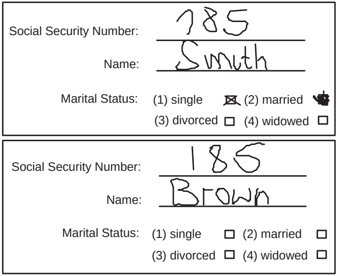
Fast and Simple Relational Processing of Uncertain Data

Lyublena Antova (Cornell), Thomas Jansen (SAP), Christoph Koch (Cornell),
Dan Olteanu (Oxford)

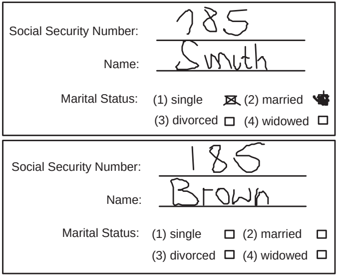
Appl?cati0n Sceanar o: Census data



We want to enter the information from forms like these into a database.

- What is the marital status of the first resp. the second person?
- What are the social security numbers? 185? 186? 785?

Appl?cati0n Sceanar o: Census data



(TID)	SSN	N	M
t_1	NULL	Smith	NULL
t_2	NULL	Brown	NULL

Much of the available information cannot be represented and is lost, e.g.

- Smith's SSN is either 185 or 785; Brown's SSN is either 185 or 186.
- Data cleaning: No two distinct persons can have the same SSN.

Main goals of the MayBMS project

Create a scalable DBMS for uncertain/probabilistic data

1. Representation and storage mechanisms
2. Uncertainty-aware query and data manipulation language
3. Efficient processing techniques for queries and constraints

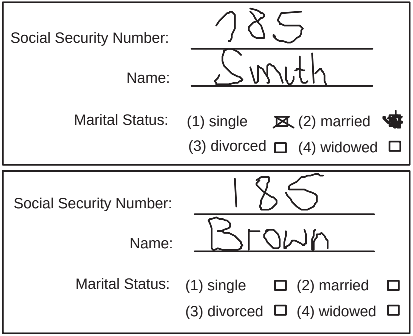
This talk will cover some aspects of (1) and (3).

Representation of uncertain data

Desiderata for a representation system

1. Succinctness/Space-efficient storage
   - Large number of independent *local* alternatives, which multiply up to a very large number of worlds.
2. Efficient real-world query processing
   - Tradeoff between succinctness and complexity of query evaluation.
     We want to do well in practice.
3. Expressiveness/Representability
   - Ability to represent all results of query and constraint processing.
   - Constraints/queries enforce dependencies across alternatives!

Quest for well-behaved representation system (1)



$R.t_1.SSN$
185
785

$\times$

$R.t_1.N$
Smith

$\times$

$R.t_1.M$
1
2

$R.t_2.SSN$
185
186

$\times$

$R.t_2.N$
Brown

$\times$

$R.t_2.M$
1
2
3
4

Properties (ICDE'07, ICDT'07)

- Relational representation of uncertainty at attribute-level
- Complete in the case of finite sets of alternatives (worlds)
- Data independence naturally supported by relational product
  Decompositions via efficient prime factorization of relations

Quest for well-behaved representation system (2)

x
$R.t_1.SSN$
185
785

$\times$

$R.t_1.N$
Smith

$\times$

v
$R.t_1.M$
1
2

$\times$

y
$R.t_2.SSN$
185
186

$\times$

$R.t_2.N$
Brown

$\times$

w
$R.t_2.M$
1
2
3
4

Equivalent column-oriented encoding with one relation per each attribute of R.

$U_{R[SSN]}$

V $\mapsto$ D	TID	SSN
$x \mapsto 1$	t_1	185
$x \mapsto 2$	t_1	785
$y \mapsto 1$	t_2	185
$y \mapsto 2$	t_2	186

$U_{R[N]}$

V $\mapsto$ D	TID	N
	t_1	Smith
	t_2	Brown

$U_{R[M]}$

V $\mapsto$ D	TID	M
$v \mapsto 1$	t_1	1
$v \mapsto 2$	t_1	2
$w \mapsto 1$	t_2	1
$w \mapsto 2$	t_2	2
$w \mapsto 3$	t_2	3
$w \mapsto 4$	t_2	4

U-Relational Databases

$U_{R[SSN]}$	V $\mapsto$ D	TID	SSN
	$x \mapsto 1$	t_1	185
	$x \mapsto 2$	t_1	785
	$y \mapsto 1$	t_2	185
	$y \mapsto 2$	t_2	186

