REPORT ON POPL'2025

52nd ACM SIGPLAN Symposium on Principles of Programming Languages
Denver, Colorado, USA, Jan 19 — 25, 2025

Vikraman Choudhury, Università di Bologna, Bologna, Italy

POPL is one of the premier conferences in the field of programming languages. The 52nd edition of the conference, **POPL'2025**, along with its co-located events, was held in Denver, CO, at the Curtis Hotel Denver, from Jan 19 to 25, 2025. According to the General Chair's report, the conference was attended by 560+ participants. As usual, this was a huge event, with conference talks, posters, workshops, tutorials, from different areas of programming languages, as well as social events and activities.

POPL attracts researchers from different areas of computer science, connected by the common theme of programming languages. The range of topics covered in the conference was quite broad, from theoretical foundations of programming languages, to practical aspects of programming languages and software engineering. I would like to highlight a couple of talks that I attended and found particularly interesting, that would be most relevant for the theoretical computer science community.

Theory at Peek-A-Boo, Fri 24 Jan, Interaction Equivalence: Adrienne Lancelot presented a fascinating talk about the equivalence of (untyped) lambda terms. The standard notion of equivalence for lambda calculus is contextual equivalence, which means two terms are equivalent if they are observationally indistinguishable in all contexts. This is an *e*quational theory, in the sense that it is a congruence relation (equivalence relation closed under contexts), and contains β-equality. However, this is too coarse of an equivalence, since it doesn't distinguish terms by the number of steps of reduction (or cost), for example, β-equivalent terms can have different numbers of reduction steps. If equivalence is considered upto equal number of reduction steps, is there an appropriate equational theory? The paper solves this puzzle by first, decomposing the lambda calculus into a *c*heckers lambda calculus, which colors term formers black or white, and then, defining a notion of equivalence called *i*nteraction equivalence, which is an equational theory that captures cost equivalence. The key idea is that reductions can be seen as, either interactions between programs and contexts, or as internal steps within programs. The development uses various technical tools, such as Böhm separability, and intersection types, to achieve this result.

Semantic models at Peek-A-Boo, Wed 22 Jan: Abstract Operational Methods for Call-by-Push-Value: The program of Mathematical Operational Semantics provides a rigorous foundation for giving operational semantics and contextual equivalence to programming languages, using a categorical framework of bialgebras. Recent work by the speaker and collaborators extends this framework to higher-order languages, that is, languages with binding, which is a significant improvement. The work presented in the talk extends this framework to call-by-push-value (CBPV) languages, allowing for languages with effects, and obtaining an abstract framework for proving theorems about applicative bisimulation and step-indexed logical relations.

While there were many other interesting talks and lots of exciting applications of programming languages to real-world problems, I noticed that most researchers base their work on the foundational ideas and techniques in the field, perhaps best summarized by a quote from Loris D'Antoni's keynote, "strong foundations transcend trends".