Foundations of Probabilistic Answers to Queries

Dan Suciu and Nilesh Dalvi

University of Washington

Databases Today are Deterministic

- An item either is in the database or is not
- A tuple either is in the query answer or is not
- This applies to all variety of data models:
  - Relational, E/R, NF2, hierarchical, XML, …

What is a Probabilistic Database ?

- “An item belongs to the database” is a probabilistic event
- “A tuple is an answer to the query” is a probabilistic event
- Can be extended to all data models; we discuss only probabilistic *relational* data

Two Types of Probabilistic Data

- Database is deterministic
  Query answers are probabilistic

- Database is probabilistic
  Query answers are probabilistic

Long History

Probabilistic relational databases have been studied from the late 80’s until today:

- Cavallo&Pitarelli:1987
- Barbara,Garcia-Molina, Porter:1992
- Lakshmanan,Leone,Ross&Subrahmanian:1997
- Fuhr&Roellke:1997
- Dalvi&S:2004
- Widom:2005

So, Why Now ?

Application pull:

- The need to manage imprecisions in data

Technology push:

- Advances in query processing techniques

The tutorial is built on these two themes

Application Pull

Need to manage imprecisions in data

- Many types: non-matching data values, imprecise queries, inconsistent data, misaligned schemas, etc, etc

The quest to manage imprecisions = major driving force in the database community

- Ultimate cause for many research areas: data mining, semistructured data, schema matching, nearest neighbor

Theme 1:

A large class of imprecisions in data can be modeled with probabilities

Technology Push

Processing probabilistic data is fundamentally more complex than other data models

- Some previous approaches sidestepped complexity

There exists a rich collection of powerful, non-trivial techniques and results, some old, some very recent, that could lead to practical management techniques for probabilistic databases.

Theme 2:

Identify the source of complexity,
present snapshots of non-trivial results,
set an agenda for future research.

Some Notes on the Tutorial

There is a *huge* amount of related work:

probabilistic db, top-k answers, KR, probabilistic reasoning, random graphs, etc, etc.

We left out many references

All references used are available in separate document

Tutorial available at: http://www.cs.washington.edu/homes/suciu

Requires TexPoint to view http://www.thp.uni-koeln.de/~ang/texpoint/index.html

Overview

Part I: Applications: Managing Imprecisions

Part II: A Probabilistic Data Semantics

BREAK

Part III: Representation Formalisms

Part IV: Theoretical foundations

Part V: Algorithms, Implementation Techniques

Summary, Challenges, Conclusions

Part I

Applications: Managing Imprecisions

Outline

1. Ranking query answers
2. Record linkage
3. Quality in data integration
4. Inconsistent data
5. Information disclosure

1. Ranking Query Answers

Database is deterministic

The query returns a *ranked list of tuples*

- User interested in top-k answers.

[Agrawal,Chaudhuri,Das,Gionis 2003]

The Empty Answers Problem

Query is overspecified: no answers

Example: try to buy a house in

```
SELECT *
FROM Houses
WHERE bedrooms = 4
    AND  style = ‘craftsman’
    AND  district = ‘View Ridge’
    AND  price < 400000
```

… good luck !

Today users give up and move to Baltimore

[Agrawal,Chaudhuri,Das,Gionis 2003]

Ranking:
Compute a similarity score between a tuple and the

```
Q = SELECT *
    FROM   R
    WHERE  A_1=v_1 AND … AND A_m=v_m
```

Query is a vector: $Q = (v_1, \ldots, v_m)$

Tuple is a vector: $T = (u_1, \ldots, u_m)$

Rank tuples by their TF/IDF similarity to the query Q

Includes partial matches

[Motro:1988,Dalvi&S:2004]

Similarity Predicates in SQL

Beyond a single table:
"Find the good deals in a neighborhood !"

```
SELECT *
FROM Houses x
WHERE x.bedrooms ~ 4  AND  x.style ~ 'craftsman' AND x.price ~ 600k
    AND  NOT EXISTS
        (SELECT *
         FROM Houses y
         WHERE x.district = y.district AND x.ID != y.ID
           AND y.bedrooms ~ 4 AND y.style ~ 'craftsman' AND y.price ~ 600k
```

Users specify similarity predicates with ~
System combines atomic similarities using <u>probabilities</u>

Types of Similarity Predicates

- String edit distances:
  - Levenstein distance, Q-gram distances
- TF/IDF scores
- Ontology distance / semantic similarity:
  - Wordnet
- Phonetic similarity:
  - SOUNDEX

[Theobald&Weikum:2002, Hung,Deng&Subrahmanian:2004]

[Hristidis&Papakonstantinou’2002,Bhalotia et al.2002]

Keyword Searches in Databases

Goal:

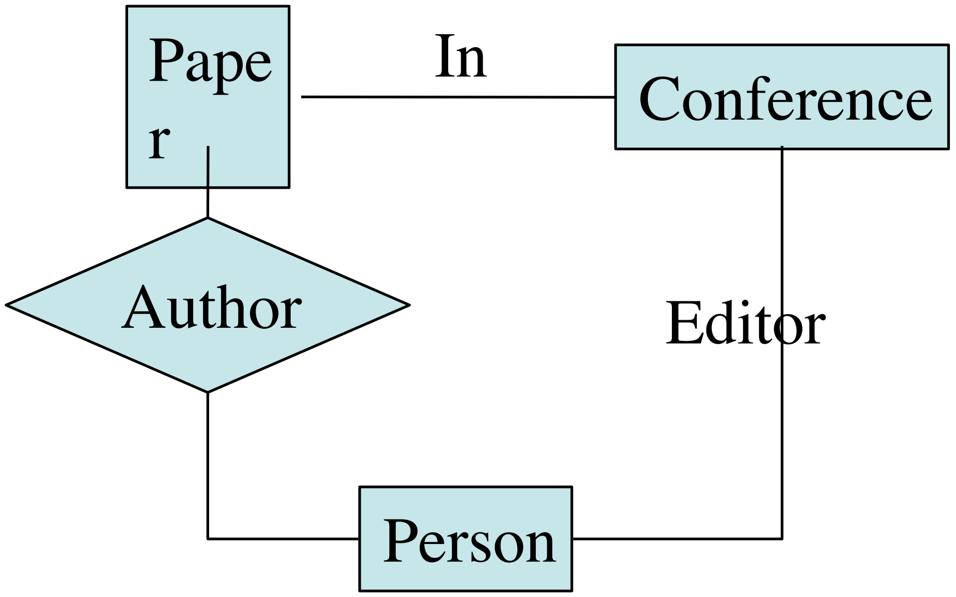
- Users want to search via keywords
- Do not know the schema

Techniques:

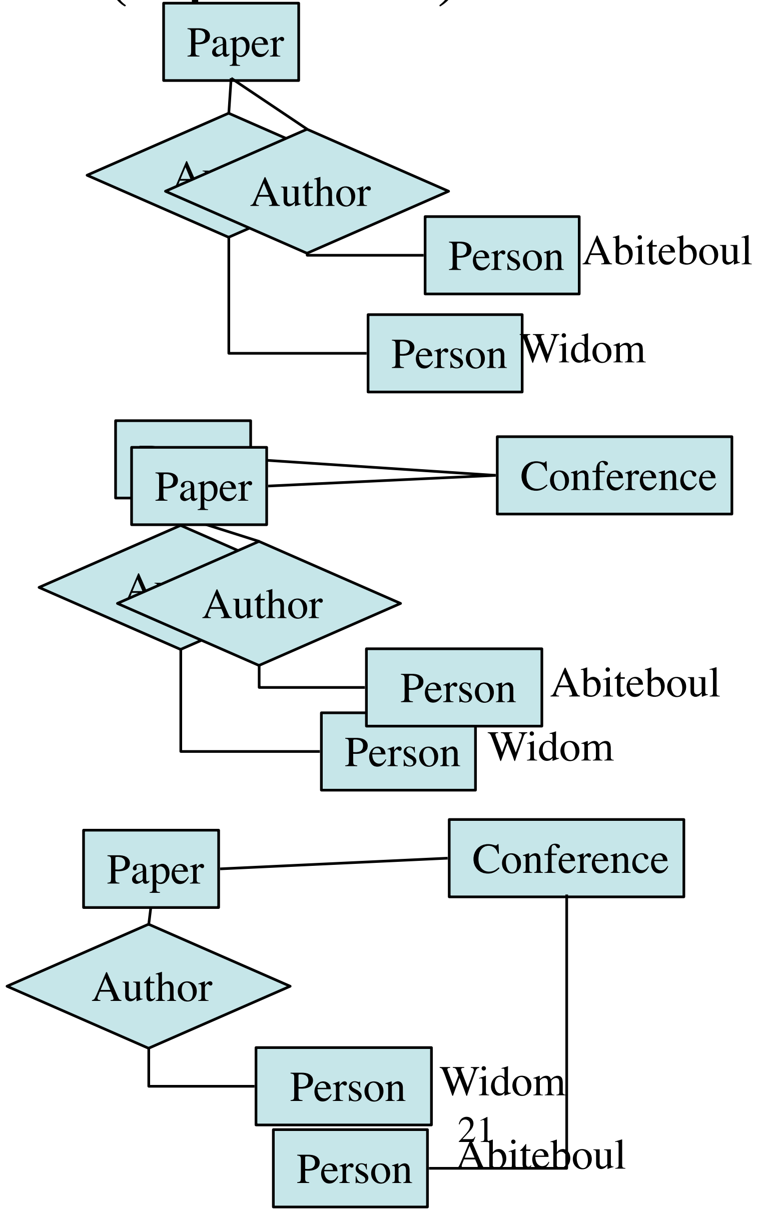
- Matching objects may be scattered across physical tables due to normalization; need *on the fly* joins
- Score of a tuple = number of joins, plus “prestige” based on indegree

[Hristidis,Papakonstantinou'2002]

Q = 'Abiteboul' and 'Widom'



Join sequences (tuple trees):



[Kiessling&Koster2002,Chomicki2002,Fagin&Wimmers1997]

More Ranking: User Preferences

Applications: personalized search engines, shopping agents, logical user profiles, “soft catalogs”

Two approaches:

- Qualitative $\Rightarrow$ Pareto semantics (deterministic)
- Quantitative $\Rightarrow$ alter the query ranking

Summary on Ranking Query Answers

Types of imprecision addressed:

Data is precise, query answers are imprecise:

- User has limited understanding of the data
- User has limited understanding of the schema
- User has personal preferences

Probabilistic approach would…

- Principled semantics for complex queries
- Integrate well with other types of imprecision

[Cohen: Tutorial]

2. Record Linkage

Determine if two data records describe same object

Scenarios:

- Join/merge two relations
- Remove duplicates from a single relation
- Validate incoming tuples against a reference

[Cohen: Tutorial; Fellegi&Sunder:1969]

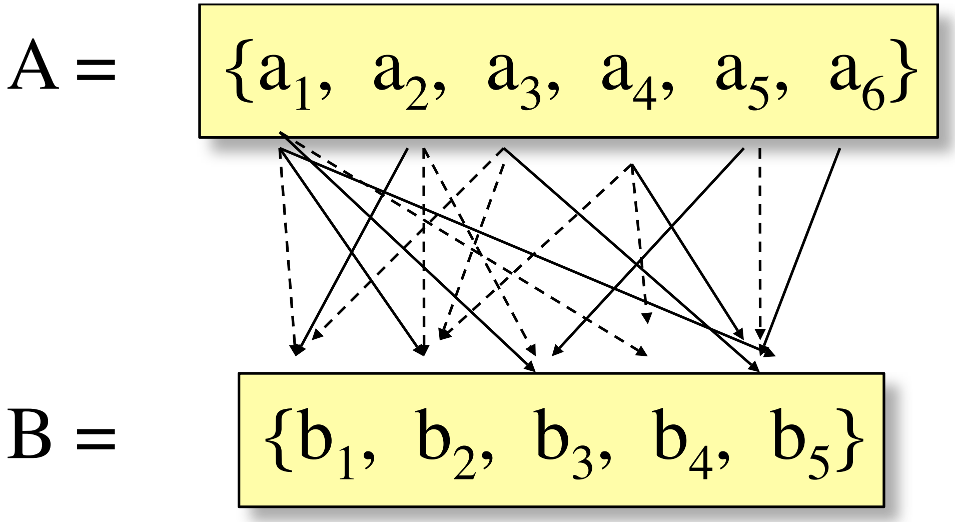
Fellegi-Sunter Model

A **probabilistic** model/framework

- Given two sets of records A, B:

Goal: partition $A \times B$ into:

- Match
- Uncertain
- Non-match



A = $\{a_1, a_2, a_3, a_4, a_5, a_6\}$
B = $\{b_1, b_2, b_3, b_4, b_5\}$

[Cohen: Tutorial]

Non-Fellegi Sunter Approaches

Deterministic linkage

- Normalize records, then test equality
  - E.g. for addresses
  - Very fast when it works
- Hand-coded rules for an “acceptable match”
  - E.g. “same SSN”;or “same last name AND same DOB”
  - Difficult to tune

Application: Data Cleaning, ETL

- Merge/purge for *large* databases, by sorting and clustering

[Hernandez,Stolfo:1995]

- Use of dimensional hierarchies in data warehouses and exploit co-occurrences

[Ananthakrishna,Chaudhuri,Ganti:2002]

- Novel similarity functions that are amenable to indexing

[Chaudhuri,Ganjam,Ganti,Motwani:2002]

- Declarative language to combine cleaning tasks

[Galhardas et al.:2001]

[Cohen:1998]

Application: Data Integration

WHIRL

- All attributes in in all tables are of type *text*
- Datalog queries with two kinds of predicates:
  - Relational predicates
  - Similarity predicates X ~ Y

Matches two sets on the fly, but
not really a "record linkage" application.

[Cohen:1998]

WHIRL

Example 1:

datalog

$$
\begin{aligned}
Q_1(*) \text{ :- } & P(\text{Company}_1, \text{Industry}_1), \\
& Q(\text{Company}_2, \text{Website}), \\
& R(\text{Industry}_2, \text{Analysis}), \\
& \quad \text{Company}_1 \sim \text{Company}_2, \\
& \quad \text{Industry}_1 \sim \text{Industry}_2
\end{aligned}
$$

Score of an answer tuple = product of similarities

[Cohen:1998]

WHIRL

Example 2 (with projection):

$Q_2(\text{Website})$:- $P(\text{Company}_1, \text{Industry}_1)$,
$Q(\text{Company}_2, \text{Website})$,
$R(\text{Industry}_2, \text{Analysis})$,
$\text{Company}_1 \sim \text{Company}_2$,
$\text{Industry}_1 \sim \text{Industry}_2$

Support(t) = set of tuples supporting the answer t

Depends on query plan !!

$$\text{score}(t) = 1 - \prod_{s \in \text{Support}(t)} (1-\text{score}(s))$$

Summary on Record Linkage

Types of imprecision addressed:

Same entity represented in different ways

- Misspellings, lack of canonical representation, etc.

A probability model would…

- Allow system to use the match probabilities: cheaper, on-the-fly
- But need to model complex probabilistic correlations: is one set a reference set ? how many duplicates are expected ?

[Florescu,Koller,Levy97;Chang,GarciaMolina00;Mendelzon,Mihaila01]

3. Quality in Data Integration

Use of probabilistic information to reason about soundness, completeness, and overlap of sources

Applications:

- Order access to information sources
- Compute confidence scores for the answers

[Mendelzon&Mihaila:2001

Global Historical Climatology Network

- Integrates climatic data from:
  - 6000 temperature stations
  - 7500 precipitation stations
  - 2000 pressure stations

<u>Soundness</u> of a data source:
what fraction of items are correct

<u>Completeness</u> data source:
what fractions of items it actually contains

[Mendelzon&Mihaila:2001

Global schema: **Temperature**
Station

Local as

S_1: V_1(s, lat, lon, c) $\neg$ **Station**(s, lat, lon c)

S_2: V_2(s, y, m, v) $\neg$
Temperature(s, y, m, v),
Station(s, lat, lon, “Canada”), y $\geq$ 1900

S_3: V_3(s, y, m, v) $\neg$
Temperature(s, y, m, v),
Station(s, lat, lon, “US”), y $\geq$ 1800

. . .

S_{8756}: . . .

[Florescu,Koller,Levy:1997;Mendelzon&Mihaila:2001]

Next, declare soundness and

S_2:

V_2(s, y, m, v) ¬
Temperature(s, y, m, v),
Station(s, lat, lon, “Canada”), y ≥ 1900

Soundness(V_2) ≥ 0.7
Completneess(V_2) ≥ 0.4

Precision

Recall

[Florescu,Koller,Levy:1997

Goal 1: completeness → order source accesses

S_5 S_{74} S_2 S_{31} . . .

[Mendelzon&Mihaila:2001]

Goal 2: soundness → query confidence

Q(y, v) :-
  Temperature(s, y, m, v), **Station**(s, lat, lon, "US"),
  $y \geq 1950, y \leq 1955, lat \geq 48, lat \leq 49$

Answer:

Year	**Value**	**Confidence**
1952	55^o F	0.7
1954	-22^o F	0.9
. . .	. . .	. . .

Summary: Quality in Data Integration

Types of imprecision addressed

Overlapping, inconsistent, incomplete data sources

- Data is probabilistic
- Query answers are probabilistic

They use already a probabilistic model

- Needed: complex probabilistic spaces. E.g. a tuple t in V_1 has 60% probability of also being in V_2
- Query processing still in infancy

[Bertosi&Chomicki:2003]

4. Inconsistent Data

Goal:
consistent query answers
from *inconsistent* databases

Applications:

- Integration of autonomous data sources
- Un-enforced integrity constraints
- Temporary inconsistencies

[Bertosi&Chomicki:2003]

The Repair Semantics

Consider all "repairs"

Key (?!?)

Name	Affiliation	State	Area
Miklau	UW	WA	Data security
Dalvi	UW	WA	Prob. Data
Balazinska	UW	WA	Data streams
Balazinska	MIT	MA	Data streams
Miklau	Umass	MA	Data security

Find people in State=WA $\Rightarrow$ Dalvi

Find people in State=MA $\Rightarrow$ $\emptyset$

Hi precision, but low recall

Alternative Probabilistic Semantics

Name	**Affiliation**	**State**	**Area**	**P**
Miklau	UW	WA	Data security	0.5
Dalvi	UW	WA	Prob. Data	1
Balazinska	UW	WA	Data streams	0.5
Balazinska	MIT	MA	Data streams	0.5
Miklau	Umass	MA	Data security	0.5

State=WA $\Rightarrow$ Dalvi, Balazinska(0.5), Miklau(0.5)

State=MA $\Rightarrow$ Balazinska(0.5), Miklau(0.5)

Lower precision, but better recall

Summary: Inconsistent Data

Types of imprecision addressed:

- Data from different sources is contradictory
- Data is uncertain, hence, arguably, probabilistic
- Query answers are probabilistic

A probabilistic would…

- Give better recall !
- Needs to support disjoint tuple events

5. Information Disclosure

Goal

- Disclose some information (V) while protecting private or sensitive data S

V=anonymized transactions

Applications:

- Privacy preserving data mining
- Data exchange
- K-anonymous data

V=standard view(s)

V=k-anonymous table

S = some atomic fact that is private

[Evfimievski,Gehrke,Srikant:03; Miklau&S:04;Miklau,Dalvi&S:05]

$\boxed{\Pr(S)}$ = a priori probability of S

$\boxed{\Pr(S \mid V)}$ = a posteriori probability of S

[Evfimievski,Gehrke,Srikant:03; Miklau&S:04;Miklau,Dalvi&S:05]

Information Disclosure

- If $\rho_1 < \rho_2$, a ρ_1, ρ_2 privacy breach:

$$\boxed{\Pr(S) \leq \rho_1 \quad \text{and } \Pr(S \mid V) \geq \rho_2}$$

- Perfect security:

$$\boxed{\Pr(S) = \Pr(S \mid V)}$$

- Practical security:

$$\boxed{\lim_{\text{domain size} ® \infty} \Pr(S \mid V) = 0}$$

Database size remains fixed

Summary: Information Disclosure

Is this a type of imprecision in data ?

- Yes: it's the adversary's uncertainty about the private data.
- The only type of imprecision that is good

Techniques

- Probabilistic methods: long history [Shannon'49]
- Definitely need conditional probabilities

Summary:
Information Disclosure

Important fundamental duality:

- Query answering: want Probability $\lesssim 1$
- Information disclosure: want Probability $\gtrsim 0$

They share the same fundamental concepts and techniques

Summary: Information Disclosure

What is required from the probabilistic model

- Don't know the possible instances
- Express the adversary's knowledge:
  - Cardinalities:
  - Correlations between values:
- Compute conditional probabilities

Size(**Employee**) $\simeq$ 1000

area-code $\rightsquigarrow$ **city**

6. Other Applications

- Data lineage + accuracy: Trio

[Widom:2005]

- Sensor data

[Deshpande, Guestrin,Madden:2004]

- Personal information management

Semex [Dong&Halevy:2005, Dong,Halevy,Madhavan:2005]
Heystack [Karger et al. 2003], Magnet [Sinha&Karger:2005]

- Using statistics to answer queries

[Dalvi&S;2005]

Summary on Part I: Applications

Common in these applications:

- Data in database and/or in query answer is uncertain, ranked; sometimes probabilistic

Need for common probabilistic model:

- Main benefit: uniform approach to imprecision
- Other benefits:
  - Handle complex queries (instead of single table TF/IDF)
  - Cheaper solutions (on-the-fly record linkage)
  - Better recall (constraint violations)

Part II

A Probabilistic Data Semantics

Outline

- The possible worlds model
- Query semantics

Possible Worlds Semantics

Attribute domains:

int, char(30), varchar(55), datetime

# values: 2^{32}, 2^{120}, 2^{440}, 2^{64}

Relational schema:

Employee(name:varchar(55), dob:datetime, salary:int)

# of tuples: $2^{440} \times 2^{64} \times 2^{23}$

# of instances: $2^{2^{440} \times 2^{64} \times 2^{23}}$

Database schema:

Employee(. . .), Projects(. . .), Groups(. . .), WorksFor(. . .)

