MayBMS - A System for Managing Large Amounts of Probabilistic Data

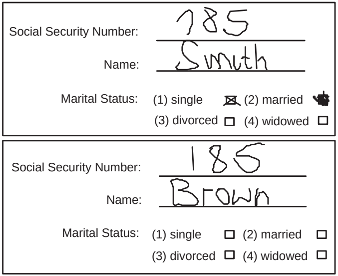
Lyublena Antova

Cornell University

February 18, 2009, A-Exam

Motivation: census data

Enter the information from **census** forms like these into a database:



Smith's SSN?
Brown's marital status?
How to make sure SSN is unique?

R	SSN	N	M
t_1	null	Smith	null
t_2	null	Brown	null

Motivation: web information extraction

Automatic extraction of structured data from the web:

Volkswagen Cars For Sale

Volkswagen : Rabbit Volkswagen 2008 VW Rabbit — **$10,750**

2008 Volkswagen Rabbit, 184 miles, Red

Location: Carmel, IN

Source: visint on eBay, 1 week ago

Details | Share | Report

2008 Volkswagen Rabbit — **$10,750**

2008 Volkswagen Rabbit, 183 miles

Location: Carmel, IN

Source: Auction Piranha, 1 week ago

Details | Share | Report

2005 Volkswagen Passat — **$10,999**

2005 Volkswagen Passat, 8,075 miles

Location: Pittsford, NY

Source: Auction Piranha, 4 days ago

Motivation: uncertain data

Data integration:

DB1:

John	$1200

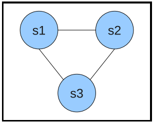
DB2:

John	$4000

John	$1200
John	$4000

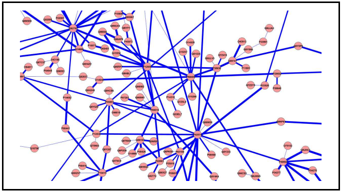
} mutually exclusive

Sensor networks:



ID	Time	Temp
s1	7:00	25
s1	8:00	27
s2	7:00	25

Scientific data:



Decision support queries:

Given sales and competitors data and a number of possible solutions which is the one that maximizes the expected profit

Managing uncertain data: motivation

- Uncertainty present in many real-world applications: information extraction, data integration, scientific data,...
- Limited support for managing uncertain data in traditional database management systems (DBMS)
- Other solutions typically not expressive or not scalable enough
- Goal of the MayBMS project: create a **scalable** probabilistic database management **system**
  - Representation system
  - Query language
  - Updates and transactions
  - ...

Outline

Representing Uncertain Information

Representing uncertain data

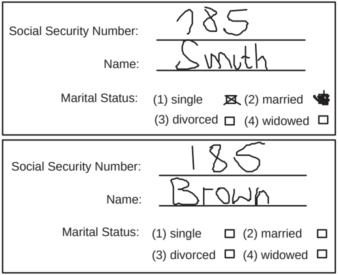
Definition

Representation system is a tuple $(\mathbf{T}, rep)$ of a a set of structures $\mathbf{T}$ and a function $rep : \mathbf{T} \longrightarrow$ sets of worlds.

Desiderata for a representation system:

1. Space-efficient storage.
2. Efficient query processing.
3. Expressiveness: represent the result of any query.

Representing uncertain data: 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

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

$U_{R[MS]}$	$V \mapsto D$	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↦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

$U_{R[MS]}$	V↦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↦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

- Table W: discrete independent (random) variables
- U-relations: the schema of each U-relation consists of
  - a tuple id column,
  - a set of column pairs (Vi , Di) representing variable assignments, and
  - a set of value columns.

U-relational databases: semantics (example)

- Pick a valuation θ that assigns a value to each variable.

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

$U_{R[SSN]}$	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

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

$U_{R[MS]}$	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: semantics (example)

- Select the tuples consistent with θ.

$U_{R[SSN]}$	V↦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

$U_{R[MS]}$	V↦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

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

W	V↦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

U-relational databases: semantics (example)

- Select the tuples consistent with θ.

