MayBMS – A System for Managing Large Amounts of Incomplete Information

Christoph Koch
Universität des Saarlandes
koch@infosys.uni-sb.de

Joint work with Lyublena Antova and Dan Olteanu

Incomplete information

- Databases with missing information
- Important in many data management applications:
  - data integration, data exchange
  - data cleaning and warehousing
  - Web information extraction
  - scientific databases
  - computational linguistics
  - management information systems, expert systems ...
- Current database management systems do not support these applications.
- Knowledge representation (AI) has come up with very interesting formalisms such as Answer Set Programming but these do not scale to these applications.

Overview of this talk

- Motivation. Possible worlds semantics.
- World-set SQL
- The MayBMS representation system: World-set Decompositions (WSDs).
  - Leverage existing relational DBMS techniques.
- Efficient query processing on WSDs.
- Minimizing representations.
- Foundations: Expressiveness and complexity, representing infinite world-sets.
- Experiments with MayBMS.
- Conclusions and outlook.

Incomplete information in SQL databases: null values

Consider the relational database table

R	A
	1

What does the following query produce?

```
select 'yes' from R where R.A = R.A;
```

Incomplete information in SQL databases: null values

Consider the relational database table

R	A
	1

What does the following query produce?

```
select 'yes' from R where R.A = R.A;
```

- Answer: one tuple: 'yes'
- Same result if we replace 1 by a different value.

Incomplete information in SQL databases: null values

Consider the relational database table with a null value ("don't know")

R	A
	null

What does the following query produce?

```
select 'yes' from R where R.A = R.A;
```

Incomplete information in SQL databases: null values

Consider the relational database table with a null value ("don't know")

R	A
	null

What does the following query produce?

```
select 'yes' from R where R.A = R.A;
```

- Answer: no tuples
- Explanation: 3-valued condition semantics (true, false, unknown).

Discussion

The semantics of SQL is unintuitive.

What we really want is a possible worlds semantics in which we can reason for query

```
select 'yes' from R where R.A = R.A;
```

as follows.

- No matter what value the null takes, there is a tuple and the condition is true.
- Thus 'yes' is a certain answer: it should be returned in all possible worlds.
- Return 'yes'.

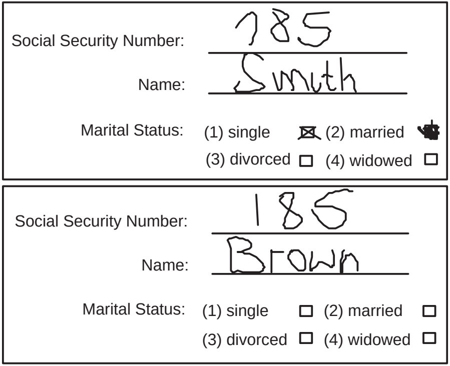
Problem: complexity

- A relational table with simple null values as in SQL but with a possible worlds semantics is called a naive table.
- Unfortunately,

  Theorem (Abiteboul, Kanellakis, Grahne). There is a fixed (and even rather simple) SQL query that is NP-hard to evaluate on naive tables.
- Reason: Naive tables are a very succinct representation of (infinitely) many possible worlds.
- Note: SQL has PTIME query evaluation (if queries are fixed).
- Nevertheless, in many applications we need possible worlds.

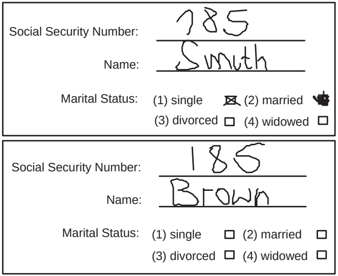
Census data scenario

Suppose we have 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?

Representation systems: naive tables (SQL)


Social Security Number: 185
Name: Smith
Marital Status: (1) single ☒ (2) married ☒
(3) divorced ☐ (4) widowed ☐

Social Security Number: 186
Name: Brown
Marital Status: (1) single ☐ (2) married ☐
(3) divorced ☐ (4) widowed ☐

(TID)	S	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.

1. Smith's SSN is either 185 or 785.
2. Brown's SSN is either 185 or 186.
3. Data cleaning: No two distinct persons can have the same SSN: The case that Smith and Brown both have SSN 185 is excluded.

Motivation: Decision Support Queries

Company_Emp	CID	EID
	Google	e1
	Google	e2
	Yahoo	e3
	Yahoo	e4
	Yahoo	e5

Emp_Skills	EID	Skill
	e1	Web
	e2	Web
	e3	Java
	e3	Web
	e4	Solve problems
	e5	Java

1. Suppose I choose to buy exactly one company.
2. Assume that one (key) employee leaves that company.
3. If I acquire that company, which skills can I obtain for certain?
4. Now list the possible acquisition targets if I want to guarantee to gain the skill "Web" by the acquisition.

Result	CID
	Google

Motivation: Decision Support Queries

Company_Emp	CID	EID
	Google	e1
	Google	e2
	Yahoo	e3
	Yahoo	e4
	Yahoo	e5

Emp_Skills	EID	Skill
	e1	Web
	e2	Web
	e3	Java
	e3	Web
	e4	Solve problems
	e5	Java

- Suppose I choose to buy exactly one company.

U ←

select	*
from	Company_Emp
choice of	CID;

U	CID	EID
	Google	e1
	Google	e2

U	CID	EID
	Yahoo	e3
	Yahoo	e4
	Yahoo	e5

Motivation: Decision Support Queries

Company_Emp	CID	EID
	Google	e1
	Google	e2
	Yahoo	e3
	Yahoo	e4
	Yahoo	e5

Emp_Skills	EID	Skill
	e1	Web
	e2	Web
	e3	Java
	e3	Web
	e4	Solve problems
	e5	Java

- Assume that one (key) employee leaves that company.

V ←

```
select   R1.CID, R2.EID
from     (select * from U choice of EID) R1, Company_Emp R2
where    R1.CID = R2.CID and R1.EID != R2.EID;
```

V	CID	EID
	Google	e1

V	CID	EID
	Google	e2

V	CID	EID
	Yahoo	e3
	Yahoo	e4

V	CID	EID
	Yahoo	e3
	Yahoo	e5

V	CID	EID
	Yahoo	e4
	Yahoo	e5

Motivation: Decision Support Queries

Company_Emp	CID	EID
	Google	e1
	Google	e2
	Yahoo	e3
	Yahoo	e4
	Yahoo	e5

Emp_Skills	EID	Skill
	e1	Web
	e2	Web
	e3	Java
	e3	Web
	e4	Solve problems
	e5	Java

- If I acquire that company, which skills can I obtain for certain?

