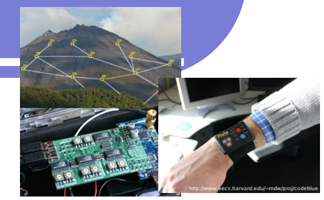
Uncertain Data Management for Sensor Networks

Amol Deshpande, University of Maryland

(joint work w/ Bhargav Kanagal, Prithviraj Sen, Lise Getoor, Sam Madden)

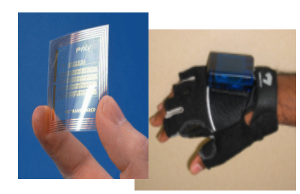
Motivation: Sensor Networks

- Unprecedented, and rapidly increasing, instrumentation of our every-day world
- Huge data volumes generated *continuously* that must be processed in *real-time*
- *Imprecise, unreliable and incomplete* data
  - Inherent measurement noises (e.g. GPS)
  - Low success rates (e.g. RFID)
  - Communication link or sensor node failures (e.g. wireless sensor networks)
  - Spatial and temporal biases
- Typically *acquisitional* environments
  - Energy-efficiency the primary concern



Wireless sensor networks

Distributed measurement networks (e.g. GPS)



RFID

Industrial Monitoring

Motivation: Uncertain Data

- Similar challenges in other domains
  - Data integration
    - Noisy data sources, automatically derived schema mappings
    - Reputation/trust/staleness issues
  - Information extraction
    - Automatically extracted knowledge from text
  - Social networks, biological networks
    - Noisy, error-prone observations
    - Ubiquitous use of *entity resolution, link prediction* etc…
- Need to develop database systems for efficiently representing and managing uncertainty

Example: Wireless Sensor Networks

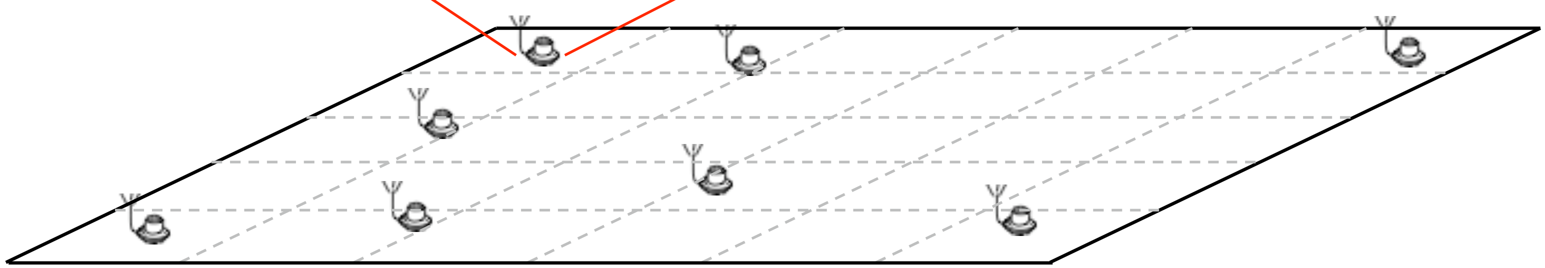
Moteiv Invent:

8Mhz uProc, 250kbps 2.4GHz Transreceiver

10K RAM, 48K program/ 512k data flash

Rechargeable Battery (USB)

Light, *temperature*, acceleration, and sound sensors



A wireless sensor network deployed to monitor temperature

Example: Wireless Sensor Networks

User

select *time, avg(temp)*
from *sensors*
epoch *1 hour*

{10am, 23.5}
{11am, 24}

time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

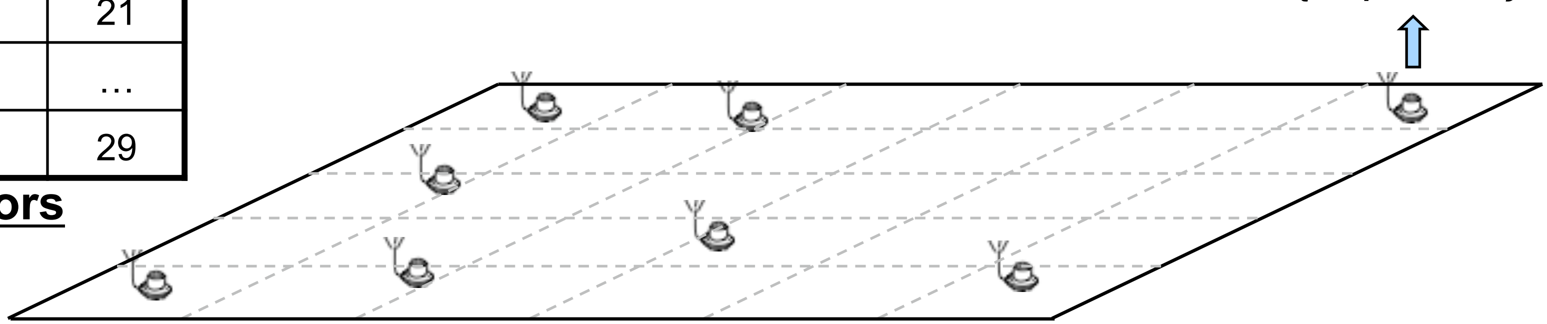
sensors

1. Spatially biased deployment
➔ these are not true averages

2. High data loss rates
➔ averages of different sets of sensors

3. Measurement errors propagated to the user

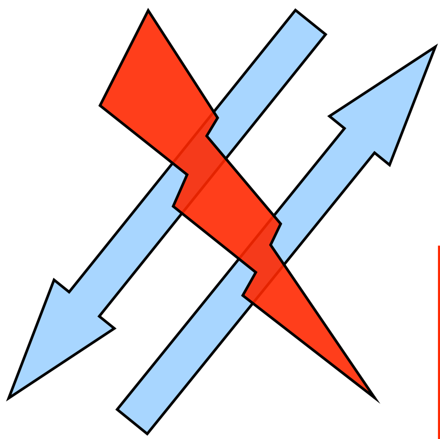
{12pm, 70}



A wireless sensor network deployed to monitor temperature

Example: Wireless Sensor Networks

User

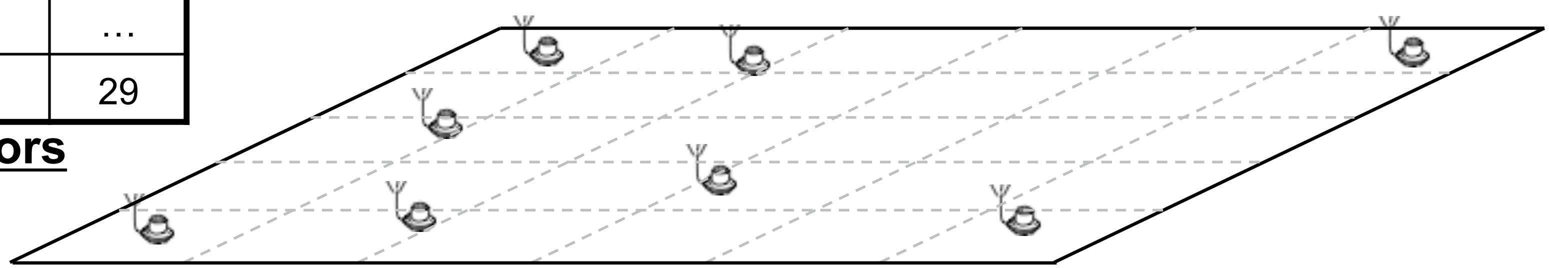


time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

sensors

Impedance mismatch

User wants to query the "underlying environment", and not the sensor readings at selected locations



A wireless sensor network deployed to monitor temperature

Example: Inferring High-level Events

- Inferring "transportation mode"/ "activities"
  - Using easily obtainable sensor data (GPS, RFID proximity data)
  - Can do much if we can infer these automatically

home

office

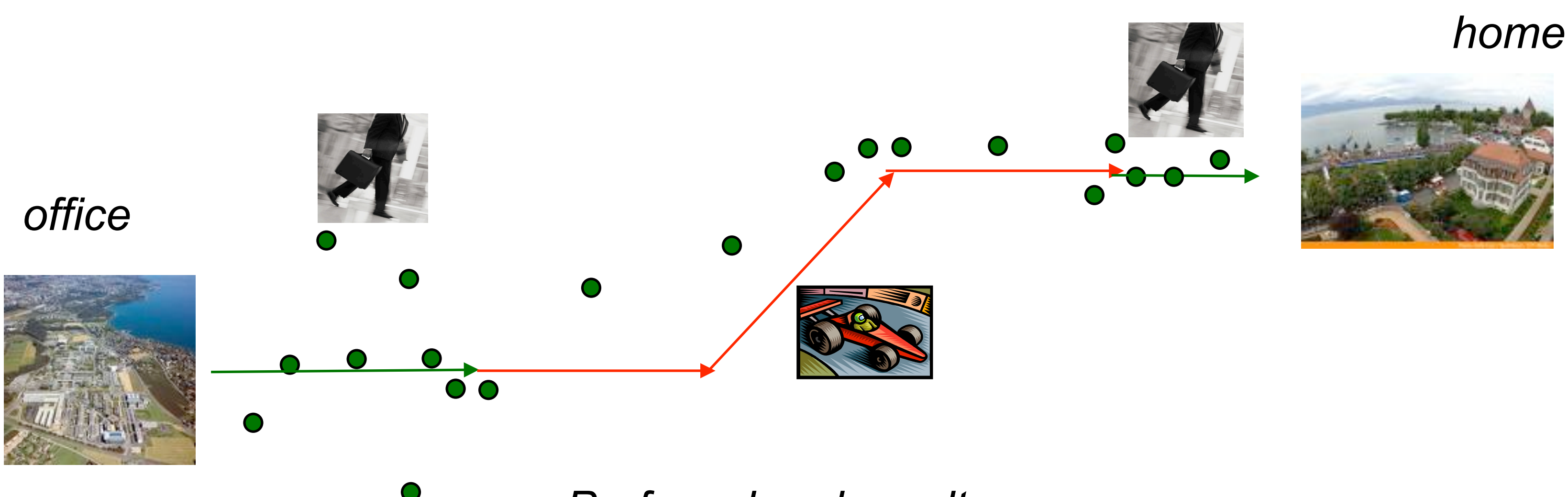
Have access to noisy "GPS" data

<u>Infer</u> the transportation mode:

walking, running, in a car, in a bus

Example: Inferring High-level Events

- Inferring “transportation mode”/ “activities”
  - Using easily obtainable sensor data (GPS, RFID proximity data)
  - Can do much if we can infer these automatically

