Optimisation of Control Flow Graphs using Graph Rewriting

Maik Marschner

Leibniz Universität Hannover
Institut für Mensch-Maschine-Kommunikation
Lehrstuhl Graphische Datenverarbeitung

www.welfenlab.de

11th May 2016

Leibniz
Universität
Hannover

Motivation

- Buggy is a language-agnostic programming framework currently in development at the *Welfenlab*

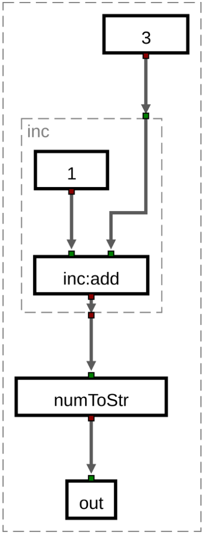
Functional code
Abstract problem descriptions
Formulas
. . .
$\rightarrow$ Buggy graph $\rightarrow$
Another language
$\rightsquigarrow$ Machine code
Graph images
. . .

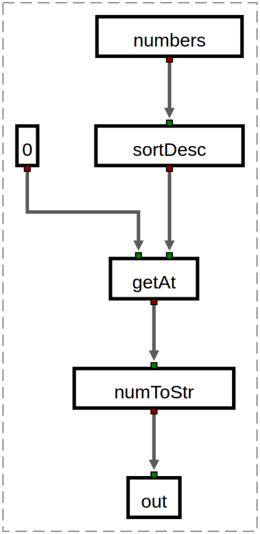
- Programs are compositions of *atomic nodes* and other programs and lambda functions (*compound nodes*)

Motivation – The current state of Buggy

- Subset of Lisp $\rightarrow$ Buggy graph $\rightarrow$ Go
- *Compound nodes* are resolved, Buggy graphs contain their implementation
- Buggy: Graphs contain the entire control and data flow, even across libraries (i.e. foreign *compound nodes*)
  Other languages: usually problematic to extract the control flow graph
- They have a lot of potential for optimisation

Motivation – Examples





Goals

- Create a convenient library for graph rewriting and a command line tool to optimise Buggy graphs
- Rewrite Buggy graphs to optimize size and runtime performance
- Investigate the possibilities and limits of graph rewriting

Schedule

Week 1-4	• Familiarise with Buggy • Start programming the rewriting tool and library • Optimise trivial cases
Week 5-8	• Optimise some non-trivial cases • Research (rewrite rules, theory of graph rewriting)
Week 9-12	• Research (efficient graph rewriting, limits of graph rewriting) • Improve performance, if possible
Week 13-16	• Write the Bachelor thesis