W ←

select **certain**	CID, Skill
from	V, Emp_Skill
where	V.EID = Emp_Skill.EID
group worlds by	(select CID from V);

W	CID	Skill
	Google	Web

W	CID	Skill
	Yahoo	Java

Motivation: Decision Support Queries

Company_Emp	CID	EID
	Google	e1
	Google	e2
	Yahoo	e3
	Yahoo	e4
	Yahoo	e5

Emp_Skills	EID	Skill
	e1	Web
	e2	Web
	e3	Java
	e3	Web
	e4	Solve problems
	e5	Java

- Now list the possible acquisition targets if I want to guarantee to gain the skill "Web" by the acquisition.

select **possible**	CID
from	W
where	Skill = 'Web';

Result	CID
	Google

Beyond SQL: World-set SQL

- The language of the Google/Yahoo example.
- Syntax of select queries:

select	[(**possible**|**certain**) [**in alternatives**]] *attribute_or_aggregate+*
from	*query+*
where	*condition*
[group by	*attribute+*
[having	*attribute_or_aggregate+*]]
[**choice of**	*attribute+*]
[**group worlds by**	*query*];

- Intuition: Queries execute within each world individually but may look outside if necessary.
- This viewpoint is important for getting a clean semantics to views and query-based updates.

Properties of World-set SQL

- World-set SQL: The language of the Google/Yahoo example.
- Goal: A language that is a natural analog to SQL, on world-sets.
  - Representation-independent.
  - Not too strong and not too weak.
- Conservativity. **Theorem (Antova, K., Olteanu)**. Each single-world to single-world query in World-set SQL is equivalent to an SQL query.
  - Thus W-S SQL is not too strong.
- Efficient reductions. Each 1W/1W query can be efficiently translated to SQL. (+ linear output size)
  - Practical evaluation technique.
  - Translation is interesting, e.g., generalizations of relational division for translating **certain [... group worlds by]**.

Main Goals of the MayBMS Project

- Create a scalable DBMS for supporting incomplete information (world-sets).
- Scalability should be comparable to current relational databases.
- Develop storage and query processing techniques.
- Design a query and data manipulation language (like SQL for RDBMS) for world-set databases.
  - The queries in the previous example were phrased in our language.
- Enable and deploy in novel data management applications.

- Representation system: way of storing sets of worlds (on disk).
- Our approach: World-set Decompositions.
- We will use the census data scenario as a guiding example.

Desiderata for a representation system

1. Succinctness/Space-efficient storage.
   - Usually there are many rather independent local alternatives, which multiply up to a very large number of worlds.
   - Suppose the US census, before cleaning, contains two possible readings for 0.1% of the answers. Then that is on the order of $2^{10,000,000}$ worlds, each one close to one Terabyte of data.
   - But then, not quite independent, or otherwise representation would not be so difficult.
2. Efficient real-world query processing. There is a tradeoff with succinctness and a lower bound from naive tables. We want to do well in practice.
3. Expressiveness/Representability. Ability to represent all query results.

World-set tables

- Tabular representation of set of possible worlds. Uses tuple ids.
- Columns: Fields of one world. Rows: Alternative worlds.

	$t_1.S$	$t_1.N$	$t_1.M$	$t_2.S$	$t_2.N$	$t_2.M$
(World #1)	185	Smith	1	186	Brown	3
(World #2)	185	Smith	1	$\perp$	$\perp$	$\perp$
(World #3)	185	Smith	2	186	Brown	1

- Pad with null values ("bombs") to get uniform arity if not all worlds have the same number of tuples in each relation.
- This represents the world-set

World #1:

(TID)	S	N	M
t_1	185	Smith	1
t_2	186	Brown	3

World #2:

(TID)	S	N	M
t_1	185	Smith	1

World #3:

(TID)	S	N	M
t_1	185	Smith	2
t_2	186	Brown	1

World-set decompositions (WSDs)

World-set table:

$t_1.S$	$t_1.N$	$t_1.M$	$t_2.S$	$t_2.N$	$t_2.M$
185	Smith	1	186	Brown	1
185	Smith	1	186	Brown	2
185	Smith	1	186	Brown	3
185	Smith	1	186	Brown	4
185	Smith	2	186	Brown	1
		⋮			
785	Smith	2	186	Brown	4

WSD: Product decomposition of world-set table.

$t_1.S$
185
785

×

$t_1.N$
Smith

×

$t_1.M$
1
2

×

$t_2.S$
185
186

×

$t_2.N$
Brown

×

$t_2.M$
1
2
3
4

To reverse, compute product of the component relations.

World-set decompositions (WSDs): Bombs

World #1:

(TID)	A	B
t_1	a	c

World #2:

(TID)	A	B
t_1	b	c

World #3:

(TID)	A	B

WSD representation:

$t_1.A$	$t_1.B$
a	c
b	c
$\perp$	$\perp$

$\Rightarrow$

$t_1.A$
a
b

$\times$

$t_1.B$
c
$\perp$

$=$

$t_1.A$	$t_1.B$
a	c
b	c
a	$\perp$
b	$\perp$

$=$

$t_1.A$	$t_1.B$
a	c
b	c
$\perp$	$\perp$
$\perp$	$\perp$

Data Cleaning on WSDs

Consider WSD

$t_1.S$
185
785

$\times$

$t_1.N$
Smith

$\times$

$t_1.M$
1
2

$\times$

$t_2.S$
185
186

$\times$

$t_2.N$
Brown

$\times$

$t_2.M$
1
2
3
4

We clean this dataset – no two persons can have the same SSN.

$t_1.S$	$t_2.S$
185	186
785	185
785	186

$\times$

$t_1.N$
Smith

$\times$

$t_1.M$
1
2

$\times$

$t_2.N$
Brown

$\times$

$t_2.M$
1
2
3
4

Note: you cannot represent this world-set using single attribute components ("or-sets").

Queries: Relational selection (select * from R where C=7)

Query $P := \sigma_{C=7}(R)$ on WSD

R.t_1.A
1
2

×

R.t_1.B	R.t_1.C	R.t_2.B
1	0	3
2	7	4

×

R.t_2.A
4
5

×

R.t_2.C
0

1. Replace all values in $R.t_i.C$ fields that are different from 7 by bombs.
2. Propagate bombs to all $P.t_i.B$, i.e., other fields of the same tuple, within the same component. Result:

P.t_1.A
1
2

×

P.t_1.B	P.t_1.C	P.t_2.B
$\perp$	$\perp$	3
2	7	4

×

P.t_2.A
4
5

×

P.t_2.C
$\perp$

3. Remove tuples that are "mined" in all worlds. Result:

P.t_1.A
1
2

×

P.t_1.B	P.t_1.C
$\perp$	$\perp$
2	7

Note: Join conditions $A = B$ require the merging of the components of $t_i.A$ and $t_i.B$ if they are different.

Queries: Relational projection (select A from R)

Given WSD

$R.t_1.A$
a

$\times$

$R.t_2.A$
b

$\times$

$R.t_1.B$	$R.t_2.B$
c	$\perp$
$\perp$	d

representing world-set

(TID)	A	B
t_1	a	c

(TID)	A	B
t_2	b	d

The projection $\pi_A(R)$ is

$P.t_1.A$	$P.t_2.A$
a	$\perp$
$\perp$	b

Although of low complexity, the algorithm is rather involved.

