Objective-Droid: Honours Project Progress Report 1

Stanislav Manilov

October 11, 2012

1 Overview

Objective-Droid is a compiler that compiles applications originally written for iOS to Android. The essence of the project is writing a backend that translates LLVM code to Dalvik bytecode.

2 Motivation

The problem was inspired by the issue of portability of mobile code. Currently, there are tools to build multiplatform mobile applications, but none to re-target, or port, applications that are already developed for a specific platform.

In addition to being an interesting project, Objective-Droid can potentially make: Objective-C more popular, Dalvik VM easier to target, developers worry less about portability, and users have access to wider range of apps. Everyone has a reason to be happy!

3 Issues

There are several potential problems that can arise for the project.

First, iOS libraries are proprietary. There have been attempts to rewrite them in open-source code, notably GNUstep, but it is still version 0.x. Using it is outside of the scope of the project, but possible.

Second, translating idioms for writing apps would not be a trivial task. This might require some work on the front-end, which is outside the scope of this project as well.

Third, and more importantly, during the first group meeting about the honours projects, the potential risk of having internal representation instructions that are hard to match to DEX instructions was identified. It was suggested that these, and ways to work around these, should be investigated.

In order to minimise potential risks, the aim of the project is narrowed down to the ability to compile command-line programs. Also, it is important to sustain good code quality and produce a software that is maintainable and easy to build on.

4 Implementation

Objective-Droid is based on the LLVM Compiler Infrastructure. It uses clang for a front-end, it's own backend, smali for assembling, and apktool for packaging. Objective-C is compiled to LLVM code by clang, the backend then translates it to smali code, smali assembles the smali code to Dalvik bytecode, and finally, apktool packages the .dex files to an .apk file.

5 Evaluation

Two aspects of the compiler are going to be tested: correctness and performance of the resulting programs.

Correctness is going to be tested by compiling a set of programs using Objective-Droid for a mobile app, and gcc for a desktop app. If the results are always the same, then there is a good chance that Objective-Droid is producing correct programs.

Performance is going to be measured by compiling a specific benchmark programs written in both Objective-C and Java. The Obj-C versions will be compiled with Objective-Droid and the Java versions will be compiled using the Android SDK. After that a set of metrics (speed, memory usage, energy usage, possibly others) is going to be measured for both programs and compared.

6 Timeline

By the end of October a big portion of the research on the DEX format and writing an LLVM backend will be researched. By the end of November there will be a first prototype of the compiler with limited functionality. By the end of December the functionality will be broaden and evaluation will have been started. By the end of January a big portion of the evaluation will be done and documentation will have been started. By the end of February the compiler will be in a finished or near-finished state. By the end of March the report will have been written.