A Unified Approach to Ranking in Probabilistic Databases

Jian Li, Barna Saha, Amol Deshpande
University of Maryland, College Park, USA

Probabilistic Databases

- Motivation: Increasing amounts of uncertain data
  - Sensor Networks; Information Networks
  - Data Integration and Information Extraction
  - …
- Probabilistic databases
  - Annotate *tuples* with existence probabilities, and/or *attribute values* with probability distributions
  - Interpretation according to the "possible worlds semantics"

Possible World Semantics

ID	Score	Prob
t_1	200	0.2
t_2	150	0.8
t_3	100	0.4

A probabilistic table
(assume tuple-independence)

→

pw1

ID	Score
t_1	200
t_2	150
t_3	100

w.p. 0.064

pw2

ID	Score
t_1	200
t_2	150

w.p. 0.096

pw3

ID	Score
t_2	150
t_3	100

w.p. 0.256

⋮

We focus on tuple uncertainty in the talk

Top-k Query Processing

Score values are used to rank the tuples in every *pw*.

ID	Score	Prob
t_1	200	0.2
t_2	150	0.8
t_3	100	0.4

A probabilistic table
(assume tuple-independence)

pw1

ID	Score
t_1	200
t_2	150
t_3	100

w.p. 0.064

pw2

ID	Score
t_1	200
t_2	150

w.p. 0.096

pw3

ID	Score
t_2	150
t_3	100

w.p. 0.256

⋮

The top-1 answer for each possible world

Top-*k* Queries: Many Prior Proposals

- Return *k* tuples *t* with the highest *score(t)Pr(t)* **[exp. score]**
- Returns the most probable top *k*-answer **[U-top-k]**

  [Soliman et al. '07]
- At rank *i*, return tuple with max. prob. of being at rank *i* **[U-rank-k]**

  [Soliman et al. '07]
- Return *k* tuples *t* with the largest $Pr\ (r(t) \leq k)$ values **[PT-k/GT-k]**

  [Hua et al. '08] [Zhang et al. '08]
- Return *k* tuples *t* with smallest **expected rank**: $\sum_{pw} Pr(pw)\ r_{pw}(t)$

  [Cormode et al. '09]

Top-*k* Queries

- Which one should we use???
- Comparing different ranking functions

Normalized Kendall Distance between two top-k answers:

Penalizes #reversals and #mismatches

Lies in [0,1], 0: Same answers; 1: Disjoint answers

	E-Score	PT/GT	U-Rank	E-Rank	U-Top
E-Score	----	0.124	0.302	0.799	0.276
PT/GT	0.124	----	0.332	0.929	0.367
U-Rank	0.302	0.332	-----	0.929	0.204
E-Rank	0.799	0.929	0.929	----	0.945
U-Top	0.276	0.367	0.204	0.945	----

Real Data Set: 100,000 tuples, Top-100

Top-*k* Queries

- Which one should we use???
- Comparing different ranking functions

Normalized Kendall Distance between two top-k answers:

Penalizes #reversals and #mismatches

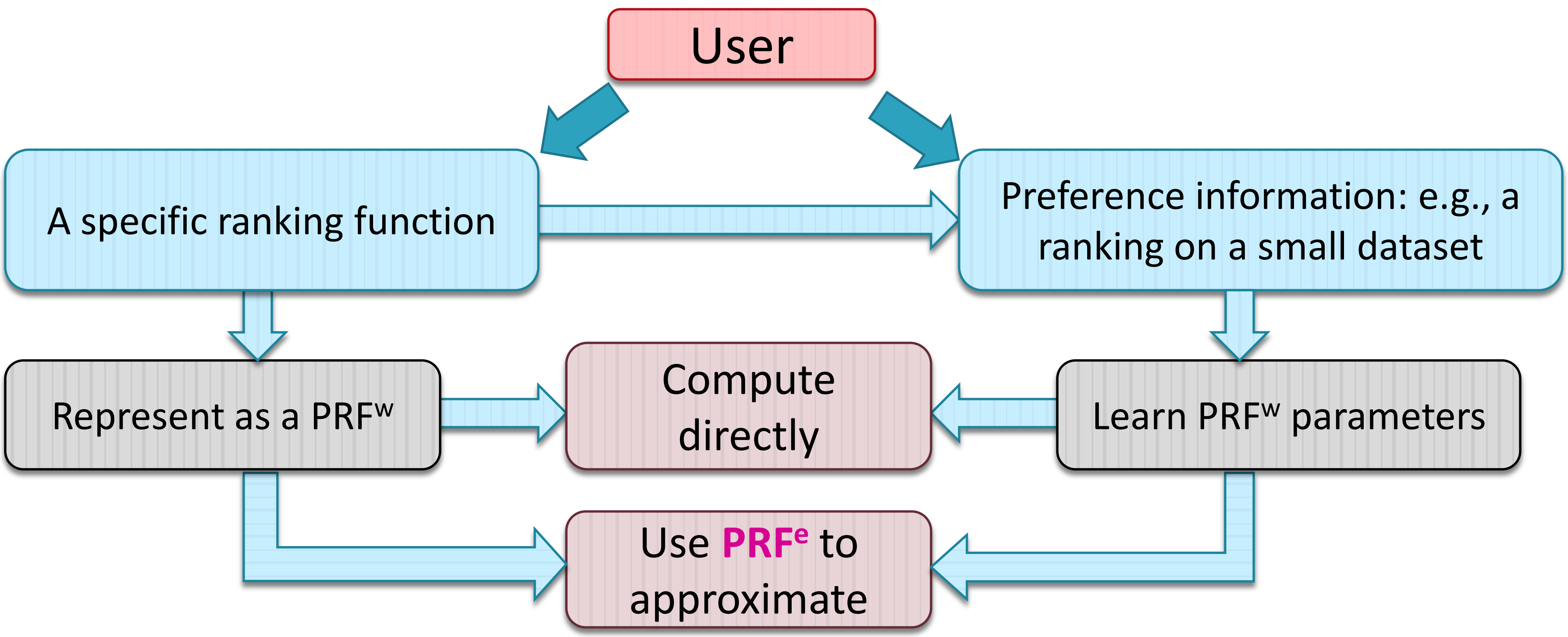
Lies in [0,1], 0: Same answers; 1: Disjoint answers

	E-Score	PT/GT	U-Rank	E-Rank	U-Top
E-Score	----	0.864	0.890	0.004	0.925
PT/GT	0.864	----	0.395	0.864	0.579
U-Rank	0.890	0.395	-----	0.890	0.316
E-Rank	0.004	0.864	0.890	----	0.926
U-Top	0.925	0.579	0.316	0.926	----

Synthetic Dataset: 100,000 tuples, Top-100

Our Approach

- Define two *parameterized* ranking functions: PRF^w; PRF^e
  - .. that can simulate or approximate a variety of ranking functions
  - PRF^e much more efficient to evaluate (than PRF^w)



Outline

- Parameterized Ranking Functions
- Computing PRF
  - Independent Tuples
  - Computing $PRF^{e}(\alpha)$
  - X-Tuples
  - Summery of Other Results
- Approximating Ranking Functions
- Experiments

Parameterized Ranking Function

- Weight Function: ω : (tuple, rank)$\rightarrow \mathbb{C}$
- Parameterized Ranking Function (PRF)

$$\Upsilon_\omega(t) = \sum_{i>0} \omega(t, i) \cdot \Pr(r(t) = i).$$

Probability that t *is ranked at position* i *across possible worlds*

Return *k* tuples with the highest $|\Upsilon_\omega|$ values.