Queries: Relational product (select * from R, S)

WSDs are product decompositions: no need to merge components.

$R.t_1.A$
1
2

×

$R.t_1.B$	$R.t_2.A$
3	5
4	6

×

$R.t_2.B$
7
8

×

$S.t_1.C$
a
b

×

$S.t_1.D$	$S.t_2.C$
c	e
d	f

×

$S.t_2.D$
g
h

(a) WSD of two relations R and S.

$t_{11}.A$	$t_{12}.A$
1	1
2	2

×

$t_{11}.B$	$t_{12}.B$	$t_{21}.A$	$t_{22}.A$
3	3	5	5
4	4	6	6

×

$t_{21}.B$	$t_{22}.B$
7	7
8	8

×

$t_{11}.C$	$t_{21}.C$
a	a
b	b

×

$t_{11}.D$	$t_{21}.D$	$t_{12}.C$	$t_{22}.C$
c	c	e	e
d	d	f	f

×

$t_{12}.D$	$t_{22}.D$
g	g
h	h

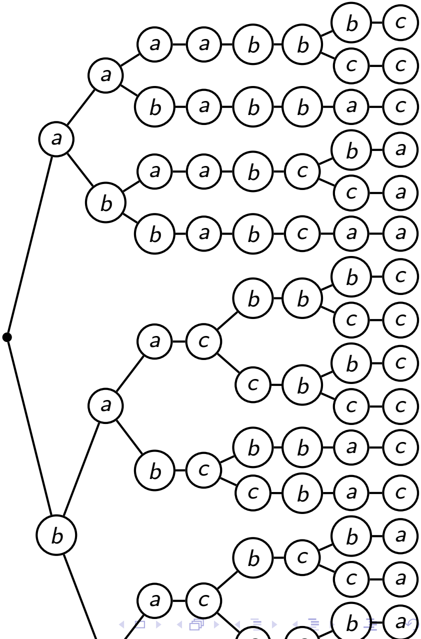
(b) WSD of their product $R \times S$.

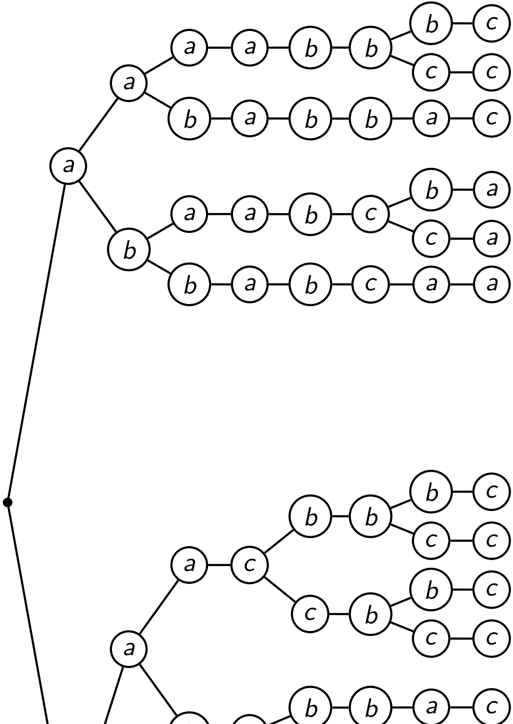
Minimizing WSDs

- Given a WSD, find an as small as possible equivalent representation for it.
- If we think of WSDs simply as product-decompositions, there is actually a unique minimal equivalent representation as a WSD: the prime factorization.
- This is closely related to work by Brayton on factorizing algebraic functions.
- **Theorem (Antova, K., Olteanu)**. The primes can be computed efficiently in time $O(n \log n)$ on disk.
- In linear time in main memory if we assume the arity of the input relation fixed.
- But WSDs have a richer structure: We will see later that the primes do not necessarily provide a minimal decomposition.

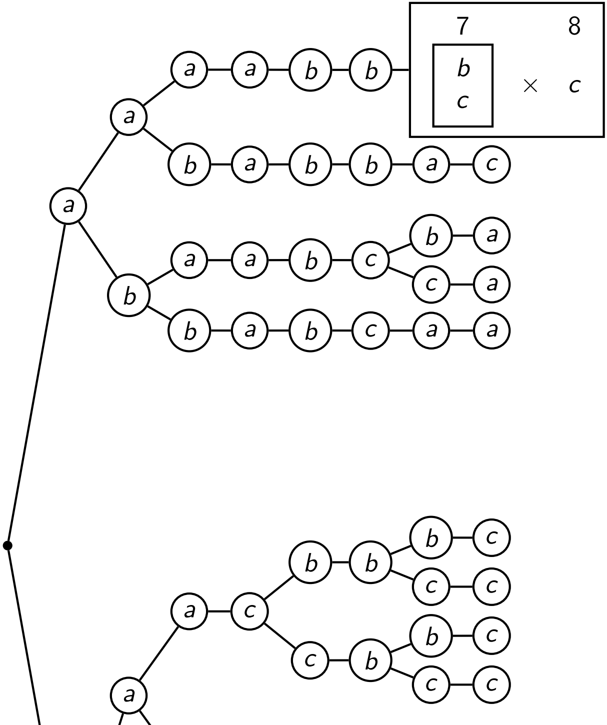
Relation and Corresponding Trie

1	2	3	4	5	6	7	8
a	a	a	a	b	b	b	c
a	a	a	a	b	b	c	c
a	a	b	a	b	b	a	c
a	b	a	a	b	c	b	a
a	b	a	a	b	c	c	a
a	b	b	a	b	c	a	a
b	a	a	c	b	b	b	c
b	a	a	c	b	b	c	c
b	a	a	c	c	b	b	c
b	a	a	c	c	b	c	c
b	a	b	c	b	b	a	c
b	a	b	c	c	b	a	c
b	b	a	c	b	c	b	a
b	b	a	c	b	c	c	a
b	b	a	c	c	c	b	a
b	b	a	c	c	c	c	a
b	b	b	c	b	c	a	a
b	b	b	c	c	c	a	a

