CLAVA: C/C++ source-to-source from CMake using LARA

João Bispo, Pedro Pinto, João M. P. Cardoso
Jefferson Delay, João A. M. P. Cardoso

Introduction

- **C++ source-to-source compiler written in Java**
  - Built on top of the LARA framework
- **Analysis and transformations written in LARA**
  - DSL based on JavaScript for source code analysis and manipulation
- **Custom AST for C/C++ code, based on the Clang AST**
  - Uses Clang to parse code

CLAVA Toolflow and Clang Usage

- **Initial idea**: use Clang AST as IR for LARA
  - Prototype with ZeroMQ and Protocol Buffers
  - However, Clang source-to-source is text-based
- **Development of CLAVA AST in Java**
  - Clang AST is used to apply transformations and generate code
  - Stand-alone C++ program (Clang AST Dump) - Uses Clang as a library to dump the necessary data for CLAVA AST
  - Simplifies flow, one-way communication through output streams

Limitations

- **Generated source-code has preprocessor elements resolved**
  - Currently only reinserts include directives
- **Does not cover all of C/C++**
- **CMake plugin is at proof-of-concept stage**

Code Base

- **CLAVA has ~36,000 SLOC-L**
- **Open-source, on GitHub**

Execution Example

- **Medium to large files**
  - NAS benchmarks
  - Large single C programs
- **Machine specs**
  - Xeon E5-1650 3.60GHz
  - 64GB RAM
- **LARA Code**
  - Inserts code to create a dynamic call graph

Selected Use Cases

- **OpenMP Auto-parallelization**
  - Static analysis of for loops
  - Uses Omega library for array analysis
  - Inserts OpenMP pragmas
- **LAT**
  - Design space exploration in LARA
  - Compiles, runs and collects results for multiple code versions
- **Code Generation for HDF5**
  - File format/library to serialize values
  - Automatic generation of the boilerplate code

LARA + CMake Example

```
// Includes

#include <iostream>
#include <string>
#include <vector>

std::vector<int> function(int x)
{
    std::vector<int> recs = std::vector<int>(1, 10, 1); // Fill the vector with 1, 10, 1

    for (int i = 0; i < recs.size(); i++)
    {
        // Function implementation
    }

    return recs;
}

int main()
{
    std::vector<int> results = function(5); // Call the function with argument 5

    // Process the results
    return 0;
}
```

CMake

```
# Simple code insertion example
- Clang & C++: 57 SLOC-L
- LARA: 2 SLOC-L
- CMake: 2 SLOC-L

# Requirements
- Stand-alone JAR does not require any installation
- Installation script for Linux (updater, CMake)
- Runs on Ubuntu, CentOS, Windows and MacOS

# Features
- Documentation generator
- Unit testing
- Standard library
- CMake integration
- Try CLAVA
  - github.com/specs/feup/clava
  - specs.fe.up.pt/tools/clava
```

Acknowledgments

João Bispo acknowledges the support provided by Fundação para a Ciência e a Tecnologia, Portugal, under Post-Doctoral grant SFRH/BPD/118211/2016.