Visualisations of Software Language Engineering Processes

(Internships: Department of Computing, Macquarie University, Sydney, Australia)

Background

Modern software development relies on powerful tools such as compilers, static analysers, integrated development environments, code transformers and build systems. All of these tools need to understand and process code that is written in a variety of languages. For example, compilers must understand programs and translate them to assembly language or virtual machine code. Static analysers have to determine the meaning of a program as a formal logical description of its effect and compare that effect to a specification. Build systems read a description of a build process and execute the build steps to construct a software artefact.

Software Language Engineering is the sub-field of Software Engineering that focuses on how to define and build language-based software tools. For example, many tools contain a parser that can read the text of a program and check that it conforms to an expected syntax. Software Language Engineers study how to formally define the syntax of a language using grammars and how to systematically turn a grammar into a parser implementation. Similarly, we study other processing needed by software tools, including building trees that represent syntactically valid programs, analysing trees to check that they obey semantic rules, and transforming trees to apply program optimisations. Technologies used for these problems include attribute grammars and rewrite rules.

One approach to Software Language Engineering is to write descriptions of the problems that we wish to solve using some formal language and then *translate* the descriptions automatically into solutions. For example, language syntax can be defined using context-free grammars and there are many methods for translating grammars into parser implementations. These methods are incorporated into parser generators such as YACC and ANTLR.

An alternative approach is to *embed* the descriptions directly in another host language using libraries designed for Software Language Engineering. For example, grammars can be embedded using parser combinator libraries such as that provided for Scala or the parsec library for Haskell. In this approach, a problem description is just a program written in the host language so all that needs to be done is to compile it and run. There is no need for a separate translation step so the implementation complexity is greatly reduced.

Research Question and Tasks

The Programming Languages and Verication Research Group at Macquarie University has developed the Kiama language processing library for Scala (`https://bitbucket.org/inkytonik/kiama`). See the following page for links to papers and presentations about Kiama: `https://bitbucket.org/inkytonik/kiama/src/default/wiki/Research.md`.

In the Kiama project we are investigating embedding of many language processing formalisms into Scala. Examples include parsing (using context-free grammars), analysis (using attribute grammars), transformation (using rewrite rules) and output (using pretty-printing). We are using Kiama to build a variety of language-based tools, most notably the Skink static analysis tool for LLVM code (`http://science.mq.edu.au/~fcassez/software-verif.html`).

While Kiama is effective for building software tools, it can be hard for developers to understand exactly what their processing is doing. For example, a collection of rewrite rules may transform trees in complex ways involving many traversals. Debugging this kind of transformation at the Scala level is possible, but that approach loses many of the advantages of the rewrite rule formalism.

Instead, we would like Kiama developers to be able to understand the operation of their code using visualisations that are tailored specifically for the formalisms. Possibilities include simple tree views showing attribute values or dynamic displays that show how trees are rewritten.

The aim of this internship is to:

- design high-level visualisation for Kiama processing formalisms,
- implement the visualisation Web technology such as d3 (`https://d3js.org`) or similar, and
- explore the effectiveness of the visualisations using realistic Kiama-based projects such as Skink.

Since Kiama contains many different language processing formalisms which might be visualised in different ways, this project may suit more than one student. For example, one internship could focus on visualising attribute grammar evaluation and another could address rewrite rules.

General Information

Macquarie University https://www.mq.edu.au is a public research university based in Sydney, Australia, in the suburb of Macquarie Park. It is the first in Australia to fully align its degree system with the Bologna Accord.

The Department of Computing interests are across a diverse selection of areas including: formal methods, programming languages, program verification, security. The Department of Computing is an essential component of the Optus Macquarie University Cyber Security Hub. In the 2015 ARC (Australian Research Council) assessment of research quality, the department was ranked at above World standard (4/5) in Computation Theory (0802), placing it in the top two institutions in Australia for research in this area. The internship is integrated in The Programming Languages and Verification Research Group, a unique and highly motivated group of experts in formal methods (software verification) and programming languages. In the last five years, we have proposed new fundamental theoretical results for automated source code analysis techniques and implemented some of them in a static analysis tool, Skink. The successful applicant will work with A. Prof. F. Cassez and A. Prof. A.M. Sloane.

Sydney is the largest and most populous city in Australia. It is located on Australia's south-east coast of the Tasman Sea. With an approximate population of 4.5 million in the Sydney metropolitan area the city is the largest in Oceania. Sydney also ranks among the top 10 most livable cities in the world according to Mercer Human Resource Consulting and The Economist.

Contact Information

If you have any questions concerning the internship, please do not hesitate to contact us:

A. Prof. Anthony Sloane
+61 2 9850 9582
email: anthony.sloane@mq.edu.au

A. Prof. Franck Cassez
+61 2 9850 9513
email:franck.cassez@mq.edu.au