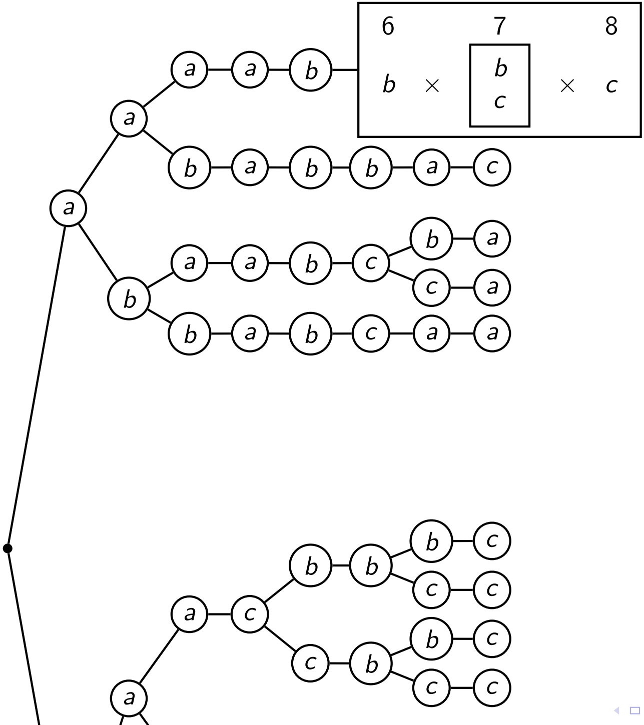


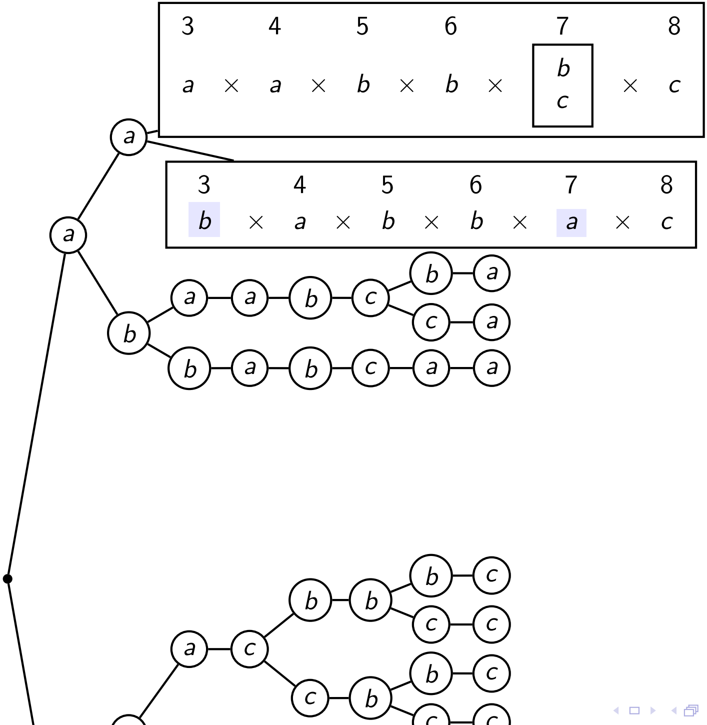


b c
a a b b
c c
a
b a b b a c
a
b a
a a b c
c a
b
b a b c a a
b c
b b
c c
a c
b c
c b
c c
a
b b a c