Parameterized Ranking Function

$$\Upsilon_\omega(t) = \sum_{i>0} \omega(t,i) \cdot \Pr(r(t) = i).$$

- $\omega(t,i)= 1$: Rank the tuples by probabilities
- $\omega(t,i)=score(t)$: E-Score
- **PRF^e(α)**: $\omega(i)=\alpha^i$ where α *can be a real or a complex number*
- **PRF$^\omega$(h)**: $\omega(t,i) = \omega(i)$ and $\omega(i) = 0\ \forall i > h$
  - Generalizes PT/GT-k and *U-Rank*

Outline

- Parameterized Ranking Functions
- Computing PRF
  - Independent Tuples
  - Computing PRF$^e(\alpha)$
  - X-Tuples
  - Summery of Other Results
- Approximating Ranking Functions
- Experiments

Computing PRF: Independent Tuples

Computing $\Pr(r(t_i) = j)$, for a given tuple t_i and a given rank j

$T_{i-1}=\{t_1,t_2,\ldots,t_{i-1}\}$ i.e., the set of tuples with scores higher than t_i

$$\begin{aligned}
\Pr(r(t_i) = j) &= \Pr(t_i) \sum_{pw: j-1 \text{ tuples in } T_{i-1} \text{ exists}} \Pr(pw) \\
&= \Pr(t_i) \sum_{\substack{S \subseteq T_{i-1} \\ |S| = j-1}} \prod_{t \in S} \Pr(t) \prod_{t \in T_{i-1} \setminus S} (1 - \Pr(t))
\end{aligned}$$

Generating Function Method

$$\mathcal{F}^i(x) = \left(\prod_{t \in T_{i-1}} \Big(1 - \Pr(t) + \Pr(t) \cdot x\Big) \right) (\Pr(t_i) \cdot x)$$

The coefficient of x^j : $Pr(r(t_i)=j)$

Computing PRF: Independent Tuples

- **Algorithm:**
  - For i=1 to n
    - Expand $\mathcal{F}^i(x) = \sum_{j=1}^{n} \Pr(r(t_i) = j) x^j$
    - $\Upsilon(t_i) = \sum_{j=1}^{n} \omega(t_i, j) \Pr(r(t_i) = j)$
  - Return k tuples with largest $|\Upsilon_\omega|$ values

Can be improved to O(n)

O(n) time

$O(n^2)$ overall

F^i and F^{i-1} differ in two multiplicative terms

$$\mathcal{F}^i(x) = \left(\prod_{t \in T_{i-1}} \left(1 - \Pr(t) + \Pr(t) \cdot x\right) \right) (\Pr(t_i) \cdot x)$$

Outline

- Parameterized Ranking Functions
- Computing PRF
  - Independent Tuples
  - Computing PRF^e(α)
  - X-Tuples
  - Summery of Other Results
- Approximating Ranking Functions
- Experiments

Computing PRF$^e(\alpha)$: Independent Tuples

- Recall $\omega(j)=\alpha^j$
- Generating Function Method
  - $\mathcal{F}^i(x) = \sum_{j=1}^{n} \Pr(r(t_i) = j)x^j$
  - $\Upsilon(t_i) = \sum_{i=1}^{n} \Pr(r(t_i) = j)\alpha^j$

$$\boxed{\Upsilon(t_i) = \mathcal{F}^i(\alpha)}$$

No need to expand the polynomial !!

- Therefore: $\mathcal{F}^i(\alpha) = \left(\prod_{t \in T_{i-1}} \left(1 - \Pr(t) + \Pr(t)\cdot\alpha\right)\right)(\Pr(t_i)\cdot\alpha)$
- Morevoer: $\mathcal{F}^i(\alpha) = \frac{\Pr(t_i)}{\Pr(t_{i-1})}\mathcal{F}^{i-1}(\alpha)\left(1 - \Pr(t_{i-1}) + \Pr(t_{i-1})\alpha\right)$

O(1)

