Pony: Evolving Languages Without (as many) Bugs

Andrew Hirsch

September 23, 2012

Introduction

- Pony is an *extensible compiler* for C
- Add features to C
- Think of slang words: we say “lol”
  - We don’t actually say that we laugh
  - We don’t write out our laughter

Changing Your Thoughts

- Traditional Languages require that you change your thoughts to fit the language
  - I.e. to write a mathematical function in C, one has to change it to imperative form
  - I.e. to write a subtype in Haskell, one has to understand `Monads` and `Kinds`.
- This can be very confusing!

Change Your Language

- Instead of changing thoughts, we should change languages!
- Pony allows us to change C
  - For data structures, add objects
  - For mathematical functions, add features like function composition
  - Add syntactic sugar for features, to add standard notation to C
    - I.e. [1, 2, 3] for the linked list containing 1, 2, and 3

Multiple Extensions

- Most programs involve more than one algorithm
  - Hence, more than one extension makes sense
- However, extensions might use the same syntax
  - If [1, 2, 3] is both a linked list and a javascript-style (prototype-based) object, which should the compiler use?

Collision Detection

- We have termed detecting this problem *Collision Detection*
- Over a single piece of code, there are ways to tell if there is collision
  - I.e. trans1 $\circ$ trans2 $\overset{?}{=}$ trans2 $\circ$ trans1
- However, is it possible to tell if two transformations will <u>never</u> collide?

Difficulty

- We have devised a high-level algorithm for detecting collision detection
- However, in the general case, it is not possible to perfectly detect collision
- Our algorithm is too conservative
  - If there is any possibility that two transformations won't work together, we reject it

Productivity for practitioners

- Pony is a useful tools for developers everywhere
  - Pony can make developers more productive
  - It may also help reduce bugs in code, by making the "thought to code" process easier

Contributions to the scientific conversation

- Pony contributes to a scientific conversation
  - Extensible parsers have not been written in pure functional ways before
  - Collision detection is new, and has not been attempted in our general framework
  - These contribute to our understanding of the limits of extensible languages