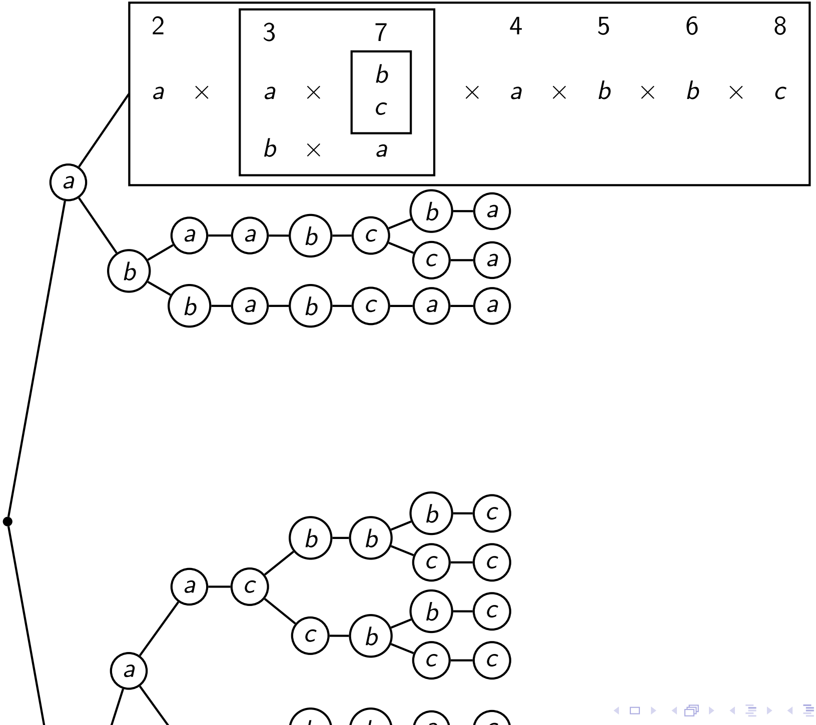


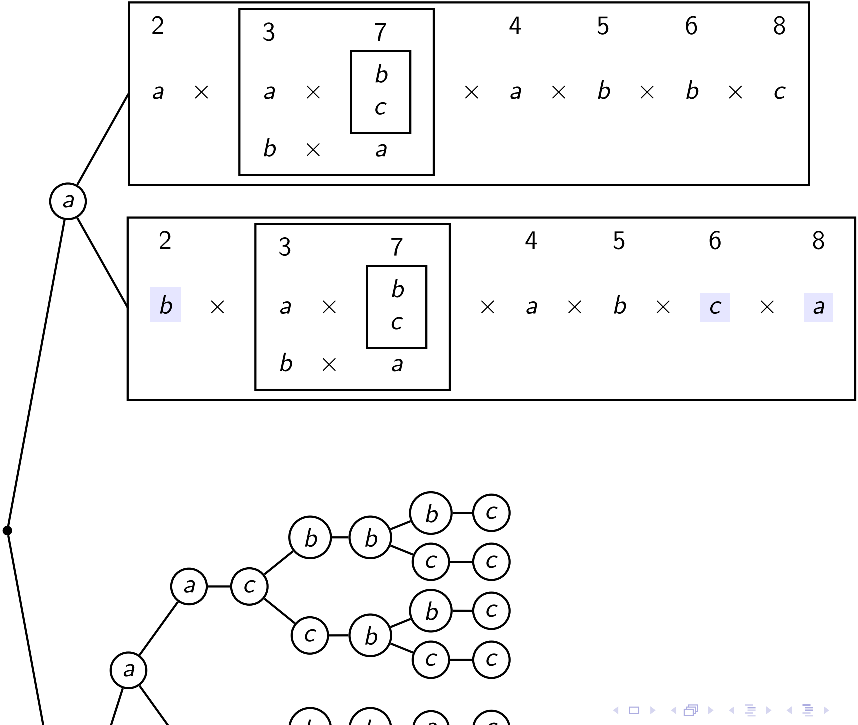
7
8
b
c
×
c

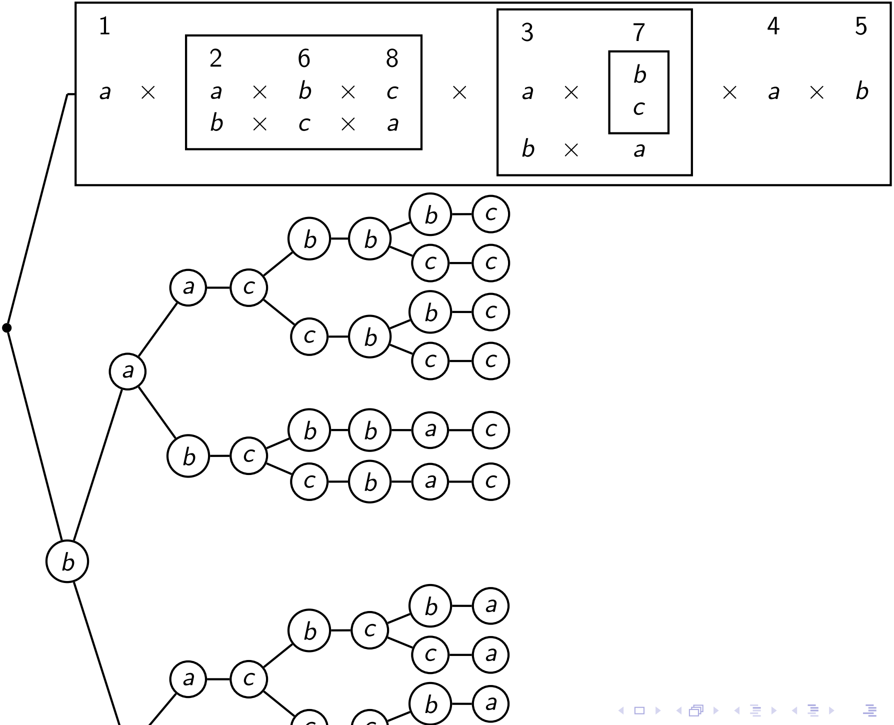

6
7
8
b × b c × c
a a b
a
b a b b a c
a
a a b c b a
c a
b
b a b c a a
b b b c
c c
a c
c b b c
c c
a



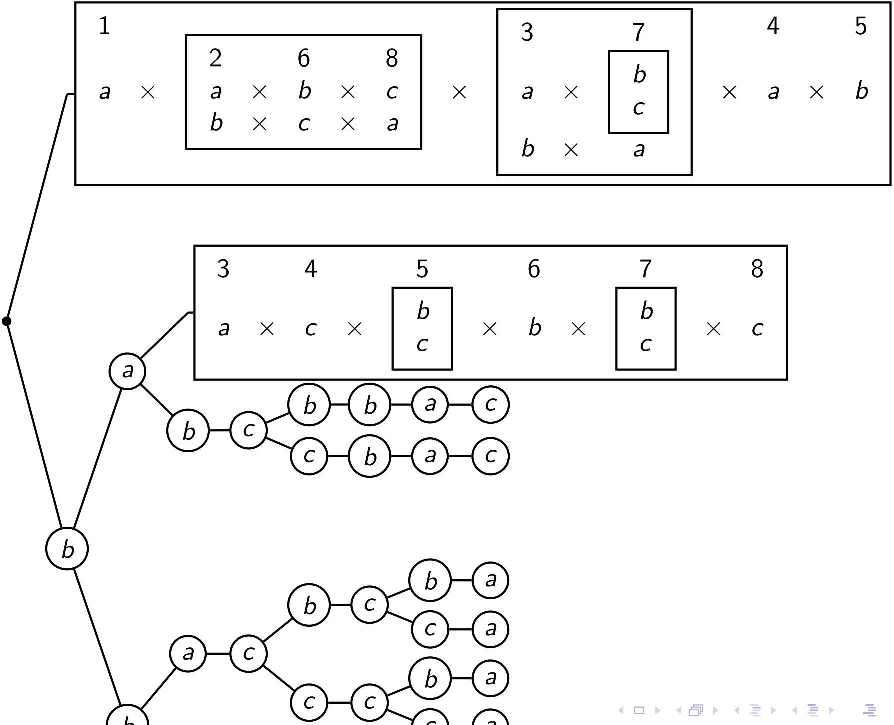
3 4 5 6 7 8
a × a × b × b × b c × c
3 4 5 6 7 8
b × a × b × b × a × c
a
a
b
a a b c b a
c a
b a b c a a
a c b b b c
c c
c b b c
c c


2
3
7
4
5
6
8

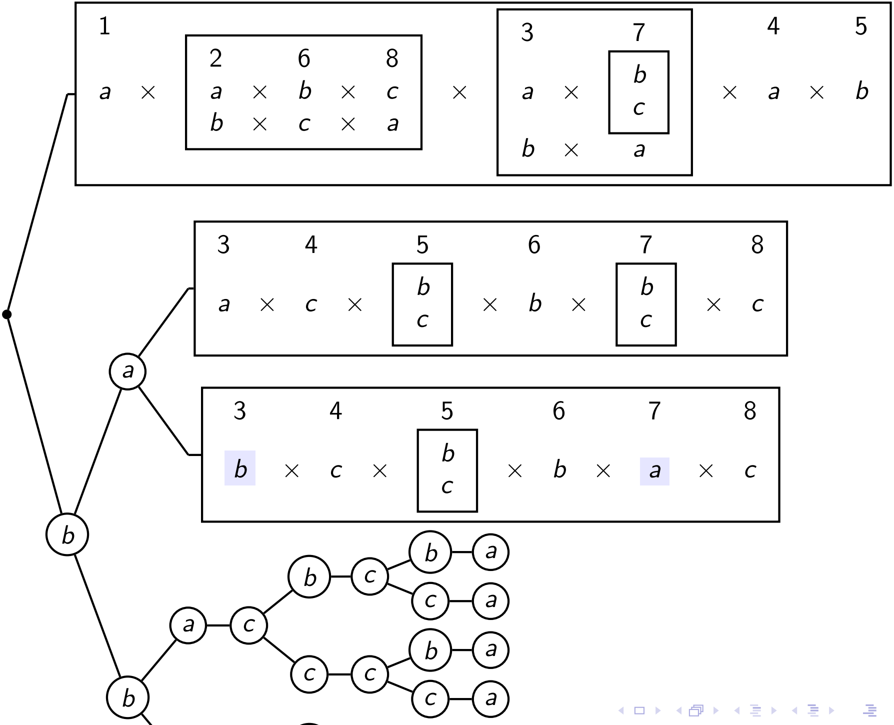

2 3 7 4 5 6 8
a × a × b c × a × b × b × c
b × a
a
2 3 7 4 5 6 8
b × a × b c × a × b × c × a
b × a
a c b b b c
c c
c b b c
c c
a



