Amy language: C Backend

Final Report

Jeniffer Lima Graf Alexander Sanchez de la Cerda

EPFL

{firtname.lastname}@epfl.ch

1. Introduction

Describe in a few words what you did in the first part of the compiler project (the non-optional labs), and briefly say what problem you want to solve with your extension.

This section should convince us that you have a clear picture of the general architecture of your compiler and that you understand how your extension fits in it.

2. Examples

Give code examples where your extension is useful, and describe how they work with it. Make sure you include examples where the most intricate features of your extension are used, so that we have an immediate understanding of what the challenges are.

You can pretty-print tool code like this:

```
object {
  def main() : Unit = { println(new A().foo(−41)); }
}

class A {
  def foo(i : Int) : Int = {
    var j : Int;
    if(i < 0) { j = 0 − i; } else { j = i; }
    return j + 1;
  }
}
```

This section should convince us that you understand how your extension can be useful and that you thought about the corner cases.

3. Implementation

This is a very important section, you explain to us how you made it work.

3.1 Theoretical Background

If you are using theoretical concepts, explain them first in this subsection. Even if they come from the course (eg. lattices), try to explain the essential points *in your own words*. Cite any reference work you used like this [Appel 2002]. This should convince us that you know the theory behind what you coded.

3.2 Implementation Details

Describe all non-obvious tricks you used. Tell us what you thought was hard and why. If it took you time to figure out the solution to a problem, it probably means it wasn't easy and you should definitely describe the solution in details here. If you used what you think is a cool algorithm for some problem, tell us. Do not however spend time describing trivial things (we what a tree traversal is, for instance).

After reading this section, we should be convinced that you knew what you were doing when you wrote your extension, and that you put some extra consideration for the harder parts.

4. Possible Extensions

If you did not finish what you had planned, explain here what's missing.

In any case, describe how you could further extend your compiler in the direction you chose. This section should convince us that you understand the challenges of writing a good compiler for high-level programming languages.

References

A. W. Appel. *Modern Compiler Implementation in Java*. Cambridge University Press, 2nd edition, 2002.