# Query Selectivity Estimation for Uncertain Data

Sarvjeet Singh[*], Chris Mayfield[*], Rahul Shah[#], Sunil Prabhakar[*] and Susanne Hambrusch[*]

[*] Department of Computer Science, Purdue University

[#] Department of Computer Science, Louisiana State University

**Presented by: Reynold Cheng**

Department of Computer Science, University of Hong Kong

ckcheng@cs.hku.hk

SSDBM 2008

# Motivation: Data Uncertainty

- Many applications need to handle uncertain data
  - Example: Sensor networks, Location-based applications, Data integration, Biological databases
- Existing databases do not provide direct support for uncertain data. Two simple options:
  - manage uncertain data outside DBMS, or
  - remove uncertainty from data
- However, there is need for managing uncertain data at the database level

# Motivation (cont.)

- DBMS have been proposed to handle uncertain data
  - Examples: Orion, Trio, MystiQ, MayBMS
- Probabilistic queries: Queries over uncertain data return answers with probabilities
  - Results with low probability of occurrence are often not desirable or meaningful
- Probabilistic Threshold Queries: Return only those answers that exceed a specified threshold

# Motivation (cont.)

- Query optimization is important
  - An essential ingredient is the ability to estimate cost of a given query plan
- New indexes have been proposed for uncertain data
  - Their effective use needs a reasonable estimate of query selectivity
  - Optimizer needs to know when to use the indexes
- **Our Contribution**

  **Efficient algorithms for selectivity estimation of probabilistic threshold queries over uncertain data**

# Related Work

- Selectivity estimation for traditional relational databases is well studied [SIGMOD96]

- Models for uncertain data
  - Attribute Uncertainty [SIGMOD03, ICDE08]
  - Tuple Uncertainty [VLDB04b, VLDB06]

- Uncertainty management systems
  - Orion [Orion], Trio [CIDR05], Mystiq [SIGMOD05], MayBMS [ICDE07], [ICDE07b]

# Related Work

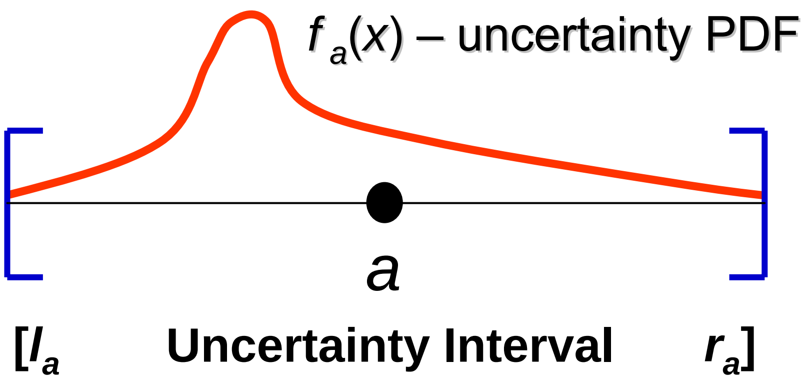
- Efficient evaluation of probabilistic queries
  - Prob. range queries [VLDB04a,VLDB04b]
  - Prob. threshold indexing [VLDB04a]
  - Prob. NN queries [SIGMOD03, ICDE07c]

- **Selectivity estimation for probabilistic threshold queries has not been addressed before**

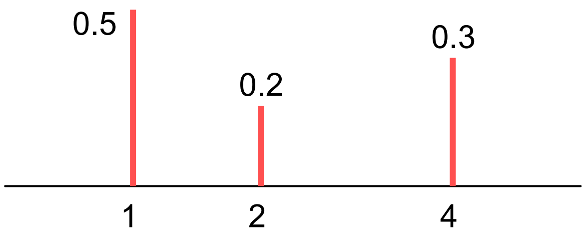
# Outline

- Motivation
- Uncertainty Model
- Selectivity estimation using Histogram
  - Unbounded Range Queries
  - General Range Queries
- Selectivity estimation using Slabs
- Experiments
- Conclusion

# Uncertainty Model




Continuous




Discrete

- Attribute Uncertainty: An uncertain attribute *a* consists of an *Uncertainty Interval* [$l_a$, $r_a$] and a pdf $f_a(x)$ (cdf $F_a(x)$) over the interval

