Recovery of Data Dependencies

Hee Beng Kuan Tan
Nanyang Technological University, Singapore

Yuan Zhao
Nanyang Technological University, Singapore

INTRODUCTION

Today, many companies have to deal with problems in maintaining legacy database applications, which were developed on old database technology. These applications are getting harder and harder to maintain. Reengineering is an important means to address the problems and to upgrade the applications to newer technology (Hainaut, Englebert, Henrard, Hick, J.-M., & Roland, 1995). However, much of the design of legacy databases including data dependencies is buried in the transactions, which update the databases. They are not explicitly stated anywhere else. The recovery of data dependencies designed from transactions is essential to both the reengineering of database applications and frequently encountered maintenance tasks. Without an automated approach, the recovery is difficult and time-consuming. This issue is important in data mining, which entails mining the relationships between data from program source codes. However, until recently, no such approach was proposed in the literature.

Recently, Hee Beng Kuan Tan proposed an approach based on program path patterns identified in transactions for the implementation of the most commonly used methods to enforce each common data dependency. The approach is feasible for automation to infer data dependencies designed from the identification of these patterns through program analysis (Muchnick & Jones, 1981; Wilhelm & Maurer, 1995).

BACKGROUND

Data dependencies play an important role in database design (Maiser, 1986; Piatetsky-Shapiro & Frawley, 1991). Many legacy database applications were developed on old generation database management systems and conventional file systems. As a result, most of the data dependencies in legacy databases are not enforced in the database management systems. As such, they are not explicitly defined in database schema and are enforced in the transactions, which update the databases. Finding out the data dependencies designed manually during the maintenance and reengineering of database applications is very difficult and time-consuming. In software engineering, program analysis has long been developed and proven as a useful aid in many areas. This article reports the research on the use of program analysis for the recovery of common data dependencies, that is, functional dependencies, key constraints, inclusion dependencies, referential constraints, and sum dependencies, designed in a database from the behavior of transactions.

RECOVERY OF DATA DEPENDENCIES FROM PROGRAM SOURCE CODES

Tan (Tan & Zhao, 2004) has presented a novel approach for the inference of functional dependencies, key constraints, inclusion dependencies, referential constraints, and sum dependencies designed in a database from the analysis of the source codes of the transactions, which update the database. The approach is based on the program path patterns for implementing the most commonly used methods for enforcing data dependencies. We believe that the approach should be able to recover majority of data dependencies designed in database applications. A prototype system has been implemented for the proposed approach in UNIX by using Lex and Yacc.

Many of the world’s database applications are built on old generation DBMSs. Due to the nature of system development, many data dependencies are not discovered in the initial system development; they are only discovered during the system maintenance stage. Although keys can be used to implement functional dependencies in old generation DBMSs, due to the effort in restructuring databases during the system maintenance stage, many of these dependencies are not defined explicitly as keys in the databases. They are enforced in transactions. Most of the conventional files and relational databases allow only the definition of one key. As such, most of the candidate keys are enforced in transactions. The feature for implementing inclusion dependencies and referential constraints in a database is only available in some of the latest generations of DBMSs.