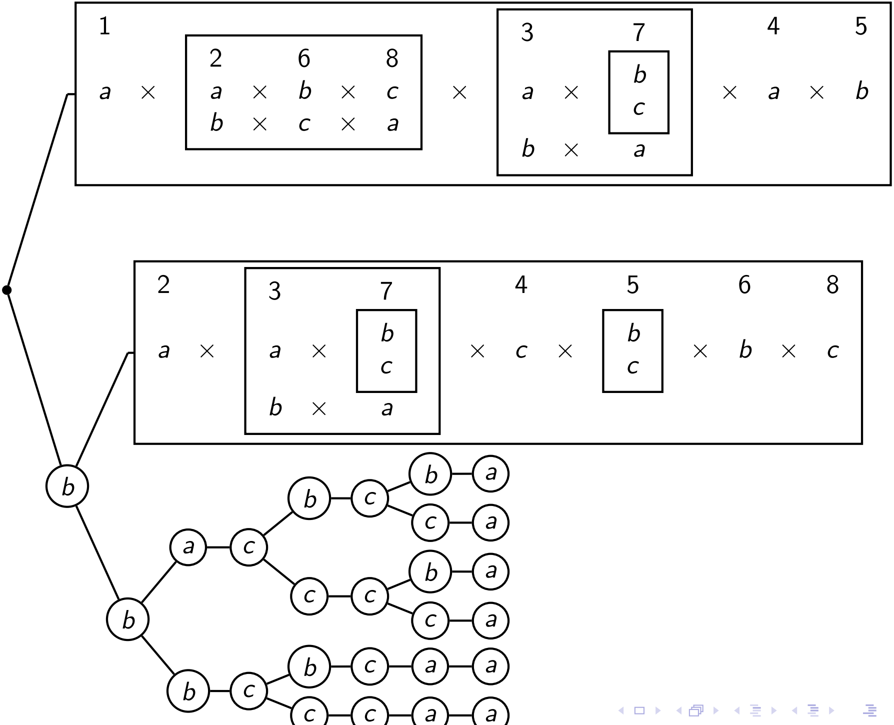
1 a × [2 6 8 / a × b × c / b × c × a] × [3 7 / a × [b c] / b × a] × 4 a × 5 b
a c b b b c
c c
c b b c
c c
b c b b a c
c b a c
b a c b c b a
c a
c c b a



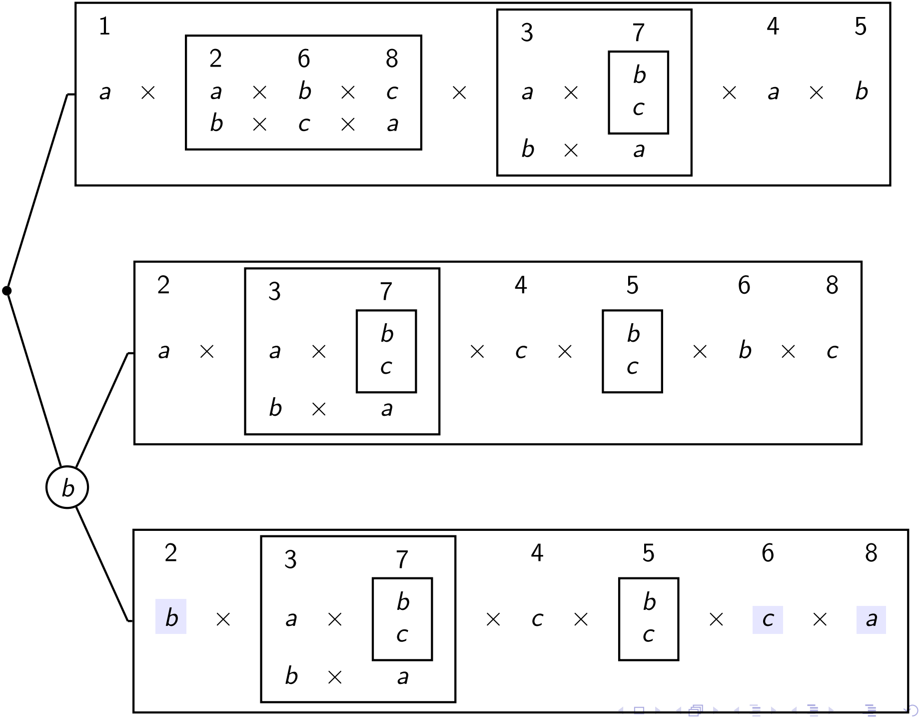
1 2 6 8 3 7 4 5
a × a × b × c × a × b × a × b
b × c × a
c
b × a
3 4 5 6 7 8
a × c × b × b × b × c
c c
a b c b b a c
c b a c
b b c b a
c a
a c c b a
c



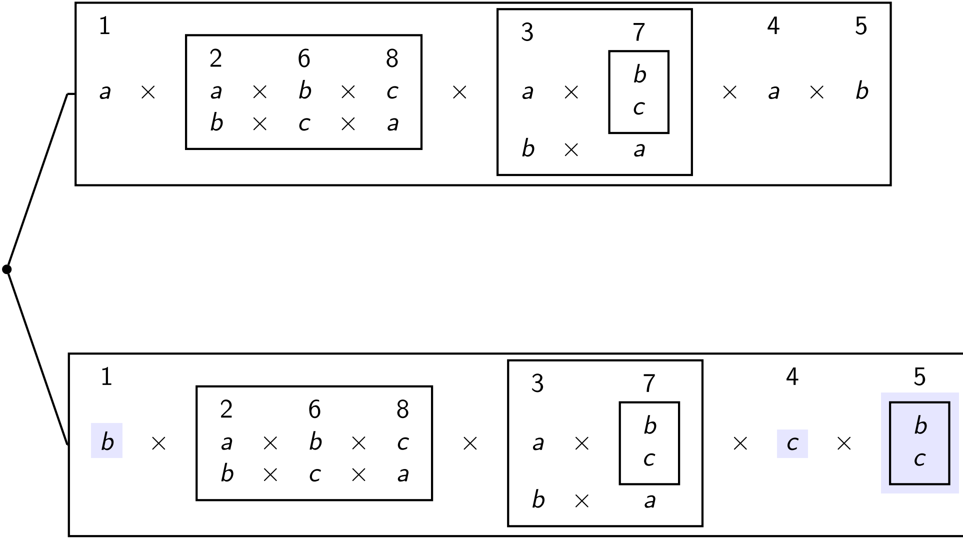
1
a ×
2 6 8
a × b × c
b × c × a
×
3 7
a × b c
b × a
× 4 a × 5 b
3 4 5 6 7 8
a × c × b c × b × b c × c
3 4 5 6 7 8
b × c × b c × b × a × c
b
a
b
a
c
b
c
b a
c a
c
c
b a
c a

