AUTOBAHN : Using Genetic Algorithms to Infer Strictness Annotations

Yisu Remy Wang, Diogenes Nunez, Kathleen Fisher

Tufts University, Medford MA, USA

September 2, 2016

Overview

- Intro.
- The Problem: Adding Strictness Annotations
- Background: Strictness Annotations & Genetic Algorithms
- The Algorithm
- Soundness
- Evaluation & Case Study
- Related Work

Once Upon a Time, 2 Eager Programmers with a Lazy Program . . .

- Example code & explanation here
- Annotated code & explanation, explain laziness / bangs informally in passing
- ?? What kind of example do we want?

The Problem

- **Need** to add strictness annotations
  - Too much laziness **slowdown** programs: runtime, allocation, GC work ...
  - ghc must be **conservative**, does not add best annotations
  - **Library writers** cannot know how much laziness is good
- **Difficult** to add strictness annotations
  - Add **helpful** annotations
  - Guarantee **soundness**

Background: Laziness & Strictness Annotations

- What's a thunk, what's WHNF
- Different kinds of annotations and what do they do (mention StrictHaskell)

Background: Genetic Algorithms

- Introduce GA

Background: GA for Strictness Annotations

- Why is it good: avoid local optima
- Works like a desperate Haskeller trying all kinds of different bangs
- Works great if bangs work with each other in a simple way (corpus analysis will answer this)
- !! bangs can only have complex relation when they share some scope, investigate this!

Algorithm: Representation

- Genes & chromosomes
- Fitness Functions: the user has the opportunity to optimize over any metric Haskell RTS reports

The Algorithm: Optimization

- Parameters
- 1st Generation
- New Generations
- Determining a Winner

The Algorithm: Pulling it All Together

- List of inputs
- Start demo and leave running?

The Algorithm: Discussion

- seq, strict app etc.

Soundness

- use example for soundness problem
- requiring representitive input is good enough

Evaluation

- Introduce benchmarks / setup

Evaluation: nofib benchmarks

- standard benchmark, make sure AUTOBAHN works
- however, not target use (programs that have unacceptable performance)

Evaluation: strict Haskell

- ?? Did we ever try StrictHaskell on aeson/gcSim?

Evaluation - Case Study: gcSimulator

Evaluation - Case Study: Aeson

- Specializing libraries

Evaluation: 10-fold Cross-validation

- Stability of optimization

Evaluation: Autobahn Performance

- room for improvement: parallelization, use GHC-API

Related Work

- static analysis
- including dynamic information
- other approaches

Future Work

- Soundness
- Other algorithms
- Improve the tool
- Learnabile?
- Learn to be annotation expert

Conclusion

Acknowledgments