Chapter III
A Paraconsistent Relational Data Model

Navin Viswanath
Georgia State University, USA

Rajshekhar Sunderraman
Georgia State University, USA

INTRODUCTION

Typically, relational databases operate under the Closed World Assumption (CWA) of Reiter (Reiter, 1987). The CWA is a meta-rule that says that given a knowledge base KB and a sentence P, if P is not a logical consequence of KB, assume ~P (the negation of P).

Thus, we explicitly represent only positive facts in a knowledge base. A negative fact is implicit if its positive counterpart is not present. Under the CWA we presume that our knowledge about the world is complete i.e. there are no "gaps" in our knowledge of the real world. The Open World Assumption (OWA) is the opposite point of view. Here, we "admit" that our knowledge of the real world is incomplete. Thus we store everything we know about the world – positive and negative. Consider a database which simply contains the information "Tweety is a bird". Assume that we want to ask this database the query "Does Tweety fly?". Under the CWA, since we assume that there are no gaps in our knowledge, every query returns a yes/no answer. In this case we get the answer "no" because there is no information in the database stating that Tweety can fly. However, under the OWA, the answer to the query is "unknown". i.e. the database does not know whether Tweety flies or not. We would obtain a "no" answer to this query under the OWA only if it was explicitly stated in the database that Tweety does not fly.

Current implementations of relational databases adopt the CWA; and for good reason. The negative facts generally turn out to be much larger than the positive facts and it may be unfeasible to store all of it in the database. A typical example is an airline database that records the flights between cities. If there is no entry in the database of a flight between city X and city Y, then it is reasonable to conclude that there is no flight between the cities. Thus for many application domains the Closed World Assumption is appropriate. However, there are a number of domains where the CWA is not appropriate. A prime example is databases that require domain knowledge. For example, consider a biological database that stores pairs of neurons that are connected to each other. If we were to ask this database the query "Is neuron N1 connected to neuron N2?" and this information was not available in the database, is "no" an appropriate answer? What if we do not know yet whether N1 is connected to N2? Then surely the answer "no" is misleading. A more appropriate answer would be "unknown" which we would obtain under the OWA.

Inconsistent information may be present in a database in various forms. The most common form of inconsistency in relational databases is due to the violation of integrity constraints imposed on the database. Under the OWA, inconsistency may also be introduced directly by having both a fact and its negation stored explicitly in the database. Such a situation may arise when data is integrated from multiples sources.

The aim of this article is to introduce a data model that allows the user to store both positive and negative information. When the user poses a query to the database under this model, he obtains both positive and negative answers. The positive answers are those for which the answer to the query is "yes" and the negative answers are those for which the answer to the query is "no". We define the data model and a relational algebra for query processing.

Since the model allows the user to store both positive and negative information explicitly, it is possible for the database to become inconsistent. The data model we introduce allows the user to deal with inconsistent information by keeping the inconsistencies local so that whenever a query is posed, we obtain an inconsistent answer only when the database is itself inconsistent. However, the consistent portion of the database remains unaffected (Grant J. and Subrahmanian V.S., 2000).

BACKGROUND

Two important features of the relational data model (Codd, 1970) for databases is its value-oriented nature and rich set of simple, but powerful algebraic operators. Moreover, a strong theoretical foundation for the model is provided by the classical first order logic. A limitation of the relational model is that it cannot be applied in non-classical situations. Null values in relational databases have been studied extensively (Maier, 1983). Incomplete information has been studied in (Imielinski T. & Lipski Jr. W., 1984, Sarma et al 2006, Antova et al 2007, Benjelloun et al 2008). Disjunctive information has been studied in (Liu K.-C & Sunderraman R., 1990; Liu K.-C & Sunderraman R., 1991, Edward Chan P.F, 1993, Vadaparty K.V and Naqvi S.A 1995). The model that we present here, the paraconsistent relational data model, is a generalization of the relational data model (Bagai R. and Sunderraman R., 1995). This model is capable of manipulating inconsistent as well as incomplete information. Paraconsistent relations are the fundamental data structures underlying the model. A paraconsistent relation essentially consists of two kinds of tuples: ones that definitely belong to the relation and others that definitely do not belong to the relation. These structures are strictly more general than ordinary relations, in that for any ordinary relation there is a corresponding paraconsistent relation with the same information content, but not vice versa.

PARACONSISTENT RELATIONS

In this section, we construct a set-theoretic formulation of paraconsistent relations. Unlike ordinary relations that can model worlds in which every tuple is known to hold an underlying predicate or not to hold it, paraconsistent relations provide a framework for incomplete or inconsistent information about tuples. The data model is *skeptical* in that the tuples that are included as answers to