Handling Uncertainty in Data Management

Jian Li

Tsinghua University, Beijing, China

WAIM 2014, Macau

TexPoint fonts used in EMF.

Uncertain Data

- Uncertain data is ubiquitous
  - Data Integration and Information Extraction
  - Sensor Networks; Information Networks

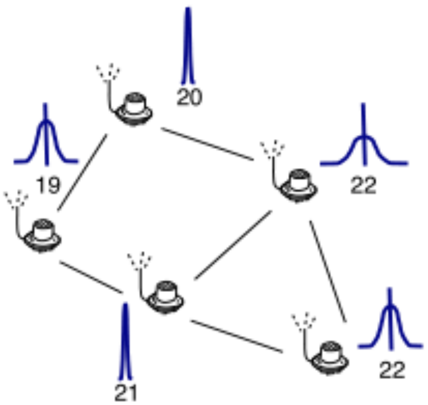
SSN	Name
208-79-4209	John Williams

SSN	Name
208-79-4209	Michael Lewin

SSN	Name	Prob
208-79-4209	John Williams	0.5
208-79-4209	Michael Lewin	0.5

Tuple uncertainty

Data integration

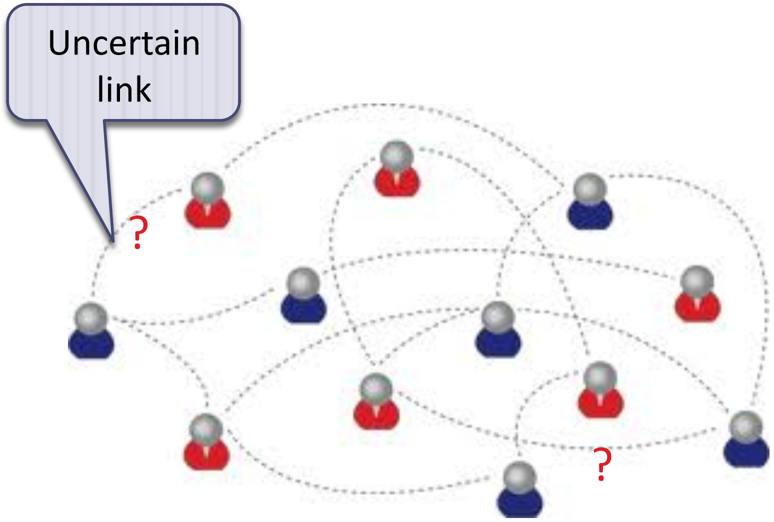


Sensor network

Sensor ID	Temp.
1	Gauss(40,4)
2	Gauss(50,2)
3	Gauss(20,9)
…	…

Attribute uncertainty

Uncertain Data



Social network

- Future data is destined to be uncertain

Uncertain Data

Decision making under uncertainty

- Many statistical/machine learning models (Graphical model etc.)
- Job Scheduling (uncertain job length)
- Online Ads assignment (uncertain intents)
- Kidney Exchange (probabilistic matching)
- Crowdsourcing (noisy answers)

Dealing with Uncertainty

- There is an increasing need for analyzing and reasoning over such data
- Handling uncertainty is a very broad topic that spans multiple disciplines
  - Economics / Game Theory
  - Finance
  - Electrical Engineering
  - Probability Theory / Statistics
  - Psychology
  - Computer Science

Outline

- Ignoring Uncertainty?
  - Examples
  - Possible world semantics
- Beyond Expectation– expected utility theory
  - St Peterburg Paradox
  - Consensus Answer
- Queries over Probabilistic Data
  - Top-k queries
  - Other queries
- Stochastic Optimization
  - Stochastic Matching
  - Stochastic Knapsack

Possible World Semantics

View a probabilistic database as probability distribution over the set of possible worlds

ID	A	Prob
t_1	1	0.2
t_2	1	0.8
t_3	2	0.4

A probabilistic table
(assume tuple-independence)