$U_{R[N]}$	TID	N
	t_1	Smith
	t_2	Brown

$U_{R[M]}$	V $\mapsto$ D	TID	M
	$v \mapsto 1$	t_1	1
	$v \mapsto 2$	t_1	2
	$w \mapsto 1$	t_2	1
	$w \mapsto 2$	t_2	2
	$w \mapsto 3$	t_2	3
	$w \mapsto 4$	t_2	4

W	V $\mapsto$ D	P
	$x \mapsto 1$	.4
	$x \mapsto 2$	.6
	$y \mapsto 1$	.7
	$y \mapsto 2$	.3
	$v \mapsto 1$	.8
	$v \mapsto 2$	.2
	$w \mapsto 1$	.25
	$w \mapsto 2$	.25
	$w \mapsto 3$	.25
	$w \mapsto 4$	.25

- Discrete independent (random) variables (x, y, v, w).
- Representation: U-relations + table W representing distributions.
- The schema of each U-relation consists of
  - a tuple id column,
  - a set of column pairs (V_i, D_i) representing variable assignments, and
  - a set of value columns.

Semantics of U-Relational Databases

- Each possible world is identified by a valuation θ that assigns one of the possible values to each variable.
- The probability of the possible world is the product of weights of the values of the variables.
- The value-component of a tuple of a U-relation is in a given possible world if its variable assignments are consistent with θ.
- Attribute-level uncertainty through vertical decomposition.

Semantics of U-Relational Databases

$U_{R[SSN]}$	V $\mapsto$ D	TID	SSN
	$x \mapsto 1$	t_1	185
	$x \mapsto 2$	t_1	785
	$y \mapsto 1$	t_2	185
	$y \mapsto 2$	t_2	186

$U_{R[N]}$	TID	N
	t_1	Smith
	t_2	Brown

$U_{R[M]}$	V $\mapsto$ D	TID	M
	$v \mapsto 1$	t_1	1
	$v \mapsto 2$	t_1	2
	$w \mapsto 1$	t_2	1
	$w \mapsto 2$	t_2	2
	$w \mapsto 3$	t_2	3
	$w \mapsto 4$	t_2	4

W	V $\mapsto$ D	P
$\longrightarrow$	$x \mapsto 1$	.4
	$x \mapsto 2$	.6
	$y \mapsto 1$	.7
$\longrightarrow$	$y \mapsto 2$	.3
$\longrightarrow$	$v \mapsto 1$	.8
	$v \mapsto 2$	.2
$\longrightarrow$	$w \mapsto 1$	.25
	$w \mapsto 2$	.25
	$w \mapsto 3$	.25
	$w \mapsto 4$	.25

- We choose possible world $\{x \mapsto 1, y \mapsto 2, v \mapsto 1, w \mapsto 1\}$.

Semantics of U-Relational Databases

$U_{R[SSN]}$	V $\mapsto$ D	TID	SSN
	$x \mapsto 1$	t_1	185
	$y \mapsto 2$	t_2	186

$U_{R[N]}$	TID	N
	t_1	Smith
	t_2	Brown

$U_{R[M]}$	V $\mapsto$ D	TID	M
	$v \mapsto 1$	t_1	1
	$w \mapsto 1$	t_2	1

W	V $\mapsto$ D	P
$\rightarrow$	$x \mapsto 1$	.4
	$x \mapsto 2$	.6
	$y \mapsto 1$	.7
$\rightarrow$	$y \mapsto 2$	.3
$\rightarrow$	$v \mapsto 1$	.8
	$v \mapsto 2$	.2
$\rightarrow$	$w \mapsto 1$	.25
	$w \mapsto 2$	.25
	$w \mapsto 3$	.25
	$w \mapsto 4$	.25

- We choose possible world $\{x \mapsto 1, y \mapsto 2, v \mapsto 1, w \mapsto 1\}$.
- Probability weight of this world: $.4 * .3 * .8 * .25 = .024$.
- Now we have a vertically decomposed version of the chosen possible world.