# of instances: N (= BIG but finite)

The Definition

The set of all possible database instances:

$$\boxed{\mathrm{INST} = \{I_1, I_2, I_3, \ldots, I_N\}}$$

will use Pr or I^p interchangeably

Definition A *probabilistic database* I^p is a probability distribution on INST

$$\boxed{\Pr : \mathrm{INST} \to [0,1]} \qquad \text{s.t. } \sum_{i=1,N} \Pr(I_i) = 1$$

Definition A *possible world* is I s.t. Pr(I) > 0

$I^p =$

Example

Customer	Address	Product
John	Seattle	Gizmo
John	Seattle	Camera
Sue	Denver	Gizmo

$\Pr(I_1) = 1/3$

Customer	Address	Product
John	Boston	Gadget
Sue	Denver	Gizmo

$\Pr(I_2) = 1/12$

Customer	Address	Product
John	Seattle	Gizmo
John	Seattle	Camera
Sue	Seattle	Camera

$\Pr(I_3) = 1/2$

Customer	Address	Product
John	Boston	Gadget
Sue	Seattle	Camera

$\Pr(I_4) = 1/12$

Possible worlds = $\{I_1, I_2, I_3, I_4\}$

Tuples as Events

One tuple t $\Rightarrow$ event $t \in I$

$$\boxed{\Pr(t) = \sum_{I:\, t \in I} \Pr(I)}$$

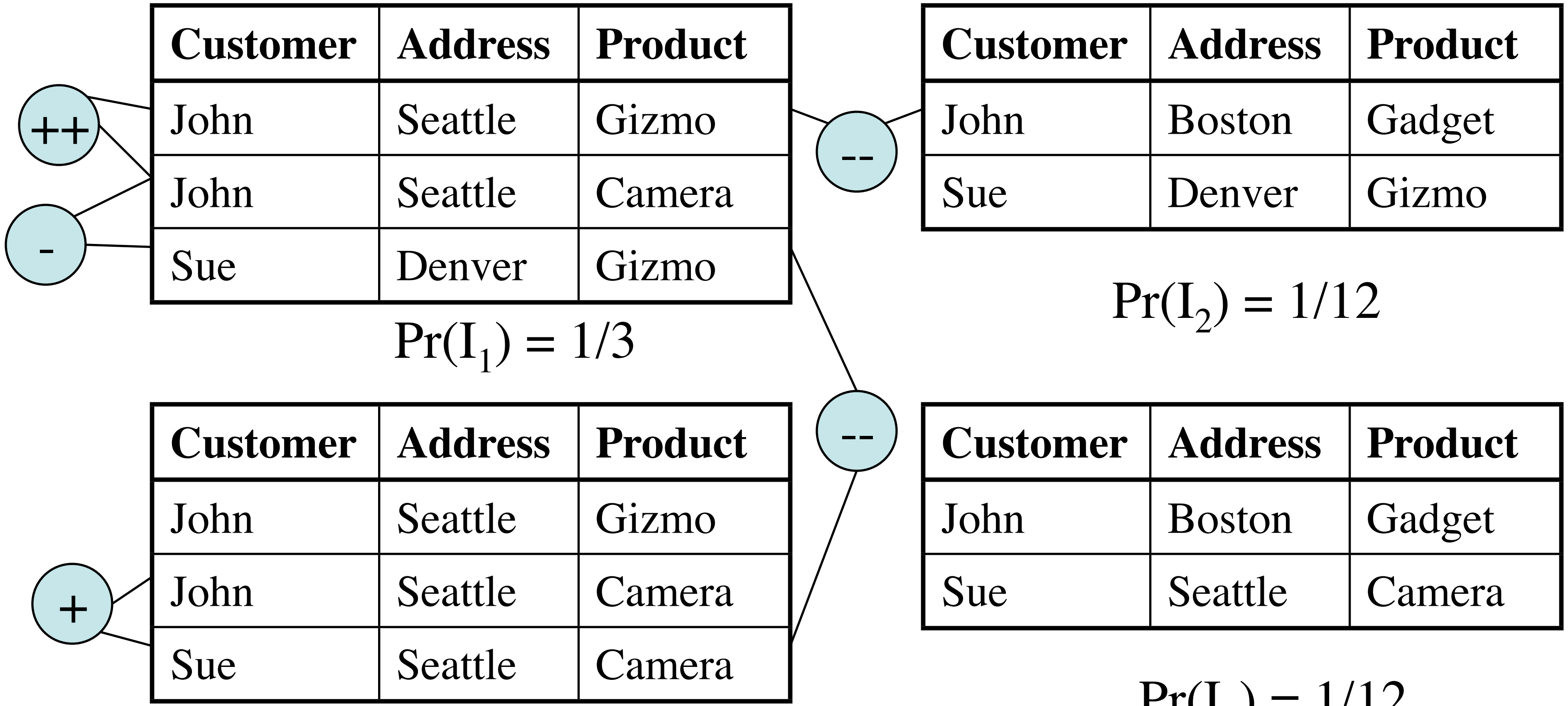
Two tuples t_1, t_2 $\Rightarrow$ event $t_1 \in I \wedge t_2 \in I$

$$\boxed{\Pr(t_1\ t_2) = \sum_{I:\, t_1 \in I \wedge t_2 \in I} \Pr(I)}$$

Tuple Correlation

Disjoint	$\Pr(t_1\ t_2) = 0$	--
Negatively correlated	$\Pr(t_1\ t_2) < \Pr(t_1)\ \Pr(t_2)$	-
Independent	$\Pr(t_1\ t_2) = \Pr(t_1)\ \Pr(t_2)$	0
Positively correlated	$\Pr(t_1\ t_2) > \Pr(t_1)\ \Pr(t_2)$	+
Identical	$\Pr(t_1\ t_2) = \Pr(t_1) = \Pr(t_2)$	++

$I^p =$ Example



Customer	Address	Product
John	Seattle	Gizmo
John	Seattle	Camera
Sue	Denver	Gizmo

$Pr(I_1) = 1/3$

Customer	Address	Product
John	Boston	Gadget
Sue	Denver	Gizmo

$Pr(I_2) = 1/12$

Customer	Address	Product
John	Seattle	Gizmo
John	Seattle	Camera
Sue	Seattle	Camera

$Pr(I_3) = 1/2$

Customer	Address	Product
John	Boston	Gadget
Sue	Seattle	Camera

$Pr(I_4) = 1/12$

Query Semantics

Given a query Q and a probabilistic database I^p, what is the meaning of $Q(I^p)$?

Query Semantics

Semantics 1: Possible Answers

A probability distributions on *sets of tuples*

$$\boxed{\forall A.\ \Pr(Q = A) = \textstyle\sum_{I \in \text{INST. } Q(I) = A} \Pr(I)}$$

Semantics 2: Possible Tuples

A probability function on *tuples*

$$\boxed{\forall t.\ \Pr(t \in Q) = \textstyle\sum_{I \in \text{INST. } t \in Q(I)} \Pr(I)}$$

Example: Query Semantics

Purchase^p

Name	**City**	**Product**
John	Seattle	Gizmo
John	Seattle	Camera
Sue	Denver	Gizmo
Sue	Denver	Camera

$Pr(I_1) = 1/3$

Name	**City**	**Product**
John	Boston	Gizmo
Sue	Denver	Gizmo
Sue	Seattle	Gadget

$Pr(I_2) = 1/12$

Name	**City**	**Product**
John	Seattle	Gizmo
John	Seattle	Camera
Sue	Seattle	Camera

$Pr(I_3) = 1/2$

Name	**City**	**Product**
John	Boston	Camera
[illegible]	[illegible]	[illegible]

$Pr(I_4) = 1/12$

```
SELECT DISTINCT x.product
FROM Purchasep x, Purchasep y
WHERE x.name = 'John'
      and x.product = y.product
      and y.name = 'Sue'
```

Possible answers semantics:

Answer set	**Probability**	
Gizmo, Camera	1/3	$Pr(I_1)$
Gizmo	1/12	$Pr(I_2)$
Camera	7/12	$P(I_3) + P(I_4)$

Possible tuples semantics:

Tuple	**Probability**	
Camera	11/12	$Pr(I_1)+P(I_3) + P(I_4)$
Gizmo	5/12	$Pr(I_1)+Pr(I_2)$

Special Case

Tuple independent probabilistic database

INST = $\mathcal{P}$(TUP)
N = 2^M

$\boxed{\text{TUP} = \{t_1, t_2, \ldots, t_M\}}$ = all tuples

$\boxed{\text{pr} : \text{TUP} \rightarrow [0,1]}$ No restrictions

$$\boxed{\Pr(I) = \prod_{t \in I} \text{pr}(t) \times \prod_{t \notin I} (1-\text{pr}(t))}$$

Tuple Prob. $\Rightarrow$ Possible Worlds

J =

Name	**City**	**pr**
John	Seattle	$p_1 = 0.8$
Sue	Boston	$p_2 = 0.6$
Fred	Boston	$p_3 = 0.9$

E[size(I^p)] = 2.3 tuples

I^p =

I_1: $\emptyset$

$(1-p_1)(1-p_2)(1-p_3)$

I_2:

Nam e	**City**
John	Seattl

$p_1(1-p_2)(1-p_3)$

I_3:

Nam e	**City**
Sue	Bosto

$(1-p_1)p_2(1-p_3)$

I_4:

Nam e	**City**
Fred	Bosto

$(1-p_1)(1-p_2)p_3$

I_5:

Nam e	**City**
John	Seattl
Sue	Bosto

$p_1p_2(1-p_3)$

I_6:

Nam e	**City**
John	Seattl
Fred	Bosto

$p_1(1-p_2)p_3$

I_7:

Nam e	**City**
Sue	Bosto
Fred	Bosto

$(1-p_1)p_2p_3$

I_8:

Nam e	**City**
John	Seattl
Sue	Bosto
Fred	Bosto

$p_1p_2p_3$

$\sum = 1$

Tuple Prob. ⇒ Query Evaluation

Name	City	pr
John	Seattle	p_1
Sue	Boston	p_2
Fred	Boston	p_3

Customer	Product	Date	pr
John	Gizmo	. . .	q_1
John	Gadget	. . .	q_2
John	Gadget	. . .	q_3
Sue	Camera	. . .	q_4
Sue	Gadget	. . .	q_5
Sue	Gadget	. . .	q_6
Fred	Gadget	. . .	q_7

```
SELECT DISTINCT x.city
FROM Person x, Purchase y
WHERE x.Name = y.Customer
      and y.Product = ‘Gadget’
```

Tuple	Probability
Seattle	$p_1(1-(1-q_2)(1-q_3))$
Boston	$1-(1-p_2(1-(1-q_5)(1-q_6))) \times (1-p_3 q_7)$

Summary of Part II

Possible Worlds Semantics

- Very powerful model: *any* tuple correlations
- Needs separate representation formalism

Summary of Part II

Query semantics

- Very powerful: *every* SQL query has semantics
- Very intuitive: from standard semantics

- Two variations, both appear in the literature

Summary of Part II

Possible answers semantics

- Precise
- Can be used to compose queries
- Difficult user interface

Possible tuples semantics

- Less precise, but simple; sufficient for most apps
- Cannot be used to compose queries
- Simple user interface

After the Break

Part III: Representation Formalisms

Part IV: Foundations

Part V: Algorithms, implementation techniques

Conclusions and Challenges

Part III

Representation Formalisms

Representation Formalisms

Problem

Need a good representation formalism

- Will be interpreted as possible worlds
- Several formalisms exists, but no winner

Main open problem in probabilistic db

Evaluation of Formalisms

- What possible worlds can it represent ?
- What probability distributions on worlds ?
- Is it closed under query application ?

