A Strategy for Semantic Integrity Enforcement in a Parallel Database Machine

Niall McCarroll, Jon Kerridge

Department of Computer Science
University of Sheffield
Regent Court, 211 Portobello Street
Sheffield S1 4DP, UK
e-mail: N.McCarroll@dcs.shef.ac.uk

Abstract. Semantic integrity constraints represent knowledge about data with which a relational database must be consistent. To enforce semantic integrity we must ensure that transactions which alter the database will preserve database consistency by accompanying each transaction with integrity tests. In this paper we address two problems. Firstly, how can we choose integrity tests that are correct and efficient? Secondly, how do we schedule and control the execution of integrity tests in the context of a parallel database machine? We use a theorem-proving method for deriving integrity tests to prove the integrity of the database which takes into account knowledge about the transaction, the partitioning strategy, and the truth of all constraints in the initial database state. Our method can derive, at schema compilation time, a range of possible sufficient or necessary tests as well as complete tests for transaction safety with respect to a constraint, and can generate separate sub-tests to independently verify changes to a section of the database. When a transaction is to be executed, tests or sub-tests are selected (from the range of alternatives generated at compile time) in order to maximise parallelism, minimise the amount of data accessed in integrity enforcement activity, and allow testing to commence as soon as possible.

1 Introduction

Static integrity constraints are an example of meta-data. Only database states which are consistent with the set of integrity constraints (all constraints are true) can be accurate models of the ‘real world’. For a given consistent initial database state, a transaction is said to be safe if and only if the new state resulting from its execution is consistent. Integrity enforcement is carried out by ensuring that only safe transactions are executed on the database. Transaction safety is evaluated by integrity tests. Integrity tests will nearly always be much cheaper to evaluate than the original constraints because they need only search for inconsistencies arising out of alterations made by a transaction. In this paper we are not concerned with the fate of transactions found to be unsafe, but
concentrate instead on finding ways of automatically generating integrity tests and of reducing the costs and delays incurred by integrity enforcement.

The context of the work is that of a scalable shared nothing parallel relational database machine, constructed using INMOS transputers. The machine provides on-line transaction processing (OLTP) services, for processing transactions and allows management information system (MIS) queries to read OLTP data [8]. We must try to keep within soft real time constraints on OLTP transaction execution time when considering how to implement semantic integrity enforcement.

This paper describes a strategy for enforcing semantic integrity in a parallel database machine. We construct integrity tests to use with a range of transaction templates, at compile time (at the time the database schema is defined). However, we do not simply insert tests into the transaction. A number of alternative (sufficient, necessary or complete) tests can often be employed to check a constraint, and so test selection and scheduling is handled separately from the management of the rest of the transaction at execution time. In particular, we propose the use of integrity sub-tests where applicable. Each sub-test independently verifies the safety of changes to a section of the database without accessing data outside that section. Sub-tests are made possible if an appropriate partitioning strategy is adopted in the database. Sub-tests allow us to exploit parallelism in integrity enforcement, minimise the amount of data that needs to be accessed and allow testing to start as soon as possible.

In section 2, previous work in this area is acknowledged. In section 3, the notation for describing partitioning information, transactions and integrity constraints is explained, and an example is given which illustrates the motivation for this work. In section 4 the methods for deriving integrity tests and sub-tests are given. In section 5 we explain in more detail how the test compiler is integrated into the DBMS architecture and how integrity tests and sub-tests are scheduled. Finally, the conclusion summarises the contribution of this work and suggests some problems which will require further investigation.

2 Previous Work

Early work by Nicolas on deriving integrity tests from constraints [13] showed the importance of the fact that the database is consistent before a transaction is executed, in the search for efficient integrity tests to prove the safety of the transaction.

Subsequent work by Henschen, McCune and Naqvi provides much of the inspiration for and starting point of this work. They present an integrity test derivation method which relies upon a theorem prover [6]. Transactions are defined by ‘transaction axioms’ which relate the original and updated states of relations. The transition axiom, the assertion that the constraint was satisfied in the original database state and the assertion that the constraint is not satisfied