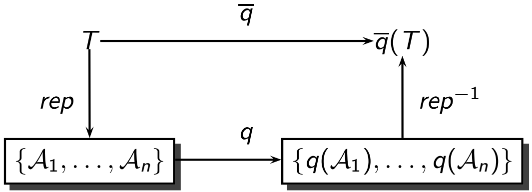
Properties of U-Relational Databases

- Complete representation system for finite sets of possible worlds
  - MystiQ: independent tuples/block-independent disjoint tables
- Often exponentially more succinct than WSDs, ULDBs, prob. databases
- A special case of c-tables
  - like all other existing representation formalisms, BUT...
- Purely relational representation of uncertainty at attribute-level
  - in contrast to probabilistic databases of MystiQ and ULDBs of Trio
- Efficient relational evaluation of many query operators (next topic)

Efficient query evaluation

Positive relational algebra

Query evaluation under *possible world semantics*:



For any positive relational algebra query q over any U-relational database T, there exists a positive relational algebra query $\overline{q}$ of polynomial size such that

$$\overline{q}(T) = rep^{-1}(\{q(\mathcal{A}_i) \mid \mathcal{A}_i \in rep(T)\}).$$

Properties

- relational evaluation using the query plan of your choice
- PTIME data complexity
- preserves the provenance of answer tuples

Query Evaluation: Example

Names of possibly married persons: $possible(\pi_{Name}(\sigma_{Status=2}(S)))$

$U_{S[Name]}$	V $\mapsto$ D	TID	Name
	$x_3 \mapsto 1$	t_1	Smith
	$x_5 \mapsto 1$	t_2	Brown

$U_{S[Status]}$	V $\mapsto$ D	TID	Status
	$x_3 \mapsto 1$	t_1	1
	$x_3 \mapsto 2$	t_1	2
	$x_6 \mapsto 1$	t_2	1
	$x_6 \mapsto 2$	t_2	2

Evaluation steps:

1. merge the U-relations storing the necessary columns:
   $Q := possible(\pi_{Name}(\sigma_{Status=2}(\text{ merge }(\pi_{Name}(S), \pi_{Status}(S)))))$

Query Evaluation: Example

Names of possibly married persons: $possible(\pi_{Name}(\sigma_{Status=2}(S)))$

$U_{S[Name]}$	V $\mapsto$ D	TID	Name
	$x_3 \mapsto 1$	t_1	Smith
	$x_5 \mapsto 1$	t_2	Brown

$U_{S[Status]}$	V $\mapsto$ D	TID	Status
	$x_3 \mapsto 1$	t_1	1
	$x_3 \mapsto 2$	t_1	2
	$x_6 \mapsto 1$	t_2	1
	$x_6 \mapsto 2$	t_2	2

Evaluation steps:

1. merge the U-relations storing the necessary columns:
   $Q := possible(\pi_{Name}(\sigma_{Status=2}(\text{ merge }(\pi_{Name}(S), \pi_{Status}(S)))))$
2. rewrite Q on column-store:
   $P := \pi_{Name}(\sigma_{Status=2}(U_{S[Name]} \bowtie_{\psi \wedge \phi} U_{S[Status]}))$, where

   ψ ensures that we only generate tuples that occur in some worlds:
   $\psi := (U_{S[Name]}.V = U_{S[Status]}.V \Rightarrow U_{S[Name]}.D = U_{S[Status]}.D)$,

   ϕ ensures that we only merge valid tuples:
   $\phi := (U_{S[Name]}.TID = U_{S[Status]}.TID)$

Query Evaluation: Example

Names of possibly married persons: $possible(\pi_{Name}(\sigma_{Status=2}(S)))$

$U_{S[Name]}$	$V \mapsto D$	TID	Name
	$x_3 \mapsto 1$	t_1	Smith
	$x_5 \mapsto 1$	t_2	Brown

$U_{S[Status]}$	$V \mapsto D$	TID	Status
	$x_3 \mapsto 1$	t_1	1
	$x_3 \mapsto 2$	t_1	2
	$x_6 \mapsto 1$	t_2	1
	$x_6 \mapsto 2$	t_2	2

Evaluation steps:

1. merge the U-relations storing the necessary columns:
   $Q := possible(\pi_{Name}(\sigma_{Status=2}(\text{merge}(\pi_{Name}(S), \pi_{Status}(S)))))$
2. rewrite Q on column-store:
   $P := \pi_{Name}(\sigma_{Status=2}(U_{S[Name]} \bowtie_{\psi \wedge \phi} U_{S[Status]}))$, where

   ψ ensures that we only generate tuples that occur in some worlds:
   $\psi := (U_{S[Name]}.V = U_{S[Status]}.V \Rightarrow U_{S[Name]}.D = U_{S[Status]}.D)$,

   ϕ ensures that we only merge valid tuples:
   $\phi := (U_{S[Name]}.TID = U_{S[Status]}.TID)$
3. feed P to *any* relational query optimizer

Query Evaluation: Example

Names of possibly married persons: $possible(\pi_{Name}(\sigma_{Status=2}(S)))$

$U_{S[Name]}$	V $\mapsto$ D	TID	Name
	$x_3 \mapsto 1$	t_1	Smith
	$x_5 \mapsto 1$	t_2	Brown

