RESEARCH INTERESTS AND PROJECTS

Constrained polymorphism (overloading / system CT)

system CT = core-ML + overloading

- studied: (constraint-set) satisfiability, principal typing, polymorphic recursion & semi-unification

Project plans:

- automatic derivation of generic functions
- compilation / separate compilation / efficiency of generated code

Monadic Parser Generation (Mímico)

- **Mímico outputs monadic parsers based on syntax description plus semantic rules that specify the result of parsing (Haskell code).**
- **Mímico outputs top-down recursive descent parsers but allows left-recursive context free grammars as input.**
- **Mímico: easy/concise way of specifying the syntax and semantics of grammars/languages, with generation of readable output (Haskell programs).**

Project plans:

- **develop interesting examples / language parsers**
- **use of monadic code / semantic rules to control parsing**
- **efficiency**

Semi-unification (RSUP)

- Rewriting system RSUP (proved equivalent to Henglein rewriting system HRS) + relation SUP instances ↔ intercell TMs:
  - RSUP terminates indicating that Γ has a solution $\leftrightarrow$ $I(\Gamma)$ is bounded (analogous to Kfoury et al's result)
  - if RSUP terminates indicating that Γ has no solution, then $I(\Gamma)$ has an eventually periodic configuration
  - if RSUP does not terminate with input Γ then $I(\Gamma)$ has a wandering (neither halting nor eventually periodic) configuration.

This gives examples for which RSUP and HRS do not terminate.

Semi-unification (RSUP)

In http://www.dcc.ufmg.br/˜camarao/SUP:

- tm2itm: TM → ITM (in Haskell)
- itm2sup: symmetric ITM → SUP instance (in Haskell)
- rsup: RSUP implementation (in Haskell)

- K2.tm: TM with no periodic and no halting configuration
- K2.itm: equivalent ITM
- K2.in: instance of SUP from symmetric closure of K2.itm

Other topics

– **Module systems and record types**

Study and develop simple (but flexible and expressive) module systems based on (extensions of) record types.

– **Subtyping / extensible types / overloading of constructors**

Group

– Carlos Camarão

– Lucília Figueiredo (system CT / generic programming)

– Cristiano Vasconcellos (system CT / compilation/ efficiency)

– João Rafael (type and module systems/record types)

– Luigi Laporte (semi-unification)

Articles on system CT

- Type Inference for Overloading without Restrictions, Declarations or Annotations, Carlos Camarão & Lucília Figueiredo, FLOPS'99, LNCS 1722, 37–52, 1999.
- Constraint-set satisfiability for Overloading, Carlos Camarão, Lucília Figueiredo & Cristiano Vasconcellos, Proc. of PPDP'04, 67-77, 2004.

Articles on extensible types and overloading of constructors

- A Type with a View, SBLP'99, 33–44, Porto Alegre, 1999.
- A View on Modular and Extensible Types, Carlos Camarão & Lucília Figueiredo, Revista Colombiana de Computacion, 3(1), 21-40, 2002.

Articles on principal typing and polymorphic recursion

- Tipos em Linguagens de Programação, SBLP'99, Porto Alegre, 1999. Tutorial.
- ML Has Principal Typings, Carlos Camarão & Lucília Figueiredo, Proc. of SBLP'2000, Recife, 231–244, 2000.
- Principal Typing and Mutual Recursion, Proc. of WFLP'2001, 157–170, 2001.

Articles on monadic parser generation

- A Monadic Combinator Compiler Compiler, Carlos Camarão e Lucília Figueiredo, Proc. of SBLP'2001, Curitiba, 64–79, 2001.
- Mímico: A Monadic Combinator Parser Generator, Carlos Camarão, Lucília Figueiredo & Hermann Rodrigues, Journal of the Brazilian Computer Society, 9(1), 2003.