SIGMOD'03
Evaluating Probabilistic Queries over Imprecise Data

Reynold Cheng, Dmitri V. Kalashnikov, Sunil Prabhakar

Department of Computer Science, Purdue University
http://www.cs.purdue.edu/place/

Sensor-Based Applications

- Sensors monitor external environment continuously
- Sensor readings are sent back to the application
- Decisions are made based on these readings
  - A moving object database monitors locations of mobile devices
  - An air-conditioning system uses temperature sensors to adjust the temperature of each room
  - Sensors are used to detect if hazardous materials are present and how they are spreading

Data Uncertainty

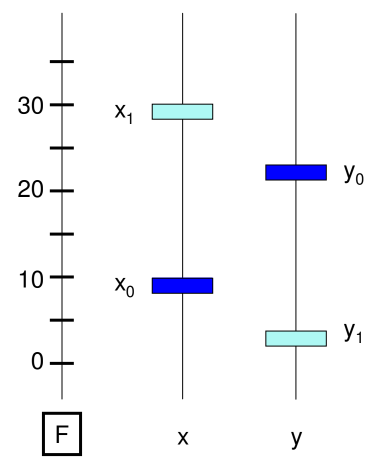
- A database/server collects readings from sensors
- The database cannot contain the exact status of an entity being monitored at every point in time
  - Limited network bandwidth
  - Scarce battery power
- Readings are sent periodically, or on-demand
- The value of entity being monitored (e.g., temperature, location) keeps changing
- At most points of the time the database stores obsolete sensor values

Answering a Minimum Query with Database Readings

Recorded Temperature

Current Temperature

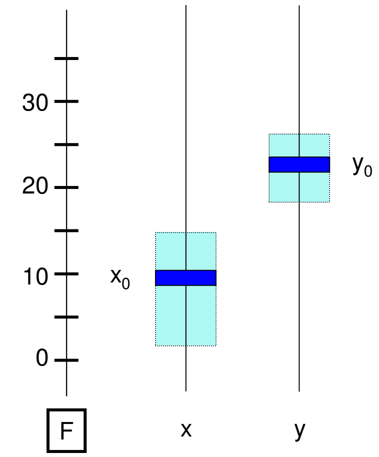
- $x_0 < y_0$: x is minimum
- $y_1 < x_1$: y is minimum
- Wrong query result



Answering a Minimum Query with Error-Bounded Readings

Recorded Temperature

Bound for Current Temperature

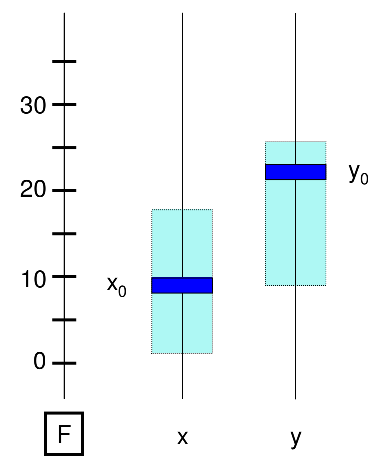


- x certainly gives the minimum temperature reading

Answering a Minimum Query with Error-Bounded Readings

Recorded Temperature

Bound for Current Temperature



- Both *x* and *y* have a chance of yielding the minimum value
- Which one has a higher probability?

Probabilistic Queries

- If the sensor value cannot change drastically over a short period of time, we can:
  1. place lower and upper bounds on the possible values
  2. define probability distribution of values within the bound
- Evaluate probability for query answers, e.g.,
  - x: 70% chance for yielding the minimum value
  - y: 30% chance for yielding the minimum value
- Probabilistic queries give us a correct (possibly less precise) answer, instead of a potentially incorrect answer

Related Work

- Few research papers discuss the evaluation of a query answer in probabilistic form
- Wolfson et al. [WS99] discussed probabilistic range queries for moving objects
- Our previous work [CPK03] presented an algorithm for evaluating probabilistic nearest neighbor query for moving objects
- Both papers only address queries in a moving object database model
- Olston and Widom [OW02] discussed tradeoff between precision and performance of querying replicated data

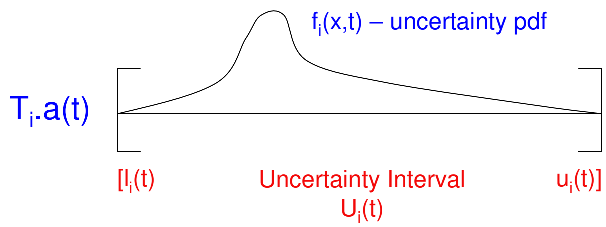
Our Contributions

- A generic uncertainty model that is applicable to any database recording imprecise values
- Classification of probabilistic queries
- Evaluation and quality of probabilistic queries
- An experimental study of proposed methods