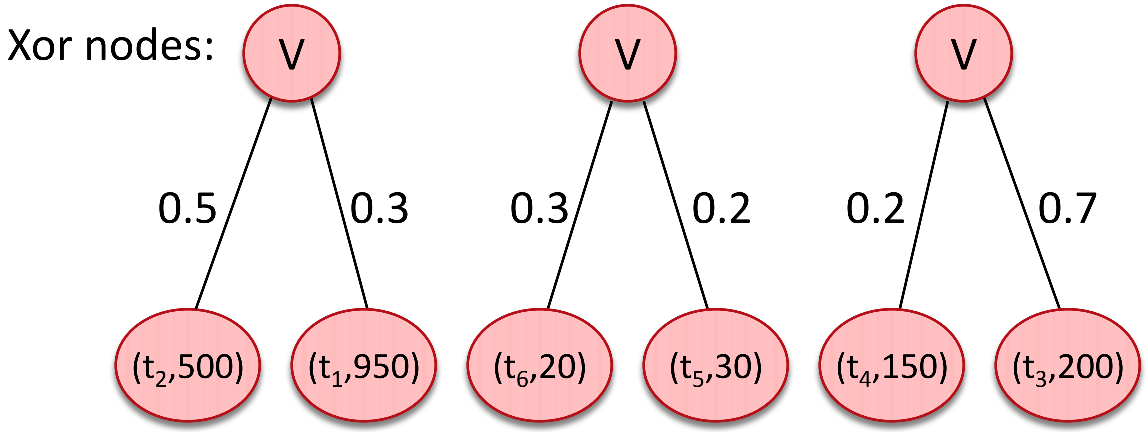
O(n) overall

Outline

- Parameterized Ranking Functions
- Computing PRF
  - Independent Tuples
  - Computing PRF$^e(\alpha)$
  - X-Tuples
  - Summery of Other Results
- Approximating Ranking Functions
- Experiments

Computing PRF: x-Tuples

- Capture mutual exclusivity correlations



Possible Worlds	Pr
t_4	0.02
t_3	0.08
…….	
t_1, t_4, t_6	0.03
t_1, t_3, t_6	0.018
…….	

