Trustworthy software systems - formal analysis tools development

Jim McCarthy, Brendan Mahony

March 5, 2018

Current proof assistant technologies build on 1960s paradigms of computer science, and with the increasing processing power of computers have at various times started to make an impact on mathematical and engineering development. A large barrier to their adoption across the STEM knowledge base is the rigidity of their language and reasoning systems, so that proficiency for each tool requires a large investment of time (or contracting of specialists). Comprehension of the tool output is significantly challenging for all but the experts, leaving its benefit limited – typically to claims of authority rather than understanding.

At the same time there is a growing awareness of security and safety throughout society. This leads to increasing demands for sufficient rigor in engineering developments to ensure correctness – including proper statements of the notion of correctness itself. In mathematics, also, there is growing concern at the state of the art with respect to "cultural" notions of proof and a search for robust means of putting the subject on a rigorous footing without sacrificing its cultural heritage. Both of these great disciplines have now devoted quite some effort to developing ways to address the issues.

It is clearly just wishful thinking to rely on indefinite increase in processing power to support large scale developments in engineering and mathematics (and so, Defence). The next generation of formal methods tools will need to leverage engineering notions such as modularisation, composition/refinement and re-use – through the use of algebraic (categorical) mathematical techniques – to attack these large scale problems. With a view to providing maximum reuse of any given development effort, this project is exploring the application of such ideas at the most fundamental levels: starting with the language underlying the tool, computational/reasoning structures, and the ability to translate between them. Such techniques can provide theories in a hierarchical "building block"-like construction, where compositing from basic constructs builds the desired complexity while allowing for different consistent abstractions. In turn, the abstractions support content comprehension for different audiences as well as clean and efficient structuring for the developers. In particular, subparts of a development should be treated just as the whole is – so that reasoning can be localised, and even targeted to different tools as appropriate.