Consistent Query Answering in Partially Consistent Databases

Fabian Pijcke (FNRS grant)

October 21, 2013

A database is called uncertain if two or more tuples of the same relation are allowed to agree on their primary key. Intuitively, such tuples act as alternatives for each other. A repair (or possible world) of such uncertain database is obtained by selecting a maximal number of tuples without ever selecting two tuples of the same relation that agree on their primary key. For a Boolean query $\mathbf{q}$, the problem $\mathbf{CERTAINTY}(\mathbf{q})$ takes as input an uncertain database $\mathbf{db}$ and asks whether $\mathbf{q}$ evaluates to true on every repair of $\mathbf{db}$. In recent years, the complexity of $\mathbf{CERTAINTY}(\mathbf{q})$ has been studied under different restrictions on $\mathbf{q}$. These complexity studies have assumed no restrictions on the uncertain databases that are inputted to $\mathbf{CERTAINTY}(\mathbf{q})$. In practice, however, it may be known that these input databases are partially consistent, in the sense that they satisfy some dependencies (e.g., functional dependencies).

In this talk, we introduce the problem $\mathbf{CERTAINTY}(\mathbf{q})$ in the presence of a set Σ of dependencies. The problem $\mathbf{CERTAINTY}(\mathbf{q}, \Sigma)$ takes as input an uncertain database $\mathbf{db}$ that satisfies Σ, and asks whether every repair of $\mathbf{db}$ satisfies $\mathbf{q}$. We focus on the complexity of $\mathbf{CERTAINTY}(\mathbf{q}, \Sigma)$ when $\mathbf{q}$ is an acyclic conjunctive query without self-join, and Σ is a set of functional dependencies and join dependencies, the later of a particular form. We provide an algorithm that, given $\mathbf{q}$ and Σ, decides whether $\mathbf{CERTAINTY}(\mathbf{q}, \Sigma)$ is first-order expressible. Moreover, we show how to effectively construct a first-order definition of $\mathbf{CERTAINTY}(\mathbf{q}, \Sigma)$ if it exists.

References

[1] Sergio Greco, Fabian Pijcke, Jef Wijsen, *Consistent Query Answering in Partially Consistent Databases*. Submitted 2013.