→

pw1

ID	A
t_1	1
t_2	1
t_3	2

w.p. 0.064

pw2

ID	A
t_1	1
t_2	1

w.p. 0.096

pw3

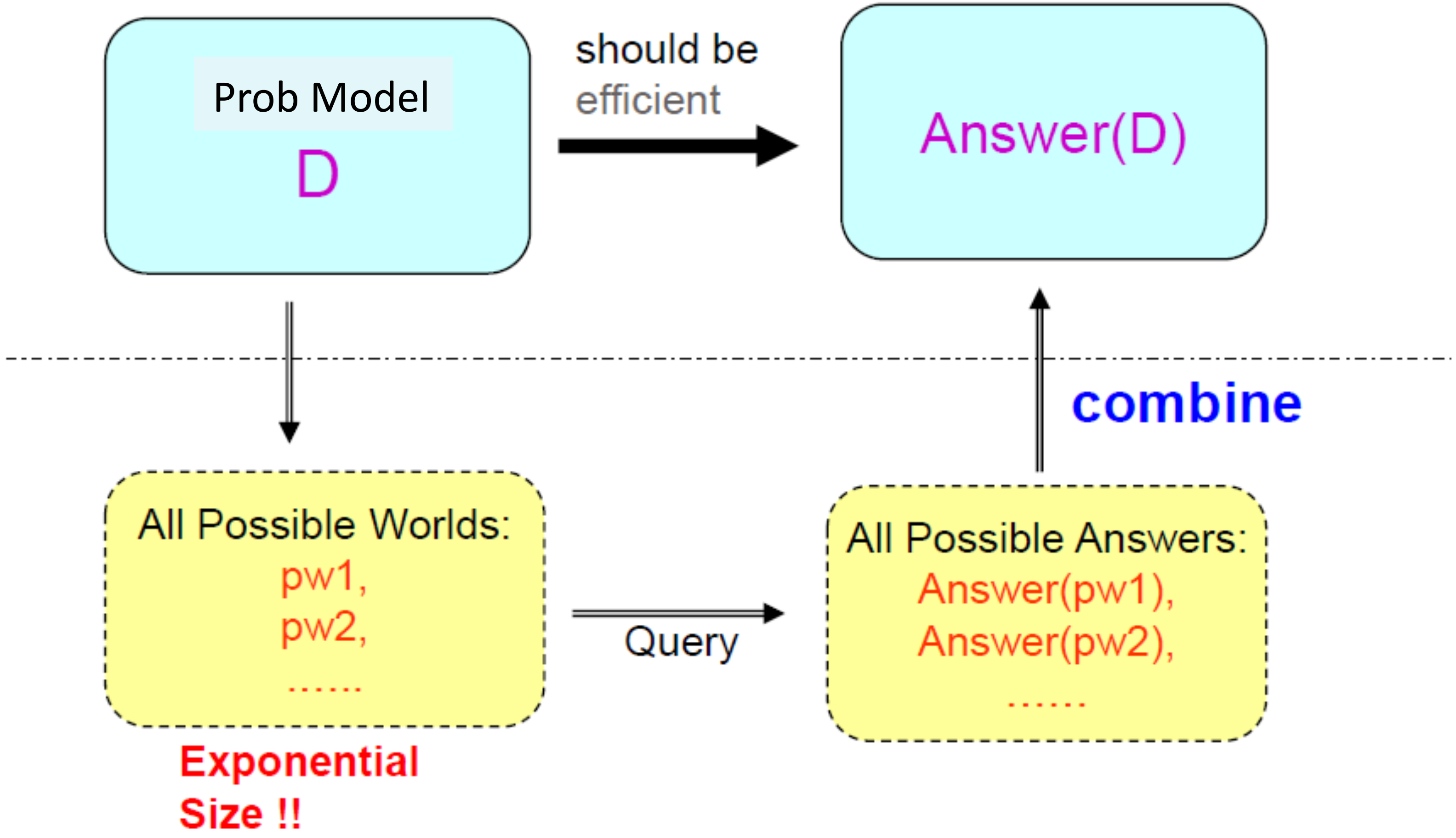
ID	A
t_2	1
t_3	2

w.p. 0.256

⋮

8 worlds

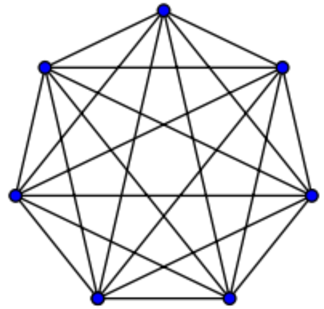
Possible World Semantics



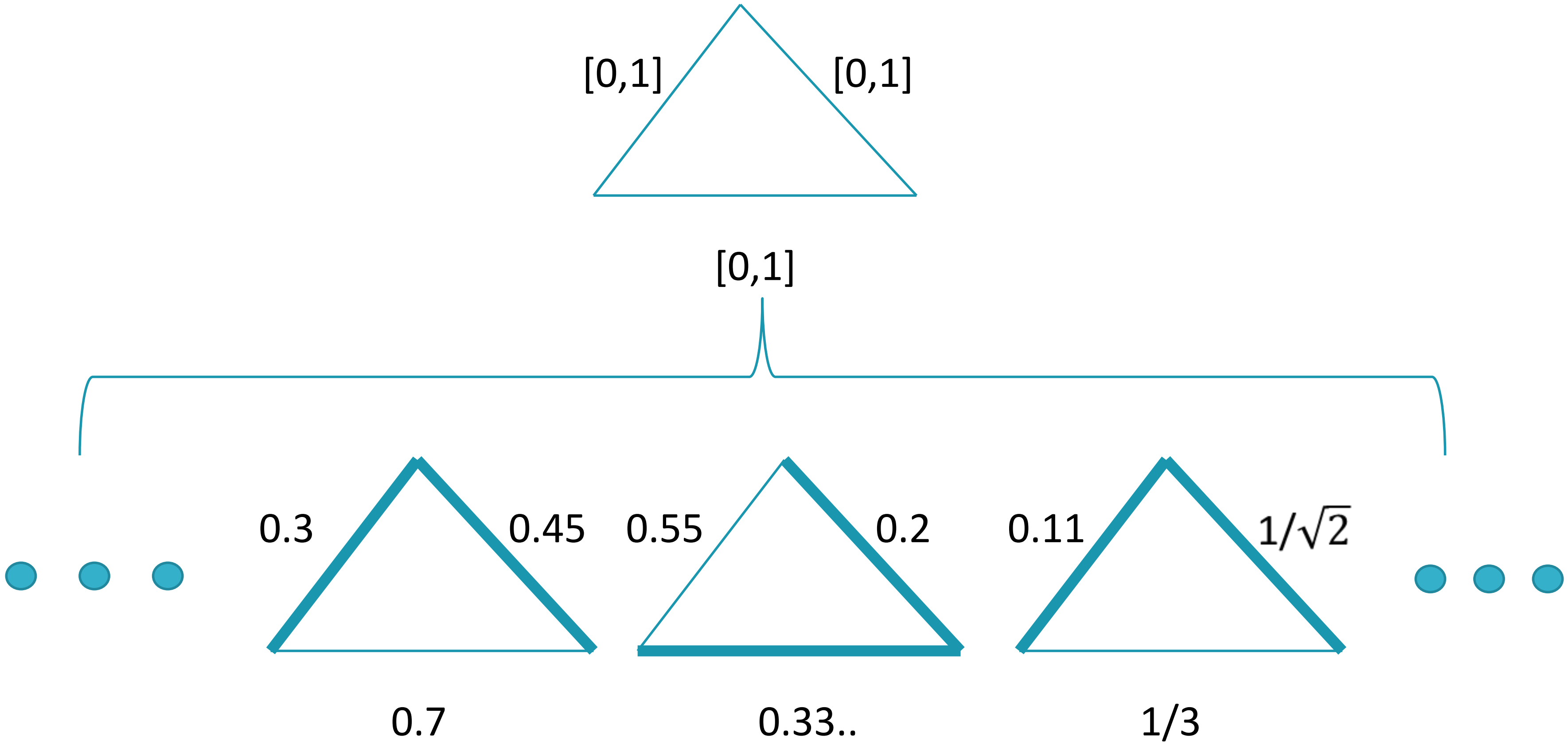
Ignoring uncertainty is not the right thing to do

- A undirected graph with n nodes
- The length of each edge: i.i.d. Uniform[0,1]

- Question: What is E[MST]?
  - MST: minimum spanning tree

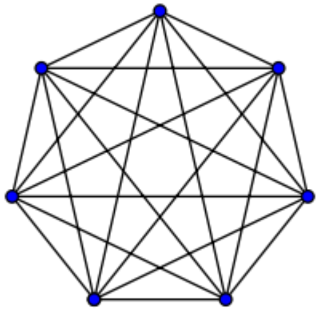


Ignoring uncertainty is not the right thing to do



Ignoring uncertainty is not the right thing to do

- A undirected graph with n nodes
- The length of each edge: i.i.d. Uniform[0,1]



- Question: What is E[MST]?
  - MST: minimum spanning tree
- Ignoring uncertainty ("replace by the expected values" heuristic)
  - each edge has a fixed length 0.5
  - This gives a WRONG answer 0.5(n-1)

Ignoring uncertainty is not the right thing to do

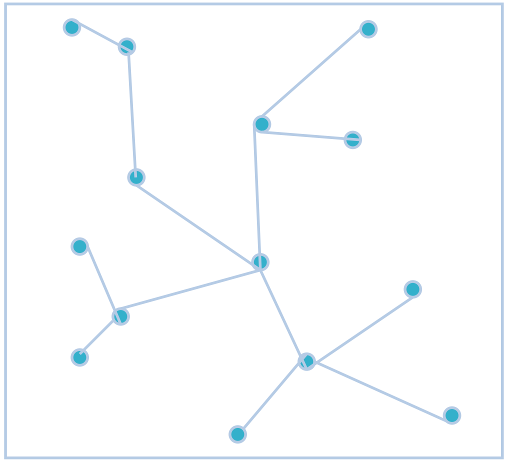
- A undirected graph with n nodes
- The length of each edge: i.i.d. Uniform[0,1]
- Question: What is E[MST]?
- Ignoring uncertainty ("replace by the expected values" heuristic)
  - each edge has a fixed length 0.5
  - This gives a WRONG answer 0.5(n-1)
- But the true answer is (as n goes to inf)

$$\zeta(3) = \sum_{i=1}^{\infty} 1/i^3 < 2$$

[McDiarmid, Dyer, Frieze, Karp, Steele, Bertsekas, Geomans]

A Similar Problem

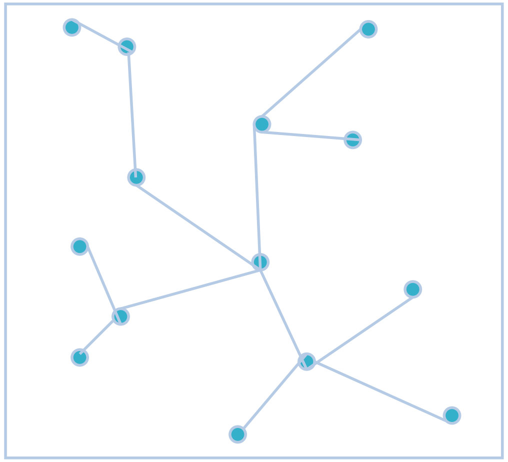
- N points: i.i.d. uniform[0,1]×[0,1]



- Question: What is E[MST] ?
- Answer:

A Similar Problem

- N points: i.i.d. uniform$[0,1] \times [0,1]$



- Question: What is E[MST] ?
- Answer: $\theta(\sqrt{n})$ [Frieze, Karp, Steele, …]

The problem is similar, but the answer is not similar…………

A more general computational problem considered in [Huang, L. ArXiv 2013]

- Similar phenomena can be found in many combinatorial optimization problems, such as matching, TSP (traveling salesman problem) etc.

- A take away message:

  Ignoring uncertainty (or simplistic replace-by-expectation heuristic) may not the right thing to do

Outline

- Ignoring Uncertainty?
  - Examples
  - Possible world semantics
- Beyond Expectation– expected utility theory
  - St Peterburg Paradox
  - Consensus Answer
- Queries over Probabilistic Data
  - Top-k queries
  - Other queries
- Stochastic Optimization
  - Stochastic Matching
  - Stochastic Knapsack

Aggregate Queries

- *Aggregate Query:*

Item	Forecaster	Profit	P
Widget	Alice	$-99K	0.99
	Bob	$100M	0.01
Whatsit	Alice	$1M	1

Profit(Item;Forecaster,Profit;P)

```
SELECT SUM(PROFIT)
FROM PROFIT
WHERE ITEM='Widget'
```

(a) Expectation Style

Answer: E[profit]=19.9K

```
SELECT ITEM
FROM PROFIT
WHERE ITEM='Widget'
HAVING SUM(PROFIT) > 0.0
```

(b) HAVING Style

Answer: Prob[profit>0] =0.01

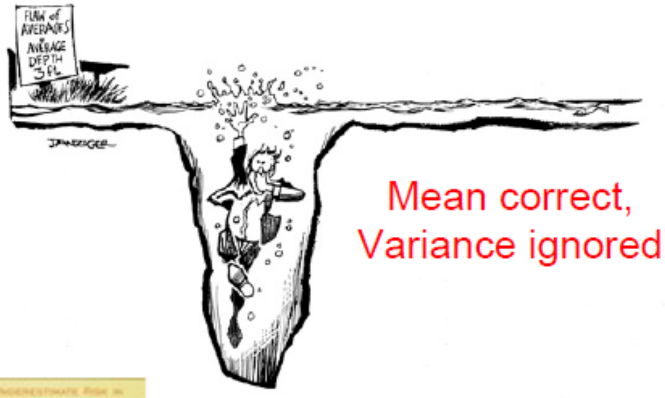
Expected value may not be sufficient!

Example taken from The trichotomy of HAVING queries on a probabilistic database, Re, C. and Suciu, D., The VLDB Journal, 2009

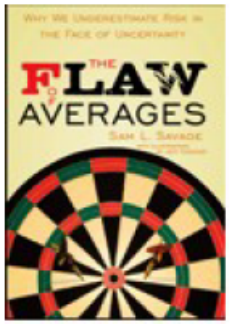
Inadequacy of Expected Value

- Be aware of risk!

Flaw of averages (weak form):



Mean correct,
Variance ignored



Flaw of averages (strong form):

Wrong value of mean:
$f(E[X]) \neq E[f(X)]$

Inadequacy of Expected Value

- Inadequacy of expected value:
  - Unable to capture risk-averse or risk-prone behaviors
    - Action 1: $100 VS Action 2: $200 w.p. 0.5; $0 w.p. 0.5
    - Risk-averse players prefer Action 1
    - Risk-prone players prefer Action 2 (e.g., a gambler spends $100 to play Double-or-Nothing)
  - St. Petersburg paradox
    - You pay x dollars to enter the game
      - Repeatedly toss a fair coin until a tail appears
      - payoff=2^k where k=#heads
    - How much should x be?
      - Expected payoff =1x(1/2)+2x(1/4)+4x(1/8)+……=
      - Few people would pay even $25 [Martin '04]

Expected Utility Maximization Principle

A : The set of valid answers

$w_{pw}(a)$: the cost of answer in pw

$u\colon R \to R$: the utility function

Expected Utility Maximization Principle:

The most desirable answer *a* is the answer that max. the exp. utility, i.e.,

$$a = \max_{a' \in A} \mathrm{E}_{pw}[\mu(w_{pw}(a'))]$$

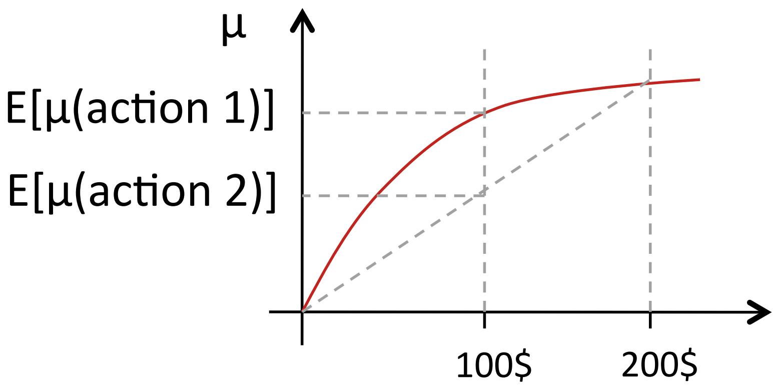
Von Neumann and Morgenstern provides an *axiomitization* of the principle (known as von Neumann-Morgenstern expected utility theorem).