Database Model

Param	Meaning
T	A set of database objects (e.g., sensors)
a	Dynamic attribute (e.g., temperature)
T_i	i^{th} object of T
$T_i.a(t)$	Value of a in T_i (e.g., temperature of a sensor) at time t

Generic Uncertainty Model



- Example: moving object uncertainty [WS99]
- Can be extended to *n* dimensions

Classification of Probabilistic Queries

1. Nature of answer
   - **Value-based:** returns a single value e.g., average query ([l,u], pdf)
   - **Entity-based:** returns a set of objects e.g., range query ($\{(T_i,p_i), p_i>0\}$)
2. Aggregation
   - **Non-aggregate:** whether an object satisfies a query is independent of others e.g., range query
   - **Aggregate:** interplay between objects decides result e.g., nearest neighbor query

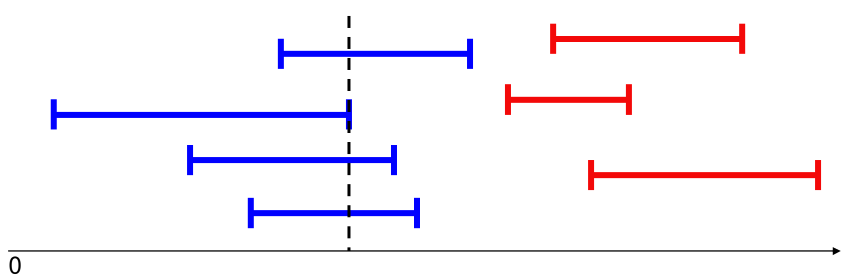
Classification of Probabilistic Queries

	Value-based answer	*Entity-based answer*
Non-aggregate	VSingleQ What is the temperature of sensor x?	ERQ Which sensor has temperature between 10F and 30F?
Aggregate	VAvgQ, VSumQ, VMinQ, VMaxQ What is the average temperature of the sensors?	ENNQ, EMinQ, EMaxQ Which sensor gives the highest temperature?

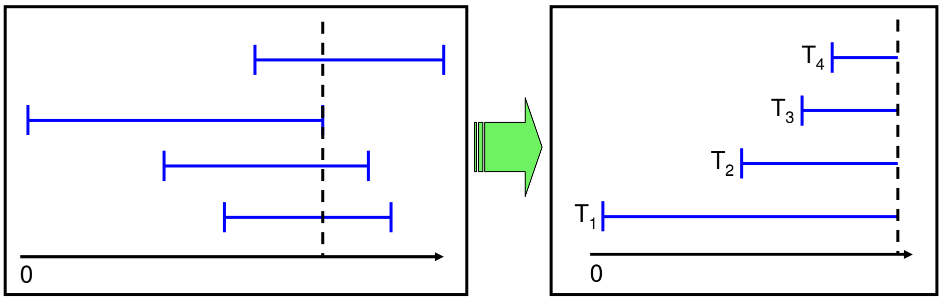
- We develop query evaluation algorithms and quality metrics for each class.

EMinQ Step 1: Interval Elimination

- Returns a set of tuples (T_i, p_i), where p_i is the (non-zero) probability that $T_i.a$ is the minimum value of *a* among all objects in T
- Eliminate objects that have zero probability of yielding the minimum value

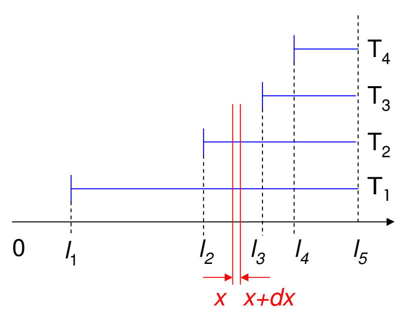


EMinQ Step 2: Sorting Uncertainty Interval



- Cut off portions that are beyond the "upper limit"
- Sort intervals using lower bounds
- Rename objects as T_1, T_2, T_3, T_4 in ascending order of lower bounds

Evaluating probability of T_2



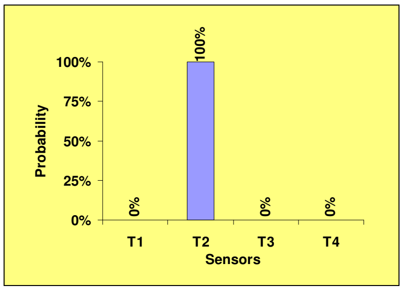
- If $T_2.a \in [l_2, l_3]$, $T_2.a$ is the min with probability $\int_{l_2}^{l_3} f_2(x) \bullet (1 - P_1(x))dx$
- p_2 is given by:

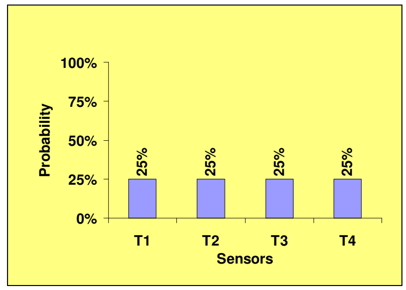
$$\int_{l_2}^{l_3} f_2(x) \bullet (1-P_1(x))dx + \int_{l_3}^{l_4} f_2(x) \bullet \prod_{k=1,3} (1-P_k(x))dx + \int_{l_4}^{l_5} f_2(x) \bullet \prod_{k=1,3,4} (1-P_k(x))dx$$

Quality of Probabilistic Result

- Notion of answer "quality"

"Which sensor, among 4, has the minimum reading?"
(assuming only 1 answer exists, if values are known precisely)



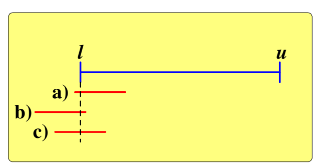


- Proposed metrics for different classes of queries

Answer Quality for Range Queries

"Is reading of sensor i in range [l,u] ?"

- regular range query
  - "yes" or "no" with 100%
- probabilistic query ERQ



a) yes with p_i = 95%: **OK**
b) yes with p_i = 5%: **OK** (95% it is not in [l, u])
c) yes with p_i = 50%: **not OK** (not certain)

$$Score = \frac{|p_i - 0.5|}{0.5}$$

$$Score\_of\_an\_ERQ = \frac{1}{|R|}\sum_{i \in R}\frac{|p_i - 0.5|}{0.5}$$

Quality for E- Aggr. Queries (1/2)

"Which sensor, among n, has the minimum reading?"

- Recall
  - Result set R = {(Ti, pi)}
    - e.g. {(T1, 30%), (T2, 40%), (T3, 30%)}
  - B is interval, bounding all possible values
    - e.g. minimum is somewhere in B = [10,20]
- Our metrics for aggr. queries Min, Max, NN
  - objects cannot be treated independently as in ERQ metric
  - uniform distribution (in result set) is the worst case
  - metrics are based on **entropy**

Quality for E- Aggr. Queries (2/2)

- $H(X)$ entropy of r.v. X ($X_1, \ldots, X_n$ with $p(X_1), \ldots, p(X_n)$)

$$H(X) = \sum_{i=1}^{n} p(X_i) \log_2 \frac{1}{p(X_i)}$$

  - entropy is smallest (i.e., 0) iff $\exists\, i : p(X_i) = 1$
  - entropy is largest (i.e., $\log_2(n)$) iff all X_i's are equally likely
- Our metric:

$$Score\_of\_Entity\_Aggr\_Query = -H(R) \times |B|$$

- Score is good (high) if
  - entropy is low (small uncertainty)
  - the width of B is small

Scores for Value- Aggr. Queries

"What is the minimum value among n sensors?"

- Recall
  - result is: l, u, {p(x) : x ∈ [l,u]}
    - e.g. minimum is in [10,20], p(x) ~ U[10,20]
- Differential entropy

$$H(X) = -\int_{l}^{u} p(x)\log_2 p(x)dx$$

  - Measures uncertainty associated with r.v. X with pdf p

$$Score\_of\_Value\_Aggr\_Query = -H(X)$$

Improving Answer Quality

- Given uncertainty, the quality of the initial answer may be unsatisfactory
- To improve quality
  - server can request updates from specific sensors
- Due to limited resources
  - important to choose right sensors to update
  - use update policies

Update Policies

- Global choice (among all sensors)
  - Glb_RR – pick random
- Local choice (among the relevant sensors)
  - Loc_RR – pick random
  - MinMin – pick such T_i that with min uncert. lower bound
  - MaxUnc – pick T_i with max uncertainty

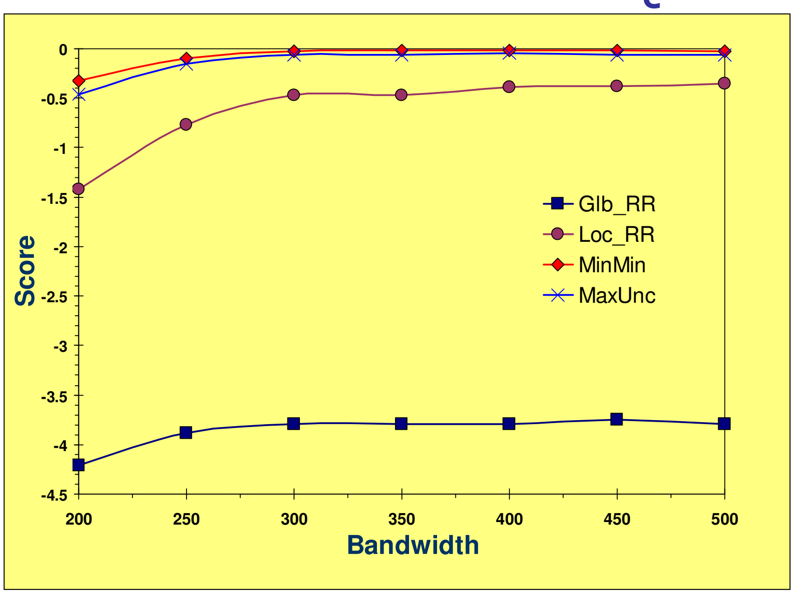
Experiments: Simulation

- Discrete event simulation
  - 1 server
  - 1000 sensors
  - limited network bandwidth
  - "Min" queries tested