Computing PRF: x-Tuples

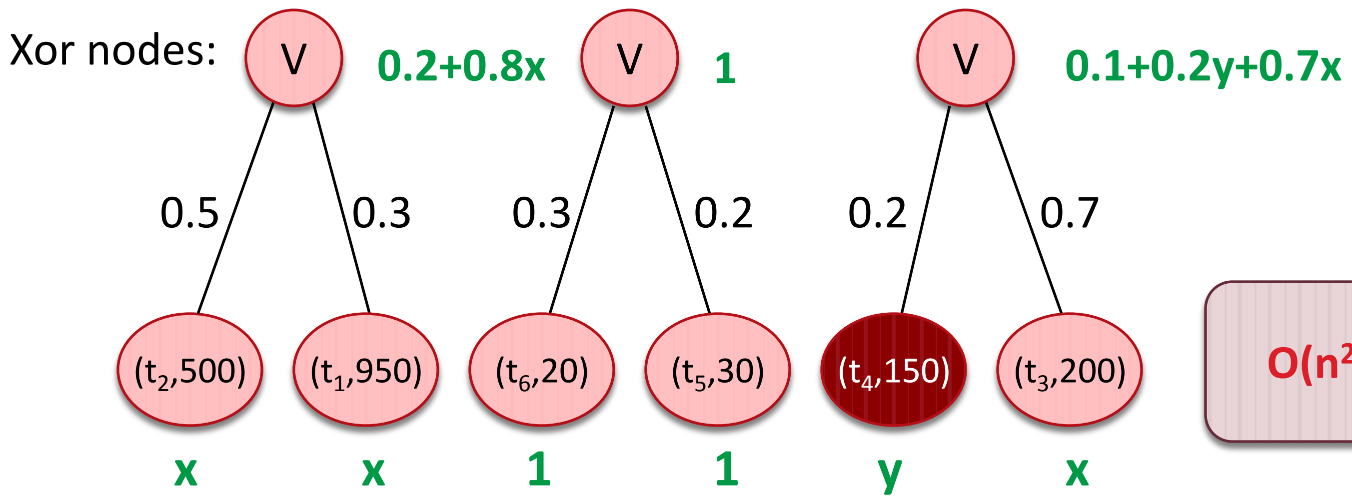
Construct generating function for t_4

$r(i)=j$ if and only if (1) j-1 tuples with higher scores appear

(2) tuple i appears

$Pr(r(t_4)=j)$ = coeff of $x^{j-1}y$

$$F(x,y)=(0.2+0.8x)(0.1+0.2y+0.7x)$$



$O(n^2)$ overall

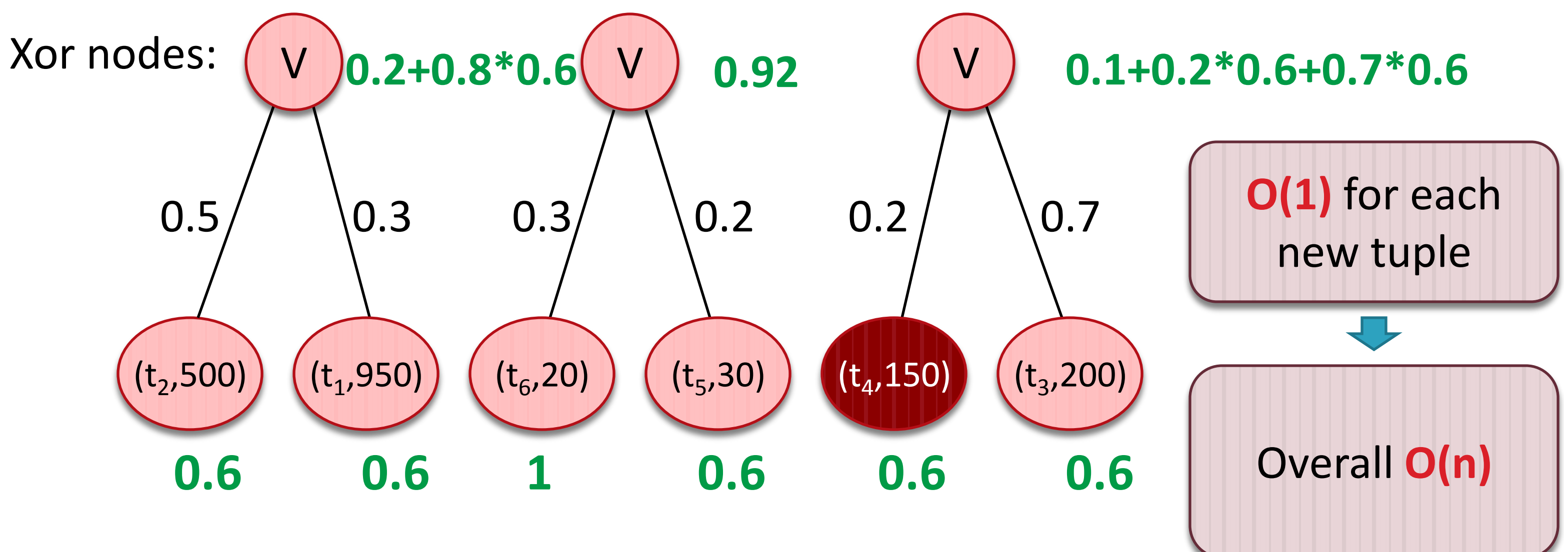
Computing PRF$^e(\alpha)$: x-Tuples

$$\boxed{\Upsilon(t_i) = \mathcal{F}^i(\alpha,\alpha) - \mathcal{F}^i(\alpha,0).}$$

We maintain only the numerical values of $F^i(\alpha,\alpha)$ and $F^i(\alpha,0)$ at each node.

E.g. α=0.6. Now we want to compute **$F^5(0.6,0.6)$**

$F^5(0.6,0.6)=0.096*0.92*0.64$



Summary of Other Results

- PRF computation for probabilistic and/xor trees
  - Generalizes x-tuples by allowing both mutual exclusivity and coexistence
  - PRF computation : $O(n^3)$
  - PRF^e computation : $O(n\log n+nd)$ (d = height of the tree)
- PRF computation on graphical models
  - A polynomial time algorithm when the junction tree has bounded treewidth
    – A nontrivial dynamic program combined with the generating function method.