Outline

A complete formalism:

- Intensional Databases

Incomplete formalisms:

- Various expressibility/complexity tradeoffs

[Fuhr&Roellke:1997]

Intensional Database

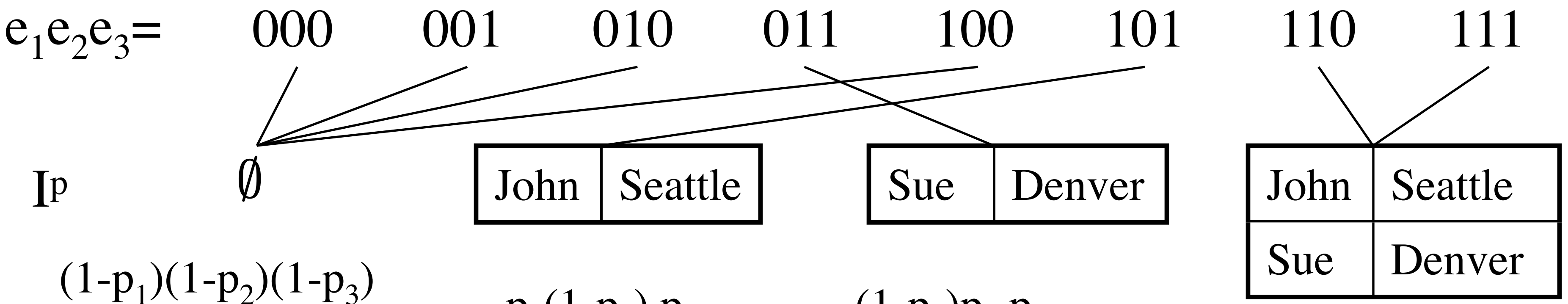
Atomic event ids	$e_1, e_2, e_3, \ldots$
Probabilities:	$p_1, p_2, p_3, \ldots \in [0,1]$
Event expressions: $\wedge, \vee, \neg$	$e_3 \wedge (e_5 \vee \neg e_2)$

Intensional probabilistic database J:
each tuple t has an event attribute t.E

Intensional DB ⇒ Possible Worlds

J =

Name	Address	E
John	Seattle	$e_1 \wedge (e_2 \vee e_3)$
Sue	Denver	$(e_1 \wedge e_2) \vee (e_2 \wedge e_3)$



$(1\text{-}p_1)(1\text{-}p_2)(1\text{-}p_3)$
$+(1\text{-}p_1)(1\text{-}p_2)p_3$
$+(1\text{-}p_1)p_2(1\text{-}p_3)$
$+p_1(1\text{-}p_2)(1\text{-}p_3$

$p_1(1\text{-}p_2)\ p_3$

$(1\text{-}p_1)p_2\ p_3$

$p_1p_2(1\text{-}p_3)$
$+p_1p_2p_3$

Possible Worlds $\Rightarrow$ Intensional DB

Name	Address
John	Seattle
John	Boston
Sue	Seattle

p_1

Name	Address
John	Seattle
Sue	Seattle

p_2

Name	Address
Sue	Seattle

p_3

Name	Address
John	Boston

p_4

$=I^p$

$E_1 = e_1$	$Pr(e_1) = p_1$
$E_2 = \neg e_1 \wedge e_2$	$Pr(e_2) = p_2/(1\text{-}p_1)$
$E_3 = \neg e_1 \wedge \neg e_2 \wedge e_3$	$Pr(e_3) = p_3/(1\text{-}p_1\text{-}p_2)$
$E_4 = \neg e_1 \wedge \neg e_2 \wedge \neg e_3 \wedge e_4$	$Pr(e_4) = p_4/(1\text{-}p_1\text{-}p_2\text{-}p_3)$

"Prefix code"

$\Rightarrow$

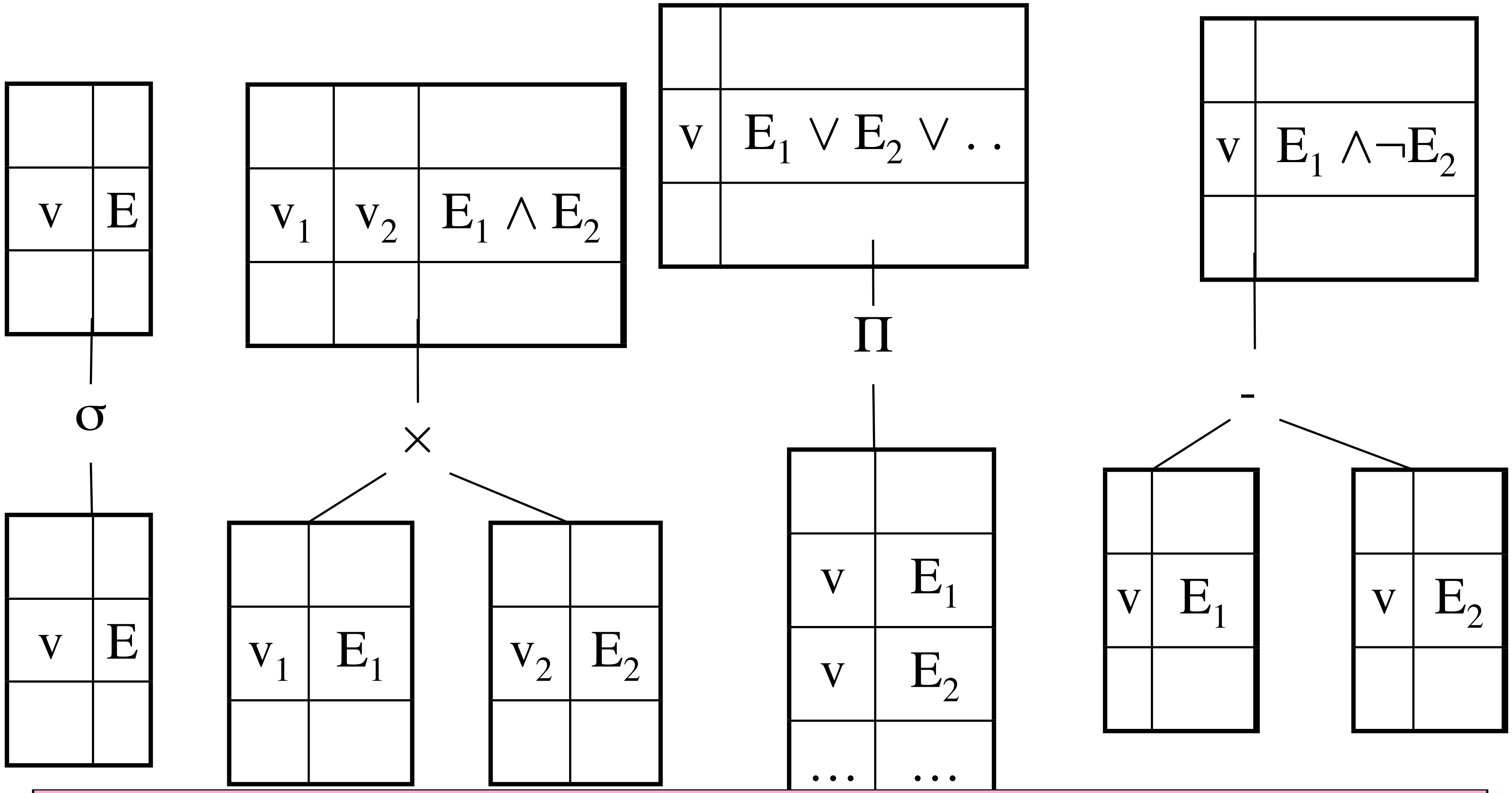
J =

Name	Address	E
John	Seattle	$E_1 \vee E_2$
John	Boston	$E_1 \vee E_4$
Sue	Seattle	$E_1 \vee E_2 \vee E_3$

Intesional DBs are complete

[Fuhr&Roellke:1997]

Closure Under Operators



One still needs to compute probability of event expression

Summary on Intensional Databases

Event expression for each tuple

- Possible worlds: any subset
- Probability distribution: any

Complete (in some sense) … but impractical

Important abstraction: consider restrictions

Related to c-tables [Imilelinski&Lipski:1984]

Restricted Formalisms

Explicit tuples

- Have a tuple template for every tuple that may appear in a possible world

Implicit tuples

- Specify tuples indirectly, e.g. by indicating how many there are

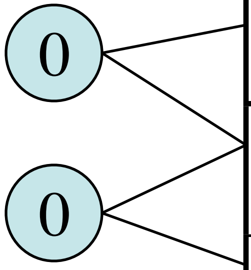
Explicit Tuples

Independent tuples

tuple = event

Atomic, distinct.
May use TIDs.

Name	**City**	**E**	**pr**
John	Seattle	e_1	0.8
Sue	Boston	e_2	0.2
Fred	Boston	e_3	0.6



independen
t

E[size(Customer)] = 1.6 tuples

Application 1: Similarity Predicates

Name	City	Profession
John	Seattle	statistician
Sue	Boston	musician
Fred	Boston	physicist

Step 1: evaluate ~ predicates

Cust	Product	Category
John	Gizmo	dishware
John	Gadget	instrument
John	Gadget	instrument
Sue	Camera	musicware
Sue	Gadget	microphone
Sue	Gadget	instrument
Fred	Gadget	microphone

```
SELECT DISTINCT x.city
FROM Person x, Purchase y
WHERE x.Name = y.Cust
        and y.Product = ‘Gadget’
        and x.profession ~ ‘scientist’
        and y.category ~ ‘music’
```

Application 1: Similarity Predicates

Name	**City**	**Profession**	**pr**
John	Seattle	statistician	p_1=0.8
Sue	Boston	musician	p_2=0.2
Fred	Boston	physicist	p_3=0.9

Step 1:
evaluate ~
predicates

Cust	**Product**	**Category**	**pr**
John	Gizmo	dishware	q_1=0.2
John	Gadget	instrument	q_2=0.6
John	Gadget	instrument	q_3=0.6
Sue	Camera	musicware	q_4=0.9
Sue	Gadget	microphone	q_5=0.7
Sue	Gadget	instrument	q_6=0.6
Fred	Gadget	microphone	q_7=0.7

SELECT DISTINCT x.city
FROM Person^p x, Purchase^p y
WHERE x.Name = y.Cust
    and y.Product = ‘Gadget’
    and **x.profession ~ ‘scientis**
    and **y.category ~ ‘music’**

Step 2:
evaluate rest
of query

Tuple	**Probability**
Seattle	$p_1(1-(1-q_2)(1-q_3))$
Boston	$1-(1-p_2(1-(1-q_5)(1-q_6)))$ $\times(1-p_3q_7)$

Explicit Tuples

Independent/disjoint tuples

Independent events: $e_1, e_2, \ldots, e_i, \ldots$

Split e_i into disjoint "shares" $e_i = e_{i1} \vee e_{i2} \vee e_{i3} \vee \ldots$

$e_{34}, e_{37} \Rightarrow$ disjoint events --

$e_{37}, e_{57} \Rightarrow$ independent events 0

Application 2: Inconsistent Data

Name	**City**	**Product**
John	Seattle	Gizmo
John	Seattle	Camera
John	Boston	Gadget
John	Huston	Gizmo
Sue	Denver	Gizmo
Sue	Seattle	Camera

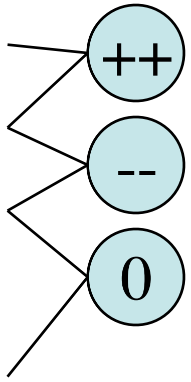
```
SELECT DISTINCT Product
FROM  Customer
WHERE City = ‘Seattle’
```

Step 1:
resolve violations

Name $\rightarrow$ City (violated)

Application 2: Inconsistent Data

Name	**City**	**Product**	**E**	**Pr**
John	Seattle	Gizmo	e_{11}	1/3
John	Seattle	Camera	e_{11}	1/3
John	Boston	Gadget	e_{12}	1/3
John	Huston	Gizmo	e_{13}	1/3
Sue	Denver	Gizmo	e_{21}	1/2
Sue	Seattle	Camera	e_{22}	1/2



```
SELECT DISTINCT Product
FROM  Customerᵖ
WHERE City = ‘Seattle’
```

Step 1:
resolve violations

Step 2:
evaluate query

Tuple	**Probability**
Gizmo	$p_{11} = 1/3$
Camera	$1-(1-p_{11})(1-p_{22}) = 2/3$

E[size(Customer)] = 2 tuples

[Barbara et al.92, Lakshmanan et al.97,Ross et al.05;Widom05]

Inaccurate Attribute Values

Name	Dept	Bonus	
John	Toy	Great	0.4
		Good	0.5
		Fair	0.1
Fred	Sales	Good	1.0

Inaccurate attributes

→

Name	Dept	Bonus	E	Pr
John	Toy	Great	e_{11}	0.4
John	Toy	Good	e_{12}	0.5
John	Toy	Fair	e_{13}	0.1
Fred	Sales	Good	e_{21}	1.0

Disjoint and/or independent events

Summary on Explicit Tuples

Independent or disjoint/independent tuples

- Possible worlds: subsets
- Probability distribution: restricted
- Closure: no

In KR:

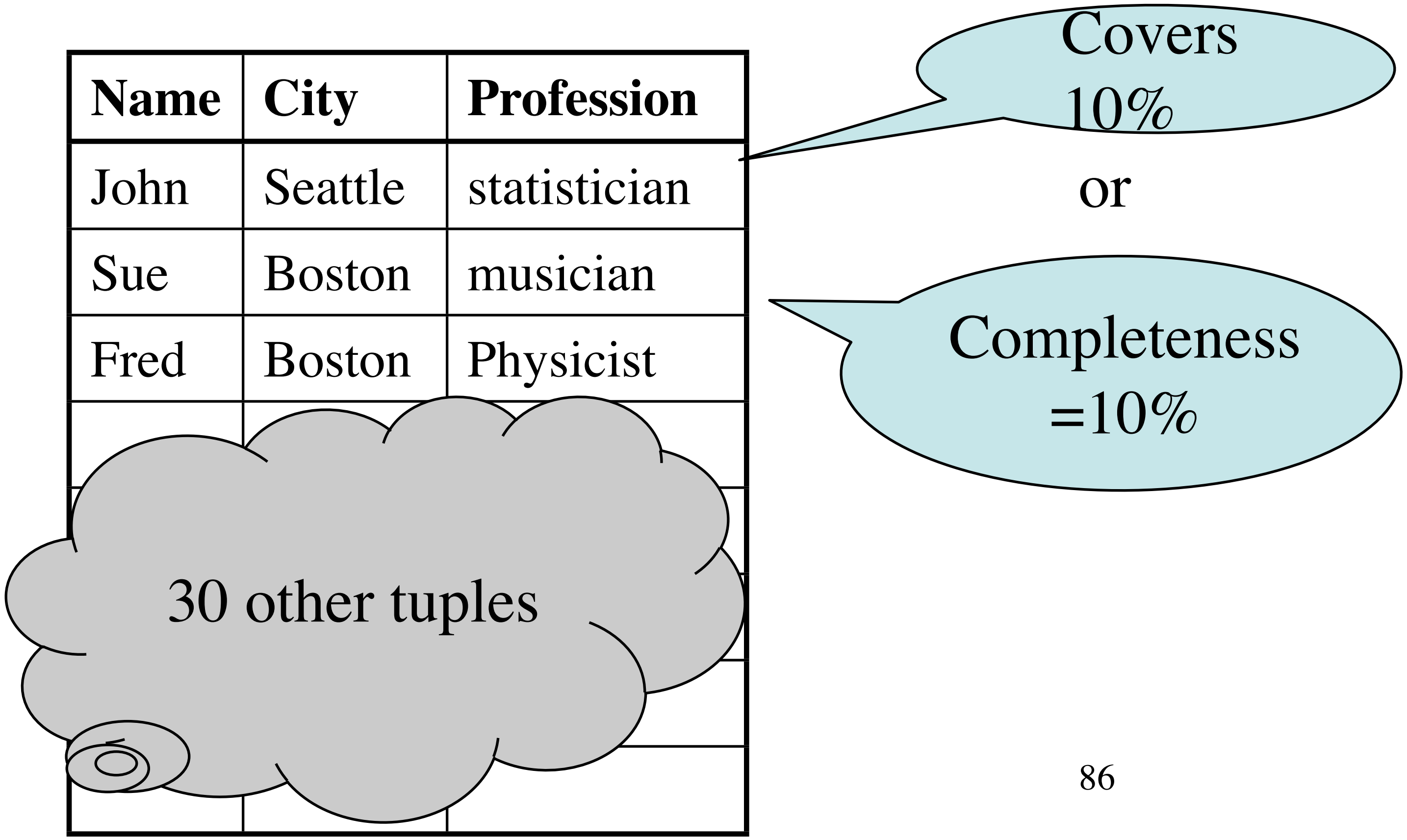
- Bayesian networks: disjoint tuples
- Probabilistic relational models: correlated tuples

[Friedman,Getoor,Koller,Pfeffer:1999]

[Mendelzon&Mihaila:2001,Widom:2005,Miklau&S04,Dalvi et al.05]

Implicit Tuples

“There are other, unknown tuples out there”

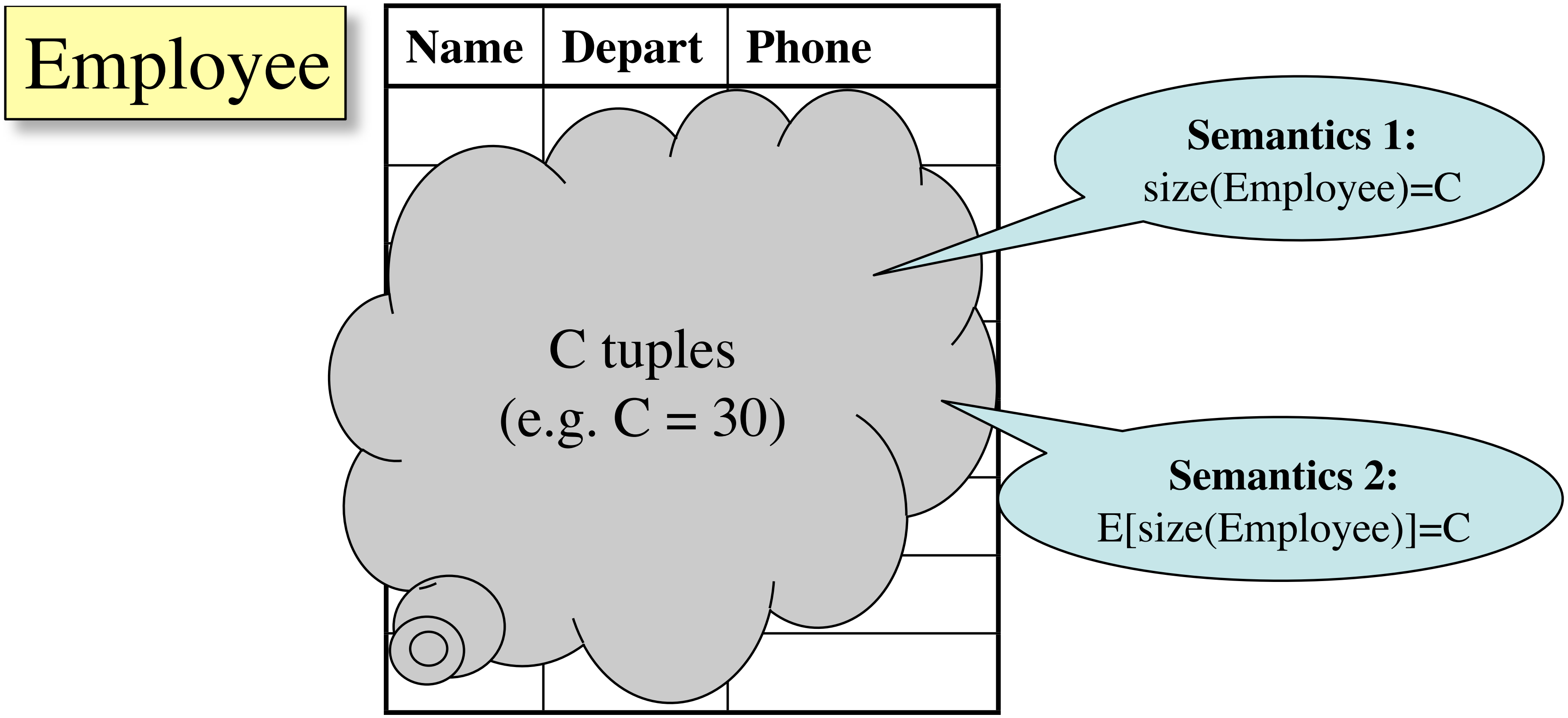


Name	City	Profession
John	Seattle	statistician
Sue	Boston	musician
Fred	Boston	Physicist

[Miklau,Dalvi&S:2005,Dalvi&S:2005]

Implicit Tuples

Statistics based:



We go with #2: the expected size is C

[Miklau,Dalvi&S:2005,Dalvi&S:2005]

Implicit Possible Tuples

Binomial distribution

Employee(name, dept, phone)

$n_1 = | D_{name} |$

$n_2 = | D_{dept} |$

$n_3 = | D_{phone} |$

$\forall t.\ Pr(t) = C / (n_1\ n_2\ n_3)$

$\Downarrow$

$E[\ Size(Employee)\ ] = C$

[Miklau,Dalvi&S:2005]

$Pr(name,dept,phone) = C / (n_1\ n_2\ n_3)$

Application 3: Information Leakage

S :- Employee("Mary", -, 5551234)

$Pr(S) \cong C/n_1n_3$

V_1 :- Employee("Mary", "Sales", -)

$Pr(S \mid V_1) \cong 1/\ n_3$

Practical secrecy

$Pr(SV_1) \cong C/n_1n_2n_3$

$Pr(V_1) \cong C/n_1n_2$

V_2 :- Employee(-, "Sales", 5551234)

$Pr(S \mid V_1V_2) \cong 1$

$Pr(SV_1V_2) \cong C/n_1n_2n_3$

$Pr(V_1\ V_2) \cong C/n_1n_2\ n_3$

Leakage

Summary on Implicit Tuples

Given by expected cardinality

- Possible worlds: any
- Probability distribution: binomial

May be used in conjunction with other formalisms

- *Entropy maximization*

[Domingos&Richardson:2004,Dalvi&S:2005]

Conditional probabilities become important

Summary on Part III: Representation Formalism

- Intensional databases:
  - Complete (in some sense)
  - Impractical, but…
  - …important practical restrictions
- Incomplete formalisms:
  - Explicit tuples
  - Implicit tuples
- We have not discussed query processing yet

Part IV

Foundations

Outline

- Probability of boolean expressions
- Query probability
- Random graphs

Needed for query processing

Probability of Boolean Expressions

$$E \;=\; X_1X_3 \vee X_1X_4 \vee X_2X_5 \vee X_2X_6$$

Randomly make each variable **true** with the following probabilities

$Pr(X_1) = p_1,\ Pr(X_2) = p_2,\ \ldots\ldots,\ Pr(X_6) = p_6$

What is Pr(E) ???

Answer: re-group cleverly

$$E = X_1 (X_3 \vee X_4) \vee X_2 (X_5 \vee X_6)$$

$$Pr(E)=1 - (1-p_1(1-(1-p_3)(1-p_4))) \\ (1-p_2(1-(1-p_5)(1-p_6)))$$

Now let's try this:

$$\boxed{E = X_1X_2 \lor X_1X_3 \lor X_2X_3}$$

No clever grouping seems possible.
Brute force:

X_1	X_2	X_3	E	Pr
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$(1-p_1)p_2p_3$
1	0	0	0	
1	0	1	1	$p_1(1-p_2)p_3$
1	1	0	1	$p_1p_2(1-p_3)$
1	1	1	1	$p_1p_2p_3$

$$\begin{aligned}\Pr(E)=&(1-p_1)p_2p_3 + \\ &p_1(1-p_2)p_3 + \\ &p_1p_2(1-p_3) + \\ &p_1p_2p_3\end{aligned}$$

Seems inefficient in general…

[Valiant:1979]

Complexity of Boolean Expression Probability

Theorem [Valiant:1979]
For a boolean expression E, computing Pr(E) is #P-complete

NP = class of problems of the form "is there a witness ?" SAT
#P = class of problems of the form "how many witnesses ?" #SAT

The decision problem for 2CNF is in PTIME
The counting problem for 2CNF is #P-complete

Summary on Boolean Expression Probability

- #P-complete
- It's hard even in simple cases: 2DNF
- Can do Monte Carlo simulation (later)

Query Complexity

Data complexity of a query Q:

- Compute $Q(I^p)$, for probabilistic database I^p

Simplest scenario only:

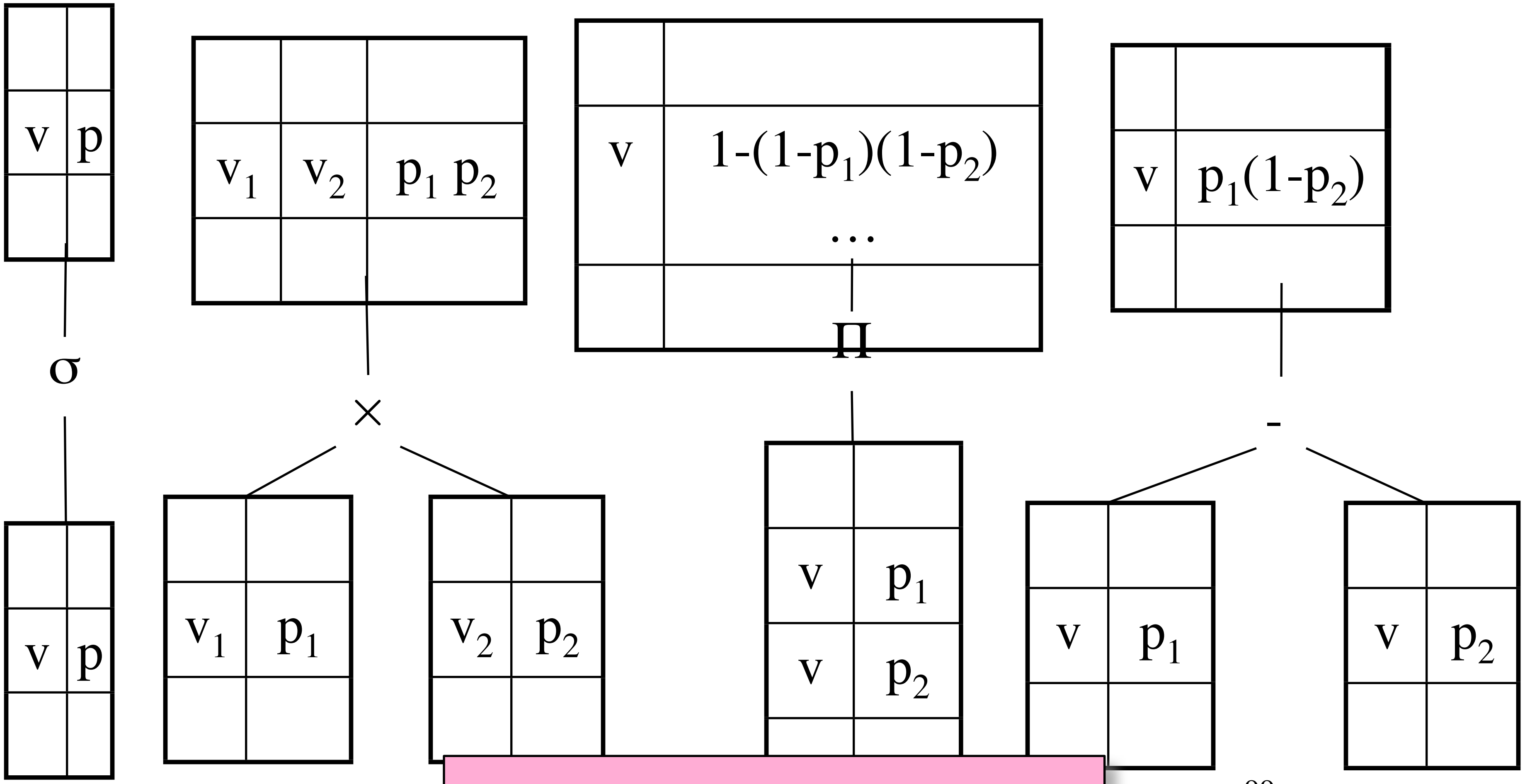
- Possible tuples semantics for Q
- Independent tuples for I^p

[Fuhr&Roellke:1997,Dalvi&S:2004]

or: $p_1 + p_2 + \dots$

Extensional Query Evaluation

Relational ops compute probabilities

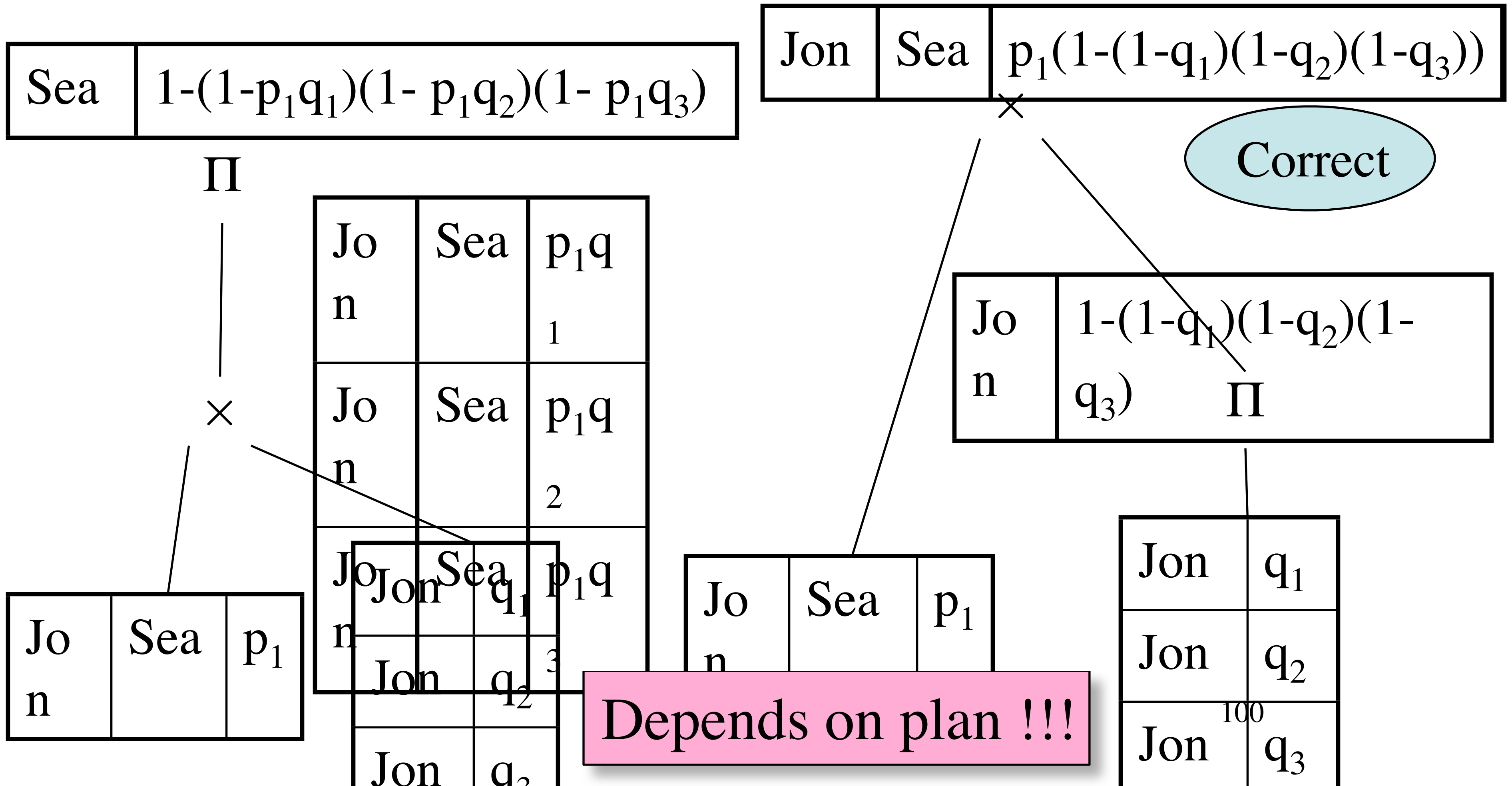


Data complexity: PTIME

[Dalvi&S:2004]

```
SELECT DISTINCT x.City
FROM Person^p x, Purchase^p y
WHERE x.Name = y.Cust
      and y.Product = 'Gadget'
```

Wrong !



Sea	$1-(1-p_1q_1)(1- p_1q_2)(1- p_1q_3)$

Jon	Sea	p_1q_1
Jon	Sea	p_1q_2
Jon	Sea	p_1q_3

Jon	Sea	p_1

Jon	q_1
Jon	q_2
Jon	q_3

Jon	Sea	$p_1(1-(1-q_1)(1-q_2)(1-q_3))$

Correct

Jon	$1-(1-q_1)(1-q_2)(1-q_3)$

Jon	Sea	p_1

Jon	q_1
Jon	q_2
Jon	q_3

Depends on plan !!!

[Dalvi&S:2004]

Query Complexity

Sometimes $\nexists$ correct extensional plan

Q_{bad} :- R(x), S(x,y), T(y)

Data complexity is #P complete

Theorem The following are equivalent
- Q has PTIME data complexity
- Q admits an extensional plan (and one finds it in PTIME)
- Q does not have Q_{bad} as a subquery

Summary on Query Complexity

Extensional query evaluation:

- Very popular
  - generalized to "strategies"

[Lakshmanan et al.1997]

- However, result depends on query plan !

General query complexity

- #P complete (not surprising, given #SAT)
- Already #P hard for very simple query (Q_{bad})

Probabilistic database have high query complexity

[Erdos&Reny:1959,Fagin:1976,Spencer:2001]

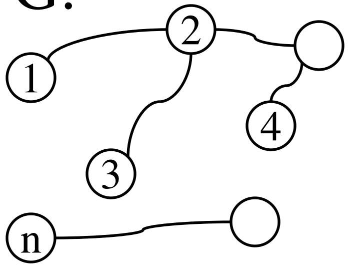
Random Graphs

Relation: $G(x,y)$

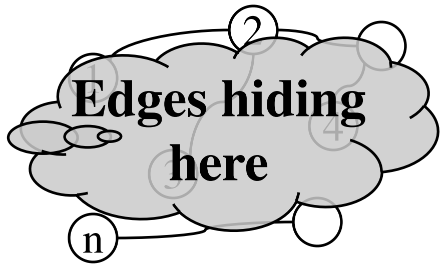
Domain: $D=\{1,2, \ldots, n\}$

G^p = tuple-independent $pr(t_1) = \ldots pr(t_M) = p$

Graph G:



Random graph G^p



Boolean query Q

What is $\lim_{n\to\infty} Q(G^p)$

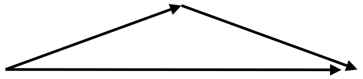
[Fagin:1976]

Fagin's 0/1 Law

Let the tuple probability be $p = 1/2$

Theorem [Fagin:1976,Glebskii et al.1969]
For every sentence Q in First Order Logic,
$\lim_{n\to\infty} Q(G^p)$ exists and is either 0 or 1

Examples

Holds almost surely: lim = 1	**Does not hold a.s.** lim = 0
$\forall x.\exists y.G(x,y)$	$\exists x.\forall y.G(x,y)$
$\exists x.\exists y.\exists z.\ G(x,y) \wedge G(y,z) \wedge G(x,z)$ 	$\forall x.\forall y.\ G(x,y)$

[Erdos&Reny:1959]

Erdos and Reny's Random Graphs

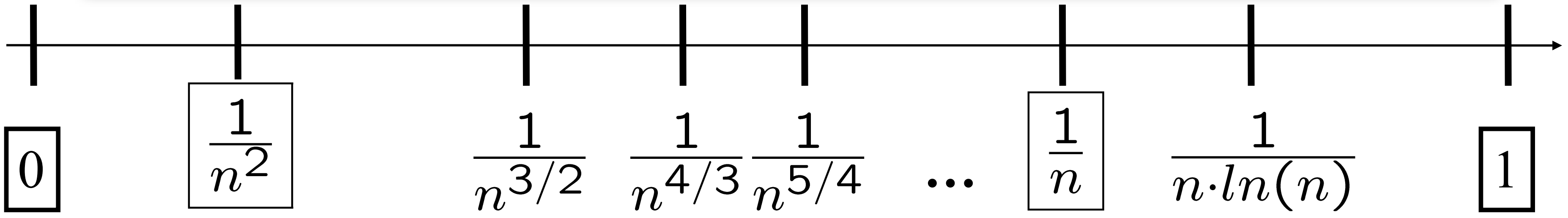
Now let $p = p(n)$ be a function of n

Theorem [Erdos&Ren]y:1959]
For any monotone Q, $\exists$ a threshold function t(n) s.t.:
if $p(n) \ll t(n)$ then $\lim_{n\to\infty} Q(G^p)=0$
if $p(n) \gg t(n)$ then $\lim_{n\to\infty} Q(G^p)=1$

[Erdos&Reny:1959; Spencer:2001]

The Evoluation of Random Graphs

The tuple probability p(n) “grows” from 0 to 1.
How does the random graph evolve ?