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

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

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

W	V↦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

U-relational databases: semantics (example)

- Undo the vertical decompositioning by rejoining the partitions.
- Possible world:

R	TID	SSN	N	MS
	t_1	185	Smith	2
	t_2	186	Brown	3

W	V↦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

- Probability of the world: $0.4 \cdot 0.3 \cdot 0.2 \cdot 0.25 = 0.006$

Representing correlations in U-relational databases

- SSN is unique:

$U_{R[SSN]}$	V$\mapsto$D	TID	SSN
	$x \mapsto 1$	t_1	185
	$x \mapsto 2$	t_1	785
	$x \mapsto 3$	t_1	785
	$x \mapsto 1$	t_2	186
	$x \mapsto 2$	t_2	185
	$x \mapsto 3$	t_2	186

- No valuation exists that results in both t_1 and t_2 having 185 for SSN.

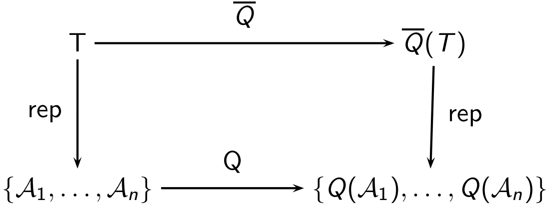
Properties of U-relational databases

Desiderata for a representation system:

1. Space-efficient storage.
   - U-relations can represent compactly an exponential number of possible worlds
   - Purely relational
2. Efficient query processing.
   - (next)
3. Expressiveness: represent the result of any query.
   - U-relations are complete for finite sets of possible worlds

Querying Uncertain Data

Possible worlds semantics



- T: probabilistic database.
- Conceptually, evaluate Q on each world
- Find a query $\overline{Q}$ on the representation that produces the representation of the result.

Query evaluation on U-relations

Names of possibly married persons: possible($\pi_N(\sigma_{MS=2}(R))$)

$U_{R[N]}$	V↦D	TID	N
	$x \mapsto 1$	t_1	Smith
	$y \mapsto 1$	t_2	Brown

$U_{R[MS]}$	V↦D	TID	MS
	$x \mapsto 1$	t_1	1
	$x \mapsto 2$	t_1	2
	$z \mapsto 1$	t_2	1
	$z \mapsto 2$	t_2	2

Evaluation steps:

Query evaluation on U-relations

Names of possibly married persons: $\text{possible}(\pi_N(\sigma_{MS=2}(R)))$

$U_{R[N]}$	V↦D	TID	N
	$x \mapsto 1$	t_1	Smith
	$y \mapsto 1$	t_2	Brown

$U_{R[MS]}$	V↦D	TID	MS
	$x \mapsto 1$	t_1	1
	$x \mapsto 2$	t_1	2
	$z \mapsto 1$	t_2	1
	$z \mapsto 2$	t_2	2

Evaluation steps:

1. Merge the U-relations storing the necessary columns
   $Q = \text{possible}(\pi_N(\sigma_{MS=2}(\text{merge}(\pi_N R, \pi_{MS} R))))$

Query evaluation on U-relations

Names of possibly married persons: possible($\pi_N(\sigma_{MS=2}(R))$)

$U_{R[N]}$	V↦D	TID	N
	$x \mapsto 1$	t_1	Smith
	$y \mapsto 1$	t_2	Brown

$U_{R[MS]}$	V↦D	TID	MS
	$x \mapsto 1$	t_1	1
	$x \mapsto 2$	t_1	2
	$z \mapsto 1$	t_2	1
	$z \mapsto 2$	t_2	2

Evaluation steps:

1. Merge the U-relations storing the necessary columns
   Q =possible($\pi_N(\sigma_{MS=2}($ merge $(\pi_N R, \pi_{MS} R)))$
2. Rewrite Q on column-stores:
   merge $(\pi_N R, \pi_{MS} R)$ $= U_{R[N]} \bowtie_{\psi \wedge \phi} U_{R[MS]}$
   - ψ: do not create inconsistent conditions:
     $\psi := (U_{R[N]}.V = U_{R[MS]}.V \Rightarrow U_{R[N]}.D = U_{R[MS]}.D)$,
   - ϕ: tuple reconstruction:
     $\phi := (U_{R[N]}.TID = UR_{[MS]}.TID)$