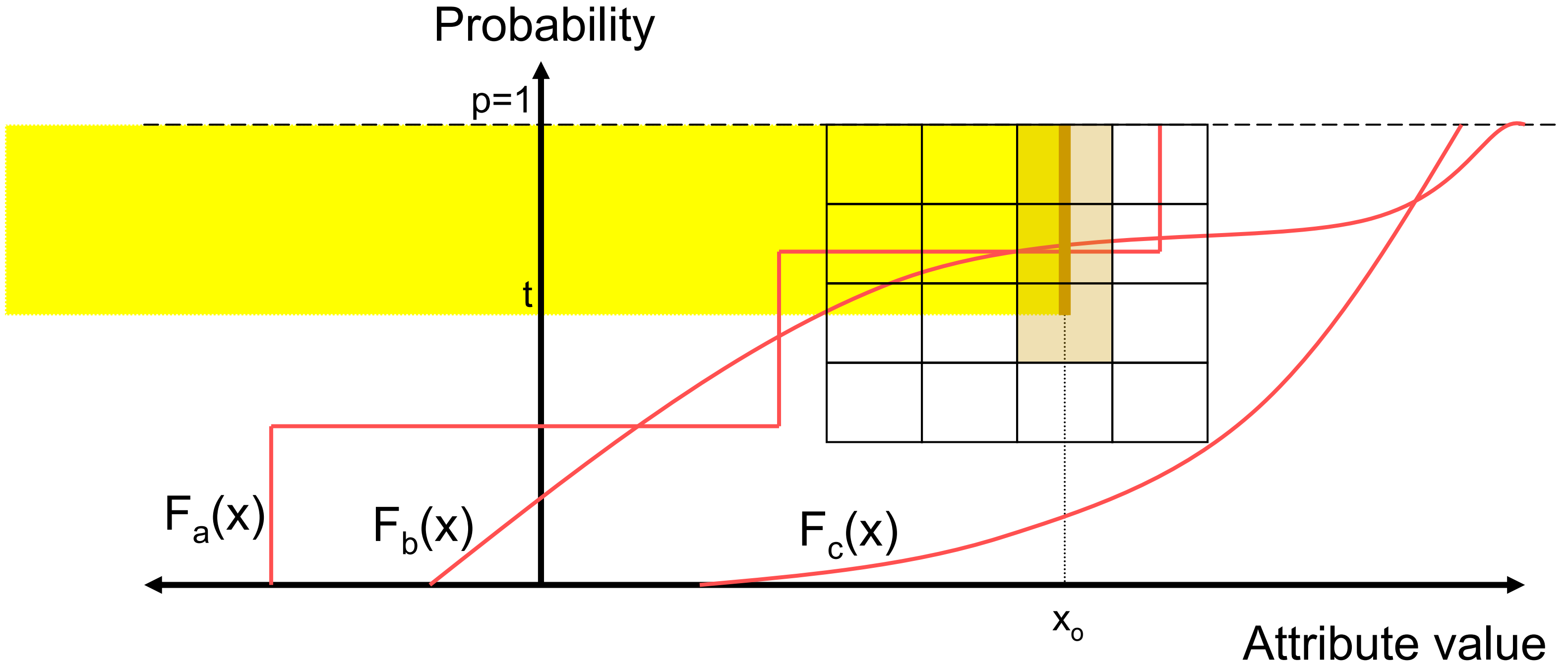
- Our techniques are also applicable to Tuple Uncertainty

# Outline

- Motivation
- Uncertainty Model
- Selectivity estimation using Histogram
  - Unbounded Range Queries
  - General Range Queries
- Selectivity estimation using Slabs
- Experiments
- Conclusion

# Selectivity of Unbounded Range Queries

$$a <_t x_o: \Pr(a < x_o) > t \Leftrightarrow \int_{-\infty}^{x_0} f_a(x)\,dx > t \Leftrightarrow F_a(x_o) > t$$



# General range queries

- General range query

  $\Pr(x_1 < a < x_2) > t \Leftrightarrow F_a(x_2) - F_a(x_1) > t$
- Instead of a 2D cdf curve, we can now plot a 3D curve for each uncertain data item:

$$G_a(x_1,x_2) = \int_{x_1}^{x_2} f_a(x)\,dx = F_a(x_2) - F_a(x_1)$$

- The algorithm is similar to the unbounded case
- Optimizations reducing construction time are possible (see paper)