Expected Utility Maximization Principle

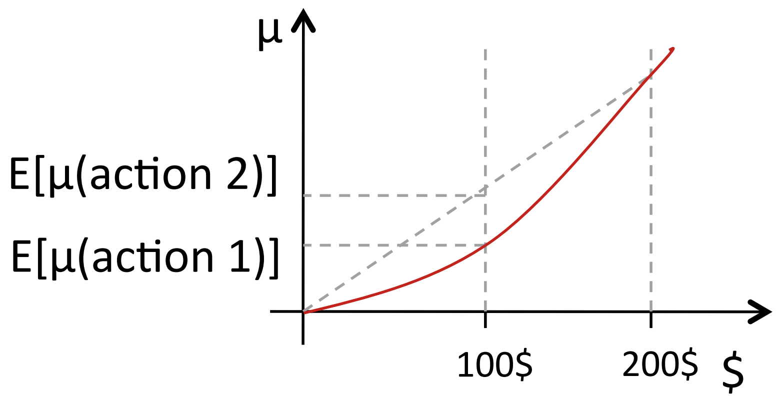
$u: R \rightarrow R:$ The utility function: profit-> utility

Expected Utility Maximization Principle: the decision maker should choose the action that maximizes the **expected utility**

- Action 1: $100
- Action 2: $200 w.p. 0.5; $0 w.p. 0.5



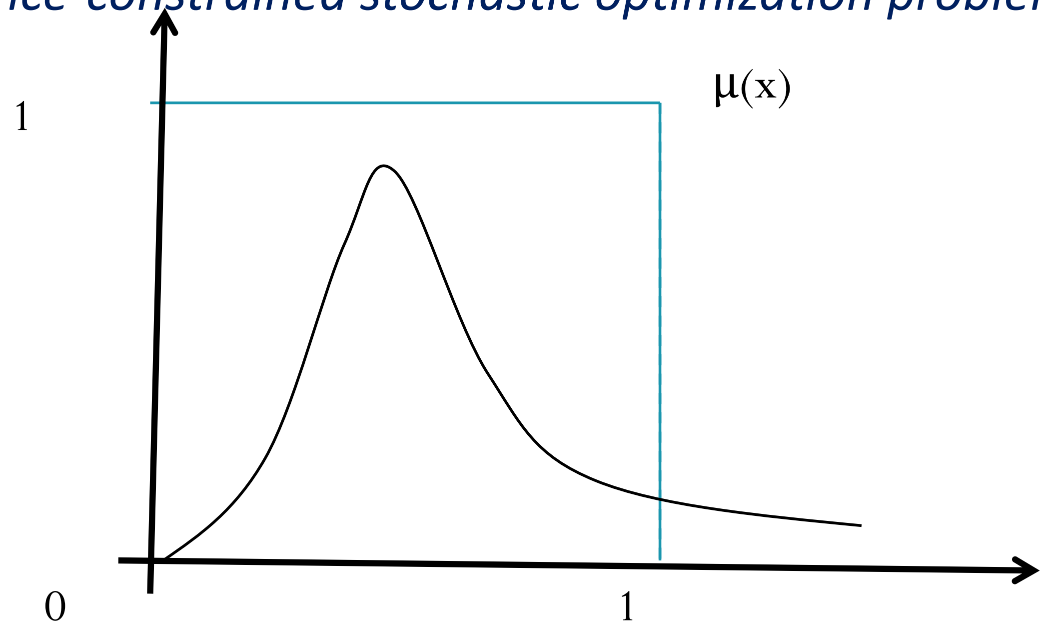
Risk-averse



Risk-prone

Threshold Probability Maximization

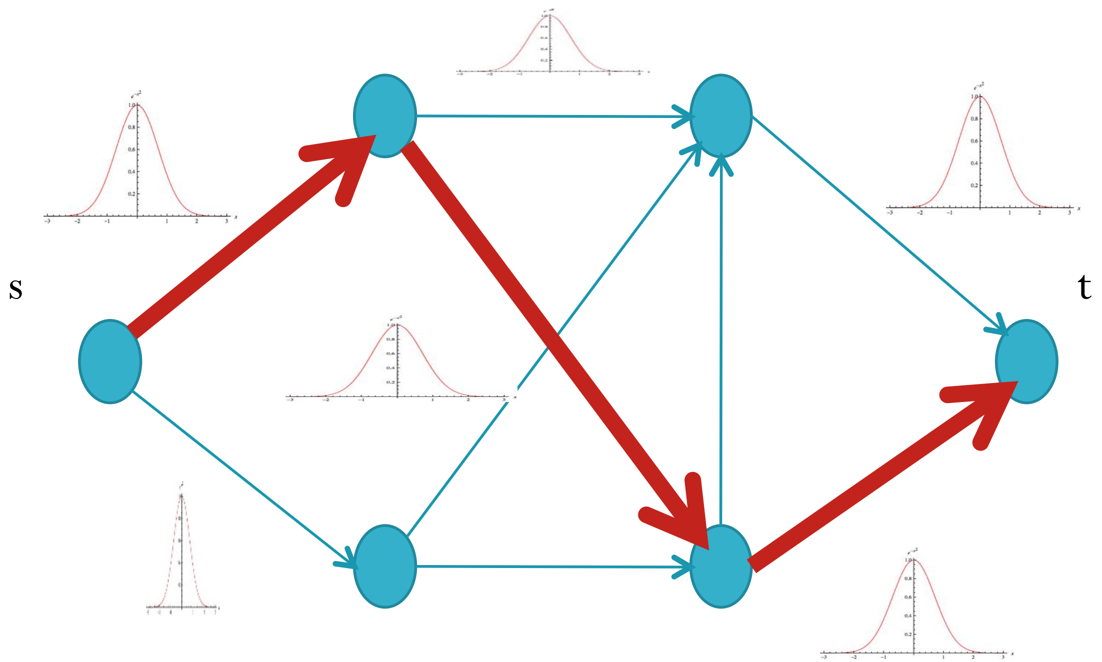
- If *μ is a threshold function, maximizing E[μ(cost)] is equivalent to maximizing* ***Pr[w(cost)<1]***
  - *minimizing overflow prob.* [Kleinberg, Rabani, Tardos. STOC'97] [Goel, Indyk. FOCS'99]
  - *chance-constrained stochastic optimization problem* [Swamy. SODA'11]



Threshold Probability Maximization

Threshold Probability Maximization

- **Stochastic shortest path** : find an s-t path P such that *Pr* [*w*(*P*)<*1*] is maximized
  - First assume Gaussian distributions (with different parameters)



in [Nikolova, Kelner, Brand, Mitzenmacher. ESA'06] [Nikolova. APPROX'10]

Threshold Probability Maximization

- **Stochastic shortest path** : find an s-t path P such that *Pr*[*w*(*P*)<*t*] is maximized
  - First assume Gaussian distributions (with different parameters)
  - Note that $N(\mu_1,\sigma_1^2)+N(\mu_2,\sigma_2^2)=N(\mu_1+\mu_2,\sigma_1^2+\sigma_2^2)$

$$\Pr\left(\sum_{i\in\pi} X_i \le t\right) = \Pr\left(\frac{\sum X_i - \sum \mu_i}{\sqrt{\sum \sigma_i^2}} \le \frac{t-\sum \mu_i}{\sqrt{\sum \sigma_i^2}}\right) = \Phi\left(\frac{t-\sum \mu_i}{\sqrt{\sum \sigma_i^2}}\right),$$

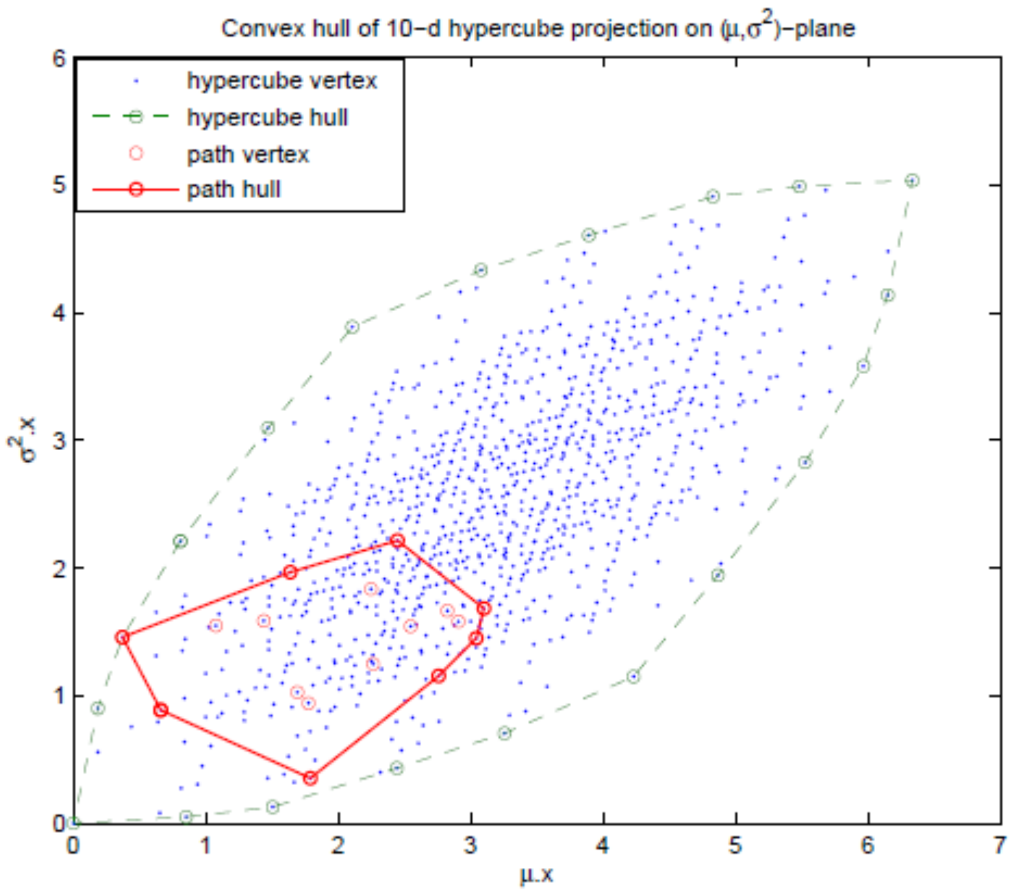
Standard Gaussian CDF

So, we want to $\max_{\pi} \frac{t-\sum_{i\in\pi}\mu_i}{\sqrt{\sum_{i\in\pi}\sigma_i^2}}.$