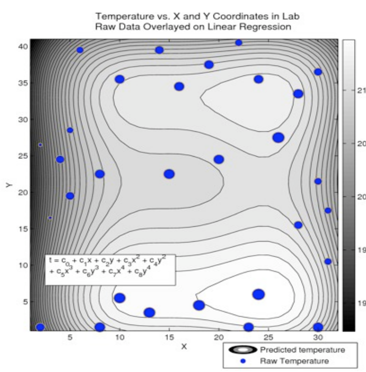


Preferred end result:
Clean path annotated with transportation mode

Data Processing Step 1

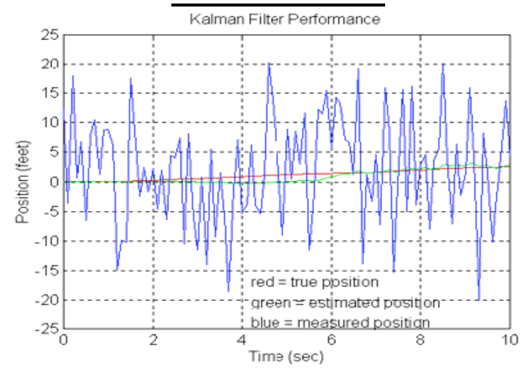
- Apply a statistical model to the data
  - Eliminate spatial/temporal biases, handle missing data through extrapolation (e.g. regression, interpolation models)
  - Filter measurement noise (e.g. Kalman Filters)
  - Infer hidden variables, pattern recognition (e.g. HMMs)
  - Fault/anomaly detection
  - Forecasting/prediction (e.g. ARIMA)
- *No support in current database systems !*

<u>Temperature monitoring</u>



Regression/interpolation models

<u>GPS Data</u>



Kalman Filters …

Sensor Data Processing: Now

Sensor Network

Database

User

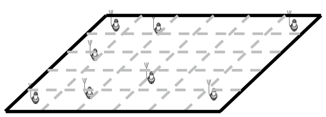


Table raw-data

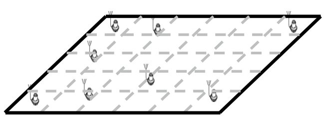
time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

1. Extract all readings into a file
2. Run MATLAB/R/other data processing tools
3. Write output to a file/back to the database
4. Write data processing tools to process/aggregate the output (maybe using DB)
5. Decide new data to acquire

Repeat

Sensor Data Processing: What we want

Sensor Network



Data

Tasks

Database

Table raw-data

time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

Table processed-data

time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

User

Models to be applied in real-time for data cleaning, forecasting, anomaly/event detection etc…

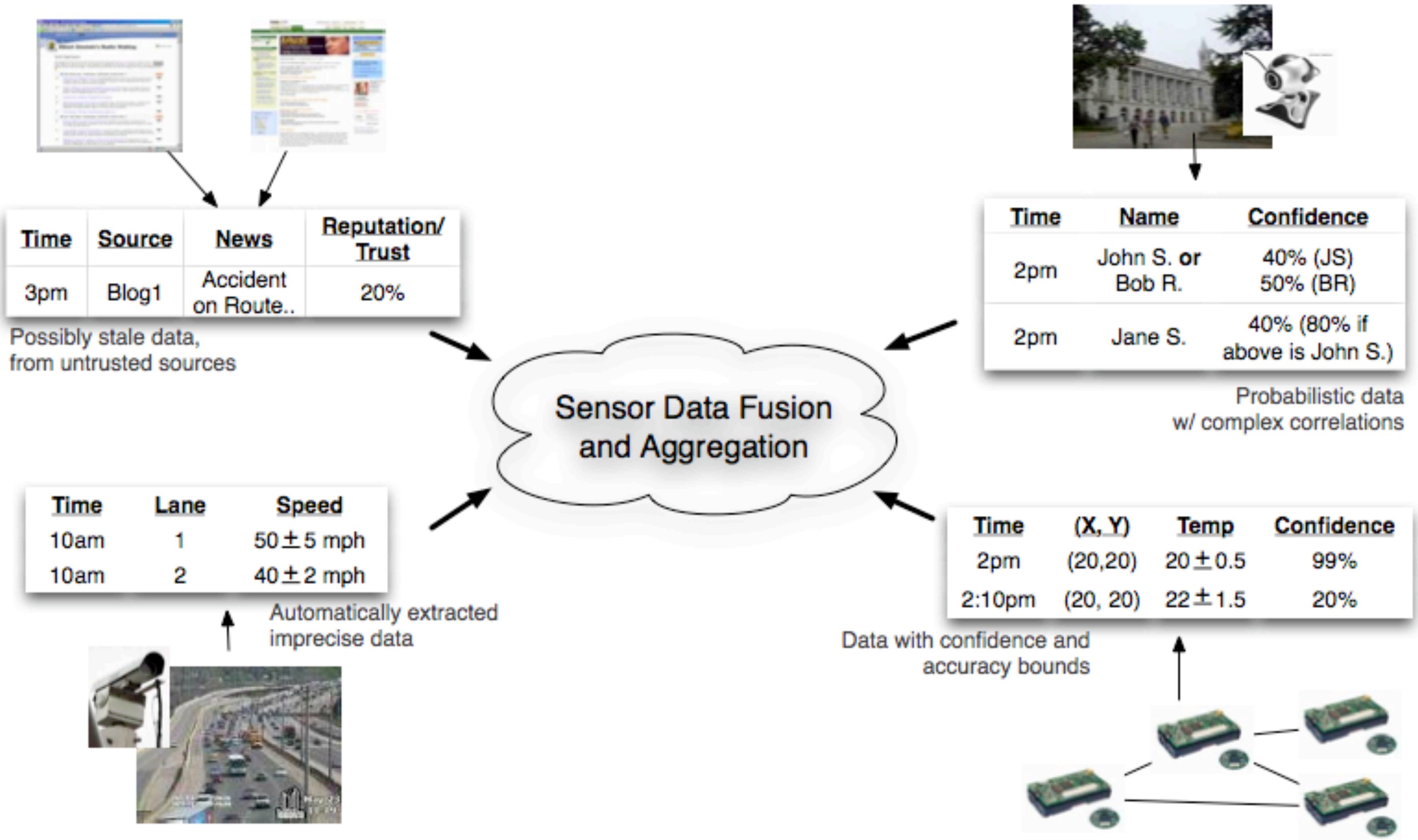
Continuous (standing) queries e.g. alert monitoring

Results to continuous queries

Ad hoc queries (possibly against processed, modeled data)

Challenges

- Abstractions and language constructs for pushing statistical models into databases
  - Large diversity in the models used in practice
- Efficiently processing high-rate data streams
- Querying over probabilistic model outputs
  - Naturally exhibit high degrees of correlations
  - Many different types of uncertainty
- Model-driven data acquisition
  - Minimize the data acquired to answer a query
- Need for in-network, distributed processing
  - Global inference needed to achieve consistency



Time	Source	News	Reputation/ Trust
3pm	Blog1	Accident on Route..	20%

Time	Name	Confidence
2pm	John S. **or** Bob R.	40% (JS) 50% (BR)
2pm	Jane S.	40% (80% if above is John S.)

Time	Lane	Speed
10am	1	50 ± 5 mph
10am	2	40 ± 2 mph

Time	(X, Y)	Temp	Confidence
2pm	(20,20)	20 ± 0.5	99%
2:10pm	(20, 20)	22 ± 1.5	20%

Caption: Even if the sensor web data sources were to publish data using intuitive well-defined interfaces, the complex and semantically disparate measures of data quality and uncertainty typically associated with it make sensor data fusion and aggregation a challenging task.

