Maestro

A language for job scheduling

Team Members

Vaggelis Atlidakis

Georgios Koloventzos

Mathias Lecuyer

Arun Swaminathan

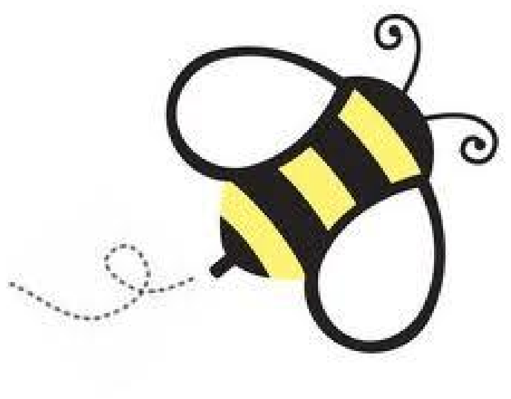
Yiren Lu

TA: Junde Huang

Agenda

- What is Maestro?
- Why Maestro?
- Who uses Maestro?
- Example # 1
- Syntax Explanation
- Example # 2
- Syntax Explanation
- System Architecture
- Testing
- Demo
- Lessons
- Thank you!!

What is Maestro?



Declarative, Interpreted Scripting Language for Job Scheduling

Dynamically Typed

Powerful Semantics for Job Distribution

Why Maestro?

Consider an experiment that is divided into 3 consecutive steps. Maestro can help:

- Express each step with a script and define a Maestro **Job.**
- Express the correlation of steps using Maestro **Job Dependencies**.
- Execute each step only after its dependencies are **Resolved**.

Who Uses Maestro?

- Research Labs like CERN that run large-scale distributed jobs
- Academics running thousands of experiments on a strict timetable
- Anyone who is tired of hand-holding a script through a conditional pipeline

Example # 1 (Hello World)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("print.rb", "Hello");
b = Job("print.rb", "World");
c = Job("print.rb", "!");
run(a -> b -> Wait(10) -> c);
```

Redis

Example # 1 (Hello World)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("print.rb", "Hello");
b = Job("print.rb", "World");
c = Job("print.rb", "!");
run(a -> b -> Wait(10) -> c);
```

Defining Jobs

Example # 1 (Hello World)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("print.rb", "Hello");
b = Job("print.rb", "World");
c = Job("print.rb", "!");
run(a -> b -> Wait(10) -> c);
```

Dependencies Syntax Operators

Example # 2 (MapReduce)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("split.rb", "/tmp/big_file_name.data");
maps = map(a, "map.rb");
red = reduce(maps, "reduce.rb");
run(red);
```

Redis

Example # 2 (MapReduce)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("split.rb", "/tmp/big_file_name.data");
maps = map(a, "map.rb");
red = reduce(maps, "reduce.rb");
run(red);
```

Define Job

Example # 2 (MapReduce)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("split.rb", "/tmp/big_file_name.data");
maps = map(a, "map.rb");
red = reduce(maps, "reduce.rb");
run(red);
```