Query evaluation on U-relations

Names of possibly married persons: possible($\pi_N(\sigma_{MS=2}(R))$)

$U_{R[N]}$	V↦D	TID	N
	$x \mapsto 1$	t_1	Smith
	$y \mapsto 1$	t_2	Brown

$U_{R[MS]}$	V↦D	TID	MS
	$x \mapsto 2$	t_1	2
	$z \mapsto 2$	t_2	2

Evaluation steps:

1. Merge the U-relations storing the necessary columns
   $Q =$possible($\pi_N(\sigma_{MS=2}($ merge $(\pi_N R, \pi_{MS} R)))$)
2. Rewrite Q on column-stores:
   merge $(\pi_N R, \pi_{MS} R)) \quad = U_{R[N]} \bowtie_{\psi \wedge \phi} U_{R[MS]}$
   - ψ: do not create inconsistent conditions:
     $\psi := (U_{R[N]}.V = U_{R[MS]}.V \Rightarrow U_{R[N]}.D = U_{R[MS]}.D)$,
   - ϕ: tuple reconstruction:
     $\phi := (U_{R[N]}.TID = UR_{[MS]}.TID)$

Query evaluation on U-relations

Names of possibly married persons: $\text{possible}(\pi_N(\sigma_{MS=2}(R)))$

$U_{R[N]}$	$V_1 \mapsto D_1$	$V_2 \mapsto D_2$	TID	N
	$x \mapsto 1$	$x \mapsto 2$	t_1	Smith
	$y \mapsto 1$	$z \mapsto 2$	t_2	Brown

Result:

Q	N
	Brown

Evaluating positive relation algebra queries on U-relations

$$
\begin{aligned}
[\![R \times S]\!] &:= \pi_{(U_R.\overline{VD} \cup U_S.\overline{VD}) \to \overline{VD}, sch(R), sch(S)} \quad (U_R \bowtie_\psi U_S) \\
[\![\sigma_\phi R]\!] &:= \sigma_\phi(U_R) \\
[\![\pi_{\vec{B}} R]\!] &:= \pi_{\overline{VD}, \vec{B}}(R) \\
[\![R \cup S]\!] &:= U_R \cup U_S \\
[\![\text{poss}(R)]\!] &:= \pi_{sch(R)}(U_R).
\end{aligned}
$$

- Simple translation, essentially preserves number of joins
- Can be fed to any relational query optimizer

Query language of MayBMS

- Relational algebra operations
- Confidence computation: $conf(Q)$ for computing tuple confidence values.
  - For each tuple that occurs in Q in at least one world, sum up the probabilities of the worlds where it occurs.
- repair-key
  - operation for introducing uncertainty.
  - turns a possible world into the set of worlds consisting of all possible maximal repairs.

Random graph example

- Random graphs as probabilistic databases
- Consider table Edge(A,B,bit,P):

Edge	A	B	bit	P
e_{10}	1	2	0	0.5
e_{11}	1	2	1	0.5
e_{20}	1	3	0	0.5
e_{21}	1	3	0	0.5

...

- Pick edges non-deterministically:

create table T **as** (**repair key** A,B in Edge **weight by** P);

create table E1 **as** (**select** A,B **from** T **where** bit = 1);

create table E0 **as** (**select** A,B **from** T **where** bit = 0);

Random graph example

- Random graphs as probabilistic databases
- Consider table Edge(A,B,bit,P):

Edge	A	B	bit	P
e_{10}	1	2	0	0.5
e_{11}	1	2	1	0.5
e_{20}	1	3	0	0.5
e_{21}	1	3	0	0.5

...

E1	V↦D, P	A	B
	$x_1 \mapsto 1, 0.5$	1	2
	$x_2 \mapsto 1, 0.5$	1	3

...