Outline

- Parameterized Ranking Functions
- Computing PRF
  - Independent Tuples
  - Computing PRF$^{e}(\alpha)$
  - X-Tuples
  - Summery of Other Results
- Approximating Ranking Functions
- Experiments

Approximating Ranking Functions

Approximating PRF^{ω} by a linear combination of PRF^{e}

- Suppose $\omega(i) \approx \sum_{l=1}^{L} u_l \alpha_l^i$

$$\Upsilon(t) = \sum_i \omega(i) \Pr(r(t) = i) \approx \sum_{l=1}^{L} u_l \left(\sum_i \alpha_l^i \Pr(r(t) = i) \right)$$

- Reduce to L individual PRF^{e} computations
- Running time : **O(nlogn+nL)** (as opposed to $O(n^2)$)

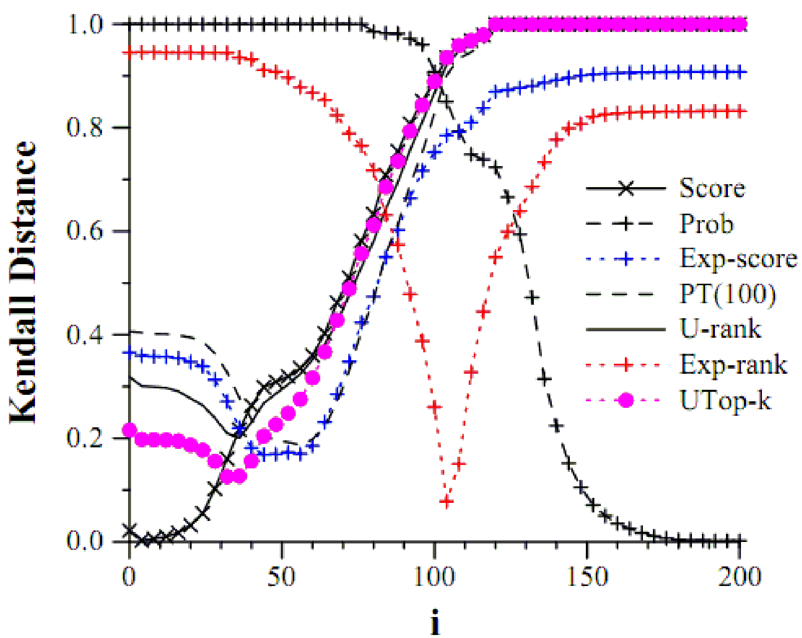
- We developed a scheme based on adapting the *discrete Fourier transformation* of ω
- Works very well for monotonically non-increasing ω
  - E.g. the step function (PT/GT-k)
- Details in the paper

Outline

- Parameterized Ranking Functions
- Computing PRF
  - Independent Tuples
  - Computing $PRF^e(\alpha)$
  - X-Tuples
  - Summery of Other Results
- Approximating Ranking Functions
- Experiments