Outline

- Motivation
- Statistical modeling of sensor data
  - Abstraction of *model-based views*
  - Regression-based views
  - Views based on dynamic Bayesian networks
- Query processing over model outputs
- Some interesting sensor network problems
  - Model-driven data acquisition
  - Distributed inference in sensor networks

Outline

- Motivation
- Statistical modeling of sensor data
  - Abstraction of *model-based views*
  - Regression-based views
  - Views based on dynamic Bayesian networks
- Query processing over model outputs
- Some interesting sensor network problems
  - Model-driven data acquisition
  - Distributed inference in sensor networks

Model-based User Views for Sensor Data; *A. Deshpande, S. Madden; SIGMOD 2006*

Abstraction: Model-based Views

- An abstraction analogous to *traditional database views*
- Provides *independence from the messy measurement details*

No difference from a user's perspective

A *traditional* database *view*
(defined using an SQL query)

User

avg-balances
select zipcode, avg(balance)
from accounts
group by zipcode

acct-no	balance	zipcode
101	a	20001
102	b	20002
..	..	
..	..	

accounts

A *model-based* database *view*
(defined using a statistical model)

User

temperatures
Use *Regression* to predict missing values and to remove spatial bias

time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

raw-temp-data

Grid Abstraction

A Regression-based View

User

Continuous Function

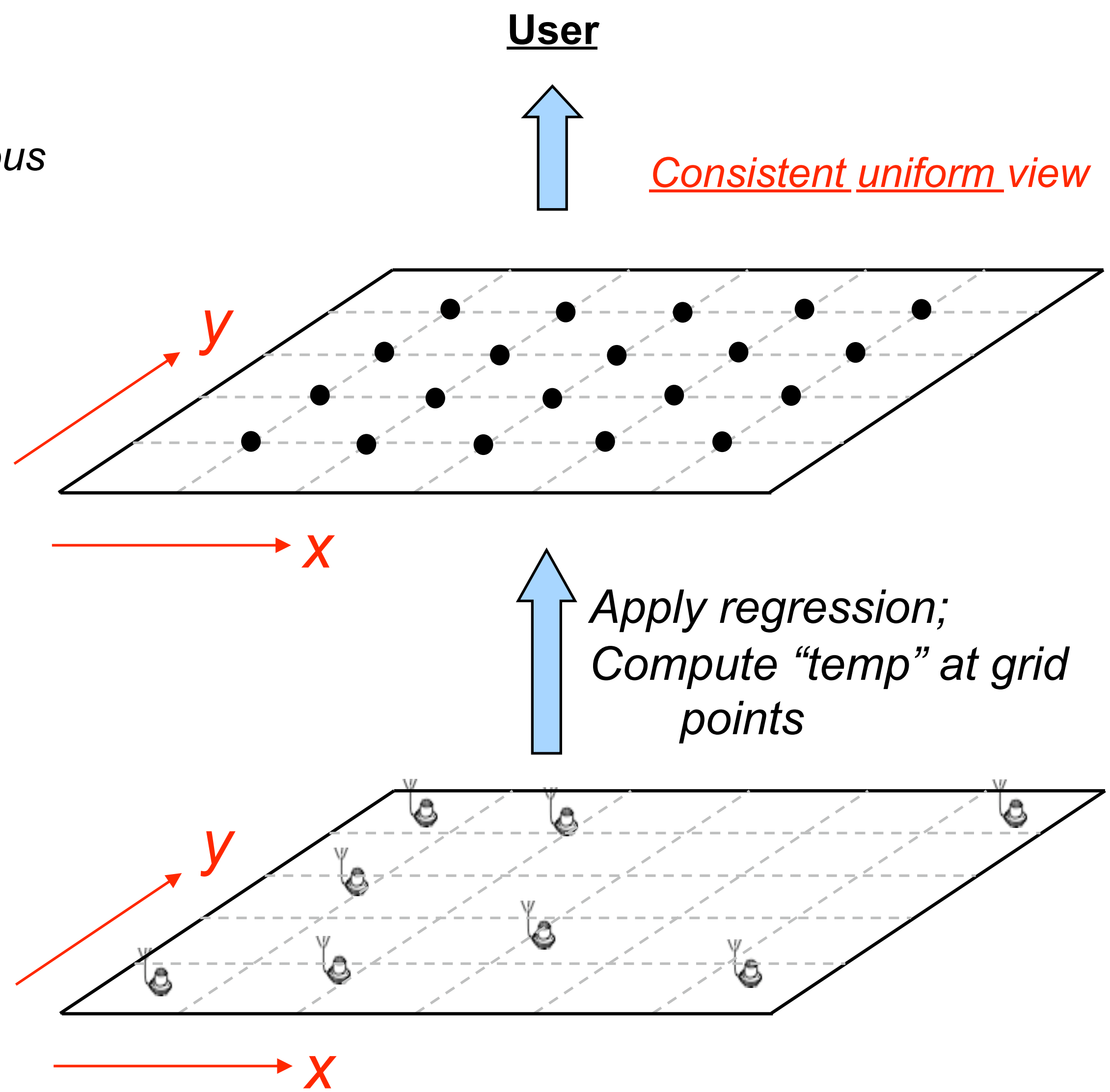
temperatures

Use *Regression* to model temperature as:

$$temp = w_1 + w_2\,x + w_3\,x^2 + w_4\,y + w_5\,y^2$$

time	id	temp
10am	1	20
10am	2	21
..	..	…
10am	7	29

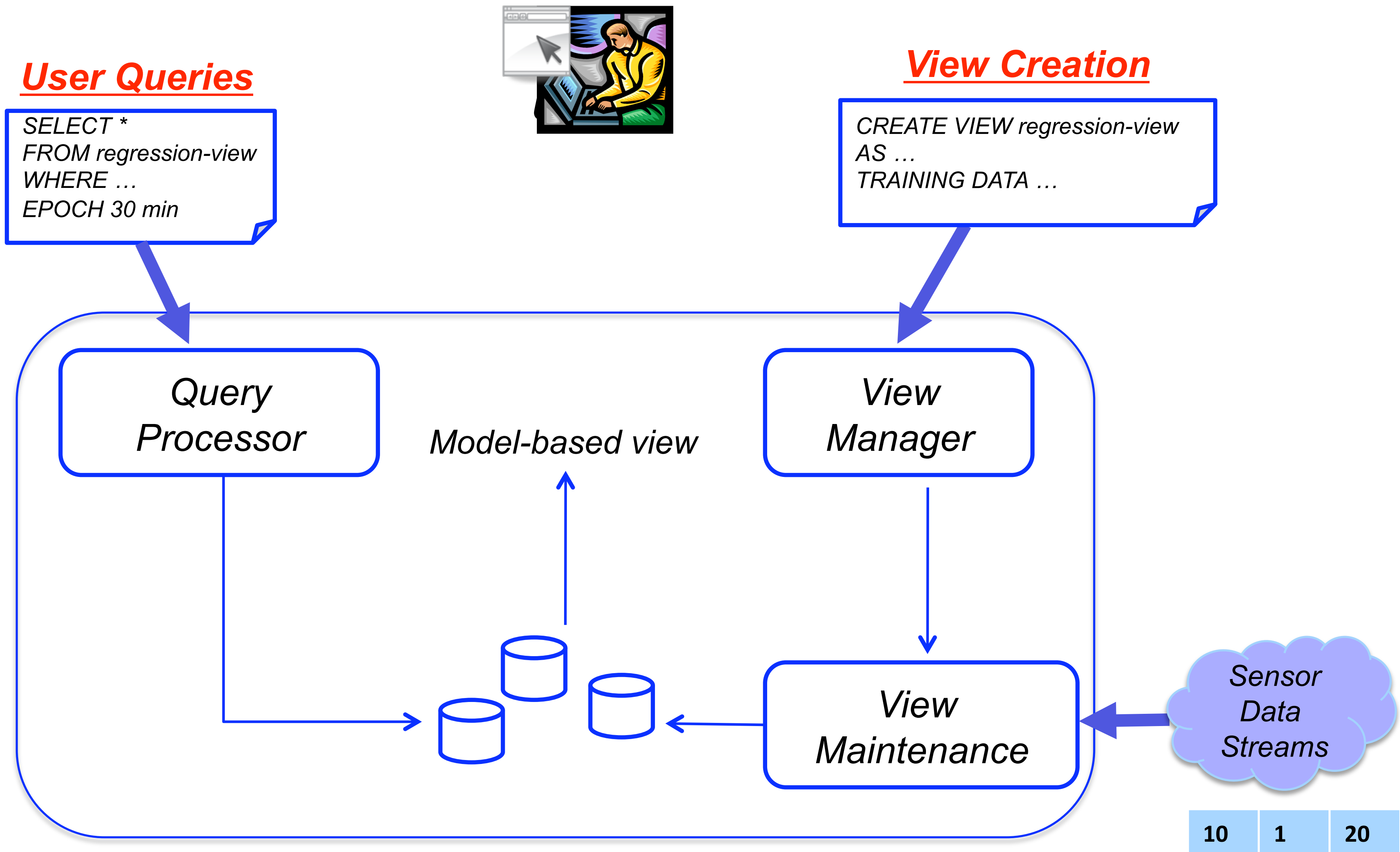
raw-temp-data



MauveDB System

- Being written using the *Apache Derby* Java open source database system codebase
- Supports the abstraction of Model-based User Views
  - Declarative language constructs for creating such views
  - SQL queries over model-based views
  - Keep the models up-to-date as new data is inserted in database

MauveDB System Architecture



MauveDB System Architecture

CREATE VIEW

RegView(time [0::1], x [0:100:10], y[0:100:10], temp)

