Provably Correct Software: Not a Pipe Dream Anymore

Andreas Eggers, BTC Embedded Systems

The use of agentic AI coding systems is revolutionizing how software is built, showing enormous potential for productivity gains, but also introducing a whole new set of quality challenges -- including code being created at speeds no human reviewer can keep up with. But the solution is not reviewing all that code. The solution is maybe not even only testing all that code. The solution might be something much older, much more thorough, and for most of computer science history being thought of more like an academic sport and maybe applicable only to the very few ultra-high-risk applications and even then only sporadically due to the enormous human effort: formalization of specifications and formal proof of program correctness. It turns out that the same AI systems that we use to build code at breakneck speed are also already astonishingly good at formalizing mathematical proofs and fixing errors in them which are pointed out by the verified kernels of automated theorem provers / proof assistants like Lean 4.

In this presentation, we want to highlight some examples of how AI is currently being used by researchers in academia and in the industry to mechanically formalize mathematics, allowing automated checks of proofs, and how the same thing is starting to happen in computer science -- with some recent success stories. The second part of the talk is about how we at BTC Embedded Systems aim to leverage these new capabilities and make them available for us internally and through our tools and solutions for our customers in their development and QA processes. We have been using symbolic methods like formalization and SAT-/SMT-solving-based model checking for over two decades, and we think AI-based proof generation and theorem proving might very well be the next big building block that allows unprecedented reasoning power and thus scalability to even more challenging applications that have emerged and are still emerging today.