[Nikolova, Kelner, Brand, Mitzenmacher. ESA'06] [Nikolova. APPROX'10]

Threshold Probability Maximization

- Objective: $$\max_{\pi} \frac{t - \sum_{i\in\pi}\mu_i}{\sqrt{\sum_{i\in\pi}\sigma_i^2}}.$$



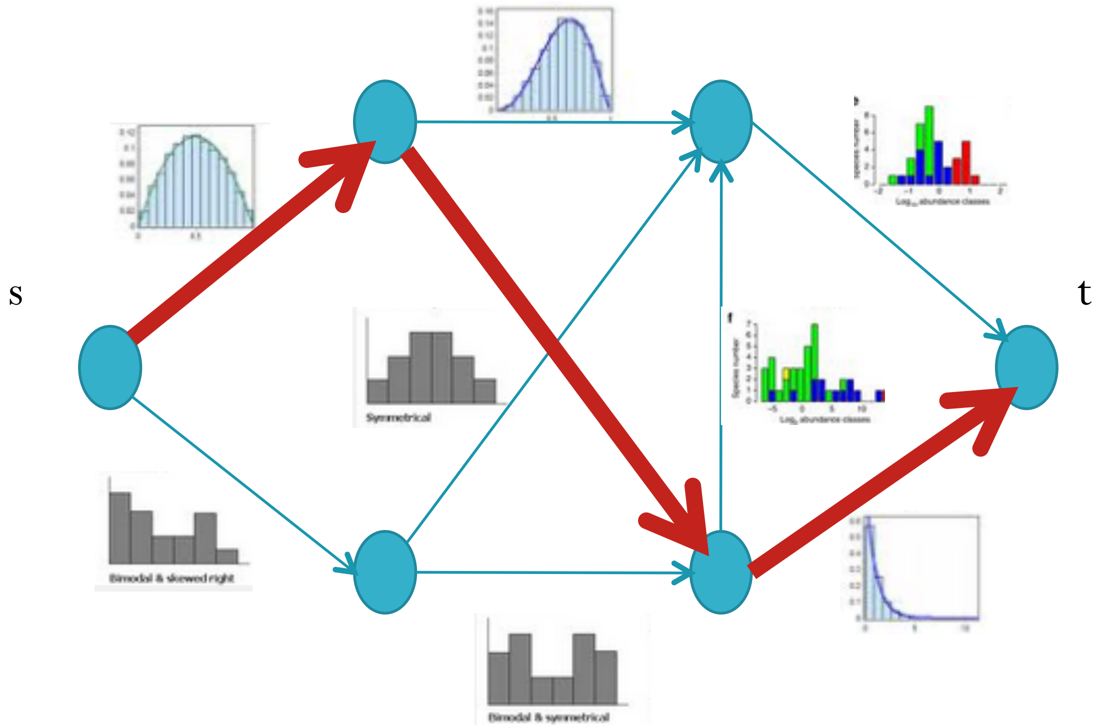
Ob: The obj is **quasi-convex**; the optimal solution must be a boundary point on the path hull

ALGO: enumerate the boundary points

- Time (worst case): $O(n^{\log n})$
- (Smoothed): polynomial
- Approximation with ϵ error: polynomial

Threshold Probability Maximization

For more general distributions, we can get the same result via more sophisticated techniques (characteristic functions, Poisson Approximation)



For more general results, see [L, Deshpande, FOCS11][L, Yuan STOC13]

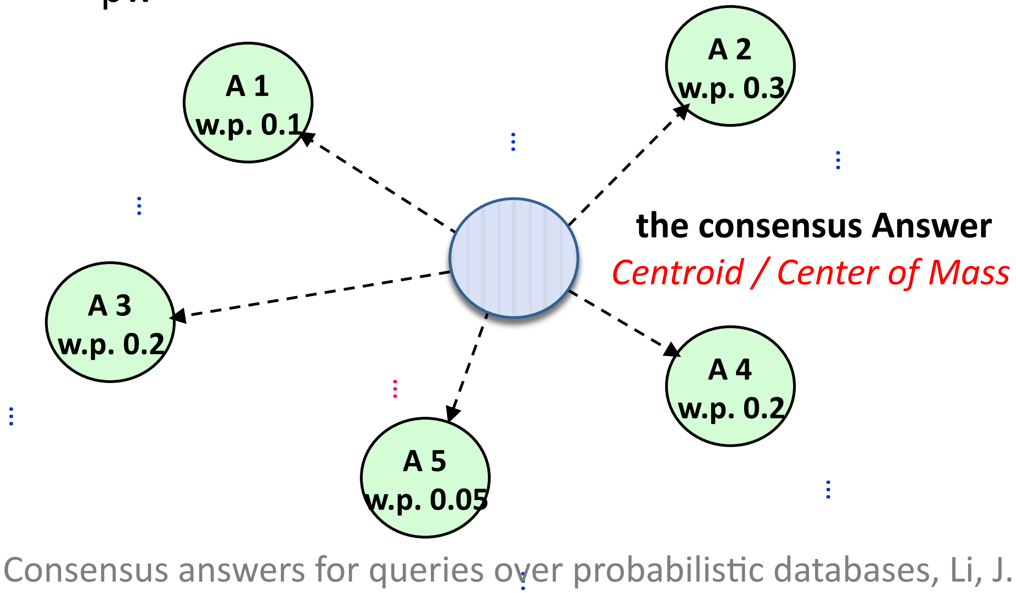
Consensus Answer

Consensus Answer:

- Think of each possible answers as a point in the space.
- Suppose d() is a distance metric between answers.
- Consensus Answer is a single deterministic answer

$$\tau = \arg\min_{\tau' \in} \{\mathbb{E}[d(\tau', \tau_{pw})]\}$$

where τ_{pw} is the answer for the possible world pw



Can be viewed as a version of the expected utility maximization principle!
(utility= - distance)

Consensus answers for queries over probabilistic databases, Li, J. and Deshpande, A., PODS, 2009

Outline

- Ignoring Uncertainty?
  - Examples
  - Possible world semantics
- Beyond Expectation– expected utility theory
  - St Peterburg Paradox
  - Consensus Answer
- Queries over Probabilistic Data
  - Top-k queries
  - Other queries
- Stochastic Optimization
  - Stochastic Matching
  - Stochastic Knapsack

Ranking over Probabilistic Data

- Our goal: support "ranking" or "top-*k*" query processing
  - Deciding which apartments to inquire about
  - Selecting a set of sensors to "probe"
  - Choosing a set of stocks to invest in
  - …
- How? Choose tuples with large scores? Or tuples with higher probabilities?
  - A complex trade-off

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 — w.p. 0.064

ID	Score
t_1	200
t_2	150
t_3	100

pw2 — w.p. 0.096

ID	Score
t_1	200
t_2	150

pw3 — w.p. 0.256

ID	Score
t_2	150
t_3	100

⋮

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. ICDE'07]

- At rank *i*, return tuple with max. prob. of being at rank *i* **[U-rank-k]**

  [Soliman et al. ICDE'07]

- Return *k* tuples *t* with the largest $Pr\ (r(t) \leq k)$ values **[PT-k/GT-k]**

  [Hua et al. SIGMOD'08] [Zhang et al. EDBT'08]

- Return *k* tuples *t* with smallest **expected rank**: $\sum_{pw} Pr(pw)\ r_{pw}(t)$

  [Cormode et al. ICDE'09]

- Return k tuples t with expected score of best available tuple **[k-selection]** [Liu et al. DASFAA'10]

Top-*k* Queries: Many Proposals

- Probabilistic Threshold (PT-k/GT-k) [Hua et al. SIGMOD'08] [Zhang et al. EDBT'08]
  - Return *k* tuples *t* with the largest $Pr(r(t) \leq k)$ values

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

Possible worlds	Prob
t_1, t_2, t_3	0.064
t_1, t_2	0.096
t_1, t_3	0.016
t_2, t_3	0.256
t_1	0.024
t_2	0.384
t_3	0.064
Φ	0.096

K=2

ID	$Prob(r(t) \leq 2)$
t_1	0.2
t_2	0.8
t_3	0.336

Ranking: t_2, t_3, t_1

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

	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

Parameterized Ranking Function

PRF$^\omega$(h): Weight Function : ω : rank$\rightarrow \mathbb{C}$

$$\Upsilon_\omega(t) = \sum_{i=1}^{h} \omega(i) \cdot \Pr(r(t) = i).$$

Positional probability:
Probability that t *is ranked at position* i

PRF^e(α): $\omega(i)=\alpha^i$ where α *can be a real or a complex*

$$\Upsilon_\omega(t) = \sum_{i \geq 1} \alpha^i \cdot \Pr(r(t) = i).$$

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

- E.g., ω(i)= 1 : Rank the tuples by probabilities
- E.g., ω(i)=1 for 1≤i≤k, ω(i)=0 for i>k: PT-k (i.e., ranking by *$Pr(r(t) \leq k)$*)
- Generalizes PT/GT-k, *U-Rank, E-Rank*
- We can easily incorporate the score as an feature

