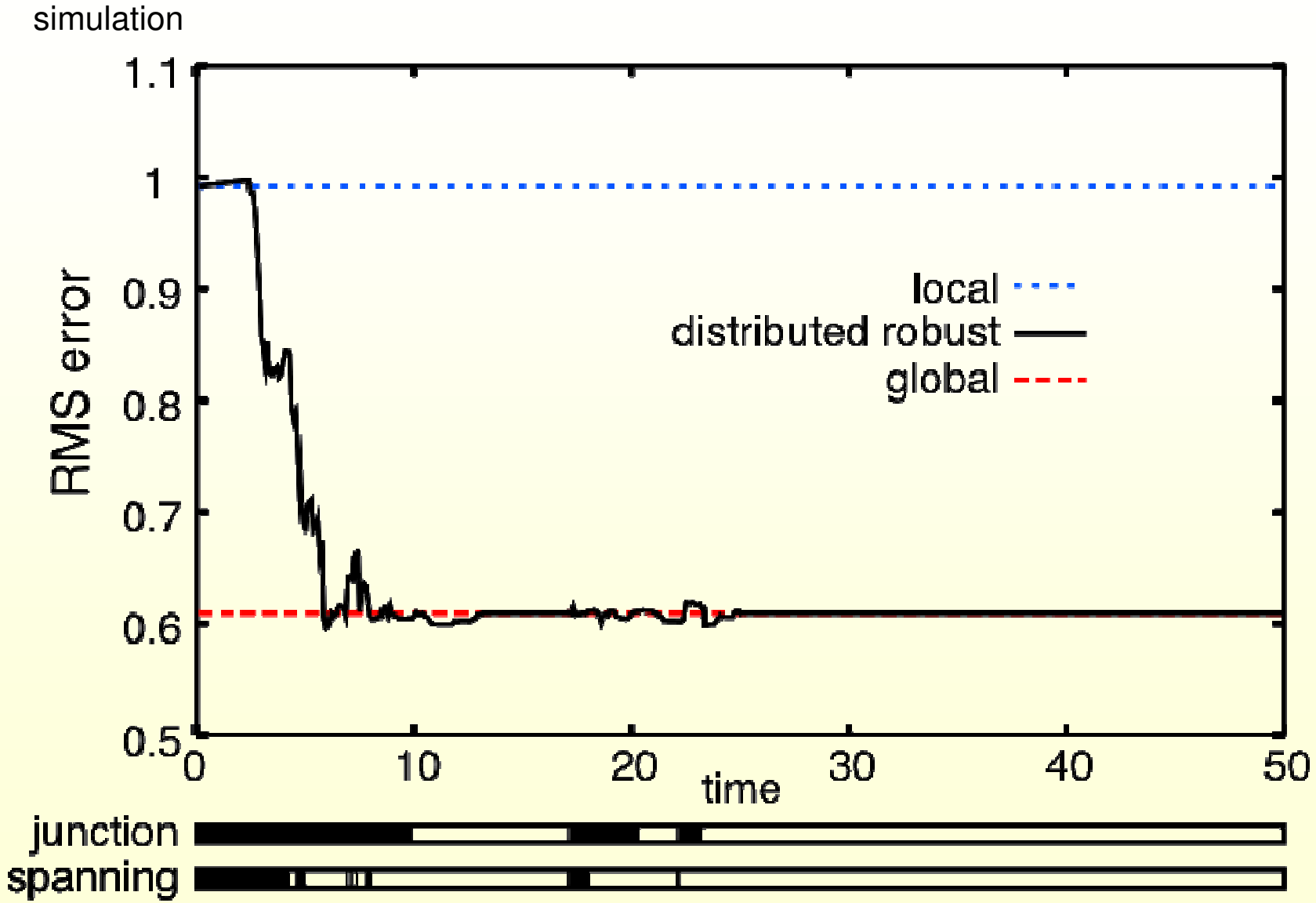


Tree Optimization

simulation

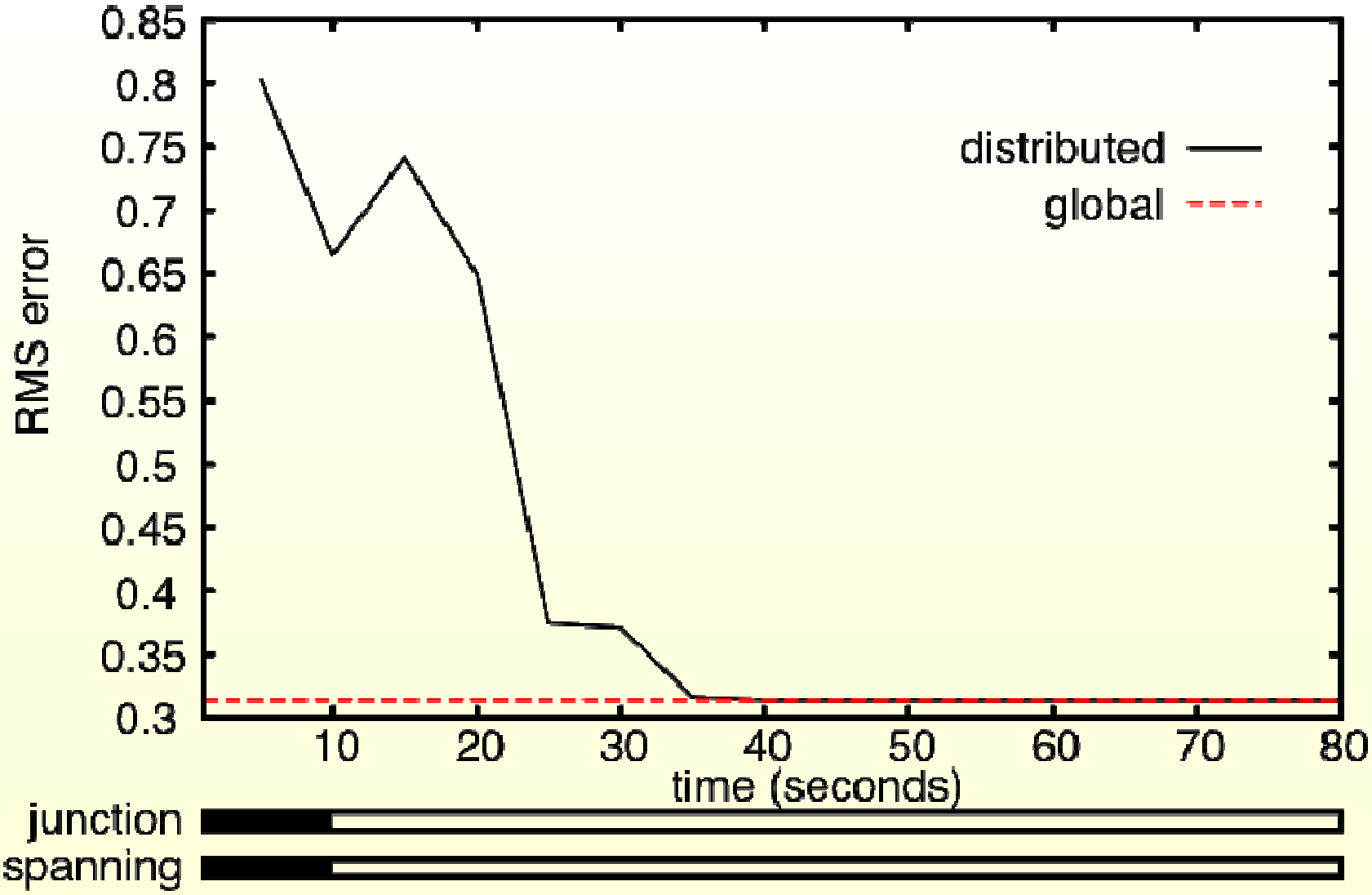


Calibration

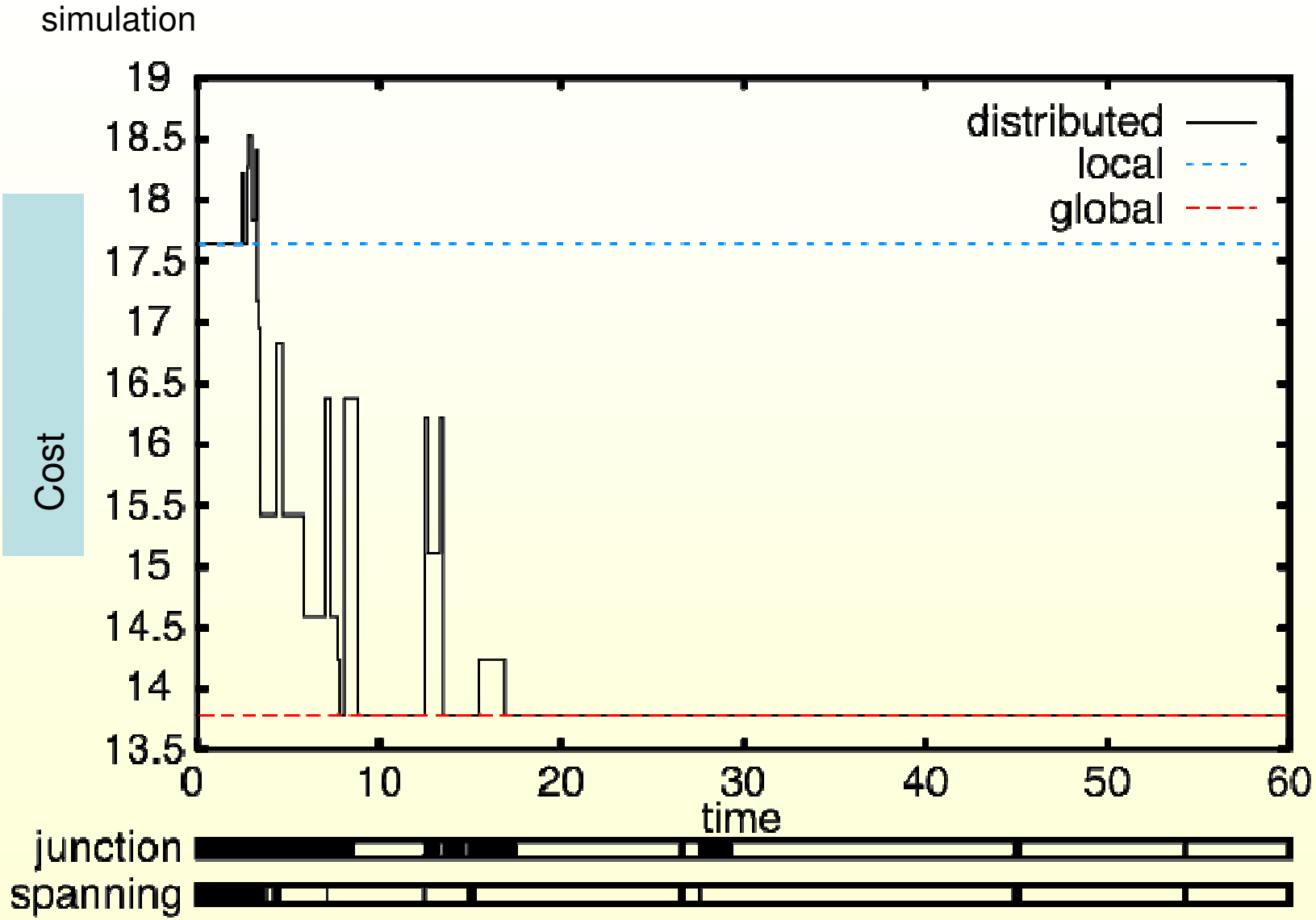