AS

FIT temp USING time, x, y

BASES 1, x, x^2*, y,* y^2

FOR EACH time T

TRAINING DATA

SELECT temp, time, x, y

FROM raw-temp-data

WHERE raw-temp-data.time = T

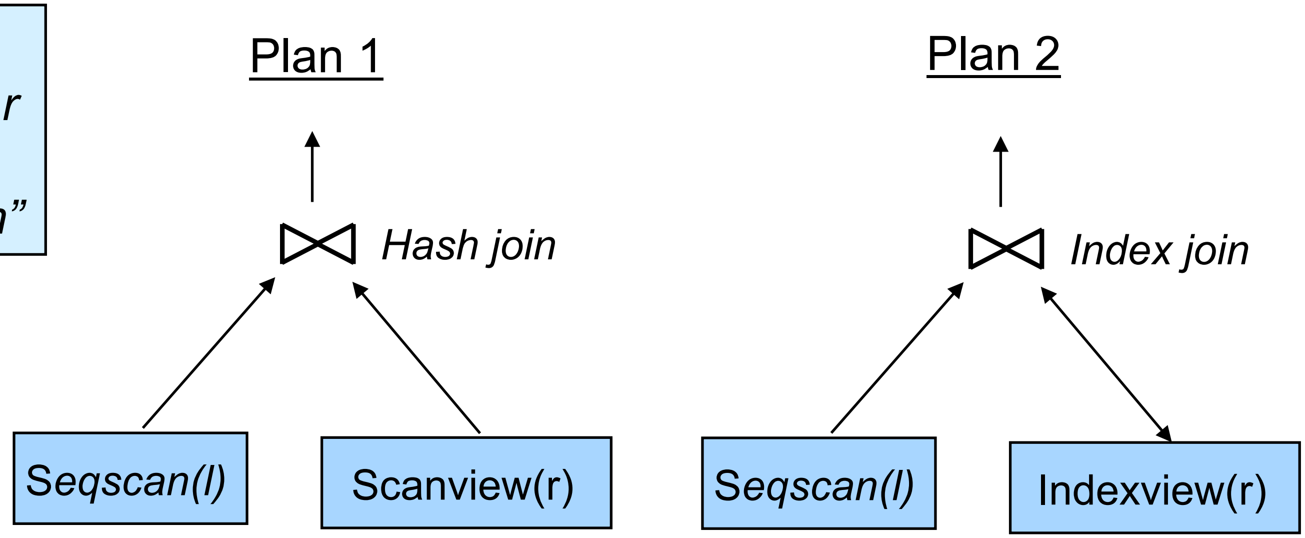
Details specific to the model being used

10 1 20

Query Processing

- *Key challenge: Integrating in a traditional database system*
- Two operators per view type that support *get_next() API*
  - *ScanView:* Returns the contents of the view one-by-one
  - *IndexView (condition):* Returns tuples that match a condition
    - e.g. return *temperature* where *(x, y) = (10, 20)*

*select **
from locations l, reg-view r
where (l.x, l.y) = (r.x, r.y)
and r.time = "10am"



View Maintenance Strategies

- Option 1: Compute the view as needed from base data
  - For regression view, scan the tuples and compute the weights
- Option 2: Keep the view materialized
  - Sometimes too large to be practical
    - E.g. if the grid is very fine
  - May need to be recomputed with every new tuple insertion
    - E.g. a regression view that fits a single function to the entire data
- Option 3: Lazy materialization/caching
  - Materialize query results as computed

- *Generic options shared between all view types*

View Maintenance Strategies

- *Option 4: Maintain an efficient intermediate representation*
- Typically model-specific
- Regression-based Views
  - Say $temp = f(x, y) = w_1 h_1(x, y) + \ldots + w_k h_k(x, y)$
  - Maintain the *weights* for $f(x, y)$ and a *sufficient statistic*
    - Two matrices ($O(k^2)$ *space*) that can be incrementally updated
  - ScanView: Execute $f(x, y)$ on all grid points
  - IndexView: Execute $f(x, y)$ on the specified point
  - InsertTuple: Recompute the coefficients
    - Can be done very efficiently using the sufficient statistic

Thoughts

- Table functions/User-defined functions
  - Can be used to apply a statistical model to a raw data table
    - Using code written in C or Java etc
  - Must be applied repeatedly as new data items arrive
  - No optimization opportunities
  - Not declarative
- Complex data analysis tasks
  - May not be doable using our primitives
  - Our focus is on easy application of statistical models to data
    - By a layperson not familiar with Matlab (or other tools)

Outline

- Motivation
- Statistical modeling of sensor data
  - Abstraction of *model-based views*
  - Regression-based views
  - Views based on dynamic Bayesian networks
- Query processing over model outputs
- Some interesting sensor network problems
  - Model-driven data acquisition
  - Distributed inference in sensor networks

Online filtering, smoothing, and modeling of streaming data*; B. Kanagal, A. Deshpande; ICDE 2008*

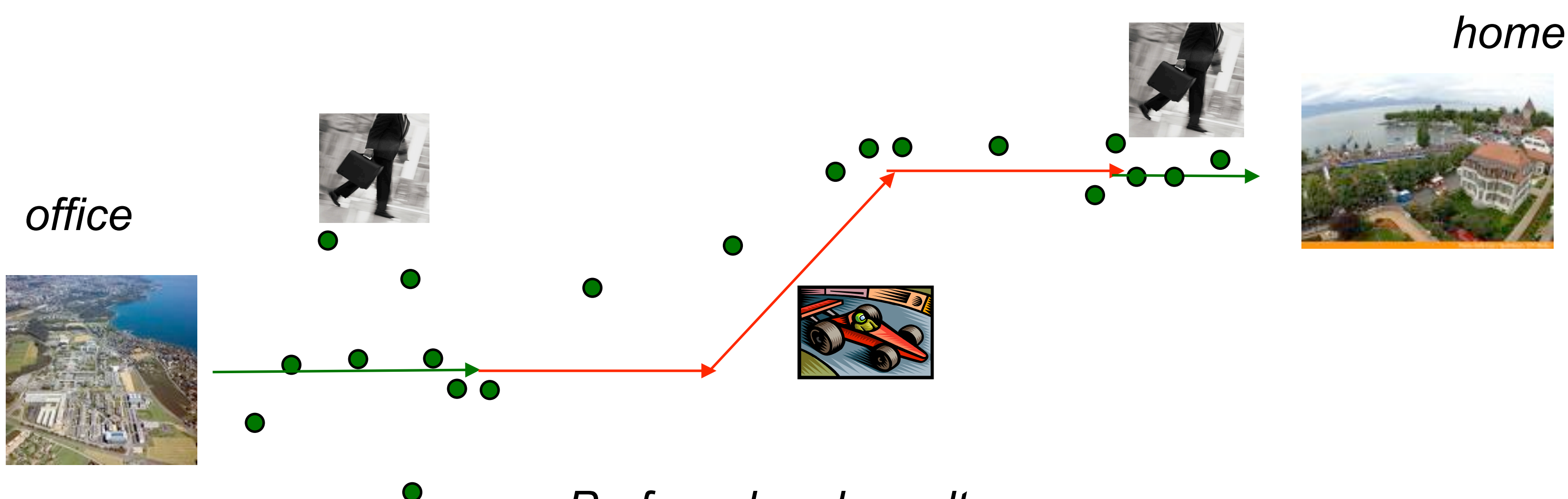
Dynamic Bayesian Networks

- A class of models that can capture *temporal evolution* of a complex stochastic process

- Widely used for many tasks
  - Eliminating measurement noise (Kalman Filters)
  - Anomaly/failure detection
  - Inferring high-level hidden variables (HMMs)
    - e.g. *working status* of a remote sensor, *activity recognition*

Example: Inferring High-level Events

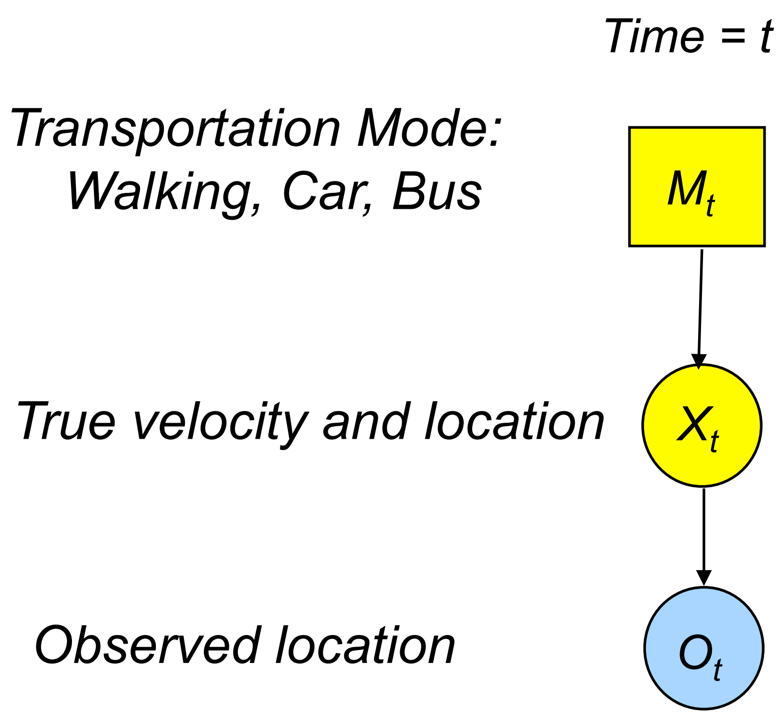
- Inferring “transportation mode”/ “activities”
  - Using easily obtainable sensor data (GPS, RFID proximity data)
  - Can do much if we can infer these automatically



