MayBMS: Managing Incomplete Information with Probabilistic World-Set Decompositions

Lyublena Antova, Christoph Koch, and Dan Olteanu

Lehrstuhl für Informationssysteme
Universität des Saarlandes, Germany

1 The MayBMS System

Managing incomplete information is important in many real world applications. In this demonstration we present MayBMS - a system for representing and managing finite sets of possible worlds - that successfully combines expressiveness and efficiency. Some features of MayBMS are

- Completeness of the representation system for finite world-sets.
- Space-efficient representation of large world-sets.
- Scalable evaluation and support for full relational algebra queries.
- Probabilistic extension of the representation system and the query language.

MayBMS is implemented on top of PostgreSQL. It models incomplete data using the so-called *world-set decompositions (WSDs)* [2]. For this demonstration, we introduce a probabilistic extension of world-sets and WSDs, where worlds or correlations between worlds have probabilities. The main idea underlying probabilistic WSDs is to use relational factorization combined with probabilistic independence in order to efficiently decompose large world-sets into a set of independent smaller relations.

Queries in MayBMS can be expressed in an SQL-like language with special constructs that deal with incompleteness and probabilities. MayBMS rewrites and optimizes user queries into a sequence of relational queries on world-set decompositions.

Efficiency and Scalability. The efficiency of MayBMS was confirmed experimentally in a large census data scenario [2]. The dataset represented a 5% extract from the 1990 US census with nearly 12.5 million records and 50 columns [3], occupying around 3GB of disk space. We introduced noise with different degree of incompleteness to the data by replacing randomly picked values with or-sets. Thus in one test scenario the resulting world-set contained more than 2^{624449} different worlds, each of the size of the original dataset. MayBMS was capable of representing them with a space overhead of only 2% over the original relation. The second part of the experiments showed how data cleaning procedures can be used in MayBMS. We cleaned the world-set from inconsistencies by enforcing real-life integrity constraints. In the third part we evaluated a number of queries on the cleaned data. The performance of query evaluation on incomplete data was compared to that of conventional query processing (that is, of processing a single world using standard database techniques). Our results showed that the processing time on large world-sets is very close to that on a single world. More information on the experiments can be found in [2].

Demonstration scenarios. In this demonstration we will present the main features of MayBMS using real world datasets. In the first part we will introduce the data representation used by our system using examples from the census data scenario, and of the medicine data collection, detailed in the next section.

We will then demonstrate the query processing of MayBMS, starting from the simple select-from-where queries, and continuing with queries that make use of the probabilities. We will also show the optimized query plans produced by MayBMS.

2 World-set Decompositions

We illustrate WSDs using a medical scenario describing diagnoses, tests, and symptoms [1]:

r_1.Diagnosis	r_1.Test	p
pregnancy	ultrasound	0.4
hypothyroidism	TSH	0.6

$\times$

r_1.Symptom	p
weight gain	0.7
fatigue	0.3

$\times$

r_2.Diagnosis	p
obesity	1

$\times$

r_2.Test	p
BMI	1

$\times$

r_2.Symptom	p
weight gain	1

The above WSD is defined as a relational product of five relations, hereafter called components. Each component defines values for a set of fields, and a world is obtained as a combination of one tuple from each of the components.

Thus our WSD represents four worlds, where each world can be seen as one possible patient record. For example, one such record states that the patient is suspect of hypothyroidism and obesity because of weight gain, and a TSH (thyroid stimulating hormone), respectively BMI (body mass index), measurements are recommended. This record is obtained by choosing the second tuple of the first component, and the first tuple of each of the remaining components.

The main principle of WSDs is to store independent tuple fields in separate components and dependent tuple fields within the same component. Thus component tuples do not necessarily correspond to tuples of a relation defined in a world, but rather to dependencies of fields across several tuples (or even across several relations). In our example, the tuples with identifiers r_1 and r_2 are independent and therefore stored in separate components. Also, the field r_1.Symptom in the r_1-tuple is independent of the other fields of r_1.