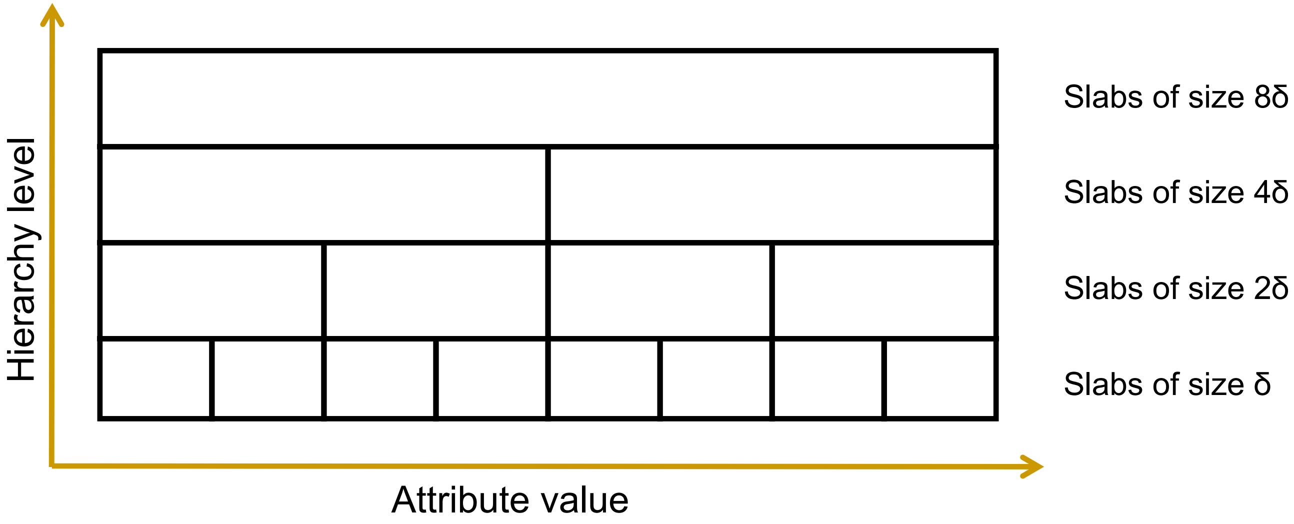
# Outline

- Motivation
- Uncertainty Model
- Selectivity estimation using Histogram
  - Unbounded Range Queries
  - General Range Queries
- Selectivity estimation using Slabs
- Experiments
- Conclusion

# General Range Queries

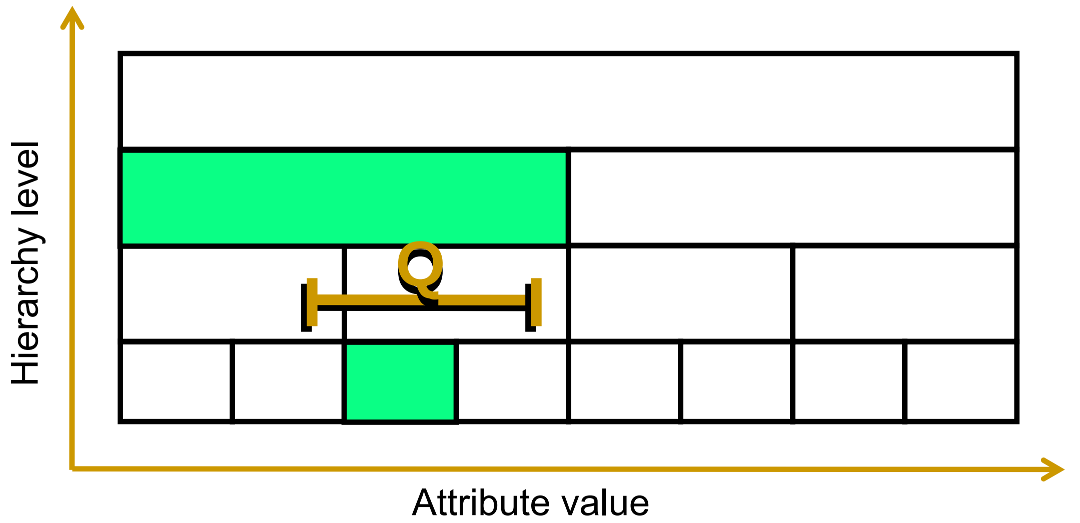
- Histogram approach for general range queries
  - Provides very good selectivity estimate
  - Initial construction time is quadratic in terms of range of input data
- General Range Queries using Slabs
  - Provides a better space-time complexity than histogram technique
  - Has a lower accuracy (in general)

# General Range queries using Slabs




- A slab $S(x_1,x_2,t)$ stores the selectivity of query $Q(x_1,x_2,t)$
- We define a hierarchy of slabs, with the size of slabs increasing exponentially
- Space and construction time complexity of this approach is linear in terms of range of input data

