
dots

— A graph programming language —

The Team

Hosanna Fuller: Manager

Rachel Gordon: Language Guru

Adam Incera: C Guru

Yumeng Liao: Tester

The Problem

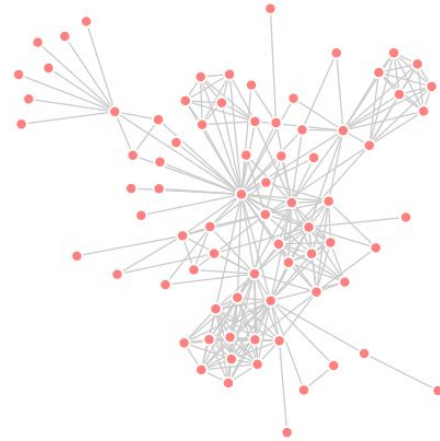
Masters student at Columbia interested in Networks

Noticed that implementing graphs or graph operations are a high cost data structure to build

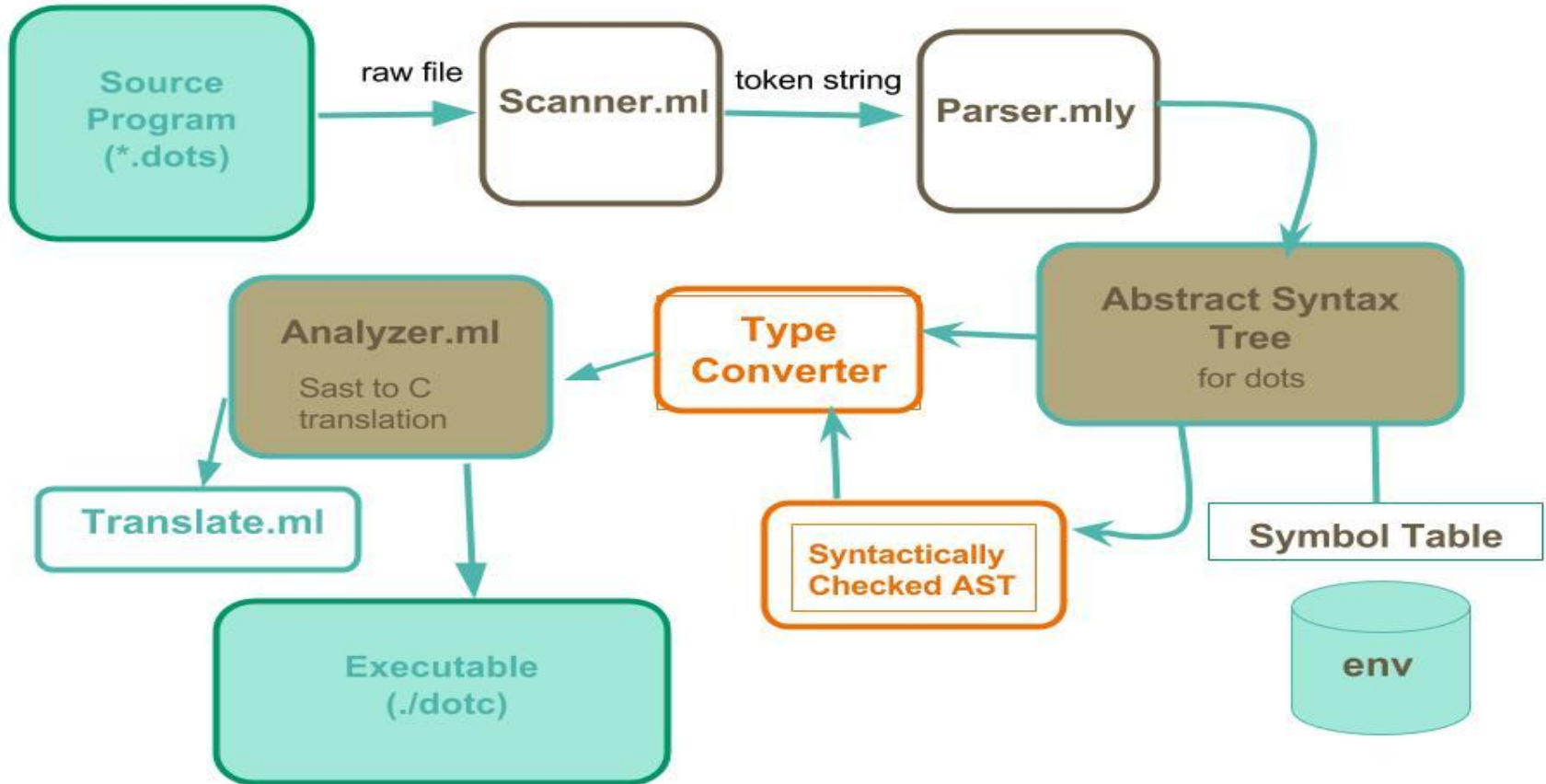
Wanted to to try to implement Graph language to help with my research

Our Goal

Create a versatile language to allow simple manipulation of graphs and their data.