- Pick edges non-deterministically:

create table T **as** (**repair key** A,B in Edge **weight by** P);

create table E1 **as** (**select** A,B **from** T **where** bit = 1);

create table E0 **as** (**select** A,B **from** T **where** bit = 0);

Random graph example

- U-relational representation of the random graph:

E0	V↦D, P	A	B
	$x_1 \mapsto 0, 0.5$	1	2
	$x_2 \mapsto 0, 0.5$	1	3
	$x_3 \mapsto 0, 0.5$	1	4
	$x_4 \mapsto 0, 0.5$	2	3

...

E1	V↦D, P	A	B
	$x_1 \mapsto 1, 0.5$	1	2
	$x_2 \mapsto 1, 0.5$	1	3
	$x_3 \mapsto 1, 0.5$	1	4
	$x_4 \mapsto 1, 0.5$	2	3

...

- Queries:
  - Probability for a triangle in the random graph:

```
select conf()
from E1 R, E1 S, E1 T
where R.A = S.B and S.A = T.A and T.B=R.B
and R.A != S.A and R.A != T.A and S.A != T.A;
```

Random graph example

- U-relational representation of the random graph:

E0	V↦D, P	A	B
	$x_1 \mapsto 0, 0.5$	1	2
	$x_2 \mapsto 0, 0.5$	1	3
	$x_3 \mapsto 0, 0.5$	1	4
	$x_4 \mapsto 0, 0.5$	2	3
	...		

E1	V↦D, P	A	B
	$x_1 \mapsto 1, 0.5$	1	2
	$x_2 \mapsto 1, 0.5$	1	3
	$x_3 \mapsto 1, 0.5$	1	4
	$x_4 \mapsto 1, 0.5$	2	3
	...		

- Queries:
  - Probability for a triangle in the random graph:

```
select conf()
from E1 R, E1 S, E1 T
where R.A = S.B and S.A = T.A and T.B=R.B
and R.A != S.A and R.A != T.A and S.A != T.A;
```

  - Probability of a 4-clique
  - Probability of a path of length 5
  - ...

Experimental Evaluation

Experimental evaluation

- Uncertain data generator
  - extend TPC-H generator 2.6 to generate U-relational databases
  - parameters: scale (s), uncertainty ratio (x), correlation ratio (z), max alternatives per field (8), drop after correlation (0.25)
- Evaluate modified TPC-H queries (without aggregation) on the generated data

Experimental results: storage

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
		$\mathbf{x} = 0.0$	$\mathbf{x} = 0.001$			$\mathbf{x} = 0.01$			$\mathbf{x} = 0.1$		

Figure: 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.

- 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)

Experimental results: evaluation of positive relational algebra queries.

Q1:
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')

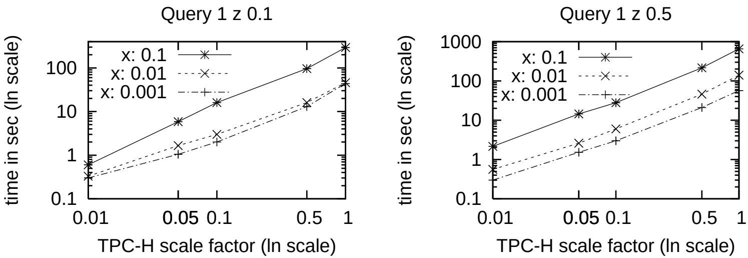
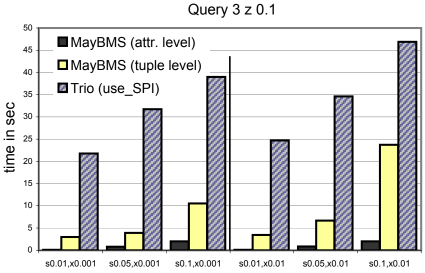


Figure: Performance of query evaluation for various scale, uncertainty, and correlation.

Experimental results: effect of vertical partitioning.

SPJ query on six relations represented by equivalent

- attribute-level U-relational databases
- tuple-level U-relational databases
- Trios ULDBs (are tuple-level only)



- 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.