$U_{S[Status]}$	V $\mapsto$ D	TID	Status
	$x_3 \mapsto 1$	t_1	1
	$x_3 \mapsto 2$	t_1	2
	$x_6 \mapsto 1$	t_2	1
	$x_6 \mapsto 2$	t_2	2

	$V_1 \mapsto D_1$	$V_2 \mapsto D_2$	TID	Name	Status
wrong Status	$x_3 \mapsto 1$	$x_3 \mapsto 1$	$t_1 \stackrel{?}{=} t_1$	Smith	1
inconsistent	$x_3 \mapsto 1$	$x_3 \mapsto 2$	$t_1 \stackrel{?}{=} t_1$	Smith	2
wrong TIDs	$x_3 \mapsto 1$	$x_6 \mapsto 1$	$t_1 \stackrel{?}{=} t_2$	Smith	1
wrong TIDs	$x_3 \mapsto 1$	$x_6 \mapsto 2$	$t_1 \stackrel{?}{=} t_2$	Smith	2
wrong TIDs	$x_5 \mapsto 1$	$x_3 \mapsto 1$	$t_1 \stackrel{?}{=} t_2$	Brown	1
wrong TIDs	$x_5 \mapsto 1$	$x_3 \mapsto 2$	$t_1 \stackrel{?}{=} t_2$	Brown	2
wrong Status	$x_5 \mapsto 1$	$x_6 \mapsto 1$	$t_2 \stackrel{?}{=} t_2$	Brown	1
	$x_5 \mapsto 1$	$x_6 \mapsto 2$	$t_2 \stackrel{?}{=} t_2$	Brown	2

Query Evaluation: Example

Names of possibly married persons: $possible(\pi_{Name}(\sigma_{Status=2}(S)))$

$U_{S[Name]}$	V $\mapsto$ D	TID	Name
	$x_3 \mapsto 1$	t_1	Smith
	$x_5 \mapsto 1$	t_2	Brown

$U_{S[Status]}$	V $\mapsto$ D	TID	Status
	$x_3 \mapsto 1$	t_1	1
	$x_3 \mapsto 2$	t_1	2
	$x_6 \mapsto 1$	t_2	1
	$x_6 \mapsto 2$	t_2	2

	$V_1 \mapsto D_1$	$V_2 \mapsto D_2$	TID	Name	Status
	$x_5 \mapsto 1$	$x_6 \mapsto 2$	t_2	Brown	2

Beyond positive relational algebra

Difference

Tuple q-possibility is NP-hard even for normalized tuple-level U-relations and queries with difference. BUT this is already true for Codd tables.

World-set Algebra [SIGMOD'07,VLDB'07]

- Possible (R)
  Implemented using projection
- Certain (R)
  Implemented using division for *normalized* tuple-level U-relations (normalization = at most one variable assignment per tuple)
- repair-key$_{\vec{A}[@P]}(R)$
  Turns a possible world into the set of worlds consisting of all possible maximal repairs of key $\vec{A}$ in R.
- conf (R)
  Computes the exact confidence of (distinct) tuples
- ...

repair-key example

Tossing a biased coin twice.

R	Toss	Face	FProb
	1	H	.4
	1	T	.6
	2	H	.4
	2	T	.6

Pr = 1

$S :=$ repair-key$_{\mathrm{Toss@FProb}}(R)$ results in four worlds:

S^1	Toss	Face	FProb
	1	H	.4
	2	H	.4

S^2	Toss	Face	FProb
	1	H	.4
	2	T	.6

S^3	Toss	Face	FProb
	1	T	.6
	2	H	.4

S^4	Toss	Face	FProb
	1	T	.6
	2	T	.6

$$\Pr(S^1) = 1 \cdot \frac{.4}{.4 + .6} \cdot \frac{.4}{.4 + .6} = .16, \;\; \Pr(S^2) = \Pr(S^3) = .24, \;\; \Pr(S^4) = .36$$

repair-key example

Tossing a biased coin twice.

R	Toss	Face	FProb
	1	H	.4
	1	T	.6
	2	H	.4
	2	T	.6

Pr = 1

$S := \text{repair-key}_{\text{Toss@FProb}}(R)$ is just a projection/copying of columns (even though we may create an exponential number of possible worlds)!

U_S	V $\mapsto$ D	Toss	Face	FProb
	1 $\mapsto$ H	1	H	.4
	1 $\mapsto$ T	1	T	.6
	2 $\mapsto$ H	2	H	.4
	2 $\mapsto$ T	2	T	.6

W	V $\mapsto$ D	P
	1 $\mapsto$ H	.4
	1 $\mapsto$ T	.6
	2 $\mapsto$ H	.4
	2 $\mapsto$ T	.6

What about probabilities?

