University Paris Sud (Orsay) site and ENS Lyon subsite

1/08/2000-31/07/2001

1 Participants

The following people participated to the TYPES project.

1.1 Paris Sud University, Orsay

Laboratoire de Recherche en Informatique, UMR 8623, CNRS-Université Paris Sud

Permanent researchers

- Judicël Courant
- Jean-Christophe Filliâtre
- Jean-Pierre Jouannaud
- Christine Paulin (Site leader)

PhD students

- Frédéric Blanqui
- Pierre Courtieu
- Pierre Letouzey

1.2 ENS Lyon

Laboratoire de l'Informatique du Parallélisme, UMR 5668, CNRS-INRIA-ENS Lyon

Permanent researchers

- Philippe Audebaud
- Jean Duprat (Site leader)
- Daniel Hirschkoff

2 Results

2.1 The Coq proof assistant

The Coq Proof assistant is conjointly developed by the Paris Sud University and the INRIA Rocquencourt site. The first release of the new version of the Proof Assistant Coq V7 happened in December 2000 [11]. It is built on a new functional type-checker for proof-terms. The new architecture is described in [7] which has been submitted to the proceedings of the TYPES meeting.

P. Courtieu adapted the Proof General Interface developed by D. Aspinall at the Edinburgh site to this new version of Coq.

2.2 Program extraction

Pierre Letouzey designed a new version of extraction which transforms Coq proofs into ML programs. This version, unlike the previous one, is able to analyse proofs using the full power of inductive definitions and universes. The implementation of this new method in Coq V7 was done by J.-C. Filliâtre and P. Letouzey.

2.3 Developments

The set of user's contributions (many of them written by members of the TYPES working group) has been adapted from older versions to Coq V7.

The work on the proof of Stålmark's algorithm done by Pierre Letouzey from LRI together with Laurent Thery from INRIA Sophia-Antipolis was presented at the TPHOLs'00 international conference [9].

Jean Duprat developed a toolkit to manipulate graphs. His work [6] was presented at the TYPES meeting in Durham and has been submitted to the proceedings.

Daniel Hirschkoff from ENS Lyon together with P. Dargenton and P. Lescanne and E. Pommateau developed in Coq a specification of a Micro-Payment Protocol [5].

J.-C. Filliâtre validated his `Correctness` tactic which proves imperative programs in Coq on several examples : the original Hoare's `Find` program [8], the Brezenham's line drawing algorithm, and an algorithm to compute the edition distance. These proofs are available as Coq's contributions.

E. Freund and C. Paulin developed a Coq theory for manipulating a special class of timed automata, designed in a commun project with France Telecom R & D. C. Paulin did a proof of a generalised version of a conformance algorithm used in the ATM network. These proofs are available as Coq's contributions.

P. Courtieu developed in Coq a proof of a self-stabilizing algorithm using a general method based on word rewriting.

2.4 Integrating higher-order rewriting to type theory

Frédéric Blanqui [1], Jean-Pierre Jouannaud [2] and Daria Walukiewicz-Chrząszcz [12] (also from Warsaw site) worked on different methods to insure confluence and strong normalisation for systems integrating strongly typed lambda-calculus like the Calculus of Inductive Constructions and higher-order rewrite rules.

F. Blanqui designed a prototype (CAC) which is an extension of the Calculus of Constructions where functions and predicates can be defined by rewrite rules. The rewrite steps are performed with the system CiME also developed at Paris Sud University.

2.5 Modules

Modules is an important feature to be added to proof assistant in order to develop large examples and reusable libraries. Judicaël Courant worked on adding explicit polymorphic universes to his modules calculus [3] and designed a proptotype of this system. Jacek Chrząszcz (also from Warsaw site) is currently integrating a module system to the `Coq` implementation that could be extended for the addition of rewrite rules.

2.6 Quotient types

P. Courtieu presented at the TYPES meeting his work [4] on integrating quotient sets to type theory in the special case where a canonical form exists for equivalence classes.

2.7 Higher-Order Abstract Syntax

Together with Christine Röckl and Stefan Berghofer from the Munich site, Daniel Hirschkoff worked on a formalisation of the π-calculus in Isabelle/HOL which uses Higher-order Abstract Syntax [10]. This work introduces well-formedness predicates in order to eliminate exotic terms and provide an induction principle.

2.8 Proofs of randomized algorithms

Type theory provides a framework for reasoning on abstract objects in mathematics like the ones used for analysing randomized algorithms. Ph. Audebaud together with R. Lassaigne (from Paris 7 U.) and C. Paulin are studying this area. Ph. Audebaud proposed an axiomatic system to reason on an imperative language extended with a `random` construction. C. Paulin is studying a functional interpretation of randomized algorithms using a monadic transformation.

3 Collaborations

Freek Wiedjik from the Nijmegen site visited Orsay for one week at the end of January 2001. Together with Jean-Christophe Filliâtre from Orsay, they adapted the proof of the Fundamental Theorem of Algebra to the new Coq version V7. This example has been added to the data-base of Coq users contributions.

Orsay and Lyon sites have strong interactions with the INRIA sites in Rocquencourt and Sophia-Antipolis. They participate together to common seminar and one-day workshops.

Orsay collaborates with the industrial site France Telecom R& D on proof environments for the development of critical algorithms in Telecommunication applications.

Lyon collaborates with the Technische Universität München site in Germany in the domain of formalising languages semantics.

Jacek Chrząszcz and Daria Walukiewicz from the University of Warsaw (Poland site) are studying for a PhD both at Paris Sud University and at Warsaw University. They are visiting Paris Sud University from February to August 2001.

Références

[1] F. Blanqui. Definitions by rewriting in the Calculus of Constructions. In *LICS'01*, 2001.

[2] F. Blanqui, J.-P. Jouannaud, and M. Okada. Inductive-data-type Systems. *Theoretical Computer Science*, 277, 2001.

[3] J. Courant. Mc2: A module calculus for pure type systems. *J. of Functional Programming*, 2001. Submitted.

[4] P. Courtieu. Normalized types. In *Proceedings of CSL2001*, LNCS, 2001. to appear.

[5] P. Dargenton, D. Hirschkoff, P. Lescanne, and E. Pommateau. Formally validated specification of a micro-payment protocol. Rapport de Recherche 2001-31, LIP-ENS Lyon, 2001.

[6] J. Duprat. A Coq toolkit for graph theory. Rapport de recherche 2001-15, LIP ENS Lyon, 2001. submitted to TYPES'2000 proceedings.

[7] J.-C. Filliâtre. Design of a proof assistant: Coq version 7. Submitted to TYPES'2000 proceedings, 2000.

[8] J.-C. Filliâtre. Formal proof of a program: Find. *Science of Computer Programming*, 2001. To appear.

[9] P. Letouzey and L. Théry. Formalizing Stålmarck's algorithm in Coq. In J. Harrison and M. Aagaard, editors, *Theorem Proving in Higher Order Logics (TPHOLs'2000)*, volume 1869 of *LNCS*, 2000.

[10] C. Roeckl, D. Hirschkoff, and S. Berghofer. Higher-order abstract syntax with induction in Isabelle/HOL: Formalizing the Pi-Calculus and mechanizing the theory of contexts. In *Proceedings FOSSACS'01*.

[11] The Coq Development Team. *The Coq Proof Assistant Reference Manual – Version V7.0*, April 2001.

[12] D. Walukiewicz-Chrząszcz. Termination of rewriting in the calculus of constructions. *J. of Functional Programming*, 2001. Submitted.