# Selectivity estimation using Slabs




- Given a query $Q(x_1,x_2,t)$, we find pairs of slabs that contains (over-estimate) and is contained (under-estimate) by the query
- We linearly interpolate the two estimates to get the final estimate

# Outline

- Motivation
- Uncertainty Model
- Selectivity estimation using Histogram
  - Unbounded Range Queries
  - General Range Queries
- Selectivity estimation using Slabs
- Experiments
- Conclusion

# Experiments

- We implemented our selectivity estimation techniques in Orion (probabilistic extension of PostgreSQL)
- Synthetic Datasets: Each dataset of random sensor readings with uniform distribution [CIKM06, VLDB04a]
    - The intervals are distributed uniformly in [0,1000]
    - Interval sizes are distributed normally
    - Database size is 250,000

# Effect on Query Plan

**PostgreSQL Query plan**

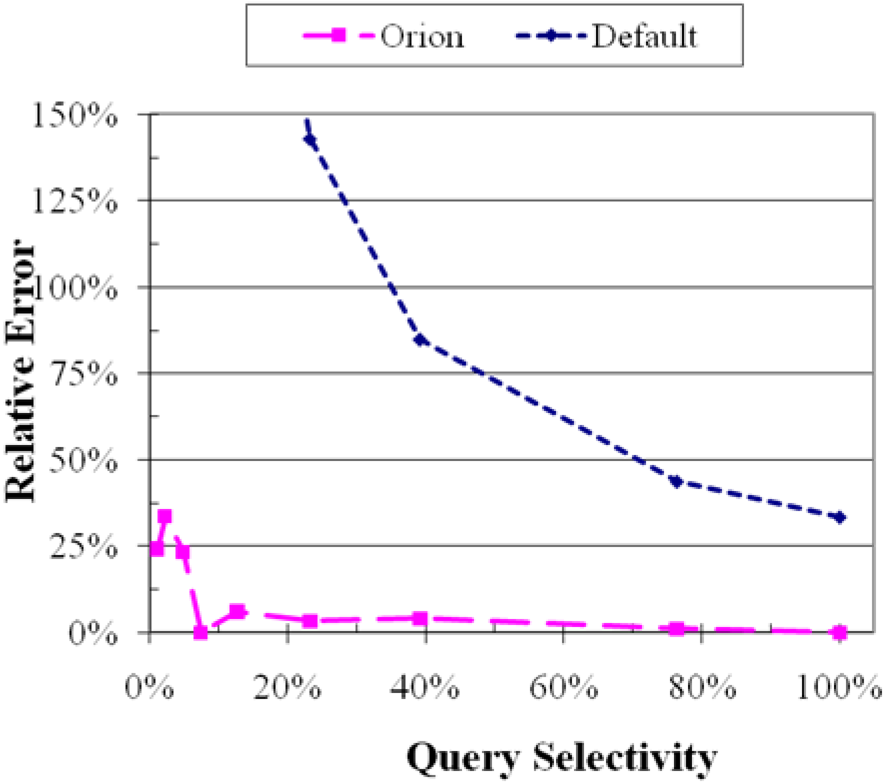
- **Without** any selectivity estimate function, PostgreSQL assumes a default (fixed) selectivity. In practice, it favors the use of un-clustered indexes

```
SELECT * FROM Readings WHERE value < 750;
-----------------------------------------
Bitmap Heap Scan on Readings
 (cost=742.33..4075.67 rows=66667 width=36)
 (actual=20379.348..20824.652 rows=153037)
 Recheck Cond: (value < 750::real)
->  Bitmap Index Scan on pti_value
 (cost=0.00..742.33 rows=66667 width=0)
 (actual=20378.677..20378.677 rows=153K)
 Index Cond: (value < 750::real)
```