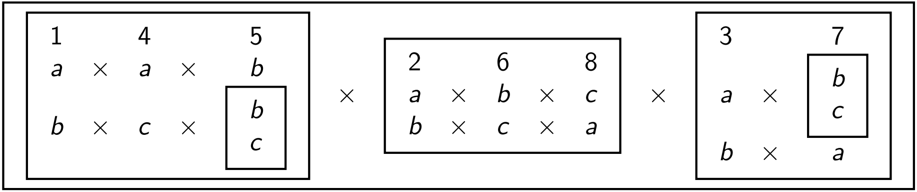


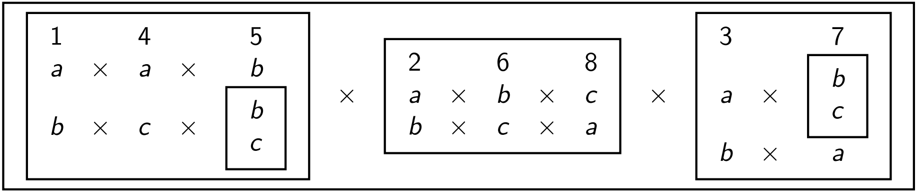
1 2 6 8 3 7 4 5
a × a × b × c × a × b × a × b
b × c × a
c
b × a
2 3 7 4 5 6 8
a × a × b × c × b × b × c
c c
b × a
b
b a c b c b a
c a
c c b a
c a
b c b c a a
c c a a



1
a × 2 6 8 a × b × c b × c × a × 3 7 a × b c b × a × 4 a × 5 b
2
a × 3 7 a × b c b × a × 4 c × 5 b c × 6 b × 8 c
b
2
b × 3 7 a × b c b × a × 4 c × 5 b c × 6 c × 8 a


1 2 6 8 3 7 4 5
a × a × b × c × a × b × a × b
b × c × a
c
b × a
1 2 6 8 3 7 4 5
b × a × b × c × a × b × c × b
b × c × a
c
c
b × a


1 4 5
a × a × b
b × c × b c
×
2 6 8
a × b × c
b × c × a
×
3 7
a × b c
b × a



Decomposition result:

1	4	5
a	a	b
b	c	b
b	c	c

×

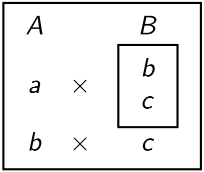
2	6	8
a	b	c
b	c	a

×

3	7
a	b
a	c
b	a

Product-Union Decompositions

We can think of



as a decomposition with respect to the operations product $\times$ and union $\cup$ of relational algebra – it corresponds to

$$\{a\} \times (\{b\} \cup \{c\}) \cup b \times c.$$

Unfortunately, as such the results of our algorithm are not necessarily minimal.
Think of the above as a term

$$a_1 \cdot (b_2 + c_2) + b_1 \cdot c_2.$$

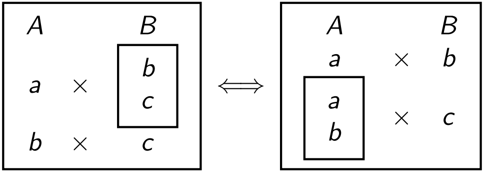
(The indices make sure that we will not by confuse the columns from which the values come.)

Factors vs. Kernels

The terms $(b_2 + c_2)$, $(a_1 + b_1)$ are known as the kernels of

$$a_1 \cdot (b_2 + c_2) + b_1 \cdot c_2 \;=\; a_1 \cdot b_2 + (a_1 + b_1) \cdot c_2 .$$

We can use the corresponding unions in our decompositions to switch between the equivalent decompositions of a relation.



- Previous work on minimizing algebraic and logic functions, and BDDs, tells us the problem we are facing is hard [cf. e.g. the work by Brayton; Bryant].
- However, there are rather few kernels in each relation, and they characterize the choices we can make during optimization.

Subtleties regarding the minimization of WSDs

- **Theorem (Antova, K., Olteanu)**. The minimization algorithm yields minimal decompositions if we have tuple ids and no bombs.
- Tuple ids are important in some applications, but may be ignored in others.
- Then the algorithm does not necessary yield a minimal result:

$t_1.A$	$t_1.B$	$t_2.A$	$t_2.B$
a	b	c	d
c	d	e	b

$\Longrightarrow$

$t_1.A$
a
e

$\times$

$t_1.B$
b

$\times$

$t_2.A$
c

$\times$

$t_2.B$
d

- Bombs sometimes give additional ways of expressing world-sets as small WSDs.
- Nevertheless, the algorithm does a good job and decomposing.
- Generally speaking, strict minimality is neither necessary nor worth the cost.

Strong representation systems

Representation system. A tuple $(\mathbf{W}, \mathit{rep})$ of a a set of structures (databases) $\mathbf{W}$ and a function *rep* that maps from $\mathbf{W}$ to sets of worlds.

Examples: *the* naive tables; *the* WSDs.

Intuition.
Strong representation systems: "Representability under queries".
Representation systems that are *closed under application of queries*.

Definition (strong representation system). A representation system $(\mathbf{W}, \mathit{rep})$ is called *strong* for a query language L if, for each query $Q \in L$ and each $\mathcal{W} \in \mathbf{W}$, there is a structure $\mathcal{W}' \in \mathbf{W}$ such that

$$\mathit{rep}(\mathcal{W}') = \{Q(\mathcal{A}) \mid \mathcal{A} \in \mathit{rep}(\mathcal{W})\}.$$

Note: many representation systems proposed in the literature are actually not strong for relational algebra/SQL: naive tables (Codd), or-set relations (Imielinski et al.), v-tables (Imielinski and Lipski), ...

Properties of WSDs

- **Observation**. Any finite world-set can be represented by a WSD.
- By construction: Each world-set can be written as a world-set table; a world-set table is a (usually bad) WSD.

- It follows that WSDs are a strong representation system for any relational query language.

- WSDs can be immediately stored in a relational database.
- WSDs are conveniently succinct and still appropriate for large-scale query processing.

Conditional tables (c-tables)

- Tables with (possibly co-occurring) variables.
- A global condition that must be satisfied in order for the world to exist.
- For each tuple a local condition that must be satisfied or else the tuple is dropped from the world.
- C-tables can represent any finite world-set and in addition many infinite ones (using variables)!

Theorem (Imielinski and Lipski). c-tables are a strong representation system for relational algebra.

T	S	N	M	Cond
	x	Smith	y	true
	z	Brown	w	true

Global cond. : $((x = 185 \wedge z = 186) \vee (x = 785 \wedge z = 185) \vee (x = 785 \wedge z = 186)) \wedge (y = 1 \vee y = 2) \wedge (w = 1 \vee w = 2 \vee w = 3 \vee w = 4)$

WSDs with variables

- WSDs can represent just finite world-sets, unlike c-tables (or even naive tables).
- gWSDs: WSDs with variables and a global conjunction of inequalities.
- gWSDs have the same expressive power as c-tables.

  Theorem (Antova, K., Olteanu). gWSDs = c-tables.
- Thus gWSDs are a strong representation system for relational algebra.
- The operations of relational algebra are slight generalizations of those for WSDs.
- The minimization algorithm works (treat variables like domain elements), but now there is another source of nonminimality.