Parameterized Ranking Function

- **Another justification/intepretation of PRF (via expected utility maximization principle or consensus answers)**

- We can show that PT-k is equivalent to Consensus-Top-k under symmetric difference $T_1 \Delta T_2 = (T_1 \backslash T_2) \cup (T_2 \backslash T_1)$

- More generally, PRFw is equivalent to Consensus-Top-k under weighted symmetric difference

Computing Positional Probability

T_{i-1}: the set of tuples with scores higher than t_i
σ : Boolean indicator vector

$$\begin{aligned}\Pr(r(t_i) = j) &= \Pr(t_i) \sum_{pw:|pw \cap T_{i-1}|=j-1} \Pr(pw) \\ &= \Pr(t_i) \sum_{\sigma:\sum_{l=1}^{i-1} \sigma_l = j-1} \prod_{l<i:\sigma_l=1} \Pr(t_l) \prod_{l<i:\sigma_l=0} (1 - \Pr(t_l))\end{aligned}$$

- Generating Function Method

$$\mathcal{F}(x) = \prod_{i=1}^{n} (a_i + b_i x)$$

- The coefficient of x^k : $\sum_{\beta:\sum_{i=1}^{n} \beta_i = k} \prod_{i:\beta_i=0} a_i \prod_{i:\beta_i=1} b_i$

Computing Positional Probability

T_{i-1}: { t_1, t_2, … … , t_{i-1} }

- Generating Function Method

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

  - The coefficient of x^k : *Pr(r(t_i)=k)*

- **Algorithm:**
  - For i=1 to n
    - Construct $\mathcal{F}^i(x)$
    - *Expand* $\mathcal{F}^i(x) = \sum_{j=1}^{n} \Pr(r(t_i) = j) x^j$
    - $\Upsilon(t_i) = \sum_{j=1}^{n} \omega(t_i, j) \Pr(r(t_i) = j)$

Expand from scratch
$O(n^2)$

$O(n^3)$ overall

Computing Positional Probability

T_{i-1}: { t_1, t_2, … … , t_{i-1} }

- Generating Function Method

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

  - The coefficient of x^k : *Pr(r(t_i)=k)*
- **Algorithm:**
  - For i=1 to n
    - Construct $\mathcal{F}^i(x)$
    - Expand $\mathcal{F}^i(x) = \sum_{j=1}^{n} \Pr(r(t_i) = j) x^j$
    - $\Upsilon(t_i) = \sum_{j=1}^{n} \omega(t_i, j) \Pr(r(t_i) = j)$

Can be improved to O(n)

O(n^2) overall

Computing PRFe

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

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

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

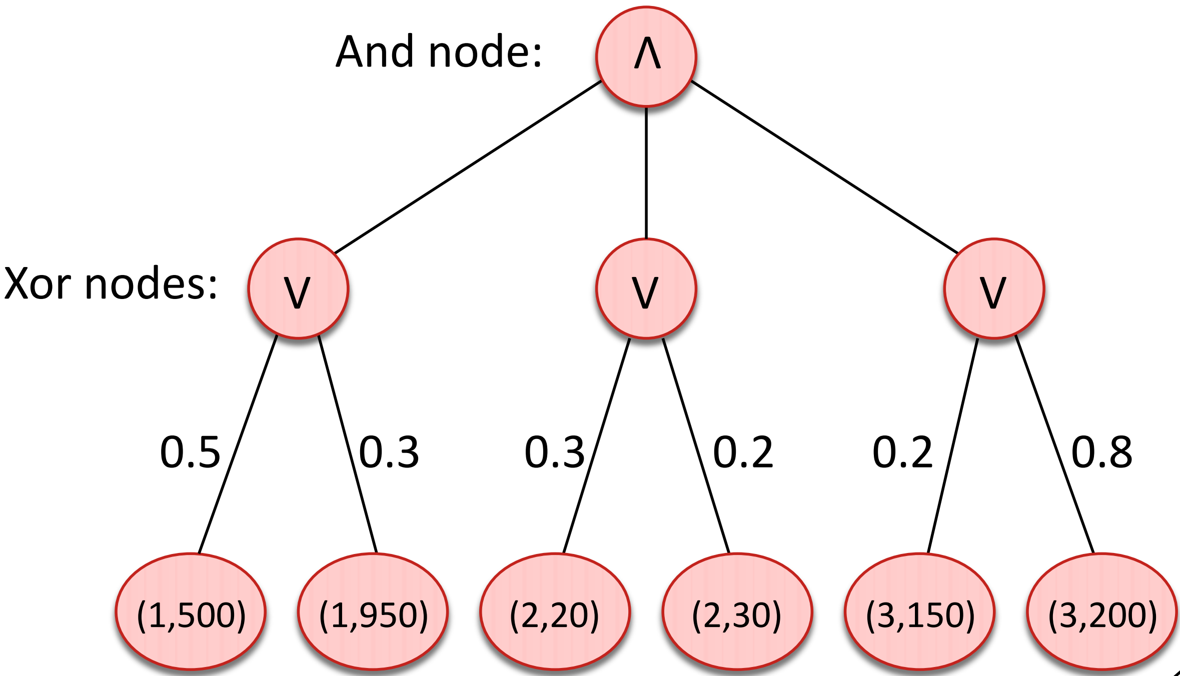
- For special weight functions, we do not even need to compute the positional probabilities Pr(r(t)=k)

- O(nlogn) for PRFe (exponential functions) and Exp-rank (linear functions) [Cormode, Li, Yi. ICDE'09]

- We can use sum of complex exponentials (Fourier transform) to approximate any weight functions.

Probabilistic And/Xor Trees

- Capture two types of correlations: mutual exclusivity and coexistence.
- Generalize x-tuples which can model only mutual exclusivity

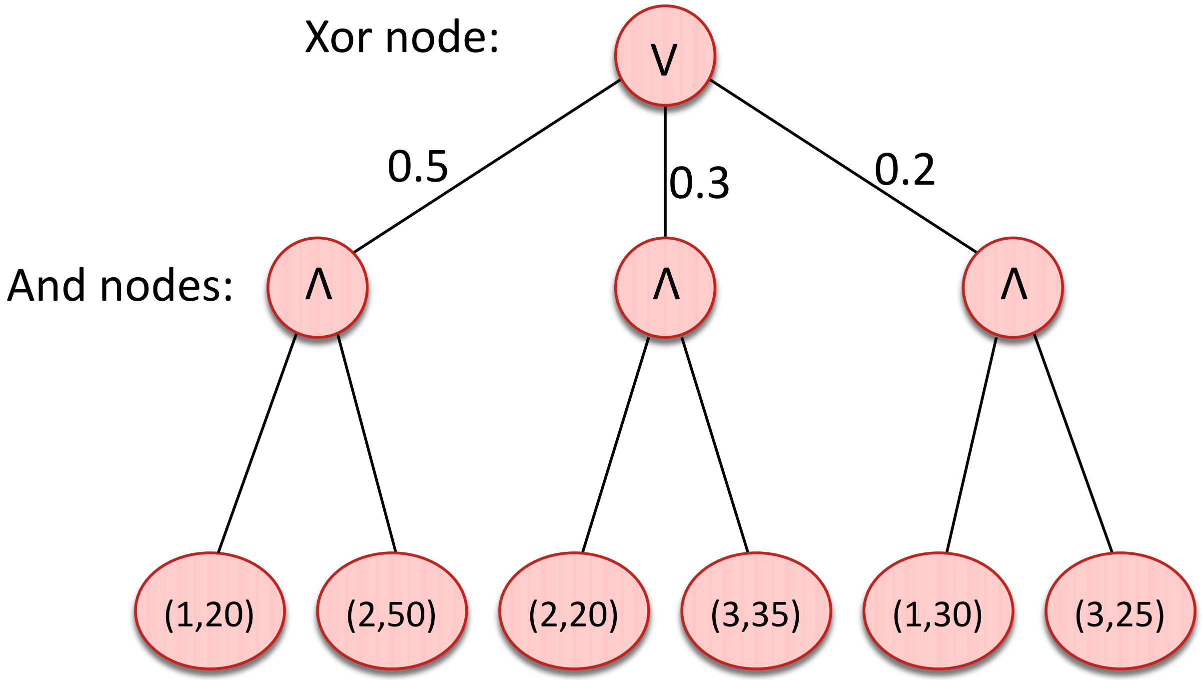


Possible Worlds	Pr
(3,150)	**0.02**
(3,200)	0.08
…….	
(1,500);(2,20); (3,150)	0.03
(1,950);(2,20); (3,150)	0.018
…….	

(1-0.5-0.3)*(1-0.3-0.2)*0.2=0.02

Probabilistic And/Xor Trees

- And/Xor trees can represent any finite set of possible worlds (not necessarily compact).

