GUEST EDITORIAL

Seventh International Workshop on Foundations of Object-Oriented Languages

The Seventh International Workshop on Foundations of Object-Oriented Languages (FOOL 7) was held on January 22, 2000, and it was colocated with the ACM Symposium on Principles of Programming Languages in Boston, Massachusetts, USA.

Martín Abadi chaired the program committee, which received 19 submissions and accepted 6 of them. After the workshop, extended versions of 4 articles were solicited for this special issue of Information and Computation. All 4 were accepted after a refereeing and selection process matching the standards of the journal.

The issue opens with “Type Inference for Variant Object Types,” by Michele Bugliesi and Santiago Pericás-Geertsen. Combining subtyping, recursive types, and method update in the same type system was a challenge only partially solved until Bugliesi and Pericás-Geertsen adapted the notion of “splitting” types to Abadi and Cardelli’s first-order object calculus.

The system of split types presented assigns two types to each method, a type used for method update and a type used for method selection. This simple idea brings an elegant solution to the problem. Furthermore, the resulting system extends the typing power of existing solutions with variant annotations by accepting more sound objects as typable. This additional expressive power does not affect the complexity of the type inference problem, as the paper shows by presenting a sound and complete $O(n^3)$ inference algorithm.

Kathleen Fisher and John Reppy, in “Inheritance-Based Subtyping,” study contrasting relations between inheritance and subtyping. In mainstream programming languages such as Java and C++, inheritance defines a subtyping hierarchy, giving rise to the phrase inheritance-based subtyping. On the other hand, most foundational studies of class-based subtyping feature a notion of inheritance that complies with but does not define the subtyping relation. This subtyping notion is known as structural subtyping.

Fisher and Reppy bridge these two approaches, showing how to implement friend functions, one of the strongest features of inheritance-based subtyping, in their programming language Moby based on structural subtyping.

“On Inner Classes,” by Atsushi Igarashi and Benjamin Pierce, is a formal study of inner classes, that is, classes defined inside other classes. The goal is to capture the essential features of inner classes as they are found in Java and to understand the subtleties that derive from their interactions with inheritance. This is achieved by adding inner classes to Featherweight Java, a fragment of Java providing class definitions, object instantiation, field access, and method invocation, by defining two different semantics and showing that they coincide.

In “Type-Safe Covariant Specialization with Generalized Matching,” Ran Rinat studies the matching relation originally introduced to model type-safe subclassing in the presence of binary methods. The paper extends this relation so that it accounts for a generalized use of covariant subtyping: besides self type covariance, which is captured by “standard” matching, the new relation allows sound covariant specialization of instance variables and method parameters.

We are grateful to the 12 colleagues who prepared the anonymous reviews for this issue. They all did an excellent job, contributing greatly to the quality of this special issue.

Giuseppe Castagna
Adriana Compagnoni
Guest Editors