Regression

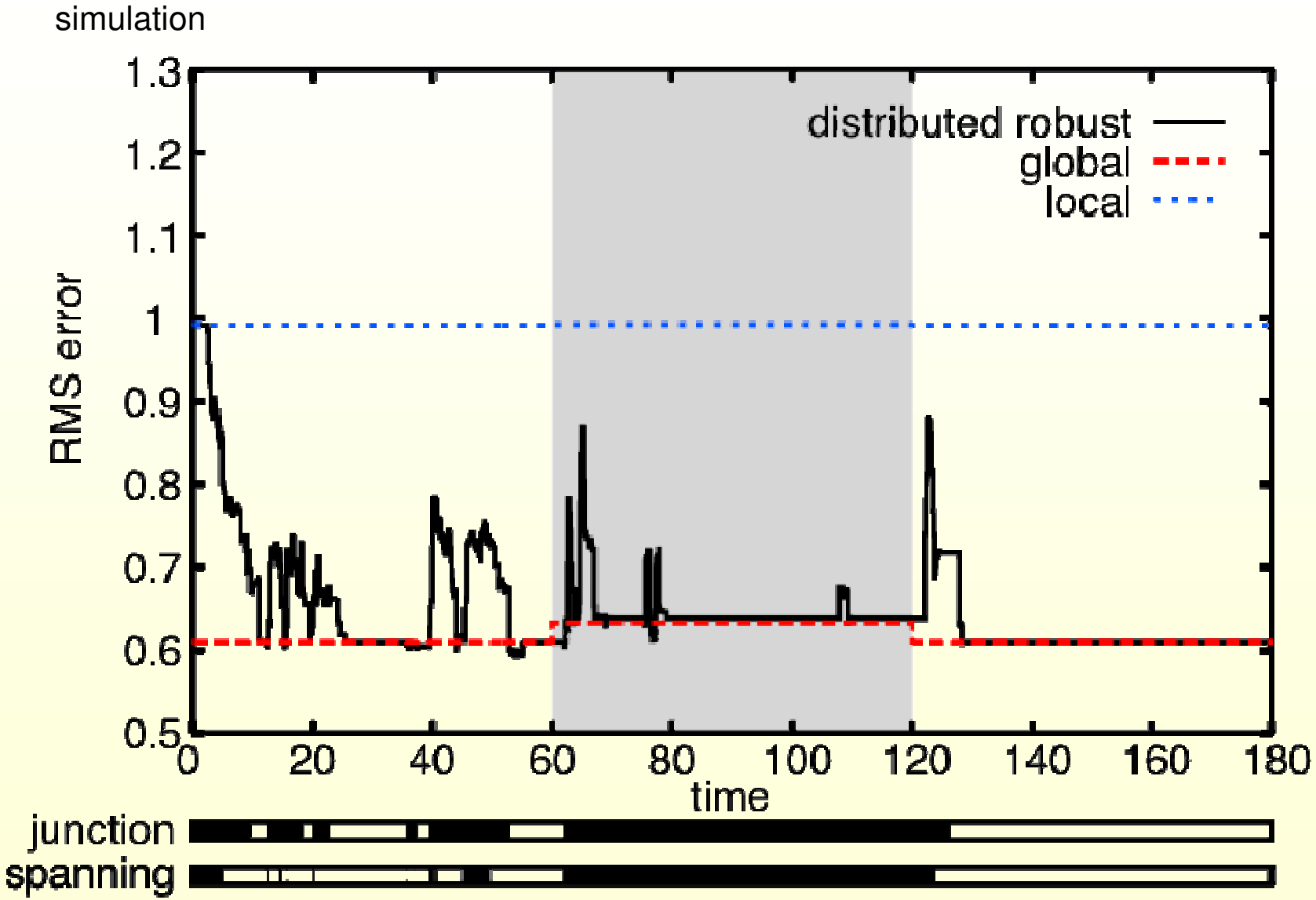
40 mica2 motes



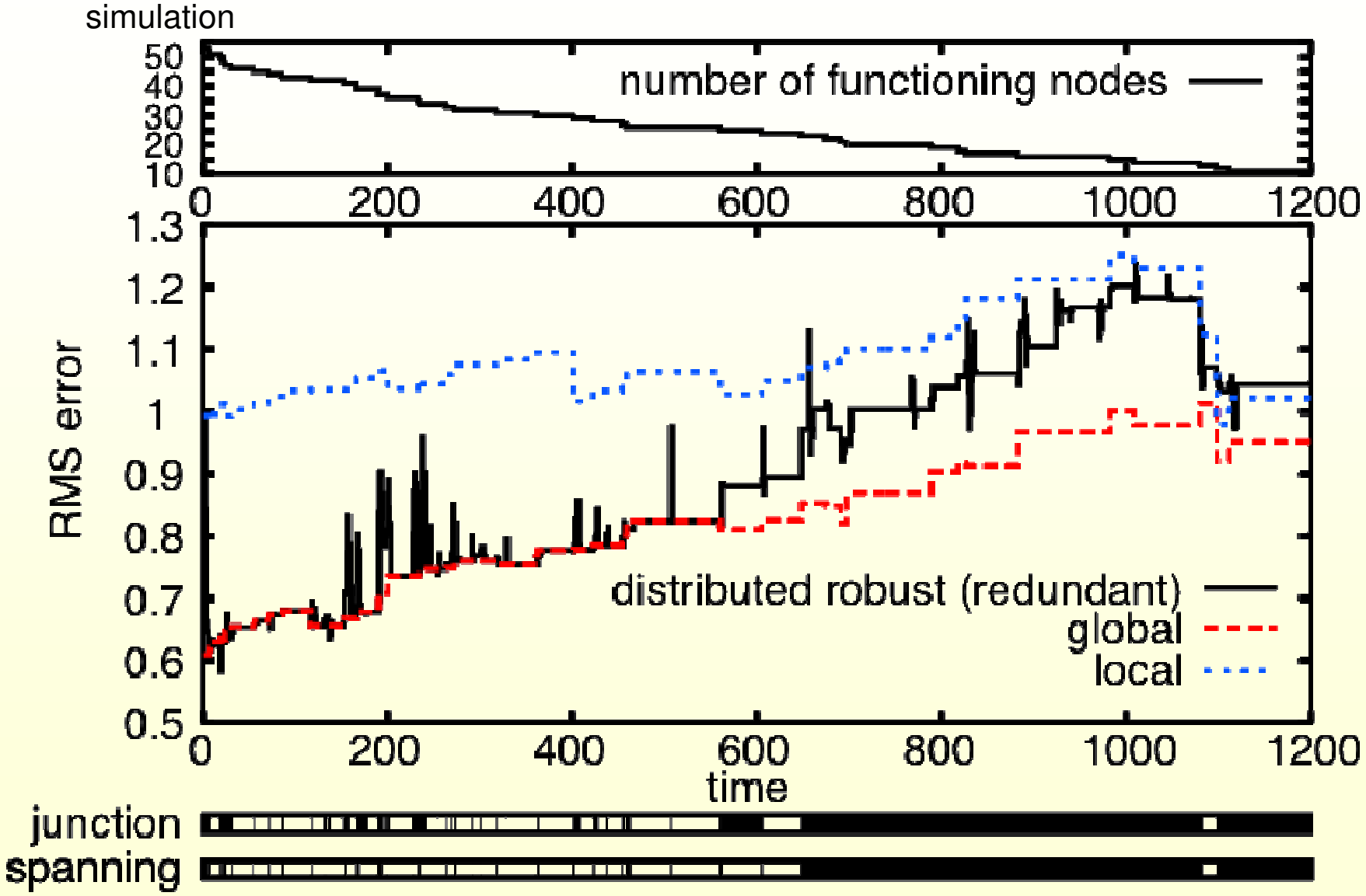
Lighting Control



