The EATCS Award 2014
Laudatio for Gordon Plotkin

We present the 2014 EATCS award to Gordon Plotkin for his lifetime contribution of a research corpus of exceptional depth and influence across a broad range of areas within theoretical computer science. Alongside his scientific contributions of the highest calibre, he helped to lay the foundations of the theoretical computer science community, shaping the careers of generations of researchers.

Scientific contributions

Plotkin is renowned for his groundbreaking contributions to programming language semantics, which have helped to shape the landscape of theoretical computer science, and which have impacted upon the design of programming languages and their verification technologies. The influence of his pioneering work on logical frameworks pervades modern proof technologies. In addition, he has made outstanding contributions in machine learning, automated theorem proving, and computer-assisted reasoning. He is still active in research at the topmost level, with his current activities placing him at the forefront of fields as diverse as programming semantics, applied logic, and systems biology.

- **Structural operational semantics.** A central concern throughout Plotkin's career has been to understand how best to furnish programming languages with precise mathematical descriptions of program behaviour. In his seminal works, Plotkin transformed the discipline by providing an elegant and very powerful approach based on simple mathematical principles. His 1981 monograph, *A Structural Approach to Operational Semantics*, revolutionised the field of programming-language research. Structural operational semantics provides a systematic method for defining operational behaviour that, by specifying the behaviour of programming constructs individually, is both mathematically elegant, and also capable of accommodating the full and ever-broadening spectrum of programming language styles and features. Nowadays, structural operational semantics is widely adopted as the standard technique for defining semantics in the programming language research community. It has been used to design and define full-scale programming languages (e.g., Standard ML, Scheme), was crucial to the development of industrial domain-specific languages (e.g., Esterel), and was used to establish key properties of them (e.g., type safety). It combines readily with theorem proving technologies to provide program verification methods. It adapts smoothly to other systems description languages, such as calculi modelling system performance or interactive behaviour. Furthermore, structural operational semantics has gained recent traction as a powerful method in the specification of biological models.

- **Denotational semantics.**

  Denotational semantics is a major approach to programming language semantics which models programs using suitably constructed mathematical structures. Plotkin saw the importance of relating such mathematical constructions to the real-world behaviour of programs, and he identified the crucial property later called *full abstraction* which captures the equivalence between denotational equality and real-world behaviour. In a seminal work, Plotkin introduced the PCF programming language and showed that Scott's denotational models are fully abstract only if programming languages are extended with certain computationally exotic parallel features. Seeking instead to model extant sequential programming practice, Plotkin posed the enormously influential problem of constructing fully abstract models for sequential programming languages. Plotkin's full abstraction problem stimulated a major international programme of fundamental research into understanding the nature of sequential computation, to which Plotkin himself, in collaboration with Kahn, made a significant early contribution. Ultimately, after two decades, the programme culminated in the development of game semantics, which offers a general toolkit for crafting fully abstract models for many programming scenarios, and which, in the guise of algorithmic game semantics, is the focus of current research into methods for software model checking.

- **Logical frameworks.** A *logical framework* is a formal system used to formally define, reason about, and manipulate other formal systems. Mechanised logical frameworks are frequently used in the verification of both hardware and software. Plotkin and his team produced the Edinburgh Logical Framework (LF). This was among the first logical frameworks, and its elegance and generality contributed enormously to the wide adoption of the notion of logical framework. LF has influenced the design of multiple computer-assisted reasoning tools.

- **Lambda-calculus.** The lambda-calculus is a fundamental computational model utilising function abstraction and application. In a seminal paper, Plotkin gave a mathematically simple reduction semantics for reducing programming languages to the lambda calculus. His pioneering use of continuation-passing style (CPS) translations influenced their subsequent wide adoption as a general method of translation between programming languages. In the same paper, he also developed the call-by-value equational theory between lambda-calculus terms, which has been extensively built upon to support extensions of the lambda-calculus with a multitude of useful programming-language features.

To acknowledge these extensive and widely-recognized contributions to theoretical computer science and his key role as a leader and educator in the field, we are delighted to present the EATCS Award 2014 to Gordon Plotkin.

The EATCS Awards Committee 2014:

- Leslie Ann Goldberg (chair)
- Kim Guldstrand Larsen
- Vladimiro Sassone