Preface

The *Coq* system is a computer tool for verifying theorem proofs. These theorems may concern usual mathematics, proof theory, or program verification.

Our main objective is to give a practical understanding of the *Coq* system and its underlying theory, the Calculus of Inductive Constructions. For this reason, our book contains many examples, all of which can be replayed by the reader on a computer. For pedagogical reasons, some examples also exhibit erroneous or clumsy uses and guidelines to avoid problems. We have often tried to decompose the dialogues so that the user can reproduce them, either with pen and paper or directly with the *Coq* system. Sometimes, we have also included synthetic expressions that may look impressive at first sight, but these terms have also been obtained with the help of the *Coq* proof assistant. The reader should decompose these expressions in practical experiments, modify them, understand their structure, and get a practical feeling for them.

Our book has an associated site,[1] where the reader can download and replay all the examples of proofs and—in cases of emergency—the solutions of the 200 exercises of the book. Our book and our site both use *Coq V8*,[2] released in the beginning of 2004.

The confidence the user can have in theorems proved with the *Coq* system relies on the properties of the Calculus of Inductive Constructions, a formalism that combines several of the recent advances in logic from the point of view of λ-calculus and typing. The main properties of this calculus are presented herein, since we believe that some knowledge of both theory and practice is the best way to use *Coq*'s full expressive power.

The *Coq* language is extremely powerful and expressive, both for reasoning and for programming. There are several levels of competence, from the ability to construct simple terms and perform simple proofs to building whole theories and studying complex algorithms. We annotate chapters and sections with

[1] `www.labri.fr/Perso/~casteran/CoqArt/`

[2] `coq.inria.fr`

information about the level of competence required to understand them, as follows:

(no annotation) accessible at first reading,
* readable by intermediate-level practitioners,
** accessible to expert users able to master complex reasoning and certify programs,
*** reserved for the specialist who is interested in exploring all the possibilities of *Coq* formalization.

A similar annotation pattern is used for the exercises, from the elementary ones (solved in a few minutes) to the extremely hard ones (which may require several days of thought). Many of these exercises are simplified versions of problems we encountered in our research work.

Acknowledgements

Many people have enthusiastically helped us in the compilation of this book. Special thanks go to Laurence Rideau for her cheerful support and her attentive reading, from the very first drafts to the last version. Gérard Huet and Janet Bertot also invested a lot of their time and efforts in helping us improve both the technical accuracy and the writing style. We are also especially grateful to Gérard Huet and Christine Paulin-Mohring for the foreword they contributed.

The *Coq* development team at large also deserves our gratitude for having produced such a powerful tool. In particular, Christine Paulin-Mohring, Jean-Christophe Filliatre, Eduardo Gimenez, Jacek Chrząszcz, and Pierre Letouzey gave us invaluable insights on the internal consistency of the inductive types, imperative program representation, co-inductive types, modules, and extraction, and sometimes they also contributed a few pages and a few examples. Hugo Herbelin and Bruno Barras were key collaborators in helping us make sure that all the experiments described in this book can be reproduced by the reader.

Our knowledge of the domain also grew through the experiments we performed with the students we were lucky to teach or to work with. In particular, some of the ideas described in this book were only understood after teaching at the École Normale Supérieure de Lyon and the University Bordeaux I and after studying the questions raised and often solved in collaborations with Davy Rouillard, Antonia Balaa, Nicolas Magaud, Kuntal Das Barman, and Guillaume Dufay.

Many students and researchers gave some of their time to read early drafts of this book, use them as teaching or learning support, and suggest improvements or alternative solutions. We wish to thank all those who sent us precious feedback: Hugo Herbelin, Jean-François Monin, Jean Duprat, Philippe Narbel, Laurent Théry, Gilles Kahn, David Pichardie, Jan Cederquist, Frédérique

Guilhot, James McKinna, Iris Loeb, Milad Niqui, Julien Narboux, Solange Coupet-Grimal, Sébastien Hinderer, Areski Nait-Abdallah, and Simão Melo de Sousa.

Our respective research environments played a key role in seeing this project through thanks to their support. We are especially grateful to the Lemme and Signes teams at INRIA and the University Bordeaux I for their support and to the European working group Types for the opportunities they gave us to meet with innovative young researchers like Ana Bove, Venanzio Capretta, or Conor McBride who also inspired some of the examples detailed in this book.

We are very grateful to the people of Springer-Verlag who made this book possible, especially Ingeborg Mayer, Alfred Hofman, Ronan Nugent, Nicolas Puech, Petra Treiber and Frank Holzwarth. Their encouragement and advice on content, presentation, editing, and typesetting have been essential. We also thank Julia Merz of KünkelLopka GmbH for the cover design with the beautiful piece of art.

Sophia Antipolis
Talence
March 2004

Yves Bertot
Pierre Castéran

Springer

http://www.springer.com/978-3-540-20854-9

Interactive Theorem Proving and Program Development
Coq'Art: The Calculus of Inductive Constructions
Bertot, Y.; Castéran, P.
2004, XXV, 472 p. 1 illus., Hardcover
ISBN: 978-3-540-20854-9