Interference

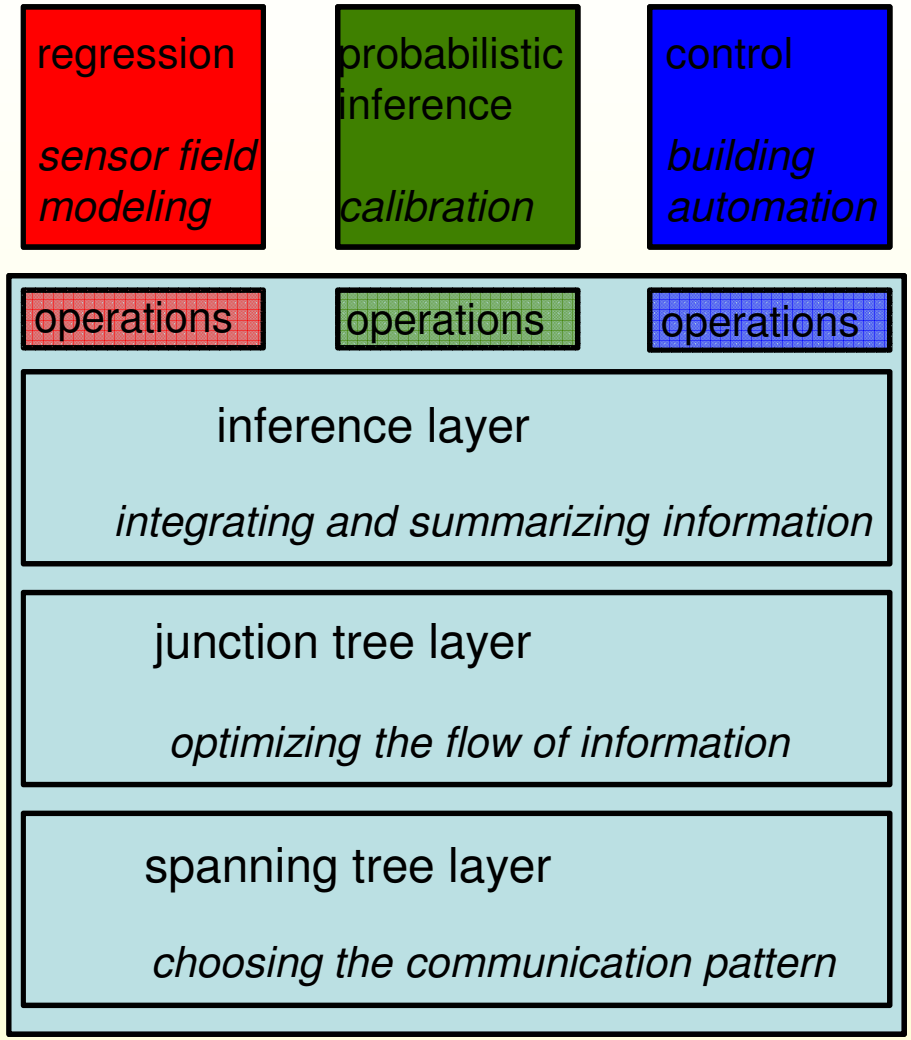


Node Failures



Summary

- An architecture for sensor network inference that is **general**, **efficient**, and **robust** to failures
- Enables rapid application of sensor networks to many challenging inference problems
- Ultimate goal: inference architecture becomes “black box” for more sophisticated sensor network algorithms



Junction Tree Conclusions

- Paper presents method for distributed inference in sensor networks using a junction tree
- Experiments show method
  - Recovers quickly from communication and node failures and produces right answer

The End