Given a tuple t with a set of valuations S, compute conf(t) by partitioning S

(a) into independent subsets (*exploit contextual independence*)

(b) by removing variables (*modified Davis-Putnam*)

(c) by removing valuations (*compute equiv. set of pairwise mutex valuations*)

Our current approach is a cost-based interplay of (a)-(c).

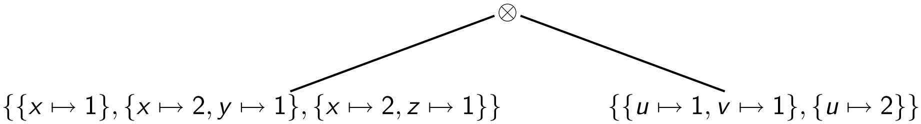
More in *Conditioning Probabilistic Databases* by Koch&Olteanu.

Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$

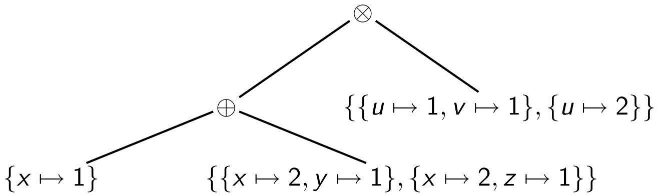
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



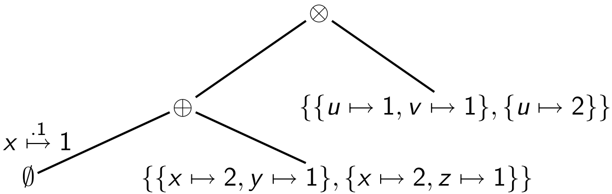
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



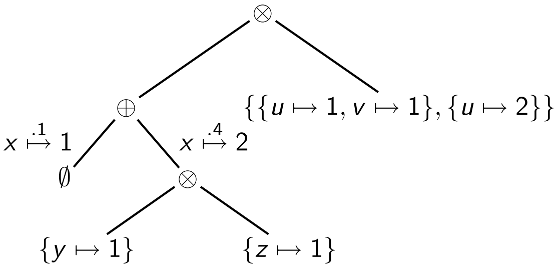
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



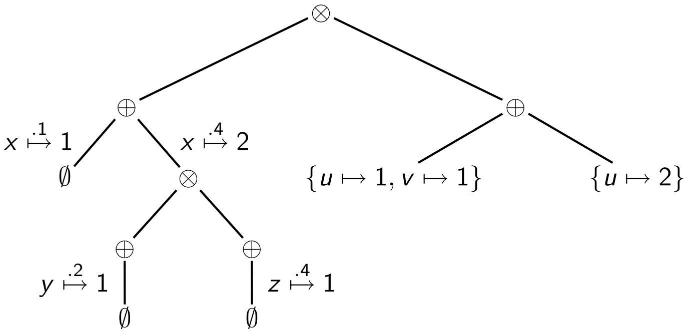
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



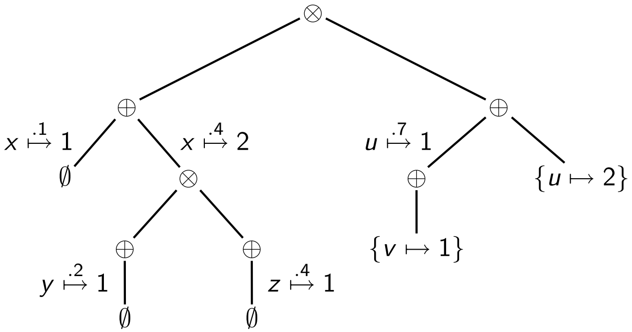
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



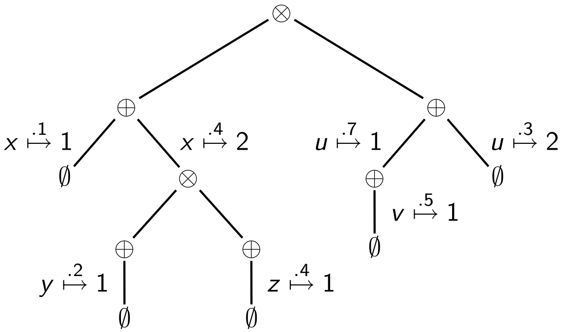
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



