Type providers

Tomas Petricek

November 14, 2012

1 Introduction

TBD

- **Information spaces** describes what?
- **Schema** describes what?

Products	
Name	**Price**
Tea	3.50
Coffee	2.25
Water	0.0

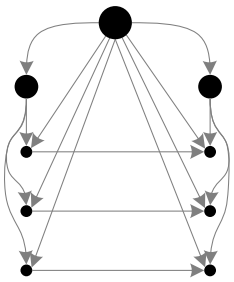

Figure 1: Mapping database table to an information space

2 Formal model

2.1 Information spaces

The first problem in formalizing type providers is to find a formal model of *information space* that is simple and amenable to formal analysis, but powerful enough to model a wide range of data sources accessible using type providers including hierarchical data (i.e. XML files), SQL databases, multi-dimensional data (such as the World Bank data set) etc.

We model information spaces simply as undirected graph with values as vertices and relations between values as edges.

Definition 1. *An* information space $(\mathcal{V}, \mathcal{R})$ *is a tuple of values* $\mathcal{V}$ *and relations between pairs of values* $\mathcal{R} \in \mathcal{V} \times \mathcal{V}$*. The relations are undirected, meaning that* $(v_1, v_2) \in \mathcal{R} \Leftrightarrow (v_2, v_1) \in \mathcal{R}$*.*

2.2 Examples

To demonstrate that the simple definition is general enough, we look at two examples that demonstrate how databases and XML data can be viewed as information spaces.

Databases. For simplicity, we only consider database with a simple table (Products) with two columns (Name and Price) where the first column is the primary key. The mapping is demonstrated in Figure 1.

The set of values is represented as dots in the figure. It is obtained by taking all data and meta-data information about the database. Assuming that P is a collection of table data indexed by I (in our example, pairs consisting of name and price), we construct the set of values by taking name of the table, names of columns and all values:

$$\mathcal{V} = \{\texttt{"Products"}, \texttt{"Name"}, \texttt{"Price"}\} \cup \bigcup_{i \in I} \{n_i | (n_1, \ldots, n_k) \in P\}$$

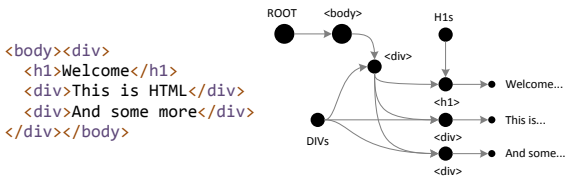

Figure 2: Stuff