Preferred end result:
Clean path annotated with transportation mode

Dynamic Bayesian Networks

Use a "generative model" that describes how the observations were generated



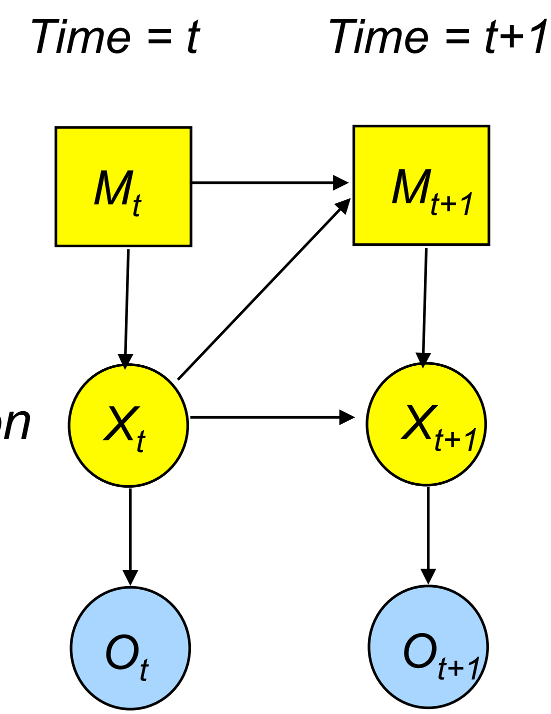
Need conditional probability distributions that capture the process

1. $p(X_t \mid M_t)$: How *(position,velocity)* depends on *mode*
2. $p(O_t \mid X_t)$: The *noise* model for observations

Prior knowledge or learned from data

Dynamic Bayesian Networks

Use a "generative model" that describes how the observations were generated



Transportation Mode: Walking, Car, Bus

True velocity and location

Observed location

Need conditional pdfs:

1. $p(M_{t+1} \mid M_t, X_{t+1})$
2. $p(X_{t+1} \mid X_t)$

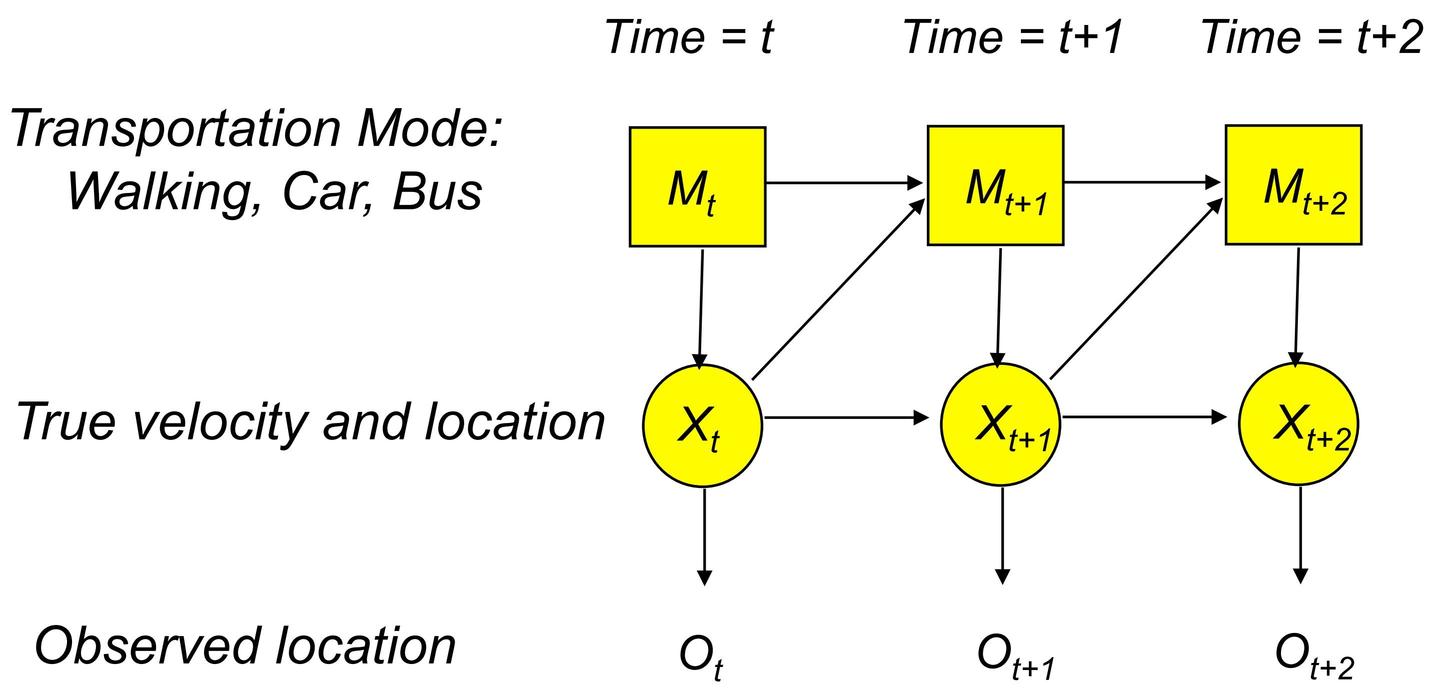
Prior knowledge or learned from data

Dynamic Bayesian Networks

Inference task:

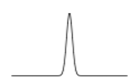
Given a sequence of observations (O_t), find most likely M_t's that explain it.

Alternatively, could provide a probability distribution on the possible M_t's.



Example DBN-based View

User

TIME	USER	MODE (INFERRED)	Location (INFERRED)
5pm	John	Walking: 0.9 Car: 0.1	
5pm	Jane	Walking: 0.9 Car : 0.1	
5:05pm	John	Walking: 0 Car: 1	

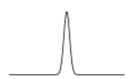
User view of the data
- *Smoothed locations*
- *Inferred variables*

Can query inferred variables:
select count(*)
group by mode
sliding window 5 min

TIME	USER	Location (Observed)
5pm	John	(x1,y1)
5pm	Jane	(x1',y1')
5:05pm	John	(x2,y2)

Original noisy GPS data

Representing DBN-based Views

TIME	USER	MODE (INFERRED)	Location (INFERRED)
5pm	John	Walking: 0.9 Car: 0.1	
5pm	Jane	Walking: 0.9 Car : 0.1	
5:05pm	John	Walking: 0 Car: 1	

id	TIME	USER	MODE	LOCATION	weight
...	...	...	…	…	…
1	5pm	John	W	(a1,b1)	0.01
2	5pm	John	W	(a2,b2)	0.02
3	5pm	John	W	(a3,b3)	0.01
4	5pm	John	C	(a4,b4)	0.01
…	…	…	…	…	…

PARTICLE TABLE

Challenges

- ❑ *Probabilistic attributes*
- ❑ *Strong spatial and temporal Correlations*

Particle-based Representation

- ❑ *Each tuple stored as a set of weighted samples*
- ❑ *Naturally ties in with inference*
- ❑ *Efficient query processing using existing infrastructure*

Architecture

```
# Node Properties
numNodes: 4
hidden: {1,3}
discrete: {1,3}
node(1): ['Wo' 'Fa']
node(3): ['Wo' 'Fa']
# Graph adjacency matrix
graph: [0 1 1 0;
        0 0 0 1;
          0 0 1;
          0 0 0]
         f Nodes
         ;0];
         val(1),[N(50,0.05);
                  U(0,100)]);
cpd(3): (val(1),[[0.99;0.01];
                 [0.01;0.99]]);
cpd(4): (val(3),[N(val(2),0.05);
                  U(0,100)]);
```

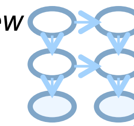
User Queries

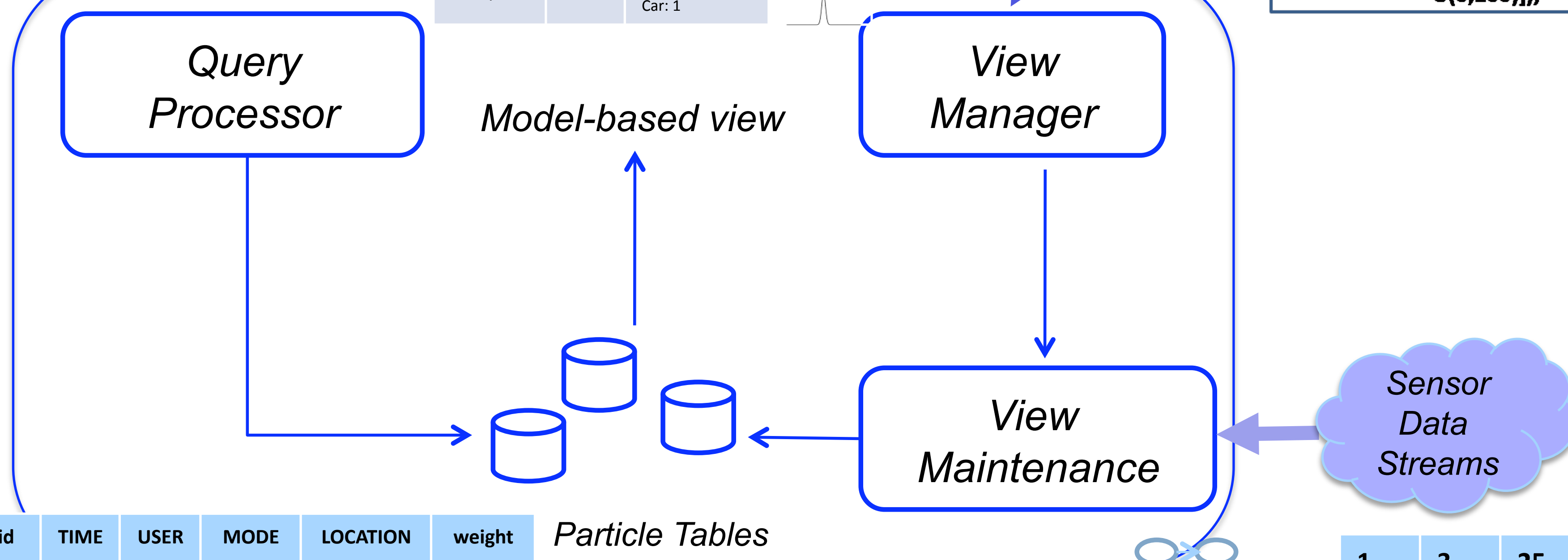
SELECT time, user, location
FROM dpmview
WHERE mode = "W"
WITH CONFIDENCE 0.95