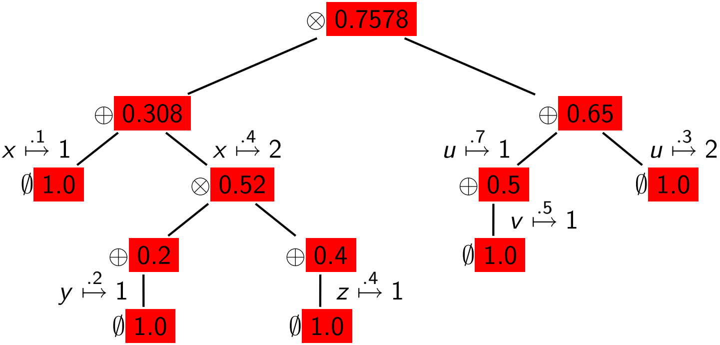
Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



Confidence computation example

$S = \{\{x \mapsto 1\}, \{x \mapsto 2, y \mapsto 1\}, \{x \mapsto 2, z \mapsto 1\}, \{u \mapsto 1, v \mapsto 1\}, \{u \mapsto 2\}\}$



$P(S) = 0.7578.$

Experiments

Uncertain data generator

- extend TPC-H population generator 2.6 to generate U-relational databases

  any generated world has the sizes of relations and join selectivities of the original TPC-H one-world case
- parameters: scale (s), uncertainty ratio (x), correlation ratio (z), max alternatives per field (8), drop after correlation (0.25)
- correlations follow a pattern obtained by chasing egds on uncertain data [ICDE'07]

Uncertainty and storage

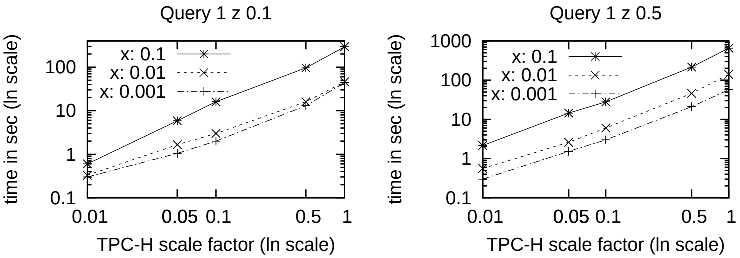
Total number of worlds, max. number of domain values for a variable (Rng), and size in MB of the U-relational database for each of our settings.

		x = 0.0	x = 0.001			x = 0.01			x = 0.1		
s	z	TPC-H dbsize	#worlds	Rng	dbsize	#worlds	Rng	dbsize	#worlds	Rng	dbsize
0.01	0.1	17	$10^{857.076}$	21	82	$10^{7955.30}$	57	85	$10^{79354.1}$	57	114
0.01	0.5	17	$10^{523.031}$	71	82	$10^{4724.56}$	901	88	$10^{46675.6}$	662	139
0.05	0.1	85	$10^{4287.23}$	22	389	$10^{39913.8}$	33	403	10^{396137}	65	547
0.05	0.5	85	$10^{2549.14}$	178	390	$10^{23515.5}$	449	416	10^{232650}	1155	672
0.10	0.1	170	$10^{8606.77}$	27	773	$10^{79889.9}$	49	802	10^{793611}	53	1090
0.10	0.5	170	$10^{5044.65}$	181	776	$10^{46901.8}$	773	826	10^{466038}	924	1339
0.50	0.1	853	$10^{43368.0}$	49	3843	10^{400185}	71	3987	$10^{3.97e+06}$	85	5427
0.50	0.5	853	$10^{25528.9}$	214	3856	10^{234840}	1832	4012	$10^{2.33e+06}$	2586	6682
1.00	0.1	1706	$10^{87203.0}$	57	7683	10^{800997}	99	7971	$10^{7.94e+06}$	113	11264
1.00	0.5	1706	$10^{51290.9}$	993	7712	10^{470401}	1675	8228	$10^{4.66e+06}$	3392	13312

- exponentially more succinct than representing worlds individually
- $10^{8 \cdot 10^6}$ worlds need 13 GBs $\approx$ 8 times the size of one world (1.4 GBs)
- case $x = 0$ is the DB generated by the original TPC-H (without uncertainty)

Evaluation of positive relational algebra queries

Q_1: **possible** (**select** o.orderkey, o.orderdate, o.shippriority **from** customer c, orders o, lineitem l **where** c.mktsegment = 'BUILDING'
and c.custkey = o.custkey and o.orderkey = l.orderkey
and o.orderdate > '1995-03-15' **and** l.shipdate < '1995-03-17')