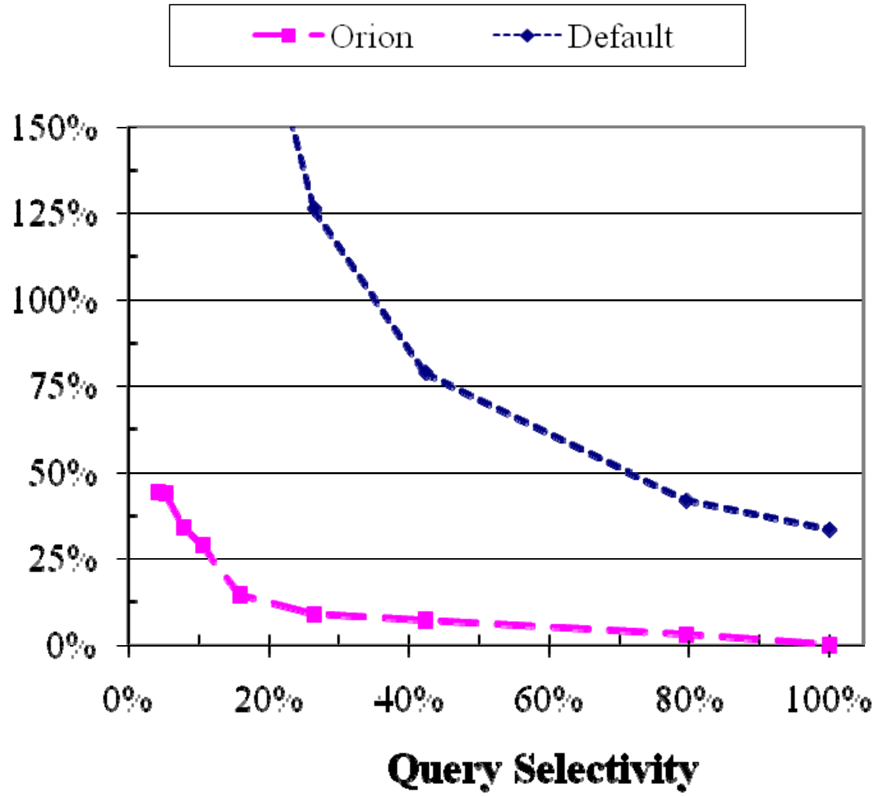
- **With** our algorithms in place, PostgreSQL correctly picks the query plan with lower I/O cost

```
(same query as before, but using our algorithms)
------------------------------------------------
 Seq Scan on Readings
   (cost=0.00..5000.00 rows=164333 width=35)
   (actual=83.841..15545.401 rows=153037)
  Filter: (value < 750::real)
```