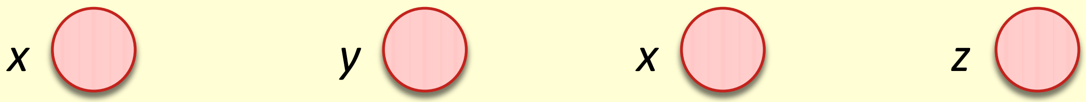


Possible Worlds	Pr
(1,20);(2,50)	0.5
(2,20);(3,35)	0.3
(1,30);(3,25)	0.2

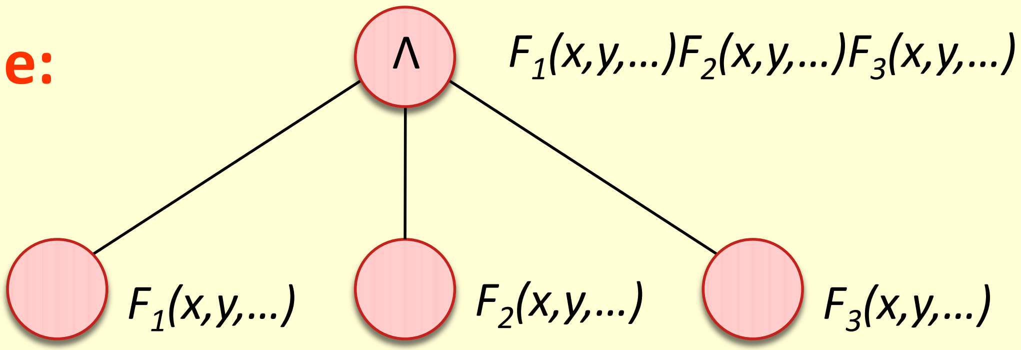
Computing Probabilities on And/Xor Trees

Generating Function Method:

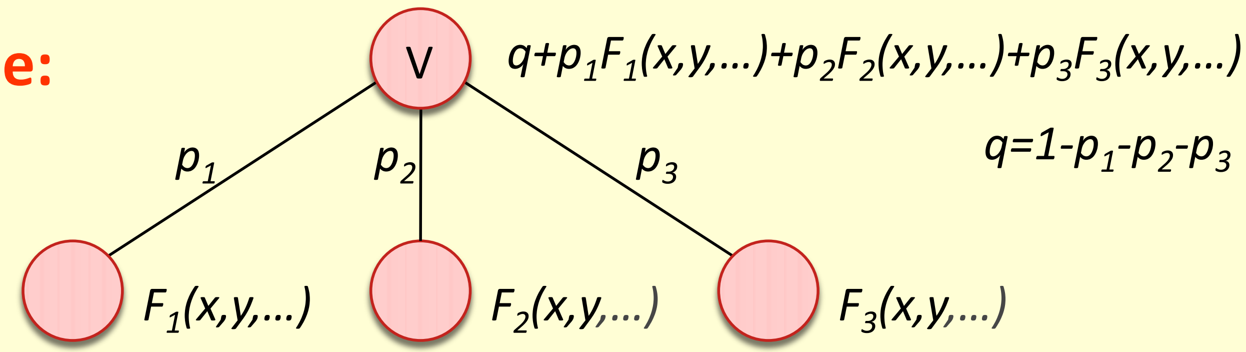
Leaves:



And Node:



Xor Node:



Computing Probabilities on And/Xor Trees

Generating Function Method:

Root: $F(x,y,\ldots)=\sum_{ij\ldots} c_{ij\ldots} x^i y^j \ldots$

THM: The coefficient $c_{ij\ldots}$ of the term $x^i y^j \ldots$

= total prob. of the possible worlds which contain

— i tuples annotated with x,

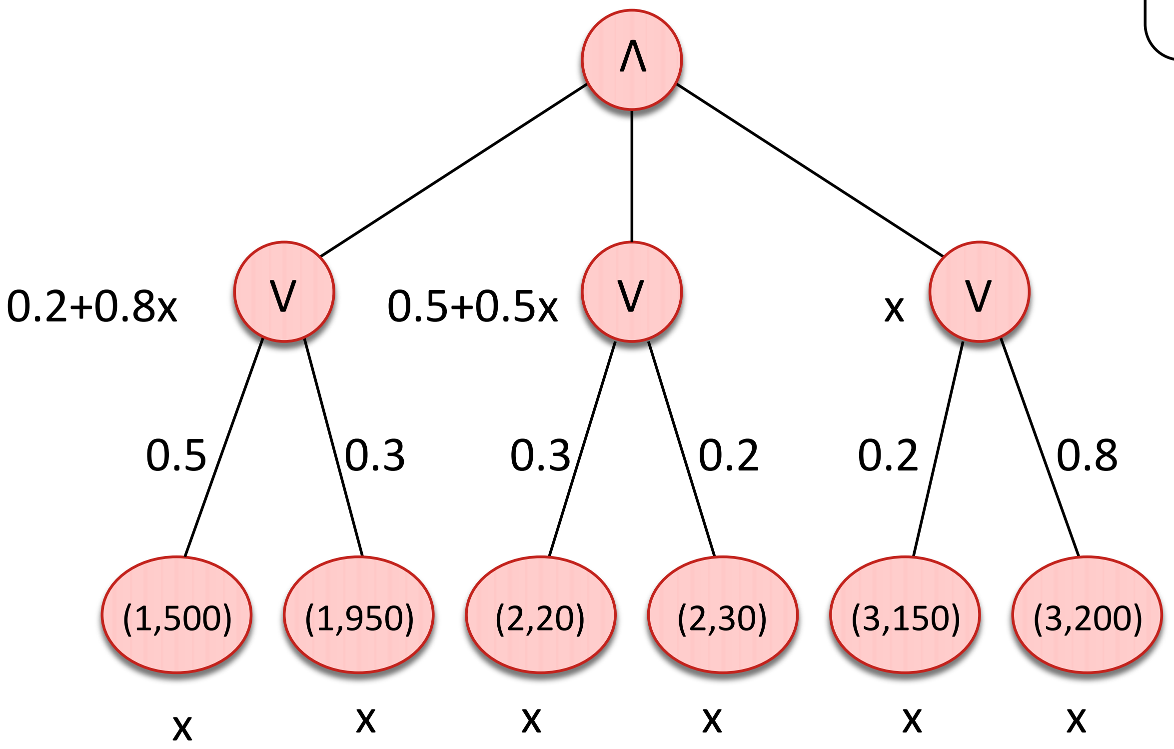
— j tuples annotated with y,......

