New Challenges for Computational Models
(Position Statement)

Yoshihito Toyama

Research Institute of Electrical Communication
Tohoku University
Katahira 2-1-1, Aoba-ku, Sendai 980-8577, Japan
toyama@nue.riec.tohoku.ac.jp

Theoretical computer science has offered various computation models like automata, Turing machine, lambda calculus, and their importance to theoretical work and to practice is widely recognized. The progress of the classical topics of formal language theory, computational complexity, and mathematical semantics of programming languages is due to these computation models. In this talk we propose two challenges that have tight connection to mathematical theory of programs and probably need new computation models to attack them.

Computation Model for Algebraic Programming

To set up algebraic equations is very common approach for many elementary mathematical problems, and we can easily find the solutions satisfying these equations by very simple algorismic method without deep knowledge of mathematics. The final goal of algebraic programming is to offer an analogous framework for deriving programs from algebraic specification for problems to that in elementary mathematics. Lambda calculus gives a useful mathematical basis in the study of lambda equations, but its theory is too mathematical and complicated for the practical use. Recent development of program transformation techniques by fusion and pairing shows an interesting direction of automatic algebraic derivation of functional programs, though their application is still too limited. The next step of algebraic programming seems to be to try to discover new computation models that unify the both directions.

Computation Model for Flexible Semantics

The most common mathematical presentations of operational semantics of programming languages are reduction systems. A reduction strategy chooses one out of many possible reductions, and can give an efficient deterministic interpreter through narrowing a given idealized nondeterministic semantics. The notion of reduction approximations is the inverse of reduction strategies, which broadens a given semantics to more flexible one. A decidable approximation, like the strongly sequential approximation, of term rewriting systems helps us to find a class of term rewriting systems having a decidable needed reduction strategy. Thus, not only narrowing but also broadening semantics is important to design an efficient deterministic interpreter for a given operational semantics.

J. van Leeuwen et al. (Eds.): IFIP TCS 2000, LNCS 1872, pp. 612–613, 2000.
© Springer-Verlag Berlin Heidelberg 2000

This observation easily leads us to an attractive idea of programming systems with flexible semantics, which can narrow or broaden the semantics of a programming language if necessary. Computation models for flexible semantics will provide new useful formal approaches to software science.