tetracom

Technology Transfer in Computing Systems

D3.35: Individual TTP35 abstract

Project no.:	609491
Funding scheme:	Collaborative project
Start date of the project:	1st September 2013
Duration:	36 months
Work programme topic:	FP7-ICT-2013-10
Deliverable type:	Report
Deliverable reference number:	ICT-609491 / D3.35
WP and tasks contributing:	WP 3 / all
Due date:	31/05/2016
Actual submission date:	28/07/2016
Responsible Organization:	IMPERIAL
Dissemination Level:	Public
Revision:	1.0

TETRACOM D3.35: CK/CLsmith: An Automated Testing Framework for Many-Core Vendor Tools

Alastair Donaldson, Andrei Lascu, Paul Thomson and Pantazis Deligiannis (Imperial College London), Grigori Fursin, Anton Lokhmotov (dividiti Ltd., UK)

The OpenCL and OpenGL programming models provide a means for achieving portability across a range of accelerator platforms for compute and graphics tasks, respectively. A key component of an OpenCL-accelerated application or an application that uses OpenGL for graphics is a programmable *kernel* (OpenCL) or *shader* (OpenGL) that executes across the processing elements of a many-core device, such as a GPU. To allow kernels and shaders to execute, many-core platform vendors must deploy compilers for kernel and shading languages that translate kernels and shaders into device-specific machine code.

Writing a reliable compiler for a kernel or shading language is challenging, and recent work at Imperial College London using random testing has led to the discovery of a large number of defects in OpenCL kernel compilers (http://multicore.doc.ic.ac.uk/publications/clsmith-pldi-15.html). This is problematic, since many-core technologies are increasingly being deployed in safety-critical contexts (e.g. the Khronos Group recently formed an Advisory Panel for Safety Critical APIs: http://www.prweb.com/releases/2016/07/prweb13579173.htm). In response to this need, Imperial developed CLsmith, a test case generator for OpenCL based on the Csmith program generator for C, which can aid in rapid discovery of OpenCL compiler bugs. Using CLsmith for testing requires executing tests across a range of graphics cards and CPUs, which complicates the testing process.

In this TTP, we focused on expanding the scope of CLsmith-based testing of many-core compilers by (a) providing support for testing compilers for the OpenGL shading language (GLSL), and (b) harnessing the Collective Knowledge (CK) framework from dividiti (a UK start-up founded in 2015 and specializing in fast, small, reliable and energy-efficient computer systems) to ease the burden of testing across a range of platforms, aid in capturing experimental data sets associated with testing campaigns, and provide "single-click" crowdsourcing, whereby testing of OpenCL- and OpenGL-capable devices from a range of end users can be automated, enabling analysis of the aggregate results. Our specific objectives were:

- Enhance a prototype test case generation framework for GLSL to enable practical testing of OpenGL compiler implementations.
- Integrate test case generation and result collection with CK for management of result data.
- Embed Imperial's test case generation tools into the CK framework to enable seamless deployment and single-click crowdsourcing.

This TTP covers partially the period of collaboration between Imperial College London and dividiti; we have firm plans for a continued relationship in the context of this project.

Testing workflow: Figure 1 shows the workflow that we adopt for many-core compiler testing. Our **program generation tools** – CLsmith for OpenCL, and GLmorph, the GLSL program generator developed during the TTP – generate families of test programs that, by construction, should be semantically equivalent. Our **test execution framework** runs the program families on an OpenCL/OpenGL platform on a specific device (typically a GPU from a major vendor). For widespread applicability we support the major desktop platforms, and during the TTP we have developed test clients for Android, iOS and WebGL. The test results for all programs in a family should match; if there is a mismatch then a minority result usually indicates a compiler bug. The relevant program is fed to our **program reducer** to produce a minimal test exposing the bug, to be filed to a vendor.

Integration with Collective Knowledge: Figure 1 illustrates our vision for integrating the testing workflow of Figure 1 is integrated with the Collective Knowledge (CK) framework, which has largely been realized during the TTP for the case of OpenGL. CK provides a unified JSON API and JSON meta for the various data items that flow through the tool chain, including information about OpenCL and OpenGL platforms, families of test cases, and details of identified bugs. We have adapted the test case generation and reduction tools of Figure 1 so that they interact directly with CK-managed repositories, allowing the testing process to be tracked in manner that is largely reproducible.