Compiler Architecture



Language Features

```
/*Types*/  
num  
string  
bool  
node  
graph  
list<type>  
dict<key,val>  
  
/*Declare Function*/  
def num foo(num bar){  
    return bar;  
}
```

```
/*Operators*/  
+ - * / > <  
>= <= == !=  
|| &&  
-->[weight]  
  
/*Some Builtin Fns*/  
enqueue(x)  
dequeue()  
node_var.in()  
node_var.out()  
min(x)  
max(x)
```

```
/*Control Flow*/  
for (x in y) {  
    something;  
}  
  
if (x == y) {  
    something;  
} else {  
    other;  
}  
  
while (x == y) {  
}
```

Language Basics - Hello World

```
/* This is a comment! Let's declare and assign some variables. */  
num w; string v = "hello world";  
node y; node z("hello");  
list<num> l = [1,2,3];  
dict<num,string> d; d[3.3] = "foo"; d[10] = "bar";  
  
/* Declare graphs with weighted edges easily */  
graph g1 = {  
    y -->[10] z  
};  
  
/* Alternatively, add nodes together to make a graph! */  
graph g2 = y + z;
```

Demo

Test Suite

- Unit/Regression testing
- Vast majority of testing on generated code
- Python script that runs all tests in a folder by opening subprocesses

```
[X] max(dtest/call-builtin-fn)
[X] member-call-enqueue(dtest/member-funcs)
[ ] member-comp(dtest/member-funcs)
[ ] member-var(dtest/member-var)
[X] min(dtest/call-builtin-fn)
[X] multi-hello-world(dtest/print)
[X] multi-line-cmnt(dtest/comments)
[X] multi-num-decl(dtest/nums)
[X] node-decl(dtest/nodes)
[X] node-decl-simp(dtest/nodes)
[ ] node-out(dtest/member-funcs)
[X] num-assign(dtest/nums)
[X] num-assign-decl(dtest/nums)
[X] num-decl(dtest/nums)
[X] num-literal(dtest/nums)
[X] out-access(dtest/member-funcs)
[ ] peek(dtest/member-funcs)
[X] print-dict(dtest/print)
[X] print-funct(dtest/functions)
[X] print-list(dtest/print)
[ ] print-node(dtest/print)
[X] print-num(dtest/print)
[X] print-string(dtest/print)
[X] relax-decl(dtest/functions)
[X] single-cmnt(dtest/comments)
[X] string-literal(dtest/strings)
[X] string-var(dtest/strings)
[X] while-simple(dtest/while-loop)
```

Tests that should fail:

```
[X] access-nonexistent(ntests)
[X] dict-literal-keyblank(ntests)
[X] dict-literal-keymiss(ntests)
[X] dict-literal-valblank(ntests)
[X] dict-literal-valmiss(ntests)
[X] double-vdecl(ntests)
[X] list-assign(ntests)
[X] list-literal(ntests)
```

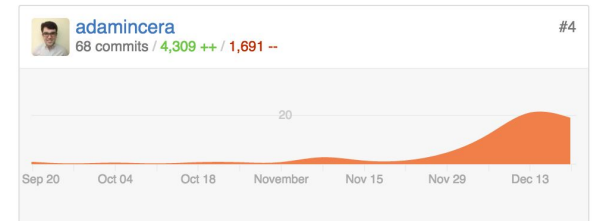
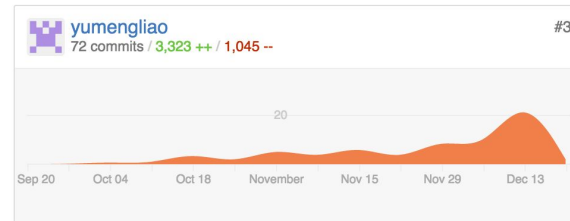
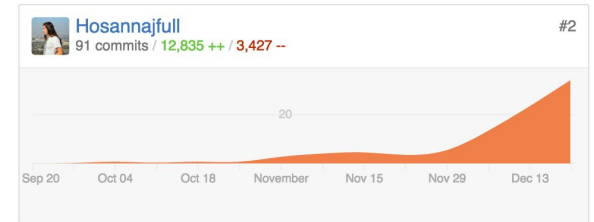
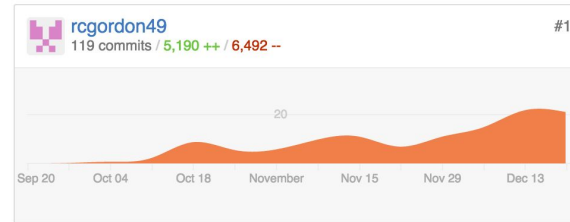
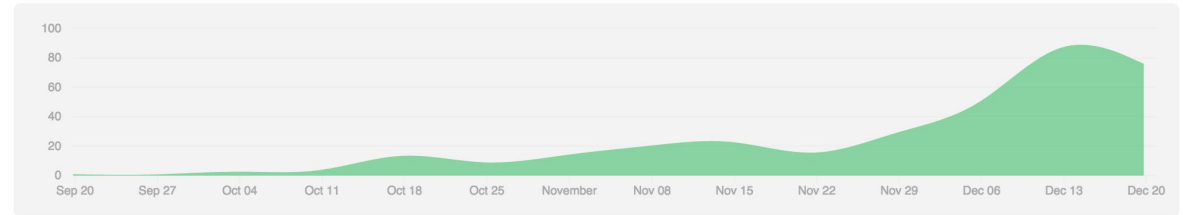
Workflow

1. Weekly Meetings
2. Biweekly Standups
3. Slack Channel
4. Github
5. Pair Programming

Sep 20, 2015 – Dec 21, 2015

Contributions to master, excluding merge commits

Contributions: **Commits** ▾



Lessons Learned

- Ambition is good, but don't try to do everything ever in the universe
- Pair programming is awesome
- Get the minimum viable product working
- Ocaml has its magical moments

Excluding merges, **4 authors** have pushed **169 commits** to master and **171 commits** to all branches. On master, **287 files** have changed and there have been **4,710 additions** and **1,119 deletions**.