MayBMS: A Probabilistic DBMS

Jiewen Huang[*,**], Lyublena Antova[*], Christoph Koch[*], and Dan Olteanu[**]

[*] Cornell University [**] Oxford University

Example: random graphs

Goal: Compute the probability that a random graph contains a triangle.

T	u	v	bit	p
	1	2	1	.5
	1	2	0	.5
	1	3	1	.5
	1	3	0	.5
	2	3	1	.5
	2	3	0	.5

```
create table E as select Q.u, Q.v
from   (repair key (u,v) in T weight by p) Q
where  Q.bit = 1;
```

8 possible worlds, one has a triangle. E not given as symmetric relation, but as subset of total order.

```
select conf() as triangle_prob
from   E e1, E e2, E e3
where  e1.v = e2.u and e2.v = e3.v
and    e1.u = e3.u and e1.u < e2.u
and    e2.u < e3.v;
```

triangle_prob
0.125

Example: hypothetical queries

Suppose I buy a company and exactly one employee leaves. Which skills do I gain for certain?

CE	CID	EID
	Google	Bob
	Google	Joe
	Yahoo	Dan
	Yahoo	Bill
	Yahoo	Fred

ES	EID	Skill
	Bob	Web
	Joe	Web
	Dan	Java
	Dan	Web
	Bill	Search
	Fred	Java

```
create table RemainingEmployees as
select CE.cid, CE.eid
from   CE, (repair key (dummy)
            in (select 1 as dummy, *
                from CE)) Choice
where  CE.cid = Choice.cid
and    CE.eid <> Choice.eid;
```

```
create table SkillGained as
select Q1.cid, Q1.skill, p1, p2, p1/p2 as p
from (select R.cid, ES.skill, conf() as p1
      from RemainingEmployees R, ES
      where R.eid = ES.eid
      group by R.cid, ES.skill) Q1,
     (select cid, conf() as p2
      from RemainingEmployees
      group by cid) Q2
where Q1.cid = Q2.cid;
```

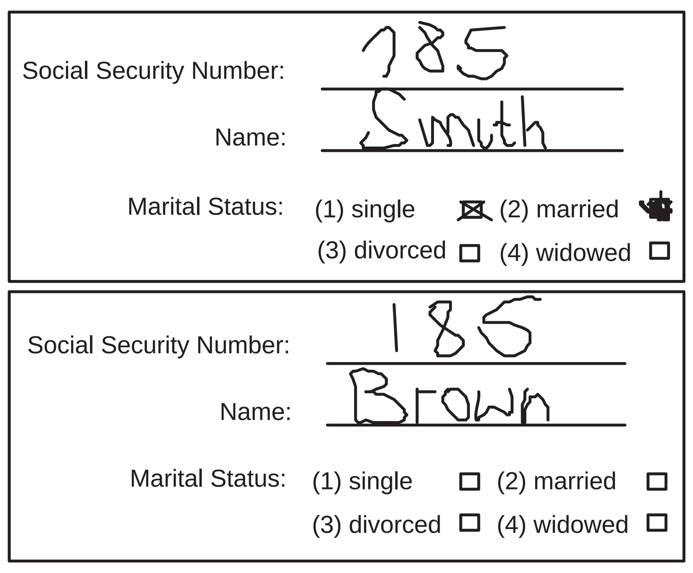
SkillGained	CID	Skill	p1	p2	p
	Google	Web	2/5	2/5	1
	Yahoo	Java	3/5	3/5	1
	Yahoo	Web	2/5	3/5	2/3
	Yahoo	Search	2/5	3/5	2/3

```
select cid, skill from SkillGained where p=1;
```

CID	Skill
Google	Web
Yahoo	Java

Representation system: U-relational databases

- c-tables optimized for discrete probability spaces.
- attribute-level uncertainty via vertical decompositioning.



$U_{R[SSN]}$	$V \mapsto D$	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

$U_{R[M]}$	$V \mapsto D$	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

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

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

Query evaluation

Positive relational algebra and repair-key on probabilistic databases are efficiently mapped to positive relational algebra on U-relations. Leverages state of the art in relational databases.

Example: Names of possibly married (M=2) persons: possible($\pi_N(\sigma_{M=2}(S))$)

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

$U_{S[M]}$	$V \mapsto D$	TID	M
	$x \mapsto 1$	t_1	1
	$x \mapsto 2$	t_1	2
	$z \mapsto 1$	t_2	1
	$z \mapsto 2$	t_2	2

Rewrite query into

$$\pi_N(\sigma_{M=2}(U_{S[N]} \bowtie_{\psi \wedge \phi} U_{S[M]}))$$

with

$$\psi := U_{S[N]}.V = U_{S[M]}.V \Rightarrow U_{S[N]}.D = U_{S[M]}.D$$

$$\phi := (U_{S[N]}.TID = U_{S[M]}.TID)$$

$V_1 \mapsto D_1$	$V_2 \mapsto D_2$	TID	N	M
$y \mapsto 1$	$z \mapsto 2$	t_2	Brown	2

Repair-key on U-relations is just a projection (even though we can create an exponential number of possible worlds)!

Example: Tossing a biased coin twice.

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

repair key Toss **in** R **weight by** FProb;

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

Acknowledgements

This work was supported by German Science Foundation (DFG) grant KO 3491/1-1, by NSF grant IIS-0812272, and by a KDD grant.

Confidence computation (#P-hard)

Three techniques implemented:

1. Exact AI heuristic search technique.
2. For hierarchical queries, PTIME techniques for exact confidence computation. Special secondary storage operator (SPROUT) that requires few sequential passes over the data. Generalizations to obtain larger PTIME query fragment via functional dependencies.
3. Monte Carlo algorithm based on the Karp-Luby FPRAS: polynomial-time approximation algorithm with relative quality guarantees.

Our techniques are the state of the art in confidence computation.

The MayBMS System

- An extension of the Postgres server backend. Compiles and runs on the same platforms as Postgres.
- Postgres APIs and middleware can be used, e.g. ODBC, JDBC, PLSQL, PHP.
- Full SQL support. Same performance as Postgres on nonprobabilistic data.
- Full support for updates, transactions and recovery.
- Secondary storage implementations for all operations.
- Open source, current release 2.1 beta: **http://maybms.sourceforge.net**

Selected publications

C. Koch, MayBMS: A System for Managing Large Uncertain and Probabilistic Databases, Chapter 6 of C. Aggarwal, ed., *Managing and Mining Uncertain Data*, Springer, 2009.

L. Antova, C. Koch, and D. Olteanu. From Complete to Incomplete Information and Back. *SIGMOD 2007*.

——— & T. Jansen. Fast and Simple Relational Processing of Uncertain Data. *ICDE 2008*.

C. Koch. Approximating Predicates and Expressive Queries on Probabilistic Databases. *PODS 2008*.

C. Koch and D. Olteanu. Conditioning Probabilistic Databases. *VLDB 2008*.

D. Olteanu, J. Huang, and C. Koch. SPROUT: Lazy versus Eager Query Plans for Tuple-Independent Probabilistic Databases. *ICDE 2009*.