# Introduction to ALF — a Background

Catarina Coquand

Chalmers Technical University
Inst. for Computer Science
Göteborg, Sweden
E-mail: `catarina@cs.chalmers.se`

We will give an introduction to type theory, focussing on its use as a logical framework for proofs and programs. We will also discuss the use of inductive definition as specifications and proofs by pattern matching. Many examples will be presented.

The basic idea behind using type theory for developing proofs and programs is the Curry-Howard isomorphism between propositions and types. Take for example the proposition ($A$ implies $B$ implies $A$). Looking at this proposition as a type we see that this is the type of the $K$ combinator. In type theory we then say that the $K$ combinator proofs the proposition ($A$ implies $B$ implies $A$).

The language that is used is a functional language with dependent types. Hence programs and proofs can be described in the same language. We use inductive definitions for our specifications. Introduction rules in logic corresponds to defining a type by its constructors. Proofs are functions defined by pattern matching over these types. We shall also mention how this can be extended to proofs about infinite objects.