Research Proposal Form

Date: 9th June, 1997

1 Contractor or Principal Investigator

- **Organization**

  Research Institute for Symbolic Computation (RISC-Linz)

- **Address of the Organization**

  Johannes Kepler University, A-4040 Linz, Austria

- **Title**

  Professor

- **Name**

  Bruno Buchberger

- **Contact**

  Address:
  Research Institute for Symbolic Computation (RISC-Linz),
  Johannes Kepler University, A-4040 Linz, Austria
  TEL: +43 7236 3231 41
  FAX: +43 7236 3231 30
  e-mail: Bruno.Buchberger@risc.uni-linz.ac.at

2 Title of the Proposed Project

(1) Title of the Proposed Project:

Distributed Constraint Solving for Functional Logic Programming

(2) Research Period (one-year project or two-year project):

Two-years project.

(3) Background of the research (about a half page):

In February 1987, at the occasion of a research visit to ICOT, Prof. Bruno Buchberger proposed using advanced methods in computer algebra (the Gröbner bases method, quantifier elimination, etc.) for dealing with nonlinear constraints in logic programming. The use of algebraic methods in non-linear constraint logic programming has meanwhile been pursued by various groups in the world. At RISC, the emphasis was on non-linear and quantifier constraint solving including the development of distributed and parallel algorithms with a new degree of efficiency in particular in practical examples. At the same time, at the SCORE group of Professor Ida at Tsukuba University, the research line of functional logic programming was developed in detail, in particular with respect to powerful narrowing techniques. Recently, a strong interaction between the two groups evolved. It is the goal of the project proposed here to bring this joint expertise into the frame of the ICOT research and application program.

(4) Purpose of the research (about a half page):

We propose the development of a distributed software system consisting of

- a functional logic language interpreter on one machine,
- a number of constraint solving engines running on other machines.

The interpreter is based on an already existing (sequential) implementation of a functional logic language on the computer system Mathematica extended in two directions:

- the possibility to specify (non-linear) constraints over real numbers;
- the possibility to specify OR-parallelism among different clauses of a predicate.

The OR parallel features of the interpreter allow the decomposition of the solution space into different subspaces denoted by various sets of constraints; the individual sets are solved by different constraint solving engines in parallel and joined together to form the total solution set. This allows to investigate problems with large solution spaces using the computational power available in large computer networks.

(5) Contents of the research (around 1 to 2 pages):

The expressiveness of most constraint logic languages is restricted to linear constraints; however substantial progress has been made to incorporate also non-linear constraints. At the Research Institute for Symbolic Computation (RISC-Linz), Dr. Hong and his research group on constraint solving have developed prototypes of systems aiming at the support of non-linear constraints into constraint logic programming [2]. In particular, they have been building the software library "Sturm" aiming at a constraint solver that takes as its input a "theory" over the real numbers (specified by function and predicate definitions involving quantifiers) and a "question" (a quantified logic formula). The solver returns as an "answer" a formula that is (within the specified theory) equivalent to the question but does not contain any quantifiers. The library has been implemented in C++ and is portable to all kinds of Unix machines; parts of it have been parallelized on top of the parallel programming interface MPI.

Furthermore, the parallel computation group at RISC-Linz has a long standing experience in the integration of parallel programming languages and systems for purposes of symbolic computation. Dr. Schreiner has developed a para-functional programming system where a functional programming language serves as a coordination language for parallel computer algebra algorithms implemented in the library SACLIB on shared memory multiprocessors [3]. Other activities in this grop have integrated the guarded Horn clause language STRAND with the computer algebra system MAPLE running on networks of computers.

In recent years efficient implementations of the execution principles for functional logic languages have been developed. Most of them are based *narrowing*, a unification-based parameter passing mechanism which subsumes rewriting and SLD resolution. The Symbolic Computation Research Group (SCORE) of Prof. Ida at the University of Tsukuba has a long experience in the area of narrowing with a number of important results [1]. Recently they have developed a new deterministic lazy conditional narrowing strategy and implemented it in the computer algebra system Mathematica.

Based on our combined experience, we propose an integration of the constraint solving package developed at RISC-Linz into the lazy narrowing system developed at SCORE such that the functional logic language serves as a "coordination language" distributing sets of constraints to a network of constraint solving engines built on top of the Sturm library. For this purpose, the functional logic language and its Mathematica implementation are extended by the mechanism to specify non-linear constraints over the real numbers and to formulate OR parallelism among different clauses of a predicate such that parts of the solution space can be investigated independently and simultaneously of each other.

The actual distribution is performed by an external "scheduling program" implemented in C++ that is linked to the Mathematica kernel via the "MathLink" facility and provides the interface between the Mathematica language interpreter and the various constraint solving engines. The scheduler receives from the Mathematica kernel OR-parallel branches of the computation represented as Mathematica expressions containing sets of constraints to be solved. The scheduler forwards each constraint set to some constraint solving engine available in the network. Each engine implements a sequential constraint solver based on the Sturm package; it communicates with the scheduler retrieving a set of constraints and delivering the corresponding solution. The scheduler holds a set of those OR-parallel branches whose constraints have been solved and returns them to the language interpreter (which in turn may generate new branches). The Mathematica language interpreter itself therefore still executes in a sequential fashion cooperating with the scheduler which is in charge of the distributed execution.

The whole approach allows to investigate problems with large solution spaces (as arise in numerous scientific and engineering problems) using the computational power available in (local- or wide-area) computer networks.

Selected Publications

[1] Hoon Hong: RISC-CLP(Real): Constraint Logic Programming over Real Numbers. In: Constraint Logic Programming: Selected Research, F. Benhamou and A. Colmerauer (eds.), MIT Press, 1993.

[2] Tetsuo Ida et al: Lazy narrowing: strong completeness and eager variable elimination, Theoretical Computer Science, 167(1996), 95-130

(Conference Version: Proceedings of the 20th Colloquium On Trees In Algebra and Programming (CAAP) Lecture Notes in Computer Science 915, 394-409,1995)

[3] Wolfgang Schreiner: A Para-Functional Programming Interface for a Parallel Computer Algebra Package. Journal of Symbolic Computation, Special Issue on Parallel Symbolic Computation, Hoon Hong (ed.), volume 21, pp. 523-614, Academic Press, 1996.