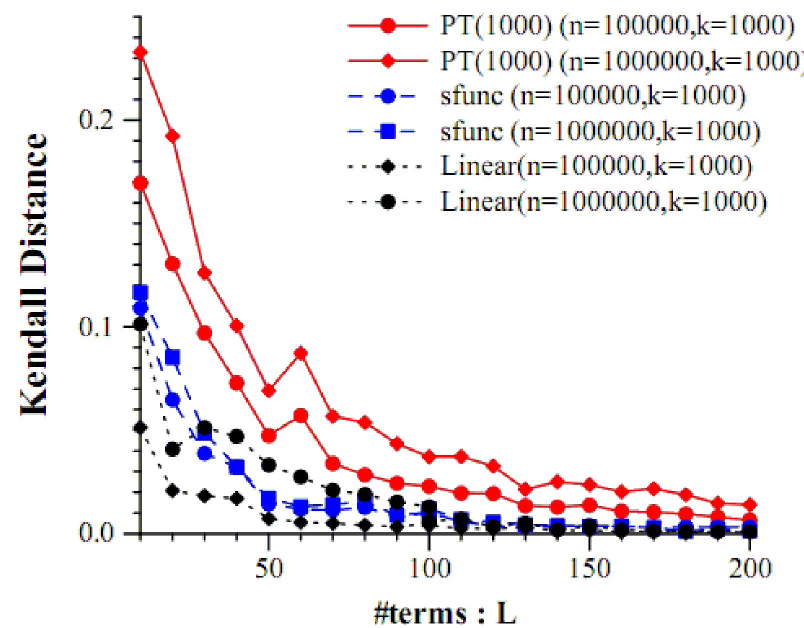
Experiments: Approximation Ability

<u>Using a single PRF^e</u>



Approximating other functions using PRF^e(α)
α= 1-0.9^i

<u>A linear combination of PRF^{e}s</u>

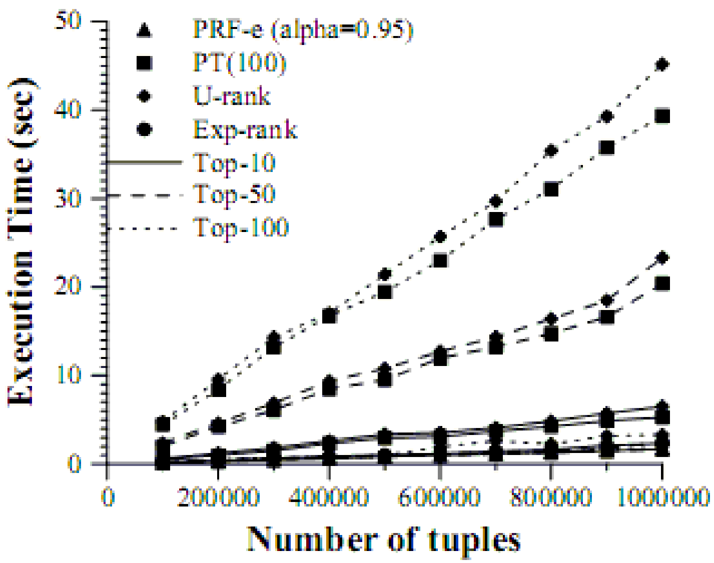


No. of Terms vs Approximation Quality

Real Icberg sighting dataset: 100,000 tuples

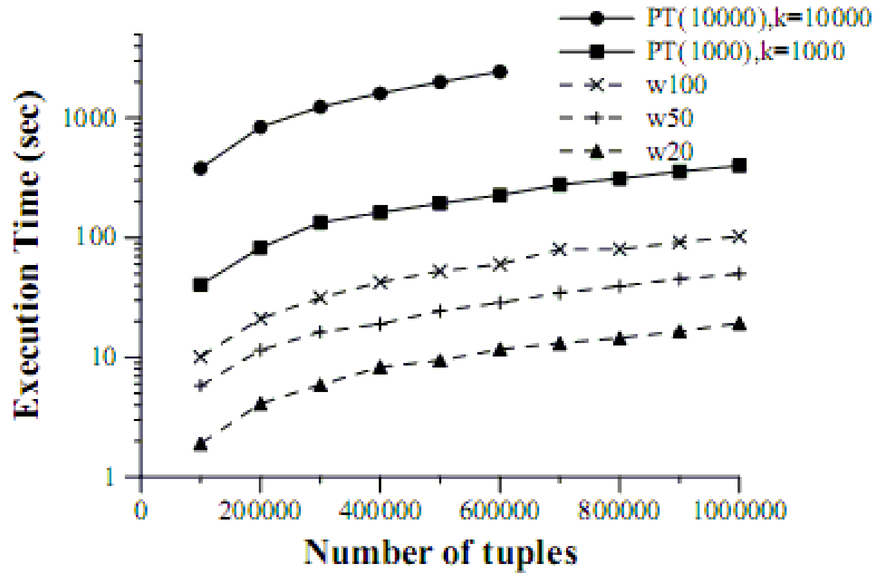
Experiments: Execution Time

PRF^e vs Others



(i)

Approx vs Exact



(ii)

Real Icberg sighting dataset

Conclusions

- Proposed a unifying framework for ranking over probabilistic databases through:
  - *Parameterized* ranking functions
  - Incorporation of user feedback
- Designed highly efficient algorithms for computing PRF and PRF^e
- Developed novel approximation techniques for approximating PRF^w with PRF^e