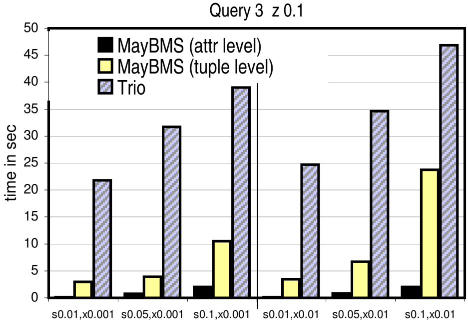


- uncertainty varies from 0.001 to 0.1 → evaluation time up to 6 times slower
- correlation varies from 0.1 to 0.5 → evaluation time up to 3 times slower
- scale varies from 0.01 to 1 → evaluation time up to 400 times slower
  scale=1: the answer size ranges from tens of thousands to tens of millions.

Attribute-level vs. tuple-level

SPJ query on six relations represented by equivalent

- attribute-level U-relational databases
- tuple-level U-relational databases
- Trio's ULDBs (are tuple-level only)
  Skipped the exponential time task of removing erroneous tuples



- Experiment only possible for small scenarios:
  1% uncertainty, lowest correlation factor 0.1, and scale up to 0.1.
- an increase in any of our parameters would create prohibitively large (exponential in the arity of relations) tuple-level representations.

Papers on MayBMS

- L. Antova, C. Koch, and D. Olteanu. From Complete to Incomplete Information and Back. In *Proc. SIGMOD 2007*.
- ———. World-Set Decompositions: Expressiveness and Efficient Algorithms. In *Proc. ICDT 2007*. Extended version conditionally accepted for *TCS*.
- ———. 10^{10^6} Worlds and Beyond: Efficient Representation and Processing of Incomplete Information. In *Proc. ICDE 2007*.
- ———. MayBMS: Managing Incomplete Information with Probabilistic World-Set Decompositions. In *Proc. ICDE 2007*. (Demo Paper.)
- ———. Query Language Support for Incomplete Information in the MayBMS System. In *Proc. VLDB 2007*. (Demo Paper.)
- Approximating Predicates and Expressive Queries on Probabilistic Databases. Christoph Koch. In *Proc. PODS 2008*.
- C. Koch, and D. Olteanu. Conditioning Probabilistic Databases. Available online.

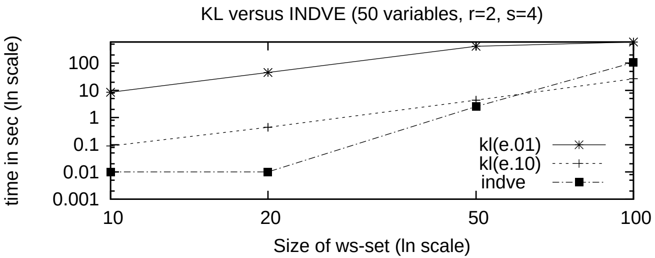
Experiments: Confidence computation

Excellent behaviour (within seconds) for

- few variables (100), many ws-descriptors (5K - 50K)
- many variables (100K), few ws-descriptors (01.K - 5K)

Heuristics for variable elimination: good variable choices are extremely valuable even if they require polynomial time

Competitive even when compared with Monte Carlo simulation based on Karp-Luby FPRAS (fully polynomial randomized approx. scheme) for #DNF.



Karp-Luby (KL): with at least 90% probability, the estimated error is within 1%, and 10% resp., from the exact value.

Query evaluation: Example 2

Violated SSN keys: $\mathit{possible}(\pi_{r_1.SSN}((R\ r_1) \bowtie_{r_1.SSN=r_2.SSN \wedge r_1.N<>r_2.N} (R\ r_2)))$

$U_{S[SSN]}$	V $\mapsto$ D	TID	SSN
	$x_1 \mapsto 1$	t_1	185
	$x_1 \mapsto 2$	t_1	785
	$x_4 \mapsto 1$	t_2	185
	$x_4 \mapsto 2$	t_2	186

$U_{S[Name]}$	V $\mapsto$ D	TID	Name
	$x_2 \mapsto 1$	t_1	Smith
	$x_5 \mapsto 1$	t_2	Brown

Rewritten query on column-store:

$S := U_{S[SSN]} \bowtie_{\psi \wedge \phi} U_{S[Name]}$

$P := \pi_{s_1.SSN\ as\ SSN}((S\ s_1) \bowtie_{s_1.SSN=s_2.SSN \wedge s_1.Name<>s_2.Name} (S\ s_2))$