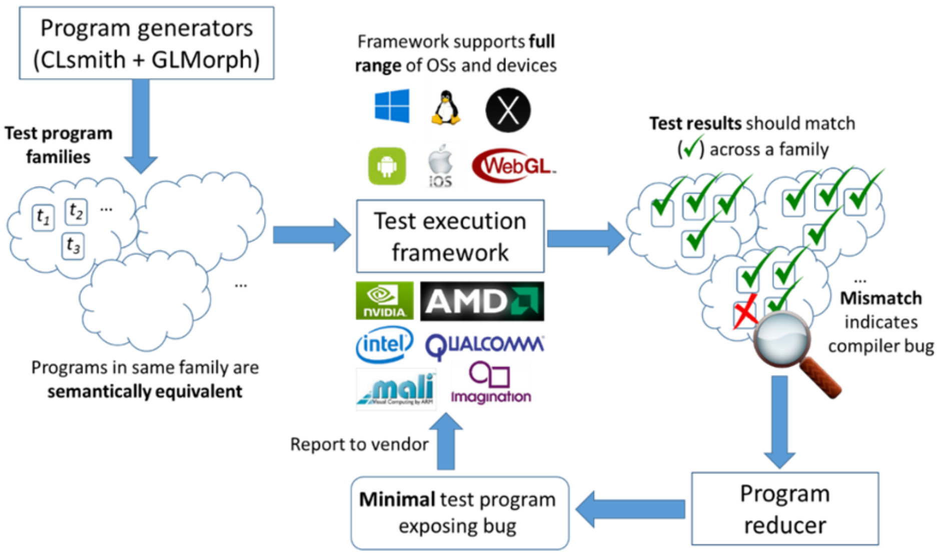
Program generators (CLsmith + GLMorph)
Test program families
Programs in same family are semantically equivalent
Framework supports full range of OSs and devices
iOS
WebGL
Test execution framework
Test results should match (✓) across a family
Mismatch indicates compiler bug
Report to vendor
Minimal test program exposing bug
Program reducer

Figure 1: Our many-core compiler testing workflow

The CK integration also facilitates easy deployment of our test execution framework on end-user devices, enabling crowdsourcing of data on many-core compiler bugs. During the TTP we have progressed the CK integration to the point where a prototype of crowdsourcing has been achieved (sourcing data from a range of devices at dividiti and Imperial). Our next steps will be to publicly deploy crowdsourcing to gather data from a wider variety of devices and platforms, which we will share via publicly hosted CK repository.

Impact: Thanks to the TTP, dividiti can offer an additional platform evaluation service for end-users (e.g. manufacturers of self-driving cars), with a focus on **robustness** as well as performance. Medium-term impact of the TTP work also includes: **improved many-core driver quality** thanks to fixes for bugs identified by our approach – Intel have fixed multiple OpenGL driver bugs during the TTP in response to our defect reports, indicating that IP vendors are interested in robustness issues; and **a robust framework** in which to conduct exciting research on crowdsourcing for software analysis and testing.

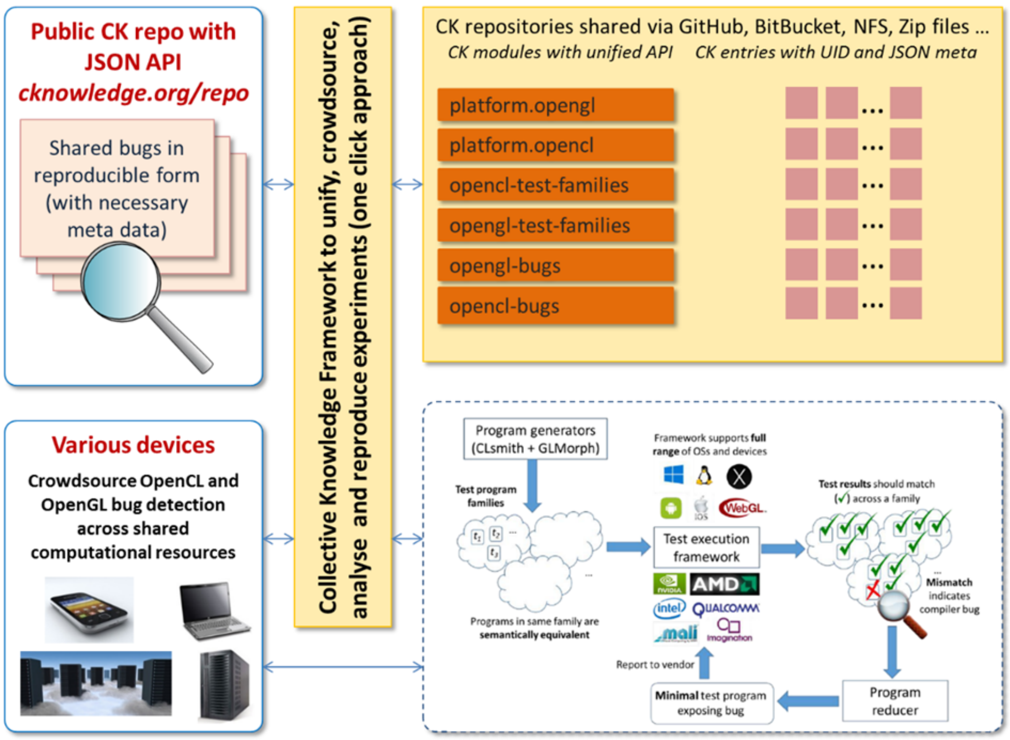
Public CK repo with JSON API
cknowledge.org/repo
Shared bugs in reproducible form (with necessary meta data)
Collective Knowledge Framework to unify, crowdsource, analyse and reproduce experiments (one click approach)
CK repositories shared via GitHub, BitBucket, NFS, Zip files ...
CK modules with unified API
CK entries with UID and JSON meta
platform.opengl
platform.opencl
opencl-test-families
opengl-test-families
opengl-bugs
opencl-bugs
Various devices
Crowdsource OpenCL and OpenGL bug detection across shared computational resources
Program generators (CLsmith + GLMorph)
Test execution framework
Minimal test program exposing bug
Program reducer

Figure 2: Integrating our testing workflow with Collective Knowledge to enable crowdsourcing of platform reliability data.