Map

Example # 2 (MapReduce)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("split.rb", "/tmp/big_file_name.data");
maps = map(a, "map.rb");
red = reduce(maps, "reduce.rb");
run(red);
```

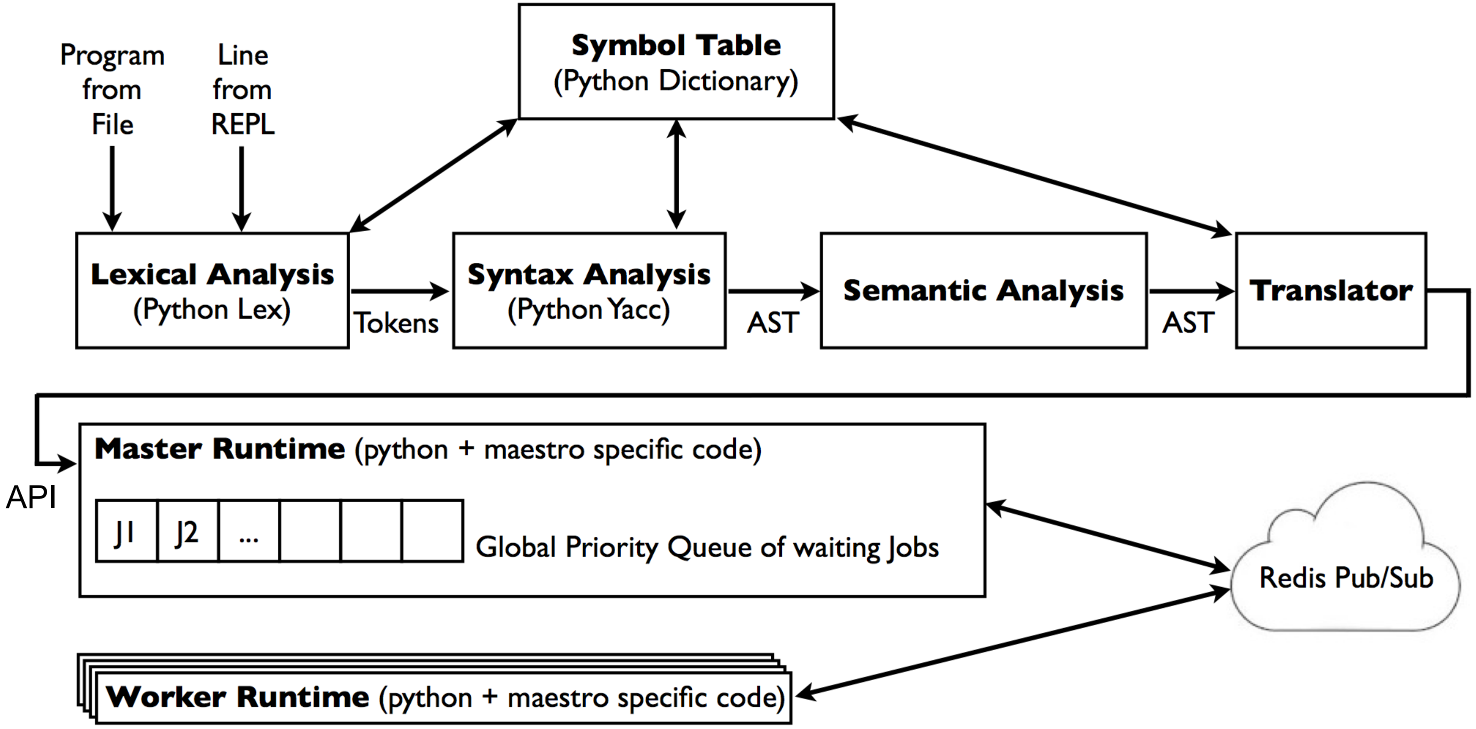
Reduce

Example # 2 (MapReduce)

```
master("systems-yellow.cs.columbia.edu:6379");
a = Job("split.rb", "/tmp/big_file_name.data");
maps = map(a, "map.rb");
red = reduce(maps, "reduce.rb");
run(red);
```

Run Command

System Architecture

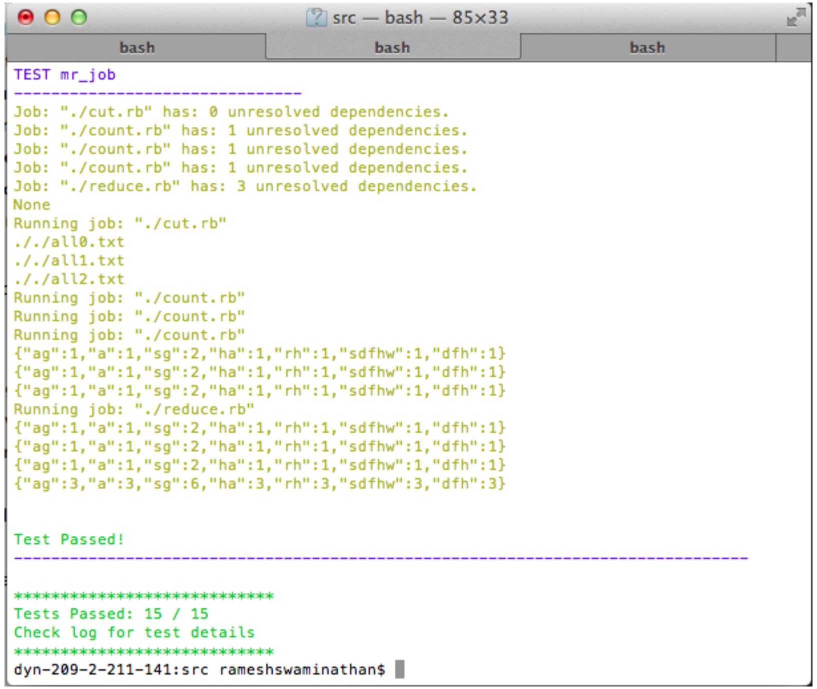

Program from File
Line from REPL
Symbol Table (Python Dictionary)
Lexical Analysis (Python Lex)
Tokens
Syntax Analysis (Python Yacc)
AST
Semantic Analysis
AST
Translator
API
Master Runtime (python + maestro specific code)
J1
J2
...
Global Priority Queue of waiting Jobs
Redis Pub/Sub
Worker Runtime (python + maestro specific code)

Testing

- Custom test engine -no library or module used
- One framework for testing all parts of the program
- Tests not only the execution of Maestro, but also the execution of the job sent to Maestro
- Supports individual or batch testing
- Logs test results for deeper analysis and to help locate/fix errors quickly

Testing

Batch tests

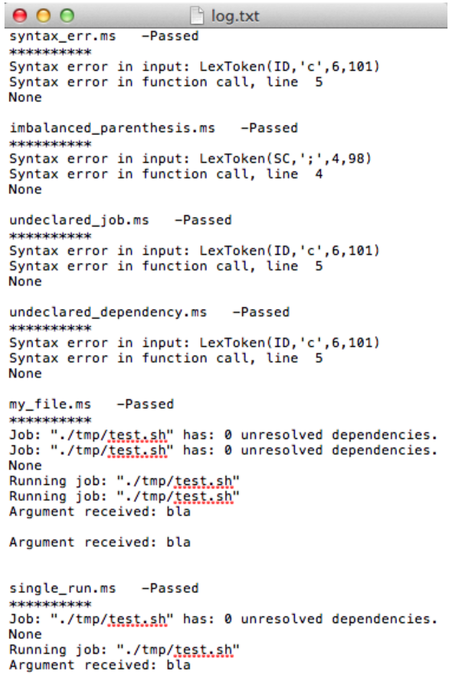