Complexity results for standard decision problems

- Relational algebra query Q fixed.
- No decompositions of tuples – "tuple-level" WSDs.
- For instance, tuple Q-certainty: Given tuple t, $\mathcal{W}$, is $t \in Q(I)$ for every world $I \in rep(\mathcal{W})$?
- Instance Q-possibility: Given instance J, $\mathcal{W}$, is there an instance $I \in rep(\mathcal{W})$ such that $J = Q(I)$?
- gWSDs: expressiveness of c-tables, only complexity of v-tables (even though they are exponentially more succinct than v-tables).

	v-tables	gWSD	c-tables
Tuple Possibility	PTIME	PTIME	NP-complete
Instance Possibility	NP-complete	NP-complete	NP-complete
Tuple Q-Possibility	PTIME*	PTIME*	NP-complete
Instance Q-Possibility	NP-complete	NP-complete	NP-complete
Tuple Certainty	PTIME	PTIME	coNP-compl.
Instance Certainty	PTIME	PTIME	coNP-compl.
Tuple Q-Certainty	coNP-compl.	coNP-compl.	coNP-compl.
Instance Q-Certainty	coNP-compl.	coNP-compl.	coNP-compl.

* Result for positive relational algebra.

A Possible Worlds Base Management System

- Currently under development, a prototype exists.
- Built on top of Postgres.
- Currently implements only WSDs – no variables. Finite world-sets.
- We have done quite extensive experiments, I will only discuss an example query.
- Data cleaning: “chasing” of integrity constraints.
- **http://www.infosys.uni-sb.de/projects/maybms/**

Two improvements over WSDs in MayBMS

WSD with Templates (WSDT):

Template	S	N	M
t_1	?	Smith	?
t_2	?	Brown	?

t_1.S	t_2.S
185	186
785	185
785	186

$\times$

t_1.M
1
2

$\times$

t_2.M
1
2
3
4

- Store the data values common to all worlds in a conventional database table (with nulls), the template relation.
- Uniform WSDTs: store component relations in a single relation with schema (component id, tuple id, local world id, attribute name, value).
  - Otherwise WSDs may require millions of relational tables of, in the worst case, unbounded arity.

Experiments

- Data set: 5% of US census, anonymized (only 5% is publicly available).
- One relation with $\approx$ 12.5 millon tuples, 50 columns.
- Noise was artificially introduced: for up to 0.1% of the fields we introduced 2 to 8 alternatives.
  - About 2^{10^6} worlds, each one several GB large.
- Data cleaning to eliminate certain possible worlds (leads to merging of components).
- Data cleaning constraints and queries were written by us.

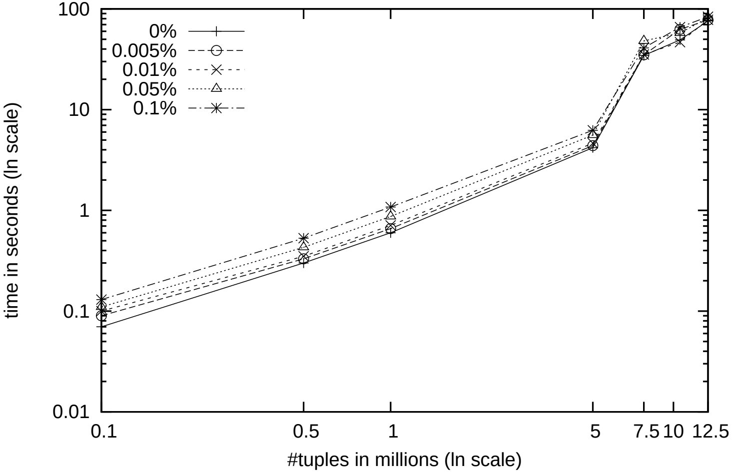
Experiments

	Density	0.005%	0.01%	0.05%	0.1%
Initial	#comp	31117	62331	312730	624449
After chase	#comp	30918	61791	309778	612956
	#comp>1	249	522	2843	10880
	$\lvert C \rvert$	108276	217013	1089359	2150935
	$\lvert R \rvert$	12.5M	12.5M	12.5M	12.5M
After Q_2	#comp	25	56	312	466
	#comp>1	0	1	8	9
	$\lvert C \rvert$	93	269	1682	2277
	$\lvert R \rvert$	82995	83052	83357	83610

Query Q_2 returns information including the place of birth of US citizens born outside to US who do not speak English well:

$$Q_2 = \pi_{\mathrm{POWSTATE,CITIZEN,IMMIGR}}(\sigma_{\mathrm{CITIZEN}<>0 \wedge \mathrm{ENGLISH}>3}(R))$$

Experiments



Summary

- MayBMS: A system for the scalable management of incomplete information.
- Foundations: Expressivenes, Representability, Complexity, Minimization.
- Leverages relational DBMS technologies.
- Experiments.
  - On selection/projection queries we in practice have a constant overhead of a factor of 3 to 5 over Postgres running the same queries on a single world.
  - Joins show exponential behaviour, but we can do them for this census database.
  - Universal operations (difference, certain aggregations) need further work.

Probabilistic WSDs

- Traditionally: very strong independence assumptions.
  - Example: MystiQ (U. Washington).
- Probabilities on worlds can be added to WSDs in a straightforward way. Decompositions manifest independence:

C_1	$t_1.A$	P
	a	0.3
	b	0.7

$\times$

C_1	$t_2.A$	P
	c	0.9
	d	0.1

Here world

R	A
	a
	d

has probability $0.3 \cdot 0.1$.

- Query processing on probabilistic WSDs: recent demo at ICDE'07.

Future work: managing data extensionally and intensionally

- Currently, MayBMS stores all its data extensionally (in WSDs).
  - Take a data cleaning rule, remove impossible worlds, never think of them again.
- Universal operations can merge large numbers of components ⇒ Representation gets large.
- Intensional approach: Trio (Stanford) stores its information in a weak representation system making strong independence assumptions and adds constraints (called lineage) to represent dependencies. For query answering, always apply all the constraints.
  - Higher complexity and overhead, but representation remains small.
  - No experiments with Trio have been reported.
- Current work: combine the advantages of the intensional and extensional approach in an intelligent way.
- Product-union decompositions are a different way of dealing with the blowup in representation size.

END

References

- MayBMS Homepage
  http://www.infosys.uni-sb.de/projects/maybms/
- L. Antova, C. Koch, and D. Olteanu. From Complete to Incomplete Information and Back. *Proc. SIGMOD 2007*, Beijing, China.
- L. Antova, C. Koch, and D. Olteanu. World-Set Decompositions: Expressiveness and Efficient Algorithms. *Proc. ICDT 2007*, Barcelona, Spain.
- L. Antova, C. Koch, and D. Olteanu. 10^{10^6} Worlds and Beyond: Efficient Representation and Processing of Incomplete Information. *Proc. ICDE 2007*, Istanbul, Turkey.
- L. Antova, C. Koch, and D. Olteanu. MayBMS: Managing Incomplete Information with Probabilistic World-Set Decompositions. *Proc. ICDE 2007*, Istanbul, Turkey. (Demo Paper.)
- L. Antova, T. Jansen, C. Koch, and D. Olteanu. Fast and Simple Relational Processing of Uncertain Data. Submitted for publication, 2007.