# Transaction Logic Programming

(or, A Logic of Procedural and Declarative Knowledge)

Anthony J. Bonner
Department of Computer Science
University of Toronto
Toronto, Ontario M5S 1A4, Canada
`bonner@db.toronto.edu`

Michael Kifer
Department of Computer Science
SUNY at Stony Brook
Stony Brook, NY 11790, U.S.A.
`kifer@cs.sunysb.edu`

**Abstract**

A new formalism, called *Transaction Logic*, is proposed to account in a clean and declarative fashion for the phenomenon of state changes in logic programs and databases. Transaction Logic extends classical logic with connectives specifically designed for combining simple actions into complex transactions and for providing control over their execution. The proposed logic has a natural model theory and a sound and complete proof theory, but unlike many other logics, it allows users to *program* transactions. This is possible because, like classical logic, Transaction Logic has a "Horn" version which has a procedural as well as a declarative semantics. In addition, the semantics leads naturally to features whose amalgamation in a single logic has proved elusive in the past. These features include both hypothetical *and* committed updates, dynamic constraints on transaction execution, nondeterminism, and bulk updates. Finally, Transaction Logic holds promise as a logical model of hitherto non-logical phenomena, including so-called *procedural knowledge* in AI, *active databases*, and the *behavior* of object-oriented databases, especially methods with side effects. Apart from the applications to Databases and Logic Programming, we also discuss applications to a number of AI problems, such as planning, temporal specifications, and the frame problem.

Technical Report CSRI-323
November 1995

(a substantial revision of Report CSRI-270 of April 1992)

Computer Systems Research Institute
University of Toronto

Related publications, a prototype implementation, and a tutorial are available at the Transaction Logic web page:
`http://www.db.toronto.edu:8020/transaction-logic.html`