Society's Reach is Limited by Tools to Implement our Imagination

Warren A. Hunt, Jr.
UT Austin, Computer Science Department

June, 2020

Society's reach is limited by the tools and processes that can be developed and used. Engineering feats like the great pyramids of Egypt, the terraced fields of the Incas, the long-span bridges of modern road systems, and contemporary information systems, are a testament to envisioning, developing, deploying, and maintaining tools that extend our capabilities. Without rigorous processes to support the visions of our people, we would not have achieved these amazing demonstrations of process and precision engineering.

The use of mathematics has supported all engineering and much of the 20th century discovery in the natural sciences. Einstein's vision and associated mathematical models have helped focus and drive discoveries in cosmology and other sciences, recently with the demonstration of gravity waves with the LIGO detector. Goedel, Kleene, von Neumann, and Turing provided mathematics suitable for specifying and analyzing models of computational systems; based on their contributions, a thriving information-system ecosystem now permeates all facets of society.

Our ability to further integrate information systems safely and reliably into our food production, transportation, energy, financial, and health-care systems, is directly dependent on tools that can support the development and analysis of mathematical models. For information systems, our ability to simulate their operation in advance of their production and deployment has been and will continue to be critical. The development of reliable information systems is foundational to reducing costs and improving precision in the design and analysis of all engineering-based enterprises. Our ability to develop, sustain, and use tools to extend our mathematical and engineering skills correlates directly to what services and products we can integrate into society.

Over the last forty years, our group has focused on the development of tools that assist with creating, analyzing, and maintaining models of computational systems. These tools have increased in capacity and speed to where they are supplanting tools supplied by computer-aided-design (CAD) vendors, offering analysis capabilities unavailable commercially. The mainstay of CAD is simulation -- an ability to confirm that some model has an expected property given a particular test case. CAD-tool vendors make much of their income licensing their simulations tools, and these simulators are deeply integrated into the CAD vendor's overall tool portfolios. Although simulation is a critical component of contemporary computer design, it alone is insufficient for assuring the reliability of modern information systems.

In contemporary information systems, each item of data is commonly stored, transmitted, and operated on using 64-bit pieces. The addition of two 64-bit numbers is a fundamental, primitive operation; hundreds of such addition mechanisms appear in every contemporary microprocessor, where this addition operation is performed billions of times each second. How can we confirm that this basic operation functions correctly? We can simulate an adder design by creating a computer-based model, and then provide test cases to confirm its correct operation on those test cases -- and this is a core problem. Note, it is easy to design an adder that fails to get the correct answer on just a single pair of inputs. Using simulation, we can confirm that the sum of a pair of numbers is correct in a millisecond or so -- and this seems fast. But, to confirm fully the correctness of one 64-bit adder design requires 340282366920938463463374607431768211456 (2^128) tests; this is far more tests than can be carried out in thousands of years even by employing all of the world's existing computers; thus, there is no 64-bit adder design that has been tested exhaustively. Instead of comparing the simulation of the adder design with mathematical addition, it is much easier to compare an adder design equation with its mathematical specification symbolically; in fact, using available mechanized mathematical tools, we can make such a symbolic comparison in a fraction of a second!

Computers include lots of operations more complicated than adders, they execute billions of operations each second, and such operations are composed to reach desired results; all operations must be correct or a computer may return an incorrect result. Validation of complex computer-based calculations is far beyond what can be done thoroughly by testing. This is where computer-based computation is much different than engineering a bridge -- if just one beam on a bridge is a bit weak, then bridge will probably function for years, but if just one bit of information in a computer is incorrect, the answer returned for an important calculation, such as dispensing medicine, might be "Yes" instead of "No" in a life-critical situation. In a very real sense, computer-based calculation is brittle, and total precision is critical.