TIME	USER	MODE (INFERRED)	Location (INFERRED)
5pm	John	Walking: 0.9 Car: 0.1	
5pm	Jane	Walking: 0.9 Car : 0.1	
5:05pm	John	Walking: 0 Car: 1	

View Creation

CREATE VIEW dpmview
DPM hmm.dpm
STREAM sensors

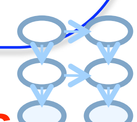




1	3	25

id	TIME	USER	MODE	LOCATION	weight
1	5pm	John	W	(a1,b1)	0.01
2	5pm	John	W	…	0.02

e.g. using particle filters



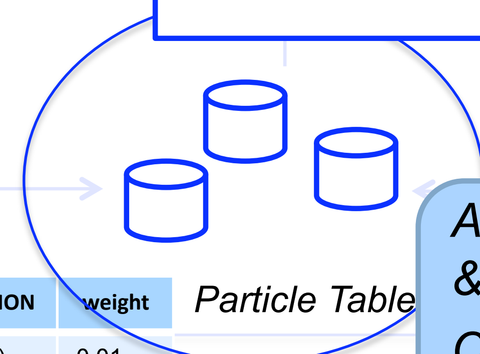
Query Processing

User Queries

SELECT time, user, location
FROM dpmview
WHERE mode = "W"
WITH CONFIDENCE 0.95

*SELECT time, user, SUM(location*weight)*
FROM particles p1
GROUP BY time
HAVING
0.95 <

SELECT SUM(weight) FROM particles p2
WHERE p1.time = p2.time AND status = "W"



id	TIME	USER	MODE	LOCATION	weight
1	5pm	John	W	(a1,b1)	0.01
2	5pm	John	W	…	0.02

Able to support single-table select, project & aggregate queries

Can reason about spatial correlations

However, temporal correlations ignored

Outline

- Motivation
- Statistical modeling of sensor data
  - Abstraction of *model-based views*
  - Regression-based views
  - Views based on dynamic Bayesian networks
- Query processing over model outputs
- Some interesting sensor network problems
  - Model-driven data acquisition
  - Distributed inference in sensor networks

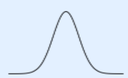
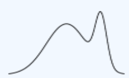
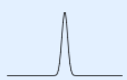
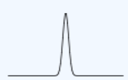
Representing and Querying Correlated Tuples in Probabilistic Databases; P. Sen, A. Deshpande; ICDE 2007
Efficient Query Evaluation over Temporally Correlated Probabilistic Streams; B. Kanagal, A. Deshpande; ICDE 2009
Shared Correlations in Probabilistic Databases; P.Sen, A. Deshpande, L. Getoor, VLDB 2008

Querying Model Outputs

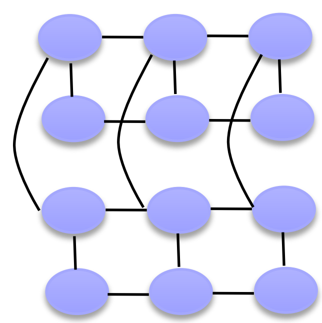
- Challenges:
  - The model outputs typically probabilistic
  - Strong spatial and temporal correlations
  - Continuous queries over streaming data
- Numerous approaches proposed in recent years
  - Typically make strong independence assumptions
  - Limited support for attribute-value uncertainty
  - In spite of that, query evaluation known to be #P-Hard
- Our goal: Develop a general, uniform framework that…
  - Captures both tuple-existence and attribute-value uncertainties
  - Can reason about correlations in the data
  - Can handle continuous queries over probabilistic streams

Overview of Our Approach

- Represent the uncertainties and correlations *graphically* using small functions called *factors*
  - Concepts borrowed from the *graphical models* literature

TIME	USER	MODE (inferred)	LOCATION (inferred)
5pm	John	Walking: 0.9 Car: 0.1	
5pm	Jane	Walking: 0.9 Car : 0.1	
5:05pm	John	Walking: 0.1 Car: 0.9	
5:05pm	Jane	Walking: 0.1 Car: 0.9	
…	…	…	…

TIME	USER	MODE (inferred)	LOCATION (inferred)
5pm	John	M^{5pm}_{John}	L^{5pm}_{John}
5pm	Jane	M^{5pm}_{Jane}	L^{5pm}_{Jane}
5:05pm	John	$M^{5:05pm}_{John}$	$L^{5:05pm}_{John}$
5:05pm	Jane	$M^{5:05pm}_{Jane}$	$L^{5:05pm}_{Jane}$
…	…	…	…



M^{5pm}_{John}	M^{5pm}_{Jane}	f()
W	W	1
W	C	0
C	W	0
C	C	1

Overview of Our Approach

- Represent the uncertainties and correlations *graphically* using small functions called *factors*
  - Concepts borrowed from the *graphical models* literature

S

	A	B	prob
s1	‘m’	1	0.6
s2	‘n’	1	0.5

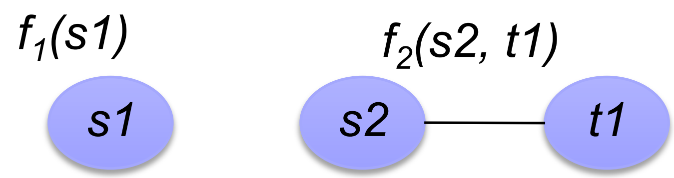
T

	C	D	prob
t1	1	‘p’	0.4

s1	***f_1(s1)***
0	0.6
1	0.4

s2	***t1***	***f_2(s2, t1)***
0	0	0.1
0	1	0.5
1	0	0.4
1	1	0

s2 and t1 mutually exclusive



Overview of Our Approach

- During query processing, add new factors corresponding to intermediate tuples
- Example query: $\pi_D(S \bowtie_{B=C} T)$

S

	A	B
s1	'm'	1
s2	'n'	1

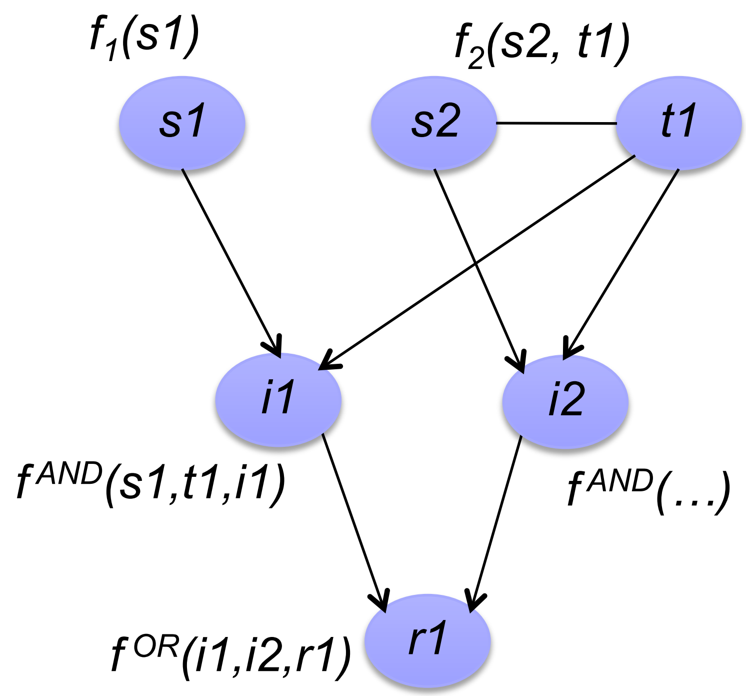
T

	C	D
t1	1	'p'

	A	B	C	D
i1	'm'	1	1	'p'
i2	'n'	1	1	'p'

π_D

	D
r1	'p'



Overview of Our Approach

See Prithvi's talk for more details

- Query evaluation ≡ Inference !!
  - Can use standard techniques like *variable elimination*
  - Can exploit the structure in probabilistic databases for scalable inference

