Project Bluex

Pattern Matching on Terms

Jan Sliacky

Do you want to build it?
Send me a message!

The Idea

A long time ago at a faculty not really that far away

- Lambdulus was born
- λ-calculus evaluator
- It could’ve been very different tool though

What Could Go Differently?

To build a language to evaluate a language

- To build a language to build a language?
- Kind of like Redex (Racket)

What Would That Look Like?

Grammars…

```
M, N, E := VAR
         |  ‘(‘ ‘λ’ VAR ‘.’ E ‘)’
         |  ‘(‘ M N ‘)’

VAR, V  := /[a-z]+/
```

… and functions

```
β “( (λ V . M) N )”
  | not’capturing V M (free’vars N)
  = substitute V M N
```

…

Visualization

The most important part

```
(λ x . ((λ y . y) x))
```

Relying on LR Analysis Again

Attributes

```
color’β-redex “( (λ V . M) N )”
  = “( (λ V . M’) N )”
  where M’ = color’free V M
```

...

So Where Are the Attributes?

Writing Colors…

- abstraction on top of very stable theory
- Synthesized vs Inherited attributes

Down to the Boring Bits

Just implementation details

```
color'β-redex "( (λ V . M) N )"

  = colored

  where colored = ( "( (λ V . M') N )"
                  , [W,Y,Y,O,Y] ++ M'.snd ++ [Y,P,W] )

        M' = color'free V M
```

...

All of This for What?

The perfect tool

- An accessible (probably web-based) app
- You give a grammar and the evaluation semantics in the form of functions
- You get a stepping, visualizing evaluator with intuitive UI

For any language you can think of!