Remark: C(n) = E[Size(G)] $\simeq$ $n^2p(n)$

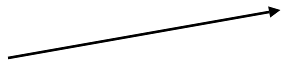
The expected size C(n) “grows” from 0 to n^2.
How does the random graph evolve ?

[Spencer:2001]

The Void

$p(n) \ll 1/n^2$

$C(n) \ll 1$

Contains almost surely	Does not contain almost surely
(nothing)	

The graph is empty

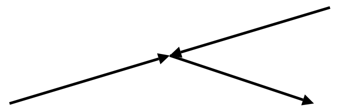
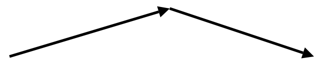
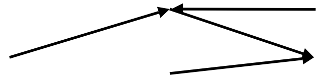
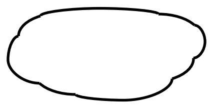
0/1 Law holds

[Spencer:2001]

On the k’th Day

$1/n^{1+1/(k-1)} \ll p(n) \ll 1/n^{1+1/k}$

$n^{1-1/(k-1)} \ll C(n) \ll n^{1-1/k}$

Contains almost surely	**Does not contain almost surely**
trees with $\leq$ k edges  	trees $>$ k edges  cycles 

The graph is disconnected

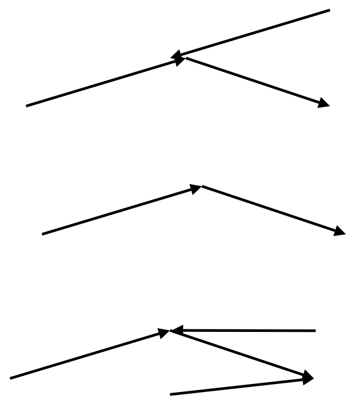
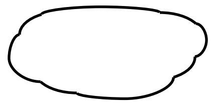
0/1 Law holds

[Spencer:2001]

On Day ω

$1/n^{1+\varepsilon} \ll p(n) \ll 1/n, \quad \forall\, \varepsilon > 0$

$n^{1-\varepsilon} \ll C(n) \ll n, \quad \forall\, \varepsilon > 0$

Contains almost surely	**Does not contain almost surely**
Any Tree 	cycles 

The graph is disconnected

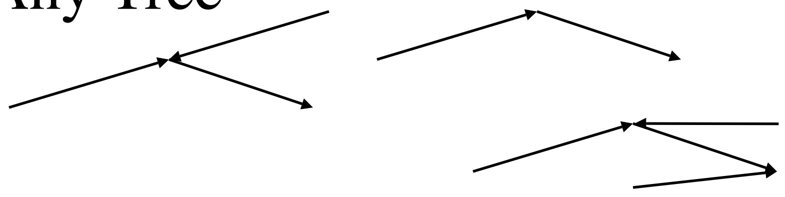
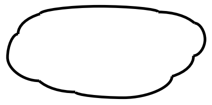
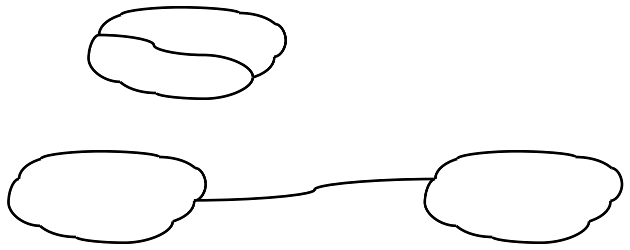
0/1 Law holds

[Spencer:2001]

Past the Double Jump (1/n)

$1/n \ll p(n) \ll \ln(n)/n$

$n \ll C(n) \ll n \ln(n)$

Contains almost surely	Does not contain almost surely
Any Tree  Any Cycle 	Any subgraph with k nodes and $\geq$ k+1 edges 

The graph is disconnected

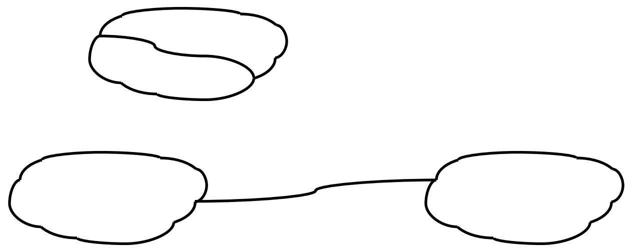
0/1 Law holds

[Spencer:2001]

Past Connectivity

$\ln(n)/n \ll p(n) \ll 1/n^{1-\varepsilon}, \forall \varepsilon$

$n \ln(n) \ll C(n) \ll n^{1+\varepsilon}, \forall \varepsilon$

Contains almost surely	**Does not contain almost surely**
Every node has degree $\geq$ k, for every k $\geq$ 0 Strange logic of random graphs !!	Any subgraph with k nodes and $\geq$ k+1 edges 

The graph is connected !

0/1 Law holds

[Spencer:2001]

Big Graphs

$p(n) = 1/n^{\alpha}, \ \alpha \in (0,1)$

$C(n) = n^{2-\alpha}, \alpha \in (0,1)$

α is irrational $\Rightarrow$ 0/1 Law holds

α is rational $\Rightarrow$ 0/1 Law does not hold

Fagin's framework: $\alpha = 0$ 0/1 Law holds

$p(n) = O(1)$

$C(n) = O(n^2)$

Summary on Random Graphs

- Very rich field
  - Over 700 references in [Bollobas:2001]
- Fascinating theory
  - Evening reading: the evolution of random graphs (e.g. from [Spencer:2001])

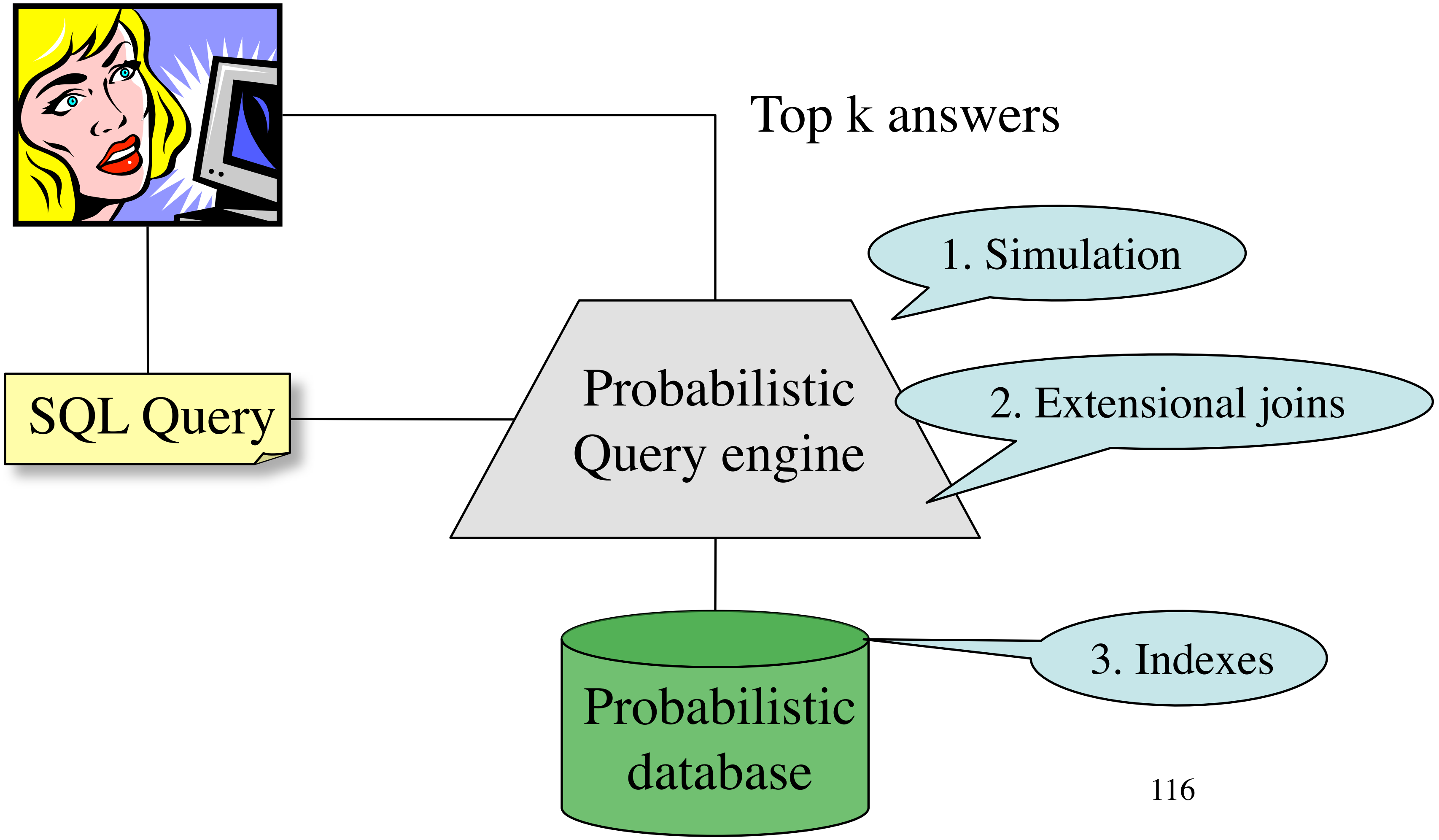
Summary on Random Graphs

- Fagin’s 0/1 Law: impractical probabilistic model
- More recent 0/1 laws for $p = 1/n^{\alpha}$ [Spencer&Shelah, Lynch]
- In practice: need precise formulas for $Pr(Q(I^p))$
  - Preliminary work [Dalvi,Miklau&S:04,Dalvi&S:05]

Part V

Algorithms, Implementation Techniques

Query Processing on a Probabilistic Database

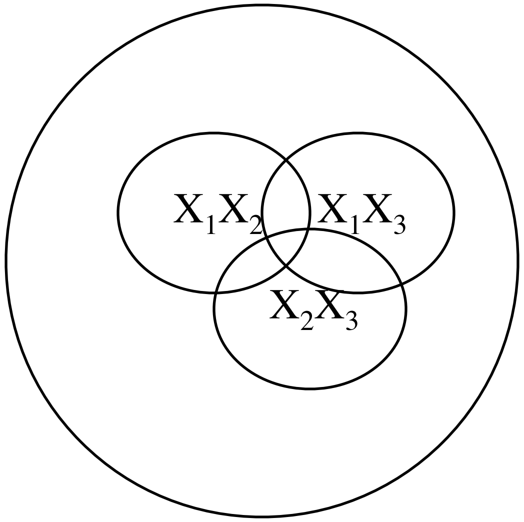


[Karp,Luby&Madras:1989]

1. Monte Carlo Simulation

Naïve:

$$E \;=\; X_1X_2 \vee X_1X_3 \vee X_2X_3$$



```
Cnt ← 0
repeat N times
    randomly choose $X_1, X_2, X_3 \in \{0,1\}$
    if $E(X_1, X_2, X_3) = 1$
        then Cnt = Cnt+1
P = Cnt/N
return P /*  $\simeq$ Pr(E) */
```

