Algorithmic Term Rewriting Systems

Isihara, A.

2010

Summary

Algorithmic term rewriting systems

In theoretical computer science, in particular the area of algebraic specification of abstract data types, sorted specifications based on the principle of induction are well-known. Here one is concerned with finite data such as natural numbers, booleans, finite trees, finite lists, and so on. More recently, a 'dual' specification method has become prominent: that of coalgebraic specifications of infinite data, also called codata. Typical codata are lazy natural numbers, infinite trees, infinite lists or streams. Whereas in the world of finite data, induction is the salient principle of definition and proof, it is replaced by coinduction in the realm of codata.

In this thesis we develop a general framework that extends both the inductive and coinductive specifications: the framework of *algorithmic term rewriting systems*. The class of algorithmic term rewriting systems provides a scheme for function specifications employing both inductive and coinductive constructions. When a function specification is given, we are concerned whether the specification is well-defined or not. This leads to the primary desired property of algorithmic term rewriting systems: all the specifications that can be given as expressions in the system should be well-defined. We shall call this property *'productivity'*. This description of productivity is still very informal. A contribution of this thesis is giving it a technically precise interpretation. The resulting notion of productivity of an algorithmic system is fundamental, on a par with the notions of termination and confluence for (finitary) term rewriting systems.

Productivity turns out to be the consequence of three secondary properties. First, *infinitary normalization* (WN) guarantees that an expression has a possibly infinite normal form. Secondly, *domain normalization* (DN) guarantees that the normal form is within the intended domain of results. Thirdly, *constructor normalization* (CN) guarantees that the normal form is built solely from constructors without defined function symbol. We give conditions for each of the three properties WN, DN, CN, and in some instances even characterizations. Together, they form conditions that ensure productivity of the algorithmic term rewriting system.

As an application of the theory developed here, we consider in the final chapter a fairly complicated infinite data type known as tree ordinals. These are important in the theory of ordinal notations in proof theory, a branch of mathematical logic. They also embody a study of the expressivity of the first order term rewriting framework, and we show that this expressivity is large: we can express ordinals far larger than the Feferman–Schütte ordinal known as Γ_0.