MayBMS

- Built inside Postgres.
- Representation system: U-relations.
- Query language: an extension of SQL with uncertainty-aware constructs.
- Supports updates.
- Exact and approximate confidence computation.
- Prototype available at http://maybms.sourceforge.net
- Project website: http://www.cs.cornell.edu/bigreddata/maybms/

APIs for Probabilistic Databases

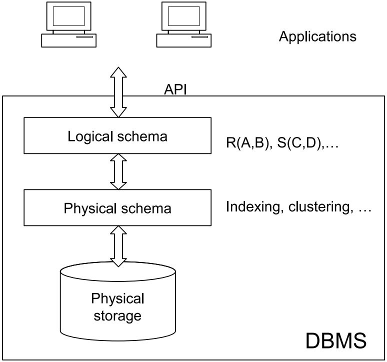
Database APIs for RDBMS

Write programs that involve:

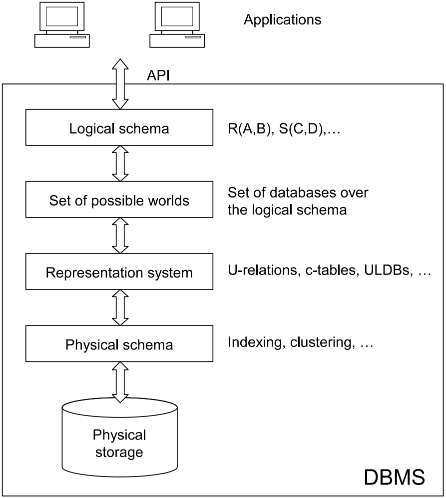
- queries
- updates
- user interaction (input/output)

- ANSI 3-layer model: abstract away from physical representation details.
- What about APIs for uncertain DBMS?

Levels of Abstraction in DBMS



(a) Traditional DBMS

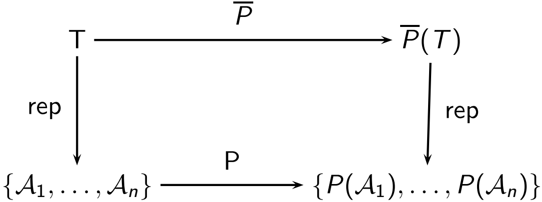


(b) DBMS for uncertain data

Data Independence

Clean reference model for uncertain DBMS:

- Sets of possible worlds.
- Any representation system can be modeled by possible worlds.



- Challenges:
  - How to find $\overline{P}$?
  - What if there is user interaction?

Programs on uncertain databases

```
read("Enter license plate:", $x);
if (exists select * from cars where num=$x){ // modify existing entry
   for($t in select * from cars where num=$x){
      write("Current entry: $t");
      read("New location and color:", $loc,$color);
      if (exists select * from cars where num=$x and loc=$loc and color=$color)
         update cars set wit=wit+1 where num=$x and loc=$loc and color=$color;
      else
         insert into cars values($x,$loc,$color,1);
   }
}
else { // entry does not exist
   write("No entry found for $x");
   read("Enter location and color:", $loc, $color);
   insert into cars values($x,$loc,$color,1);
}
```

Programs on uncertain databases

$Cars^1$	num	color	loc	wit
1	S87	red	MN	1
2	M34	blue	PA	1

$Cars^2$	num	color	loc	wit
1	S87	red	TX	1
2	M34	blue	MD	1

$Cars^3$	num	color	loc	wit
1	B87	red	TX	1
2	M34	blue	PA	1

$Cars^4$	num	color	loc	wit
1	B87	red	TX	1
2	M34	blue	MD	1

(a) Possible worlds

```
Current entry: S87 red MN
New location and color: _
```

```
Current entry: S87 red TX
New location and color: _
```

```
No entry found for S87
Enter location and color: _
```

```
No entry found for S87
Enter location and color: _
```

(b) Output of the program

$Cars^1$	num	color	loc	wit
1	S87	red	MN	2
2	M34	blue	PA	1

$Cars^2$	num	color	loc	wit
1	S87	red	TX	1
2	M34	blue	MD	1
3	S87	red	MN	1

$Cars^3$	num	color	loc	wit
1	B87	red	TX	1
2	M34	blue	PA	1
3	S87	red	MN	1

$Cars^4$	num	color	loc	wit
1	B87	red	TX	1
2	M34	blue	MD	1
3	S87	red	MN	1

(c) Result of the program

Running possible worlds in "parallel"

- Control flow can be different in each possible world.
- User interaction that is different in each world unintuitive/infeasible

- Our approach:
  - Exclude "unsafe" programs (next)
  - All other programs: rewrite into bulk queries and updates (set-at-a-time processing for world-sets).