P	$V_1 \mapsto D_1$	$V_2 \mapsto D_2$	$V_3 \mapsto D_3$	$V_4 \mapsto D_4$	T_{s_1}	T_{s_2}	SSN
	$x_1 \mapsto 1$	$x_2 \mapsto 1$	$x_4 \mapsto 1$	$x_5 \mapsto 1$	t_1	t_2	185
	$x_5 \mapsto 1$	$x_4 \mapsto 1$	$x_1 \mapsto 1$	$x_2 \mapsto 1$	t_2	t_1	185

Uncertainty-aware query language

Desiderata for a Query Language for Uncertain Data

- genericity – declarative queries, independent from representation details
  - Trio's TriQL is **not** generic
- ability to transform data
  - beyond the filtering of world-sets as in MystiQ
- ability to introduce additional uncertainty (!!!)
  - To make it a natural query language for the possible worlds model: compositionality
  - Decision support queries/hypothetical queries
  - Probabilistic databases: extending the hypothesis space to use evidence
- right degree of expressive power – not too strong and not too weak
- efficient query evaluation

World-set Algebra

- The operations of relational algebra.
  - Evaluated individually, in "parallel" in all possible worlds.
- An operation $\mathrm{conf}(R)$ for computing tuple confidence values.
  - Computes, for each tuple that occurs in R in at least one world, the sum of the probabilities of the worlds in which it occurs.
- An operation $\mathrm{assert}_{\phi}(R)$ that conditions the database using a constraint ϕ.
  - Removes those worlds that violate ϕ.
- An operation $\text{repair-key}_{\vec{A}[@P]}(R)$ for introducing uncertainty.
  - Turns a possible world into the set of worlds consisting of all possible maximal repairs of key $\vec{A}$ in R.
  - We will also look at a special case of repair-key called choice-of.
- An operation for grouping worlds based on common properties
  - property = answer to a given query
  - (we will not discuss this one here)

Operation choice-of

- Introducing uncertainty using the choice-of operation allows to extend the hypothesis space.

R^1	A	B	C
	a	1	c
	a	1	d
	b	3	e

Pr = .5 ... (further worlds)

$S := \text{choice-of}_{A@B}(R)$

$S^{1.1}$	A	B	C
	a	1	c
	a	1	d

Pr = .5 * 1/4 = 1/8

$S^{1.2}$	A	B	C
	b	3	e

Pr = .5 * 3/4 = 3/8

... (further worlds)

There must be a functional dependency $R : A \rightarrow B$.

- Necessary if we want to introduce evidence.

Operation repair-key

Example: Tossing a biased coin twice.

R	Toss	Face	FProb
	1	H	.4
	1	T	.6
	2	H	.4
	2	T	.6

Pr = 1

$S := \text{repair-key}_{\mathrm{Toss@FProb}}(R)$ results in four worlds:

S^1	Toss	Face	FProb
	1	H	.4
	2	H	.4

S^2	Toss	Face	FProb
	1	H	.4
	2	T	.6

S^3	Toss	Face	FProb
	1	T	.6
	2	H	.4

S^4	Toss	Face	FProb
	1	T	.6
	2	T	.6

$$\Pr(S^1) = 1 \cdot \frac{.4}{.4 + .6} \cdot \frac{.4}{.4 + .6} = .16, \;\; \Pr(S^2) = \Pr(S^3) = .24, \;\; \Pr(S^4) = .36$$

Operation conf

$R^{\mathcal{A}}$	A	B
	a	b
	b	c

.3

$R^{\mathcal{B}}$	A	B
	a	b
	c	d

.2

$R^{\mathcal{C}}$	A	B
	a	c
	c	d

.5

conf(R) gives the probability of each tuple across all worlds:

$conf(R)$	x	z	P
	a	b	.5
	a	c	.5
	b	c	.3
	c	d	.7

For a Boolean query Q and a world-set $\mathbf{W}$, $conf(Q)$ gives us one number, the probability of the event $\{I \in \mathbf{W} \mid I \models Q\}$, which is the confidence of tuple $\langle\rangle$.

Conditioning using assert

Example: enforcing a key constraint on SSN.

$U_{R[SSN]}$	V	D	TID	SSN
	x	1	t_1	185
	x	2	t_1	785
	y	1	t_2	185
	y	2	t_2	186

$T := assert_{fd:SSN \to TID}(R)$.
We drop the worlds where both tuples t_1 and t_2 occur with SSN $= 185$.

$U_{T[SSN]}$	V_1	D_1	V_2	D_2	TID	SSN
	x	1	y	2	t_1	185
	x	1	y	2	t_2	186
	x	2	y	1	t_1	785
	x	2	y	1	t_2	185
	x	2	y	2	t_1	785
	x	2	y	2	t_2	186