```
TEST mr_job
--------------------------------
Job: "./cut.rb" has: 0 unresolved dependencies.
Job: "./count.rb" has: 1 unresolved dependencies.
Job: "./count.rb" has: 1 unresolved dependencies.
Job: "./count.rb" has: 1 unresolved dependencies.
Job: "./reduce.rb" has: 3 unresolved dependencies.
None
Running job: "./cut.rb"
././all0.txt
././all1.txt
././all2.txt
Running job: "./count.rb"
Running job: "./count.rb"
Running job: "./count.rb"
{"ag":1,"a":1,"sg":2,"ha":1,"rh":1,"sdfhw":1,"dfh":1}
{"ag":1,"a":1,"sg":2,"ha":1,"rh":1,"sdfhw":1,"dfh":1}
{"ag":1,"a":1,"sg":2,"ha":1,"rh":1,"sdfhw":1,"dfh":1}
Running job: "./reduce.rb"
{"ag":1,"a":1,"sg":2,"ha":1,"rh":1,"sdfhw":1,"dfh":1}
{"ag":1,"a":1,"sg":2,"ha":1,"rh":1,"sdfhw":1,"dfh":1}
{"ag":1,"a":1,"sg":2,"ha":1,"rh":1,"sdfhw":1,"dfh":1}
{"ag":3,"a":3,"sg":6,"ha":3,"rh":3,"sdfhw":3,"dfh":3}


Test Passed!
-------------------------------------------------------------------------------

*****************************
Tests Passed: 15 / 15
Check log for test details
*****************************
dyn-209-2-211-141:src rameshswaminathan$
```

Test log



```
syntax_err.ms    -Passed
**********
Syntax error in input: LexToken(ID,'c',6,101)
Syntax error in function call, line  5
None

imbalanced_parenthesis.ms    -Passed
**********
Syntax error in input: LexToken(SC,';',4,98)
Syntax error in function call, line  4
None

undeclared_job.ms    -Passed
**********
Syntax error in input: LexToken(ID,'c',6,101)
Syntax error in function call, line  5
None

undeclared_dependency.ms    -Passed
**********
Syntax error in input: LexToken(ID,'c',6,101)
Syntax error in function call, line  5
None

my_file.ms    -Passed
**********
Job: "./tmp/test.sh" has: 0 unresolved dependencies.
Job: "./tmp/test.sh" has: 0 unresolved dependencies.
None
Running job: "./tmp/test.sh"
Running job: "./tmp/test.sh"
Argument received: bla

Argument received: bla


single_run.ms    -Passed
**********
Job: "./tmp/test.sh" has: 0 unresolved dependencies.
None
Running job: "./tmp/test.sh"
Argument received: bla
```

Demo

MY PROGRAM WORKS PERFECT
EXCEPT WHEN I DO A DEMO
memegenerator.net

Lessons Learned

- Start early
- Pick scope of project wisely
- Constantly reprioritize
- Integrate continuously and often
- Modularize intelligently

Thank you!!

Google

LaTeX