S

	A	B
s1	'm'	1
s2	'n'	1

T

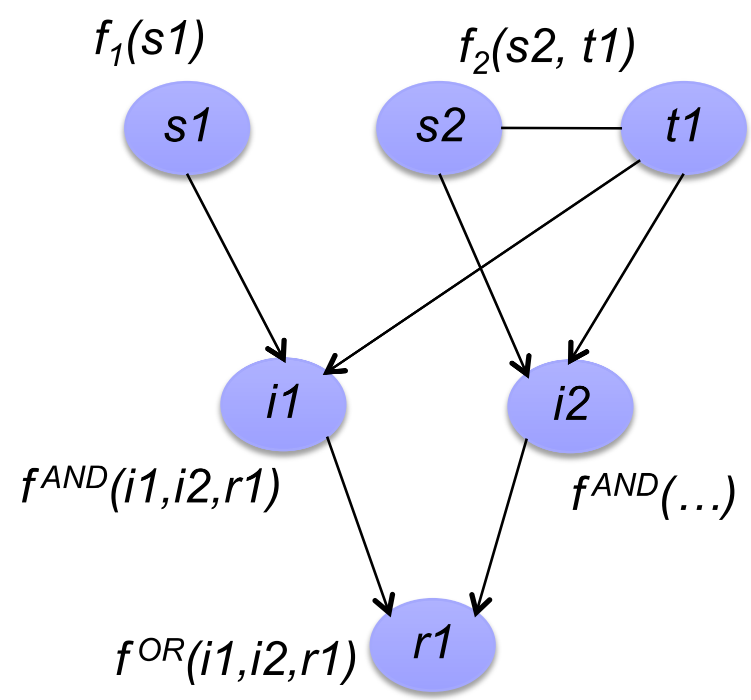
	C	D
t1	1	'p'

$\bowtie$

	A	B	C	D
i1	'm'	1	1	'p'
i2	'n'	1	1	'p'

π_D

	D
r1	'p'



Querying Probabilistic Streams

- Need to support "continuous" queries over "sliding windows"
  - *"alert me when the number of people in a mall exceeds 1000"*
    - Must take spatial correlations into account
  - *"how many people drove for at least one hour yesterday"*
    - Can't ignore the temporal correlations in the data
- Observations:
  - Probabilistic streams typically obey "Markovian" property
    - *Variables at times "t" and "t+2" are independent given the values of the variables at time "t+1"*
  - Although the actual parameters change, the correlation "structure" remains unchanged across time
    - *At every instance, we get the same set of input factors with different probability numbers*

Querying Probabilistic Streams

- Brief summary of the key ideas:
  - Extend the query language to support MAP (using Viterby's algorithm) and ML operations over probabilistic streams
  - Augment the "schema" of the probabilistic streams to include the correlation structure
  - Implement the operators to support the *iterator* interface
    - Only the parameters are transferred from operator to operator
    - Enables efficient, incremental processing of new inputs
  - Choose query plans that postpone generation of intermediate non-Markovian streams as long as possible

Ongoing and Future Work

- Developing APIs for adding arbitrary models
  - Minimize the work of the model developer
  - Identify intermediate representations useful across classes of models
- Designing index structures for querying, updating large collections of uncertain facts
- Approximate inference techniques for more efficient query processing

Outline

- Motivation
- Statistical modeling of sensor data
  - Abstraction of *model-based views*
  - Regression-based views
  - Views based on dynamic Bayesian networks
- Query processing over model outputs
- Some interesting sensor network problems
  - Model-driven data acquisition
  - Distributed inference in sensor networks

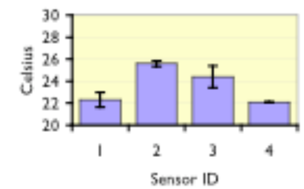
Model-Driven Data Acquisition in Sensor Networks; A. Deshpande et al., VLDB 2004

Model-based Query Processing

<u>Declarative Query</u>
Select nodeID,
temp ± .1C, conf(.95)
Where nodeID in {1..6}

<u>USER</u>

<u>Query Results</u>
1, 22.73, 100%
…
6, 22.1, 99%

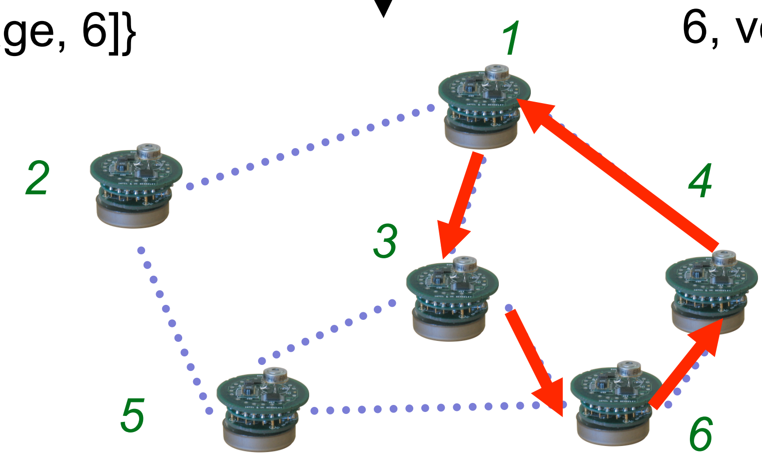


Probabilistic Model

Query Processor

<u>Observation Plan</u>
{[temp, 1],
[voltage, 3],
[voltage, 6]}

<u>Data</u>
1, temp = 22.73,
3, voltage = 2.73
6, voltage = 2.65



<u>SENSOR NETWORK</u>

Model-based Query Processing

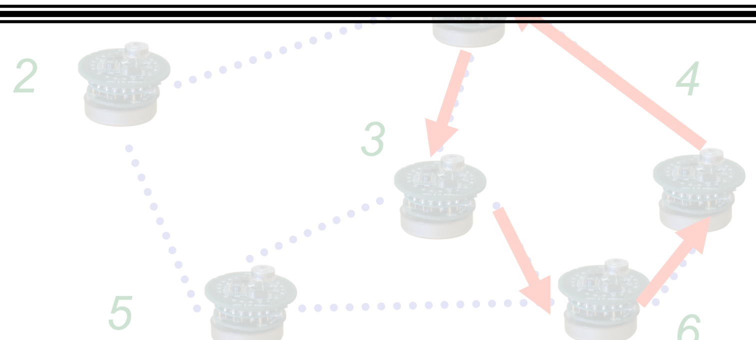
Declarative Query
Select nodeID,
temp ± .1C, conf(.95)
Where nodeID in {1..6}

USER

Query Results
1, 22.73, 100%
…
6, 22.1, 99%

Advantages:

- Exploit correlations for efficient approximate query processing
- Handle noise, biases in the data
- Predict missing or future values



SENSOR NETWORK

Model-based Query Processing

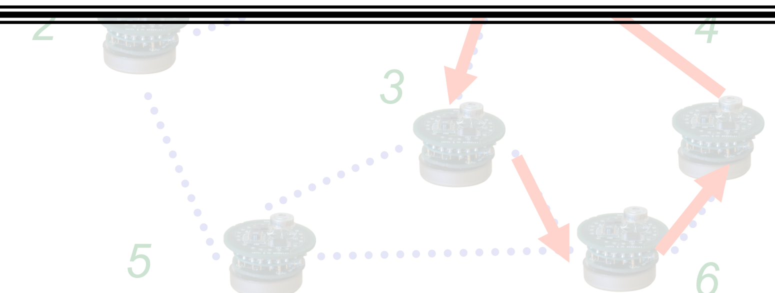
Declarative Query
Select nodeID,
temp ± .1C, conf(.95)
Where nodeID in {1..6}

USER

Query Results
1, 22.73, 100%
…
6, 22.1, 99%

Many interesting research challenges:

- Finding optimal data collection paths
- Different type of queries (max/min, top-k)
- Learning, re-training models
- Long-term planning, Continuous queries
- …