May be very big

0/1-estimator theorem

Theorem. If $N \geq (1/\Pr(E)) \times (4\ln(2/\delta)/\varepsilon^2)$ then:

$$\Pr[\,|P/\Pr(E) - 1| > \varepsilon\,] \quad < \quad \delta$$

Works for any E
Not in PTIME

[Karp,Luby&Madras:1989]

Monte Carlo Simulation

Improved:

$E \;=\; C_1 \vee C_2 \vee \ldots \vee C_m$

```
Cnt ← 0;    S ← Pr(C_1) + … + Pr(C_m);
repeat N times
    randomly choose i ∈ {1,2,…, m}, with prob. Pr(C_i) / S
    randomly choose X_1, …, X_n ∈ {0,1} s.t. C_i = 1
    if C_1=0 and C_2=0 and … and C_{i-1} = 0
        then Cnt = Cnt+1
P = Cnt/N  * 1/
return P /*  ≃ Pr(E) */
```

Now it's better

Theorem. If $N \geq (1/m) \times (4\ln(2/\delta)/\varepsilon^2)$ then:

$$\Pr[\,|\,P/\Pr(E) - 1\,| > \varepsilon\,] \;<\; \delta$$

Only for E in DNF
In PTIME

Summary on Monte Carlo

Some form of simulation is needed in probabilistic databases, to cope with the #P-hardness bottleneck

- Naïve MC: works well when Prob is big
- Improved MC: needed when Prob is small

[Nepal&Ramakrishna:1999,Fagin,Lotem,Naor:2001; 2003]

2. The Threshold Algorithm

Problem:

```
SELECT *
FROM Rp, Sp, Tp
WHERE Rp.A = Sp.B and Sp.C = Tp.D
```

Have subplans for R^p, S^p, T^p returning tuples sorted by their probabilities x, y, z

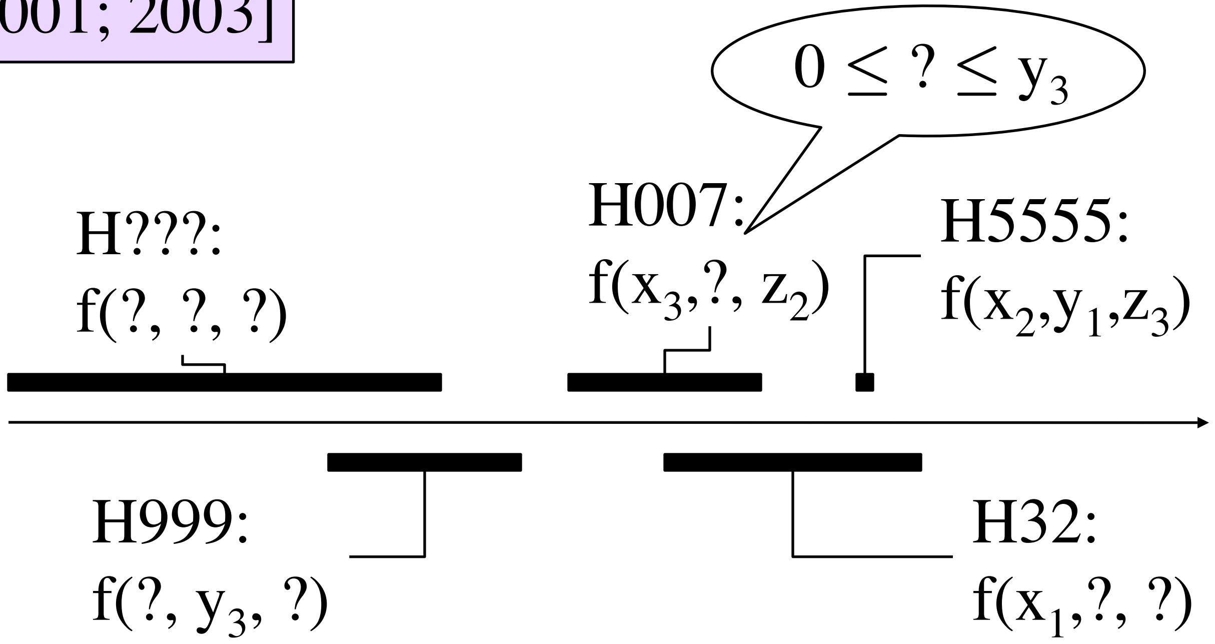
Score combination: $f(x, y, z) = xyz$

How do we compute the top-k matching records ?

[Fagin,Lotem,Naor:2001; 2003]

“No Random Access” (NRA)

Total score:



$R^p =$

H32	x_1
H5555	x_2
H007	x_3
?	?
. . .	

$1 \geq x_1 \geq x_2 \geq \ldots$

$S^p =$

H5555	y_1
H44	y_2
H999	y_3
?	?

$1 \geq y_1 \geq y_2 \geq \ldots$

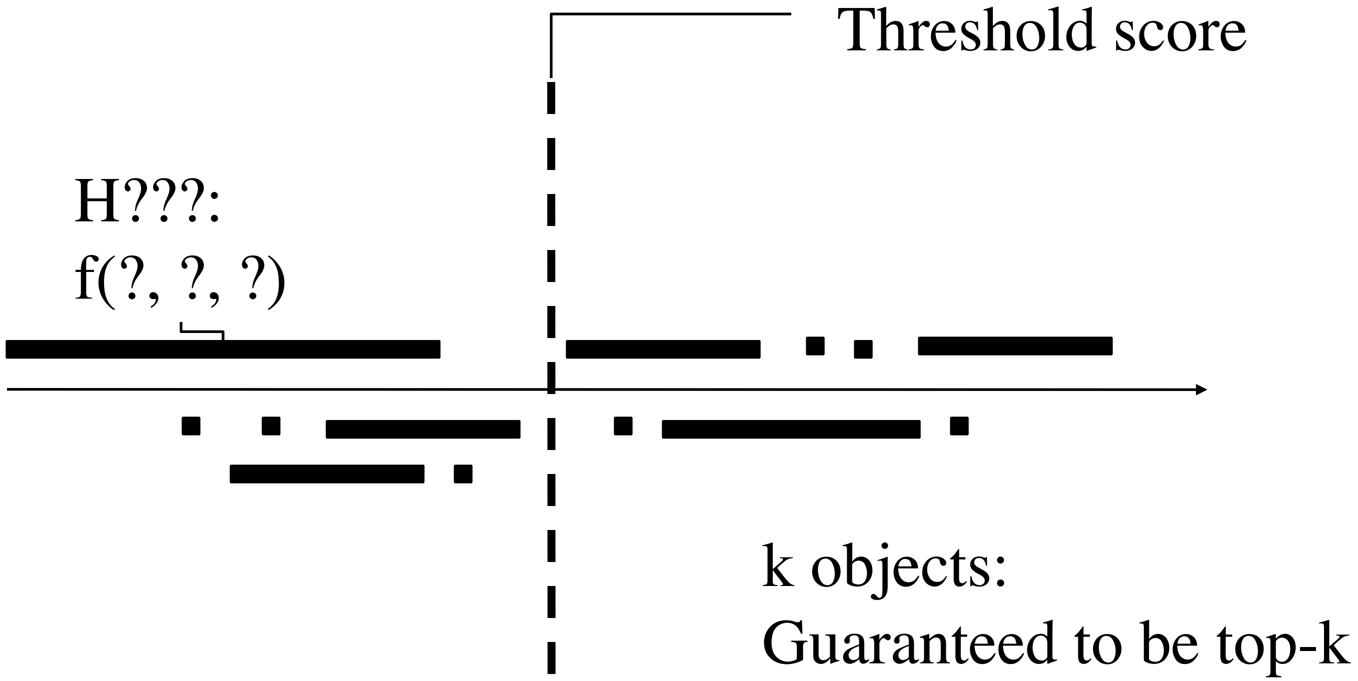
$T^p =$

H44	z_1
H007	z_2
H5555	z_3
?	?
. . .	

$1 \geq z_1 \geq z_2 \geq \ldots$

[Fagin,Lotem,Naor:2001; 2003]

Termination condition:



The algorithm is “instance optimal”
strongest form of optimality

Summary on the Threshold Algorithm

- Simple, intuitive, powerful
- There are several variations: see paper
- Extensions:
  - Use probabilistic methods to estimate the bounds more aggressively

    [Theobald,Weikum&Schenkel:2004]
  - Distributed environment

    [Michel, Triantafillou&Weikum:2005]

[Gravano et al.:2001]

Approximate String Joins

Problem:

```
SELECT *
FROM R, S
WHERE R.A ~ S.B
```

Simplification for this tutorial:
A ~ B means "A, B have at least k q-grams in common"

[Gravano et al.:2001]

Definition of q-grams

String: John_Smith

$\Downarrow$

Set of 3-grams:

##J #Jo Joh ohn hn_ n_S _Sm Smi mit ith th# h##

[Gravano et al.:2001]

```
SELECT *
FROM R, S
WHERE R.A ~ S.B
```

Naïve solution,
using UDF
(user defined function)

```
SELECT *
FROM R, S
WHERE common_grams(R.A, S.B) ≥ k
```

[Gravano et al.:2001]

A q-gram index:

R

Key	**A**	**…**
…		
k743	John_Smith	…
…		

RAQ

Key	**G**
…	
k743	##J
k743	#Jo
k743	Joh
…	

[Gravano et al.:2001]

Solution using
the Q-gram Index

```
SELECT *
FROM R, S
WHERE R.A ~ S.B
```

$\Downarrow$

```
SELECT R.*, S.*
FROM R, RAQ, S, SBQ
WHERE R.Key = RAQ.Key and S.Key=SBQ.Key
        and RAQ.G = RBQ.G
GROUP BY RAQ.Key, RBQ.Key
HAVING count(*) ≥ k
```

Summary on Part V: Algorithms

A wide range of disparate techniques

- Monte Carlo Simulations (also: MCMC)
- Optimal aggregation algorithms (TA)
- Efficient engineering techniques

Needed: unified framework for efficient query evaluation in probabilistic databases

Conclusions and Challenges Ahead

Conclusions

Imprecisions in data:

- A wide variety of types have specialized management solutions
- Probabilistic databases promise uniform framework, but require full complexity

Conclusions

Probabilistic databases

- Possible worlds semantics
  - Simple
  - Every query has well defined semantics
- Need: expressive representation formalism
- Need: efficient query processing techniques

Challenge 1: Specification Frameworks

The Goal:

- Design framework that is usable, expressive, efficient

The Challenge

- Tradeoff between expressibility and tractability

[Domingos&Richardson:04,Sarma,Benjelloun,Halevy,Widom:2005]

Challenge 1: Specification Frameworks

Features to have:

- Support probabilistic statements:
  - Simple: (Fred, Seattle, Gizmo) $\in$ Purchase has probability 60%
  - Complex: “Fred and Sue live in the same city” has probability 80%
- Support tuple corrleations
  - “t_1 and t_2 are correlated positively 30%”
- Statistics statements:
  - There are about 2000 tuples in Purchase
  - There are about 100 distinct Cities
  - Every customer buys about 4 products

Challenge 2: Query Evaluation

Complexity

- Old: = f(query-language)
- New: = f(query-language, specification-language)

Exact algorithm: #P-complete in simple cases

Challenge: characterize the complexity of approximation algorithms

Challenge 2: Query Evaluation

Implementations:

- Disparate techniques require unified framework
- Simulation:
  - Client side or server side ?
  - How to schedule simulation steps ?
  - How to push simulation steps in relational operators ?
  - How to compute subplans extensionally, when possible ?
- Top-k pruning:
  - How can we "push thresholds" down the query plan ?

Challenge 3: Mapping Imprecisions to Probabilities

- One needs to put a number between 0 and 1 to an uncertain piece of data
  - This is highly nontrivial !
  - But consider the alternative: ad-hoc management of imprecisions at all stages

- What is a principled approach to do this ?
- How do we evaluate such mappings ?

The Endp

Questions ?