# Accuracy at Varying Selectivities

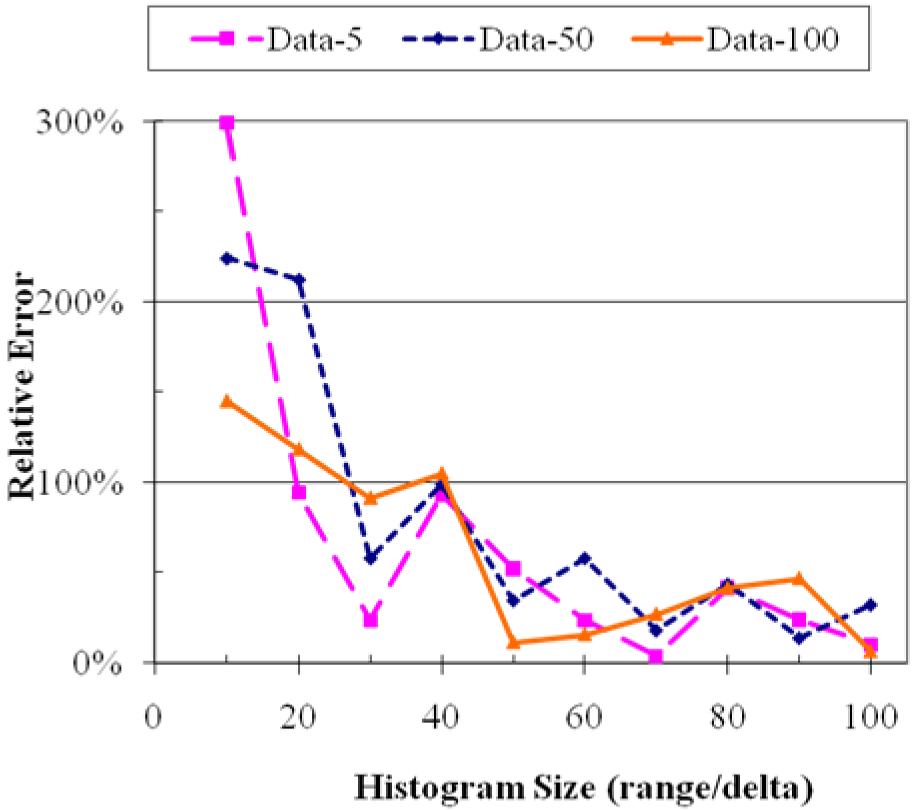



**Selectivities (2D)**

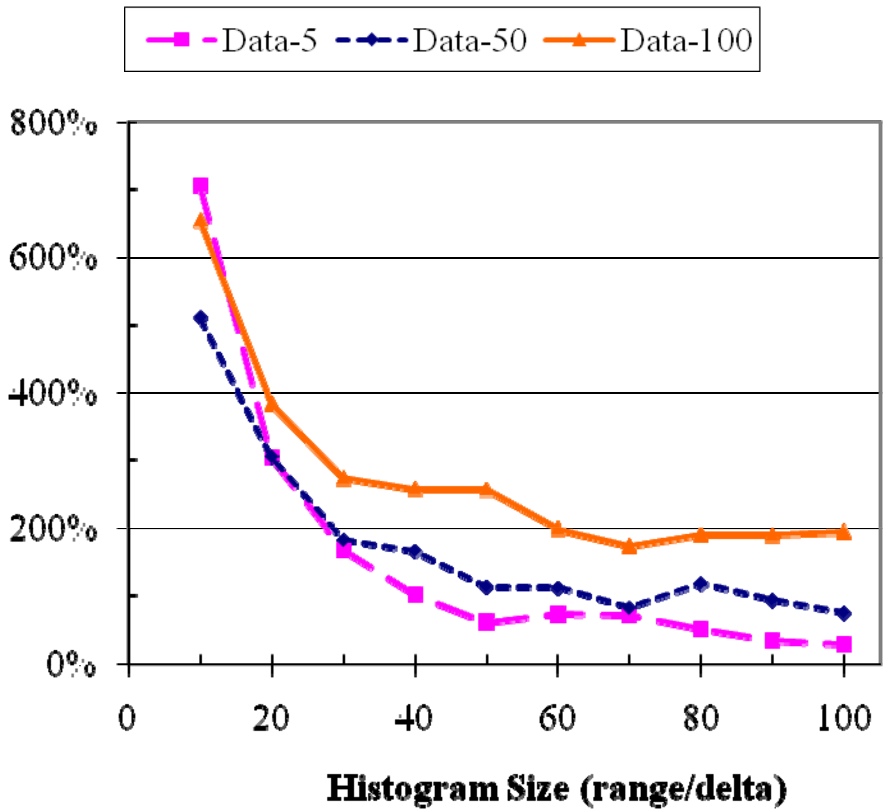



**Selectivities (3D)**

# Accuracy at Varying Precisions




Precision (2D)




Precision (3D)

# Conclusion and Future work

- Developed efficient algorithms for selectivity estimation of probabilistic threshold queries
- The algorithms were implemented in a real database system
- Experiments show that the algorithms are efficient and provide good estimates for query selectivities
- The algorithms can be further improved by combining them with standard cost estimation techniques such as equi-depth binning and sampling

# References

**[SIGMOD96]** Poosala et al. Improved histograms for selectivity estimation of range predicates. In SIGMOD 1996.

**[SIGMOD03]** R. Cheng et al. Evaluating probabilistic queries over imprecise data. In *SIGMOD* 2003.

**[ICDE08]** Singh et al. Database Support for Probabilistic Attributes and Tuples. In ICDE 2008.

**[Orion]** http://orion.cs.purdue.edu, 2006.

**[CIDR05]** J. Widom. Trio: A system for integrated management of data, accuracy and lineage. In CIDR, 2005.

**[SIGMOD05]** J. Boulos et al. MYSTIQ: A system for finding more answers by using probabilities, IN SIGMOD 2005

**[ICDE07]** Antova et al. 10^10^6 Worlds and beyond: Efficient representation and processing of incomplete information. In ICDE 2007

**[ICDE07b]** Sen et al. Representing correlated tuples in Probabilistic databases, ICDE 2007

**[VLDB04a]** R. Cheng et al. Efficient indexing methods for probabilistic threshold queries over uncertain data. In VLDB 2004

**[VLDB04b]** N. Dalvi et al. Efficient Query Evaluation on Probabilistic Databases. In VLDB 2004.

**[ICDE07c]** Ljosa et al. APLA: Indexing arbitrary probability distributions. In ICDE 2007.

**[VLDB06]** O. Benjelloun et al. ULDBs: Databases with uncertainty and lineage, In VLDB 2006.

**[CIKM06]** Reynold et al. Effiecient Join processing over uncertain data. In CIKM 2006.

# Thank you

Questions?

Sarvjeet Singh
sarvjeet@purdue.edu