Computing Probabilities on And/Xor Trees

Example: Computing the prob. dist. of the size of the pw

$(0.2+0.8x)(0.5+0.5x)x = 0.4\,x^3+0.5\,x^2+0.1\,x \Longrightarrow$

Pr(|pw|=3)=0.4
Pr(|pw|=2)=0.5
Pr(|pw|=1)=0.1



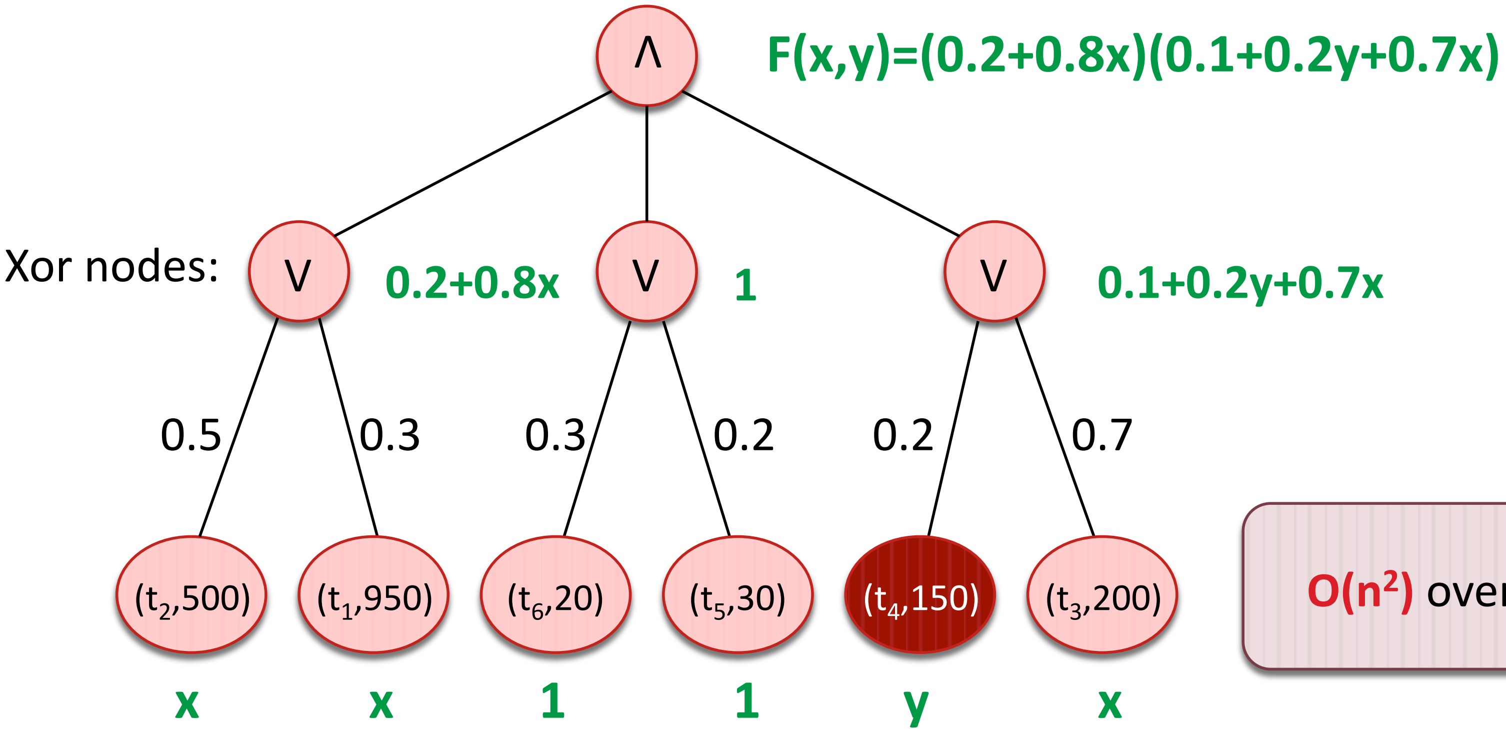
Computing PRF: And/Xor Trees

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 $x^{j-1}y$



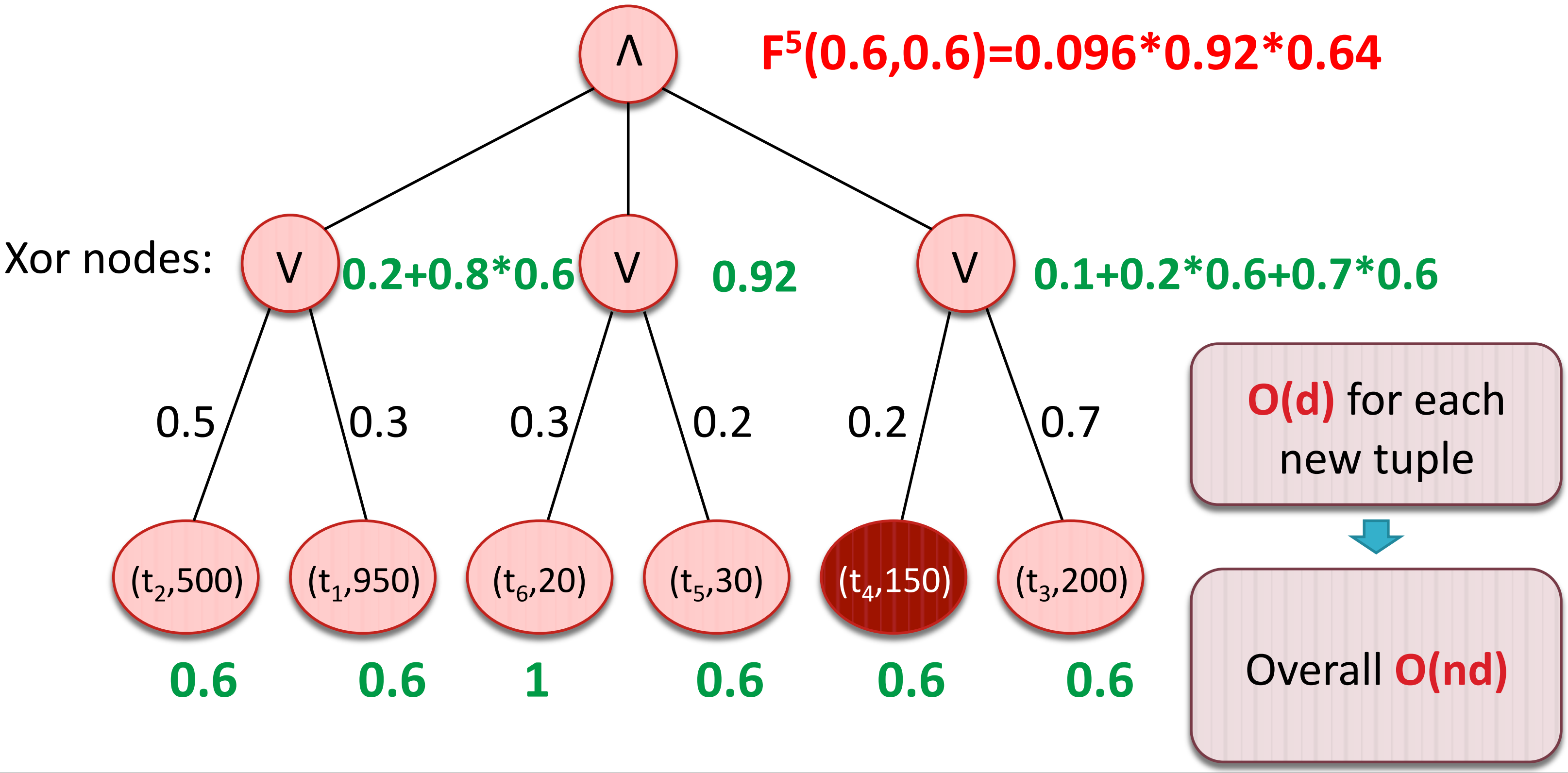
O(n²) overall

Computing $PRF^e(\alpha)$: And/Xor Trees

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

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 **$F^5(0.6,0.6)$**



Summary of Results

$PRF^w(h)$:

- Independent tuples: $O(nh+n\log n)$
  - Previous results for U-Rank: $O(n^2h)$ [Soliman et al. ICDE'07], $O(nh+n\log n)$ [Yi et al. TKDE'09]
  - Previous results for PT-k: $O(nh+n\log n)$ [Hua et al. SIGMOD'08]
- And/Xor trees: $O(dnh+n\log n)$ (d is the height of the tree, d=2 for x-tuples)
  - Previous results for U-Rank over x-tuples: $O(n^2h)$ [Soliman et al. ICDE'07], $O(n^2h)$ [Yi et al. TKDE'09]
  - Previous results for PT-k over x-tuples: $O(n^2h)$ [Hua et al. SIGMOD'08]

PRF^e:

- Independent tuples: $O(n\log n)$
- And/Xor trees: $O(nd+n\log n)$

Outline

- Ignoring Uncertainty?
  - Examples
  - Possible world semantics
- Beyond Expectation– expected utility theory
  - St Peterburg Paradox
  - Consensus Answer
- Queries over Probabilistic Data
  - Top-k queries
  - Other queries
- Stochastic Optimization
  - Stochastic Matching
  - Stochastic Knapsack

Problem Definition

Stochastic Matching

Given:

- A probabilistic graph $G(V,E)$.
- Existential prob. p_e for each edge e.
- Patience level t_v for each vertex v.

- **Probing** $e=(u,v)$: The only way to know the existence of e.
  - We can probe (u,v) only if $t_u>0, t_v>0$.
  - If e indeed exists, we should add it to our matching.
  - If not, $t_u = t_u - 1$, $t_v = t_v - 1$.

[Chen, Immorlica, Karlin, Mahdian, and Rudra. 'ICALP09]

[Bansal, Gupta, L, Mestre, Nagarajan, Rudra. ESA 10, Algorithmica 11]

Problem Definition

- Output: A strategy to probe the edges
  - Edge-probing: an (adaptive or non-adaptive) ordering of edges.
  - Matching-probing: k rounds; In each round, probe a set of disjoint edges
- Objectives:
  - Unweighted: Max. *E[cardinality of the matching]*.
  - Weighted: Max. *E[weight of the matching]*.

Motivations

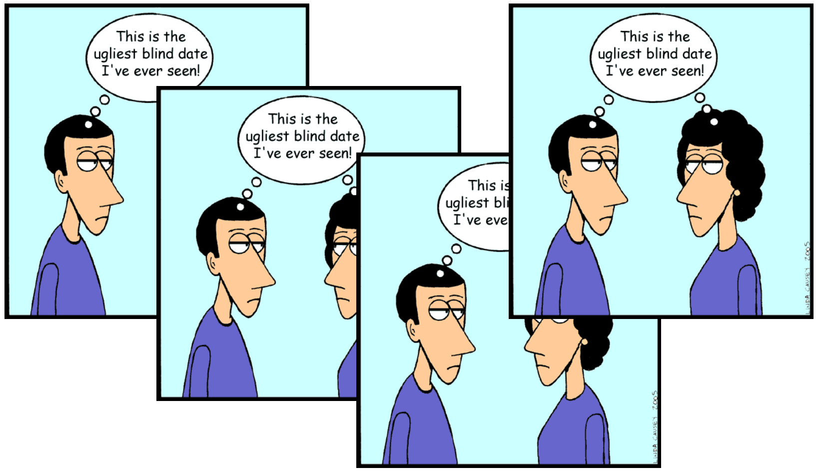
- **Online dating**
  - Existential prob. p_e : estimation of the success prob. based on users' profiles.

Motivations

- **Online dating**
  - Existential prob. p_e : estimation of the success prob. based on users' profiles.
  - Probing edge $e=(u,v)$: u and v are sent to a date.

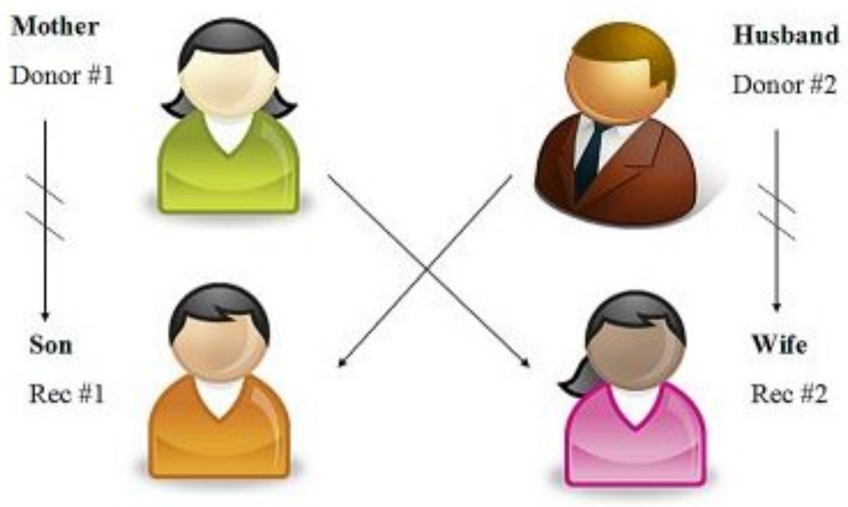
Motivations

- **Online dating**
  - Existential prob. p_e : estimation of the success prob. based on users' profiles.
  - Probing edge $e=(u,v)$: u and v are sent to a date.
  - Patience level: obvious.



Motivations

- **Kidney exchange**
  - Existential prob. p_e : estimation of the success prob. based on blood type etc.
  - Probing edge $e=(u,v)$: the crossmatch test (which is more expensive and time-consuming).