We can also assign probabilities to dependencies in the data by simply extending each component with a special probability column and assigning probabilities to each component tuple. The probability of a world defined by a set of tuples, with each tuple from a different component, is then the product of the probabilities of all tuples in this set. To ensure that the probabilities for all worlds in the world-set sum up to one, we just need to enforce that the probabilities for the tuples of each component sum up to one. This probabilistic extension of WSDs fits nicely the explicit modeling of data independence on which WSDs are based. In our example, the two alternatives for r_1.Diagnosis and r_1.Test have probabilities 0.4 and 0.6 respectively. The patient record described above represents a world with probability $0.6 \cdot 0.7 \cdot 1 \cdot 1 \cdot 1 = 0.42$.

The query language of MayBMS is a natural extension of SQL with special constructs that deal with incompleteness and probabilities. We sketch query evaluation on WSDs using an SQL query. Consider the following query asking for the tests recommended in pregnancy diagnosis (we assume that the above WSD defines a relation R)

```
select Test from R where Diagnosis='pregnancy'
```

The semantics of query evaluation on world-sets is to evaluate the query in each of the worlds. On WSDs, the evaluation is more involved, as WSDs can be exponentially more succinct than the sets of worlds they represent. Selecting only the tuples of R that satisfy the selection criterion is a fairly subtle operation, as a tuple from R spreads over several components and the deletion of a component tuple does not necessarily correspond to the deletion of a tuple in R. Thus a selection must not delete component tuples, but should mark Diagnosis fields as belonging to deleted tuples of R using the special value $\perp$. To compute the selection, we scan the components that define values for the attribute Diagnosis of tuples in R and replace the values different from 'pregnancy' by $\perp$:

r_1.Diagnosis	r_1.Test	p
pregnancy	ultrasound	0.4
$\perp$	TSH	0.6

$\times$

r_1.Symptom	p
weight gain	0.7
fatigue	0.3

$\times$

r_2.Diagnosis	p
$\perp$	1

$\times$

r_2.Test	p
BMI	1

$\times$

r_2.Symptom	p
weight gain	1

This answer represents three worlds: the first world is {(pregnancy, ultrasound, weight gain)}, the second world is {(pregnancy, ultrasound, fatigue)}, and the third world is empty. Note that none of these worlds contains r_2-tuples. Thus, we can safely drop all components columns defining values for r_2-tuples. Also, the second tuple of the first component contains a $\perp$ value for the Diagnosis field of the r_1-tuple, which means that r_1 does not exist in this world. Using these observations, we normalize the above WSD and obtain:

r_1.Diagnosis	r_1.Test	p
pregnancy	ultrasound	0.4
$\perp$	$\perp$	0.6

$\times$

r_1.Symptom	p
weight gain	0.7
fatigue	0.3

After the projection, we obtain the WSD with two worlds (out of which the world encoded by the second component tuple is empty)

r_1.Test	p
ultrasound	0.4
$\perp$	0.6

The interpretation is that the ultrasound test is recommended in pregnancy diagnosis with probability 0.4.

The above query does not explicitly access the probability information stored with the data. MayBMS also allows SQL-like queries with probability constructs in the select and where clauses. For example, asking for the probability of the ultrasound test being recommended in pregnancy diagnosis would retrieve in all worlds a unary relation with one tuple corresponding to the value 0.4. This query uses the probability construct `prob()` in the select clause. In case the ultrasound test is recommended in several worlds, then the answer to our query would be computed by summing up the probabilities of this event over all such worlds.

References

[1] `http://www.medicinenet.com`.

[2] L. Antova, C. Koch, and D. Olteanu. $\mathbf{10^{10^6}}$ worlds and beyond: Efficient representation and processing of incomplete information. In *Proc. ICDE*, 2007.

[3] S. Ruggles, M. Sobek, T. Alexander, C. A. Fitch, R. Goeken, P. K. Hall, M. King, and C. Ronnander. Integrated public use microdata series: V3.0, 2004. http://www.ipums.org.