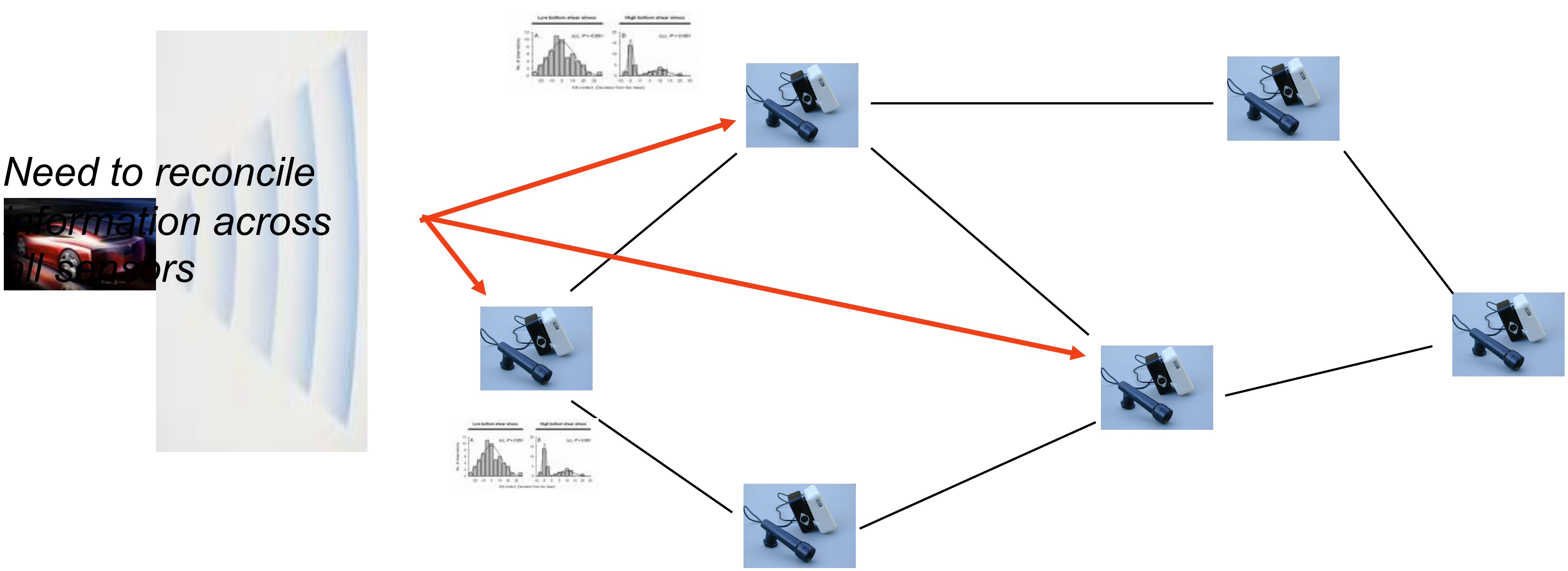
SENSOR NETWORK

Outline

- Motivation
- Statistical modeling of sensor data
  - Abstraction of *model-based views*
  - Regression-based views
  - Views based on dynamic Bayesian networks
- Query processing over model outputs
- Some interesting sensor network problems
  - Model-driven data acquisition
  - Distributed inference in sensor networks

Distributed, In-network Inference

- Often need to do in-network, distributed inference
  - Target tracking through information fusion
  - Optimal control (for actuation)
  - Distributed sensor calibration (using neighboring sensors)
  - In-network regression or function fitting

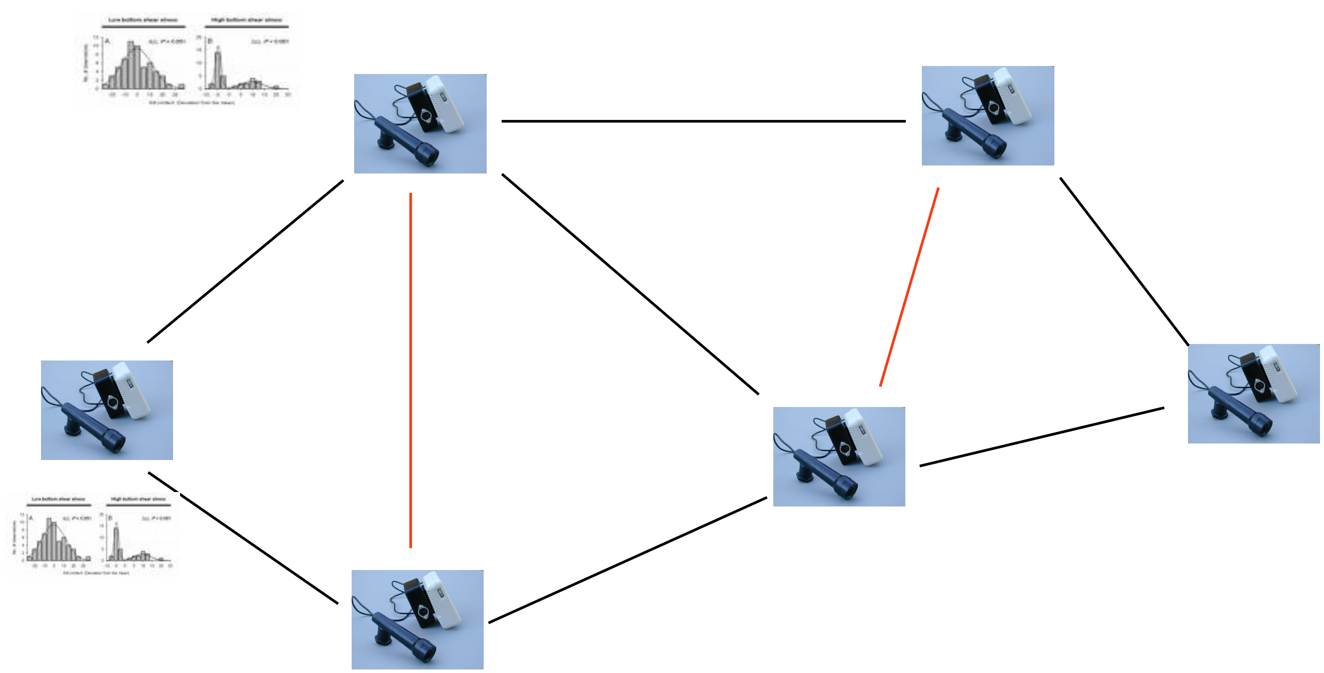


Distributed, In-network Inference

- Often need to do in-network, distributed inference
  - Target tracking through information fusion
  - Optimal control (for actuation)
  - Distributed sensor calibration (using neighboring sensors)
  - In-network regression or function fitting
- Obey a common structure:
  - Each sensor has/observes some *local* information
  - Information across sensors is correlated
    - … must be combined together to form a *global picture*
  - The global picture (or relevant part thereof) should be sent to each sensor

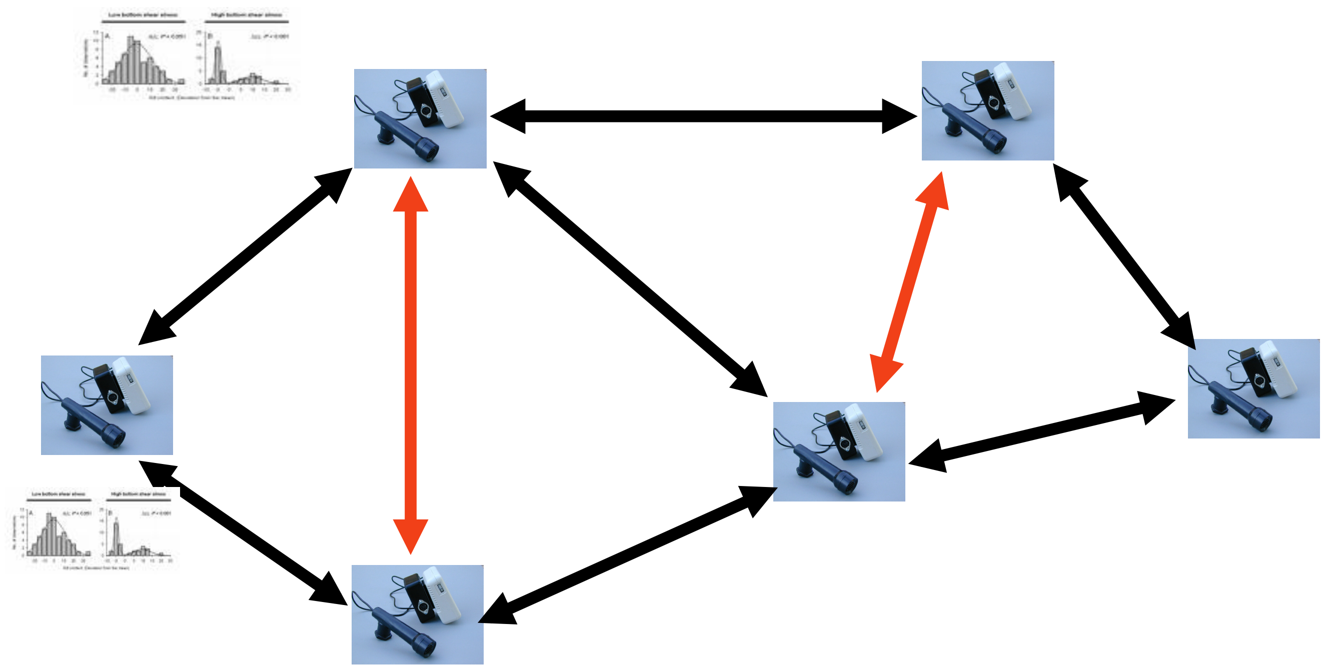
Distributed, In-network Inference

- Naïve option:
  - Collect all data at the centralized base station – too expensive
- Using graphical models
  - Form a junction tree on the nodes directly
  - Use message passing/loopy propagation for globally consistent view



Distributed, In-network Inference

- Naïve option:
  - Collect all data at the centralized base station – too expensive
- Using graphical models
  - Form a junction tree on the nodes directly
  - Use message passing/loopy propagation for globally consistent view



Conclusions

- Increasing number of applications generate and need to process uncertain data
- Statistical/probabilistic modeling provide an elegant framework to handle such data
  - But little support in current database systems
- MauveDB
  - Supports the abstraction of Model-based User Views
  - Enables declarative querying over noisy, imprecise data
  - Exploits commonalities *to define, to create, and to process queries* over such views

Conclusions

- Prototype implementation
  - Using the Apache Derby open source DBMS
  - Supports *Regression-, Interpolation-, and DBN-based views*
  - Supports many different view maintenance strategies
- Probabilistic databases
  - Increasingly important research area
  - Designed a uniform and general framework for representing and querying uncertain data with correlations
  - New inference techniques that exploit the structure in probabilistic databases

Thank you !!

- Questions ?