Observational determinism

Definition

A program is called **observationally deterministic** (o.d.) if its user interaction is identical in all possible worlds.

- User interaction: input and output of the program.
- We consider programs that do not satisfy this property unsound.

Checking observational determinism

- Model relations R as two disjoint sets of tuples, the certain and the uncertain ones, R^c, R^u.
- Propagate pairs (R^c, R^u) conservatively through program operations.
- A program is o.d. if we never output or condition on an uncertain tuple.

Propagation of uncertainty during querying

Let R – relation name, ϕ – boolean condition

Q, Q_1, Q_2 – queries in $RA^+ \cup \{\text{conf}\}$

$$
\begin{aligned}
[\![R]\!] &:= (R^c, R^u) \\
[\![\pi_U(Q)]\!] &:= (\pi_U([\![Q]\!]^c), \pi_U([\![Q]\!]^u)) \\
[\![\sigma_\phi(Q)]\!] &:= (\sigma_\phi([\![Q]\!]^c), \sigma_\phi([\![Q]\!]^u)) \\
[\![Q_1 \bowtie_\phi Q_2]\!] &:= ([\![Q_1]\!]^c \bowtie_\phi [\![Q_2]\!]^c, \\
&\quad [\![Q_1]\!]^u \bowtie_\phi [\![Q_2]\!]^u \cup [\![Q_1]\!]^c \bowtie_\phi [\![Q_2]\!]^u \cup [\![Q_1]\!]^c \bowtie_\phi [\![Q_2]\!]^u) \\
[\![Q_1 \cup Q_2]\!] &:= ([\![Q_1]\!]^c \cup [\![Q_2]\!]^c, [\![Q_1]\!]^u \cup [\![Q_2]\!]^u) \\
[\![\text{conf}(Q)]\!] &:= ([\![Q]\!]^c \cup \text{possible}([\![Q]\!]^u), \emptyset)
\end{aligned}
$$

Checking observational determinism

- Example: let $R = (Rc, \sigma_{A!=1}(R_u))$.

  for ($t in select * from R where A=1)
    write($t);

- $Q = (\sigma_{A=1}(Rc), \sigma_{A=1}(\sigma_{A!=1}(R_u)))$.

This program is observationally deterministic.

Results

- MayBMS: open-source system for managing probabilistic data
- U-relational databases
  - Compact representation of large sets of possible worlds
  - Expressive
  - Efficient query evaluation
- Query language and API for probabilistic databases.
- Ongoing and future work
  - Approximate confidence computation
  - Official release of the system

Bibliography

- Representation system and query processing
  - L. Antova, T. Jansen. C. Koch, and D. Olteanu. Fast and Simple Relational Processing of Uncertain Data. *ICDE 2008*
- Query language and API for probabilistic databases.
  - L. Antova, C. Koch, and D. Olteanu. From Complete to Incomplete Information and Back. *SIGMOD 2007*
  - L. Antova, C. Koch. On APIs for probabilistic databases. *MUD 2008*
- Demonstrations
  - L. Antova, C. Koch, and D. Olteanu. MayBMS: Managing Incomplete Information with Probabilistic World-Set Decompositions *ICDE'07*
  - L. Antova, C. Koch, and D. Olteanu. Query Language Support for Incomplete Information in the MayBMS System. *VLDB 2007*
  - http://maybms.sourceforge.net

Acknowledgments:
Christoph Koch, Dan Olteanu, Thomas Jansen and Jiewen Huang.

Thanks!