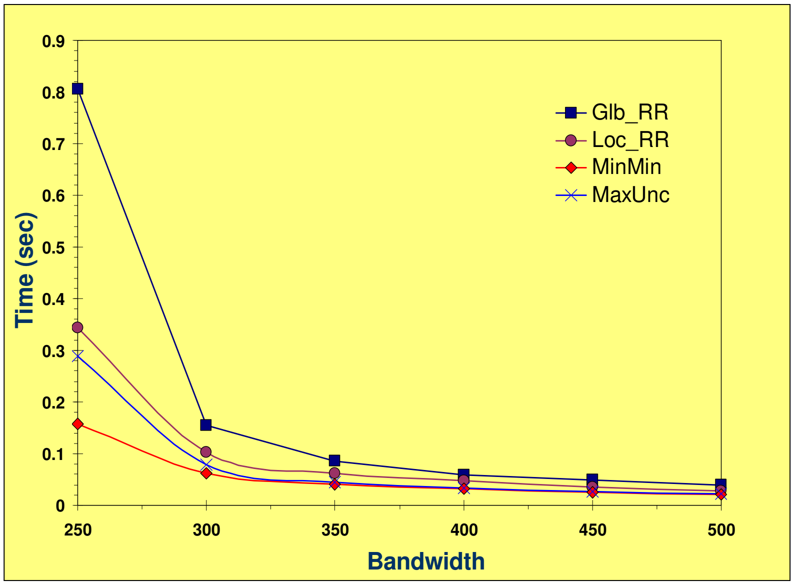
Experiments: Uncertainty Model

- Uncertainty model
  - Sensor sends update at time t_0
    - time t_0
    - current value a_0
    - rate of uncertainty growth r_0
    - at time t, a is uniform in its uncertainty interval
- Queries
  - arrival ~ Poisson(λ)
  - each over a random subset of 100 sensors

Effect of Bandwidth on EMinQ Score



Effect of Bandwidth on Response Time



Conclusions

We proposed:

a) probabilistic queries for handling inherent uncertainty in sensor databases
b) a flexible model of uncertainty defined
c) a classification of probabilistic queries
d) algorithms for computing typical queries in each class
e) metrics for quantifying the quality of answers to probabilistic queries for each class
f) various update heuristics to improve answer quality under resource constraints

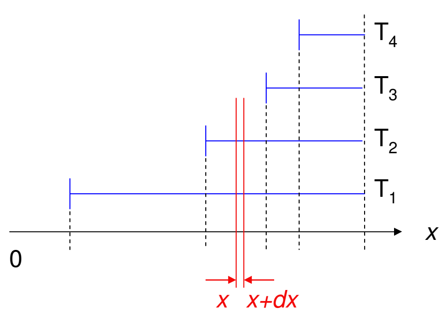
Contact Information

Reynold Cheng
www.cs.purdue.edu/homes/ckcheng
ckcheng@cs.purdue.edu

Dmitri V. Kalashnikov
www.ics.uci.edu/~dvk
dvk@ics.uci.edu

Sunil Prabhakar
www.cs.purdue.edu/homes/sunil
sunil@cs.purdue.edu

EMinQ Step 3: Evaluating p_i for T_i



- Let $f_2(x)$ be the pdf of T_2.a
- If T_2.a $\in [x, x+dx]$, T_2.a is the minimum iff T_1.a > T_2.a with the probability $f_2(x) * (1-P_1(x))\, dx$
- $P_i(x)$ is the cumulative probability density function of T_i.a

References

1. **[WS99]** O. Wolfson and A. Sistla. Updating and Querying Databases that Track Mobile Units. In *Distributed and Parallel Databases*, 7(3), 1999.
2. **[CPK03]** R. Cheng, S. Prabhakar and D. V. Kalashnikov. Querying imprecise data in moving object environments. In *Proc. of the 19th IEEE ICDE,* India, 2003.
3. **[OW03]** C. Olston and J. Widom. Best-effort cache synchronization with source cooperation. In *Proc. Of the ACM SIGMOD 2002.*

Effect of Bandwidth on Uncertainty