Editorial

The 20th century gave birth to a computer technology that has dominated our lives. Such technology may be expensive to build and/or human lives may depend on it. We have overwhelming evidence from just under a century's work that the right logic and the right notion of symbolic manipulation (rewriting) can guarantee the safety and correctness of this technology saving money, and human lives and efforts. For this reason, we have seen and will continue to see new different logics and rewriting systems, extensions of old systems and the study of their theory and applications will be as thrive as it was in the last century. This is not surprising because the twentieth century was indeed a *century of complexity* and this complexity will be carried to this century. This complexity of information, the increasing interdependency of systems, the faster and more automatic travel of information, and the disastrous consequences of failure, lead to the need for establishing **Correctness**. Moreover, modern technological systems are just too complicated for humans to reason about unaided, so **automation** is needed. Furthermore, because modern systems have so many possible states, testing is often impractical. It seems that **proofs** are needed to cover infinitely many situations. The last century is evidence that formalisms needed to *aid in design* and to *ensure safety* must accommodate some *rewriting* and *automatic* search for and checking of proofs. These ideas were present long before the 20th century. In fact, Leibniz (1646–1717) conceived of *automated deduction*, when he wanted to find:

- a language L in which arbitrary concepts could be formulated, and
- a machine to determine the correctness of statements in L.

Such a machine can not work for every statement according to Gödel and Turing. Nevertheless, the need for automation has been overwhelming and its exploration in both the safe grounds and the dangerous borderlines continues to be challenging. The relevance of rewriting and automation is witnessed by the number of international conferences and events devoted to the subject. We cannot mention all these events and refer to the usual references. This volume however, is a selection of various papers that were presented at a collection of events on rewriting, automation and theorem proving that took place in year 2000 and were funded by different sources including: the European Union's IHP High Level Scientific Conferences support, the European Educational Forum, the UK Engineering and Physical Research Council EPSRC, the Royal Society and the Dutch research council NWO. The support of all these sources is greatly appreciated. These events are as follows:

- Winter Workshop in Logics, Types and Rewriting '00 on 2 February 2000.
  See `http://www.cee.hw.ac.uk/~fairouz/inaugural-workshop2000/`
- The EEF Foundations School in Deduction and Theorem Proving'00 on 6-16 April 2000.
  See `http://www.cee.hw.ac.uk/~fairouz/ukiischool2000/ukiischool.html`
- Festival Workshop in Foundations and Computing, FC'00 on 17-18 July 2000.
  See `http://www.cee.hw.ac.uk/~fairouz/festival/workshop1/`

Due to the succes of the above events, it was decided that a special issue should be published on the above themes. Some of the lecturers and speakers agreed to write

their material as articles for this volume. Of the submitted articles, five were selected for this volume.

The article of Cirstea and Kirchner is in two parts and is concerned with a new calculus called the ρ-calculus. The characteristic feature of the ρ-calculus is that it has an operator $\rightarrow$ used to build abstractions as in the λ-calculus as for instance $x \rightarrow x$ for the identity. Abstractions can also contain patterns as in first-order rewriting. For instance the rewrite rule $a \rightarrow b$ is in the ρ-calculus represented by the abstraction $a \rightarrow b$. The application of an abstraction to an argument is as in the λ-calculus, but now denoted by for instance $[x \rightarrow x](y)$ for the identity applied to a variable y. If the pattern of the left-hand side of the abstraction is not present in its argument, the application is rewritten to $\emptyset$, representing failure. For instance $[a \rightarrow b](b) \longrightarrow \emptyset$. If the pattern of the left-hand side is present in the argument, then the application is rewritten to the set consisting of the corresponding right-hand side. For instance, we have $[x \rightarrow x](y) \longrightarrow \{y\}$ and $[a \rightarrow b](a) \longrightarrow \{b\}$. Also sets consisting of more elements are used to represent non-determinism.

In the first part, the calculus is introduced and motivated and its syntax and evaluation rules for any theory are presented. Then, the encoding of the λ-calculus is presented and a discussion of confluence is given. In the second part, conditional rewriting is encoded and the calculus is extended with a *first* operator whose purpose is to detect rule application failure. This extension enables the encoding of strategy based rewriting processes and is used to give an operational semantics to ELAN which is an environment for specifying and prototyping deduction systems in a language based on labelled conditional rules and strategies to control rule application.

The article of Jan van Eijck and Juan Heguiabehere and Breanndán Ó Nualláin presents a tableau system for dynamic first-order logic (DFOL for short), a formalism originally introduced by Groenendijk and Stokhof to account for certain aspects of natural language semantics and anaphora. The language presented in this paper contains explicit substitutions and the choice operator $\cup$. The language is further extended with the finite iteration *-operator (DFOL*). Soundness and completeness of the tableau method for DFOL and then DFOL* is proved. The authors illustrate through significant examples the usefulness of DFOL and DFOL* and of the related tableau method to represent program execution and to derive pre/post conditions in the style of Hoare logic. They also show the potential benefit of their tableau method as a tool in computational semantics of natural language.

The article of Jacques Fleuriot, reports the formalisation in the theorem prover Isabelle of a theory of non-standard geometry based on infinitely small and large reals. The theory is based on so-called hyperreal vectors which are sequences of real vectors with two such sequences being equal if they coincide on an element in an ultrafilter (an abstract way to express that they are equal almost everywhere). The paper uses the full power of the Isabelle-HOL formalism in order to get a smooth development. It can be seen as a reference paper on the basis of infinitesimal geometry. As mentioned by the author, extending usual operations to infinitely small or large objects is very subtle and can easily be done the wrong way. The fact that the theory is completely developed in Isabelle-HOL is consequently really useful.

The article of Paulson presents a short and natural mechanisation of the proof of the mutilated chessboard problem in Isabelle. This exercise is used to demonstrate some important principles in the manipulation of systems of this kind. Particular emphasis

is put on the use of inductive definitions. These are of interest both because they allow the user to give intuitive definitions of e.g. what is a domino and what is a tiling, and because they inherently capture the essence of the concepts being formalised (therefore, for example, no erroneous tiling can be generated in the development of a proof). Moreover, Isabelle's tactics technology, together with the conciseness offered by inductive definitions, makes it possible to derive a formalisation that is much shorter than in similar works based on other provers.

Fairouz Kamareddine