Our particular effort is focused on the creation, deployment, extension, maintenance and promulgation of the ACL2 mathematical analysis system. This system allows simulation, as offered by vendor tools, but also uses techniques based on mathematical proof to obtain the assurance that all cases work properly, an assurance unavailable by simulation because exhaustive simulation is impossibly expensive. Our group has been developing such tools for nearly 50 years, and we continue to enhance and extend our tools which are now in use by companies including AMD, Arm, Centaur Technology, IBM, and Intel. Other similar tools are being developed and/or used by Amazon, Apple, Facebook, and others. Why? Because the use of these tools allows models of information systems to be subjected to far more complete analyses than test-based methods.

To deploy a mathematical proof-based system to support industrial information-system development requires several critical characteristics that contemporary CAD systems offer:

- the capacity to manage enormous models such as the design database for entire an entire hardware design;

- an ability to integrate such a proof system with existing design and development tools;

- an environment where the use of such a proof system can be scripted to provide continuous integration with existing design processes; and

- a means to examine thoroughly the operation and results of such proof-based tools.

Over the last two decades, we have devoted a significant portion of our time to extending the capacity of our tools and to making our tools inter-operate with other tools in continuous-integration frameworks.

Tools based on mathematical proof provide a disruptive technology for the analysis of information systems, because in many cases it is far faster and far less expensive to use such technology, as indicated by the adder example describe above. As such, it is outside the interest of tool vendors to provide such tooling because such tools would significantly undercut their primary simulation-tool-based revenue stream -- and this, in part, is preventing the promulgation and deployment of more advanced technology by the very commercial organizations best positioned to offer and service such capabilities.

One mechanized proof run on one computer in minutes can replace thousands of computers running licensed simulators for years, and, in fact, at one company, our tools have displaced the use of many thousands of computers and the associated costs of running those computers. Because of the tremendous cost savings and productivity enhancements that are enabled by such technology, large companies with sufficient technical talent have begun this transition, but it will require decades of development and training to make mechanized mathematics readily available for specifying, developing, analyzing, deploying, and maintaining information-management systems.

Our ability to create, use, and maintain future information systems depends directly on our ability to force-multiply our skills with systems that mechanize and accelerate the use of mathematical proof techniques. The safety and security of our nation depends on our staying at the forefront of this mathematical technology. Just like staying at the forefront of integrated circuit technology for the production of hardware computing technology, we must stay at the forefront of tools for the design and validation of future information systems. Our nation's ability to control the "high ground" of deploying reliable and secure information systems will determine our place in the world's economic, security, and health care order. We must continue to invest in the tooling required to allow us to fulfill our imagination.

Addendum from J Strother Moore (UT Austin)

The fact that bugs creep into hardware and software creep in because people rarely write down precisely what is expected of the inputs and what is promised about the outputs. Then, as they think about whether the design ``works'' they trace out some execution paths and argue that the outputs are as claimed provided the inputs are. But these input/output specifications and the designs are complicated by the need for speed, low power consumption, and provision of many features. It is practically impossible for a human to then trace out all the combinations of possible inputs and features and confirm that the outputs comply with the claims in every combination of features. Typically, designers just test their designs with simulation.

What we do is unique because (a) we provide a precise language in which to write down the assumptions on the input and promises about the output, (b) we provide mechanical tools to analyze all possible combinations and paths through the design, (c) we provide the designers with a mathematical language powerful enough to express any computational relationship, and (d) we provide a tool that can check -- and quite often automatically generate -- an abstract argument or ``proof.'' As you point out, by shifting to a symbolic approach we can often check every possible input leads to a correct output in a fraction of the time it takes to test a tiny slice of the space.

But then the reader might be left with the question ``If what these guys do is so superior to the commercial alternatives, why don't they create a company to sell the technology?'' Put another way, ``why aren't customers flocking to their door?'' The answer is that our tools require a higher level of education and a different kind of education than the potential customers have. On the positive side, a relatively small cadre of people trained in our tools can support a much larger design group. So we are working to make the tools more automatic and to improve their capacity, and make it easier to integrate them into the design workflow. We are also working with industry and training experts that then get hired by industry to use the tools.