Motivations

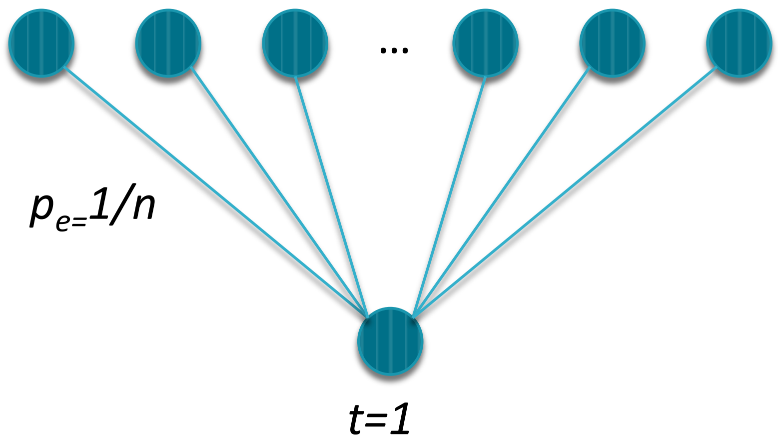
- This models the online AdWords allocation problem.

- This generalizes the stochastic online matching problem of [Feldman et al. '09, Bahmani et al. '10, Saberi et al '10] where p_e={0,1}.

Approximation Ratio

- We compare our solution against the optimal (adaptive) strategy (not the offline optimal solution).

- An example:



E[offline optimal] = $1-(1-1/n)^n \approx 1-1/e$

E[any algorithm] = $1/n$

A LP Upper Bound

- Variable y_e : Prob. that any algorithm probes *e.*

$$\text{maximize} \quad \sum_{e \in E} w_e \cdot x_e$$

$$\text{subject to} \quad \sum_{e \in \partial(v)} x_e \leq 1 \quad \forall v \in V$$

At most 1 edge in $\partial(v)$ is matched

$$\sum_{e \in \partial(v)} y_e \leq t_v \quad \forall v \in V$$

At most t_v edges in $\partial(v)$ are probed

$$x_e = p_e \cdot y_e \quad \forall e \in E$$

x_e: Prob. e is matched

$$0 \leq y_e \leq 1 \quad \forall e \in E$$

A Simple 8-Approximation

An edge (u,v) is *safe* if $t_u>0$, $t_v>0$ and neither u nor v is matched

Algorithm:

- Pick a permutation π on edges uniformly at random
- For each edge *e* in the ordering π, do:
  - If *e* is not safe then do not probe it.
  - If *e* is safe then probe it w.p. y_e/α.

A Simple 8-Approximation

Analysis:

Lemma: For any edge *(u,v)*, at the point when *(u,v)* is considered under π, *Pr(u loses its patience) ≤1/2α* .

Proof: Let *U* be #probes incident to *u* and before *e*.

$$\begin{aligned}\mathbb{E}[U] &= \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi \text{ AND } e \text{ is probed}] \\ &= \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi \text{ AND } e \text{ is safe}] \cdot \frac{y_e}{\alpha} \\ &\le \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi] \cdot \frac{y_e}{\alpha} \\ &= \textstyle\sum_{e\in\partial(u)} \frac{1}{2} \cdot \frac{y_e}{\alpha} \quad \le \quad \frac{t_u}{2\alpha}.\end{aligned}$$

$$\sum_{e\in\partial(v)} y_e \le t_v$$

By the Markov inequalit $\Pr[U \ge t_u] \le \frac{\mathbb{E}[U]}{t_u} \le \frac{1}{2\alpha}.$

A Simple 8-Approximation

Analysis:

Lemma: For any edge e=*(u,v)*, at the point when *(u,v)* is considered under π, *Pr(u is matched) ≤1/2α* .

Proof: Let *U* be #matched edges incident to u and before *e*.

$$\begin{aligned}\mathbb{E}[U] &= \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi \text{ AND } e \text{ is matched}] \\ &= \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi \text{ AND } e \text{ is safe}] \cdot \frac{y_e}{\alpha} \cdot p_e \\ &\le \textstyle\sum_{e\in\partial(u)} \Pr[\text{edge } e \text{ appears before } (u,v) \text{ in } \pi] \cdot \frac{y_e}{\alpha} \cdot p_e \\ &= \textstyle\sum_{e\in\partial(u)} \frac{1}{2} \cdot \frac{y_e}{\alpha} \cdot p_e \quad \le \quad \frac{1}{2\alpha}.\end{aligned}$$

$$\sum_{e\in\partial(v)} x_e \le 1$$

By the Markov inequality: $\Pr[U \ge 1] \le \mathbb{E}[U] \le \dfrac{1}{2\alpha}$

A Simple 8-Approximation

Analysis:

Theorem: The algorithm is a 8-approximation.

Proof: When e is considered,

Pr(e is not safe) ≤ Pr(u is matched)+ Pr(u loses its patience)+

Pr(v is matched)+ Pr(v loses its patience)

≤ 2/α

Therefore,

$$\mathbb{E}[\text{Our Solution}] = \sum_e w_e \Pr(e \text{ is safe}) \frac{y_e}{\alpha} p_e$$

$$\geq (1 - \frac{2}{\alpha}) \frac{1}{\alpha} \sum_e w_e y_e p_e$$

$$\geq \frac{1}{8} OPT \qquad (\alpha = 4)$$

Recall $\Sigma_e\, w_e\, y_e\, p_e$ is an upper bound of *OPT*

- We can improve the algorithm to achieve a 3-approximation (by a more careful selection of which edges to probe and a more careful analysis)

[Bansal, Gupta, L, Mestre, Nagarajan, Rudra. ESA 10, Algorithmica 11]

Outline

- Ignoring Uncertainty?
  - Examples
  - Possible world semantics
- Beyond Expectation– expected utility theory
  - St Peterburg Paradox
  - Consensus Answer
- Queries over Probabilistic Data
  - Top-k queries
  - Other queries
- Stochastic Optimization
  - Stochastic Matching
  - Stochastic Knapsack

Stochastic Knapsack

- A knapsack of capacity C
- A set of items, each having a fixed profit
- Known: Prior distr of size of each item.
- Each time we choose an item and place it in the knapsack irrevocably
- The actual size of the item becomes known after the decision
- Knapsack constraint: The total size of accepted items <= C
- Goal: maximize E[Profit]

[L, Yuan STOC13]

Motivation

- Scheduling with stochastic job length
  - The length/profit of each job is a random variable
  - The actual length/profit is unknown until we schedule to run it
  - Maximize the profit
- Related to the prophet inequality and secretary problem
  - Prophet inequality: We can decide to choose or discard a job AFTER we see its actual length/profit
    - Simplest case: choose only one job. E[our profit] >= E[max profit]/2
  - Secretary problem: We do NOT assume that the jobs follow any prob. distr. But instead assume they comes in a random order
    - Simplest case: choose only one job: Pr[we pick the best job]>= 1/e

Secretary Problem

- N candidates.
- Arrive in a random order. Must decide hire or not right away

Algo:

- Interview the first R=N/e candidates, but do not choose any one. Let x be the best candidate.
- Hire the first candidate who is better than x.

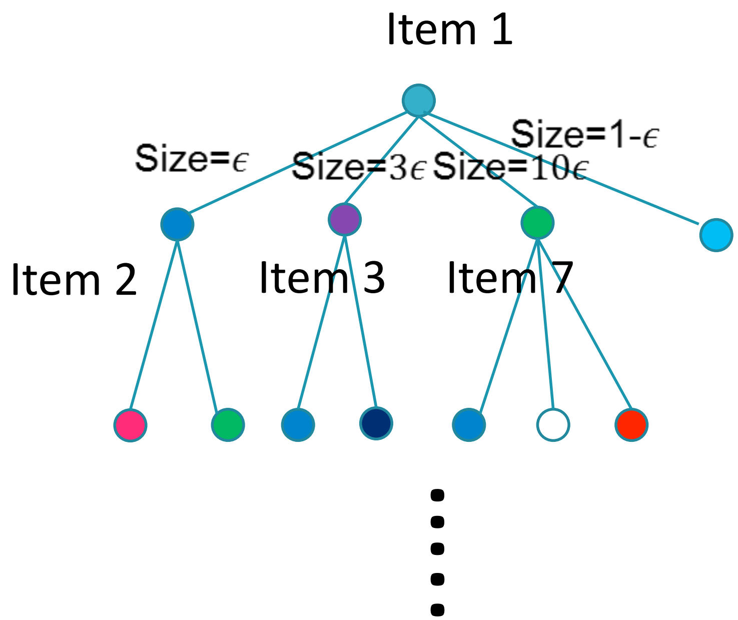
We can show Pr[we pick the best candidate] $\approx$ 1/e

A one line proof:

- Pr[we pick the best candidate] $\geq$ $\sum_{i=R+1 \text{ to } N} \Pr[i \text{ is the best}] \Pr\left[\text{the 2nd best of first } i \text{ candidates is in } [1, R]\right]$ $= \sum_{i=R+1 \text{ to } N} \frac{1}{n} \frac{R}{i} \approx 1/e$

Stochastic Knapsack

- Decision Tree



Exponential size!!!! (depth=n)

How to represent such a tree? Compact solution?

The problem is P-space complete

Stochastic Knapsack

Previous work

- 5-approx [Dean, Goemans, Vondrak. FOCS'04]
- 3-approx [Dean, Goemans, Vondrak. MOR'08]
- $(1+\epsilon, 1+\epsilon)$-approx [Bhalgat, Goel, Khanna. SODA'11]
- 2-approx [Bhalgat 12]
- 8-approx (size&profit correlation, cancellation) [Gupta, Krishnaswamy, Molinaro, Ravi. FOCS'11]

Our result:

$(1+\epsilon, 1+\epsilon)$-approx (size&profit correlation, cancellation)

2-approx (size&profit correlation, cancellation)

[Yuan, L. STOC'13]

Thanks.

Questions/Comments, please send to lijian83@mail.tsinghua.edu.cn

Prob. DB Research

- Our strength: support declarative queries, query processing and optimization techniques (indexing etc.).

- Current issues
  - Independence assumption.
  - Expressiveness/scalability trade off.
  - Different existing prototypes excels at different aspects (but not all).
  - Semantics not rich enough (need to go beyond expected values and probabilistic thresholds).