

Functional Data Structures

Mattox Beckman

Illinois Institute of Technology
Computer Science

Who we are / October 6, 2010

Who is Mattox?

Name Mattox Beckman

History PhD, Fall 2003, University of Illinois at Urbana-Champaign

Area Programming Languages

Specialty Partial Evaluation, Functional Programming

Professional Interests Teaching; Partial Evaluation; Interpreters;
Functional Programming; Semantics and Types;
Continuations

Personal Interests Home-brewing; Theology; Investing; Plants;
Tarantulas; Evolution; Travel; Korean Culture... and many
many more ...

Teaching philosophy is available

<http://dijkstra.cs.iit.edu/media/mattox/teaching-philosophy.pdf>.

Models of Computation

- Why do programming languages look the way they do?

- Imperative Languages: control the machine

```
a := a + 1
```

- OO Languages: model the agents

```
a.increment()
```

- Functional Languages: express the mathematics

$$a + 1$$

- Logic Language: express the constraints

```
inc(a,b) :- b is a+1
```

Why Functional Programming

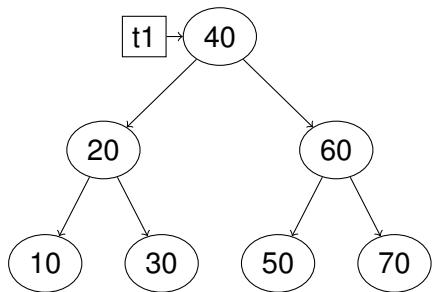
- Economy of Expression

```
guess [] = []  
guess (x:xs) = guess [y | y <- xs, y < x]  
               ++ [x] ++  
               guess [y | y <- xs, y >= x]
```

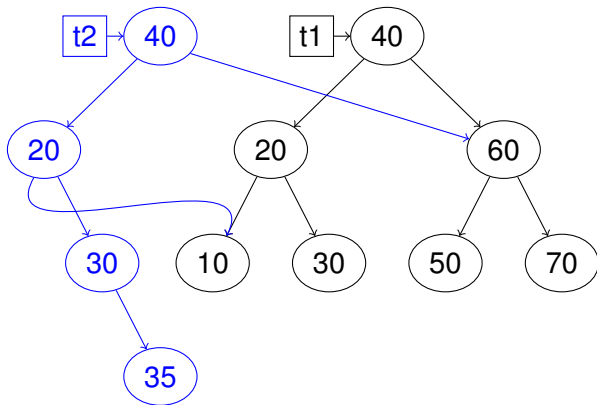
- Lazy Evaluation

- No Assignment!!

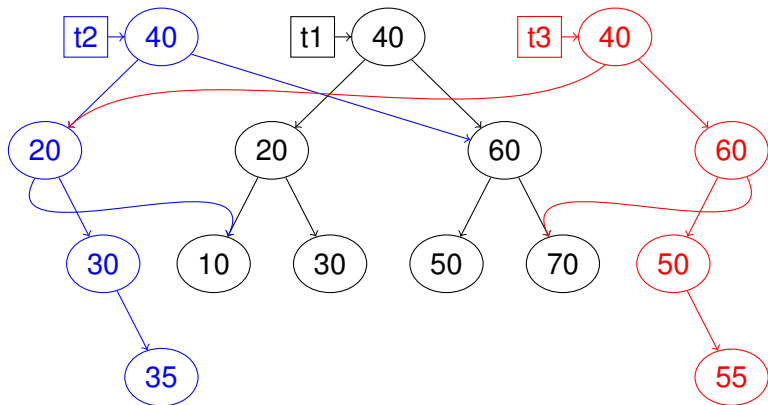
- - May use more memory
- + Easier to verify
- + Easy to parallelize



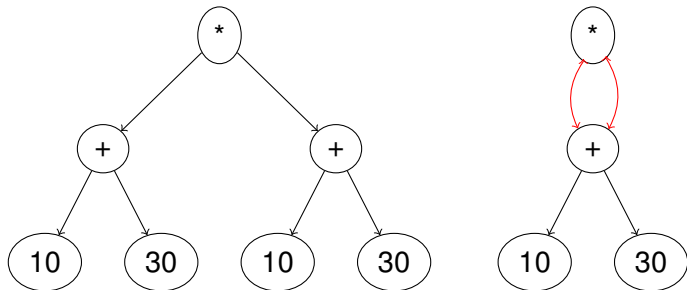
What if we run $t2 = \text{add}(t1, 35)$ on this?



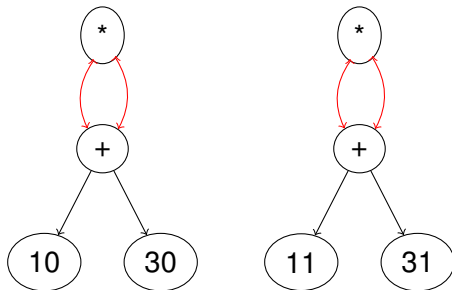
What if we run $t_3 = \text{add}(t_2, 55)$ on this?



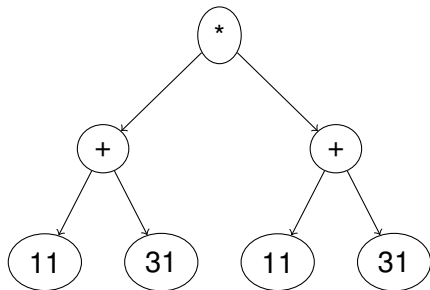
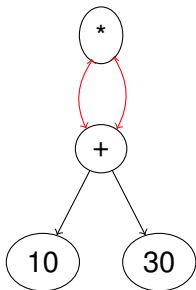
Shared References and Mathematical Purity



How to do this?



And not this?



Some Solutions

- Hard code the functions. (Yuck)
- Use a “trace”. (Works, but cumbersome.)
- Build a new language construct!