PipelineScript

data workflows made simple
Target Domains

- Natural language processing
- Machine learning
- Network analysis
The Problem

data analysis can be complicated

Data Formats

Programming Languages
The Problem
The Problem

Data analysis can be complicated by different data formats and different algorithm implementations.
The Solution

A new layer of abstraction that allows different data formats and different algorithm implementations to be used interchangeably.
Key Ideas

- Function import from third-party algorithms
- Easy data file read/write
- Parallel processing
Syntax

● Function import

   function f = !"scrape.py"

● File read/write

   "id,date,cost,quant" -> "data.csv"
   table t = @"data.csv"

● Parallel processing

   &get_names("text#.txt") => "names#.txt"
Project Management

- Version control
  Git
- Code hosting & task management
  GitHub: https://github.com/danvegeto/pipelinescript
- Document collaboration
  Google Docs
Development Process

1. Java-side functionality
   *Dan & Burak*

2. Grammar and Translation
   *Pedro & Rachel*

3. Testing system
   *David*
Design Choices

Translator: Python

- Easy
- Compact
- Integrated
- Powerful
Design Choices

Target Language: Java

- Fast
- Cross-Platform
- Manageable
Translator Design

Source code → Lex → Tokenized code → Yacc → Java Code → Java Compiler → Bytecode → JVM → Output

Python Lex-Yacc

Symbol Table

Helper Files → Java Code
Directory Structure

pipelinescript/
  PipelineScript  Java helper classes
  plugins         third-party algorithms
  data            data storage
  tests           test pipelines and testing system
  examples        example pipelines
  doc             documentation
  ply.py          Python translator
  ply.sh          Shell script
Environment Setup

• git clone https://github.com/danvegeto/pipelinescript.git
• Install PLY (Python - lex - yacc) and Java
• Run the self tests by tester.py in the /tests folder
• Run your first pls program by ./pls.sh hello_world.pls
• External plugins -> pipelinescript/plugins/
• Data files (txt,csv) -> pipelinescript/data/
Testing System

- Initial dynamic approach to study corner cases
- Easy updation (addition) of new tests
- Coverage extended to all features
The actual output is ['9']
The expected output is ['9']

The actual output is ['2']
The expected output is ['2']

The actual output is ['foo bar']
The expected output is ['foo bar']
Example 1: File I/O

"this is some example text, and it is exemplary" -> "text.txt"

print @"text.txt"
Example 2: Shell Commands

"1
2
3
4
5
6
7
8
9" -> "data.txt"

function head = !"head -n 3"

head("data.txt") -> "head.txt"

print @"head.txt"
Example 3: Simple Scripts

function words = !"tokenize.py"
function counts = !"count.py"

"the boy and the girl played with the dog and the cat" -> "text.txt"

words("text.txt") -> "words.txt"
counts("words.txt") -> "counts.csv"

print @"counts.csv"
Example 4: Newspaper

function head = !"head"
function scrape = !"newspaper/scrape.py"
function download = !"newspaper/download.py"
function tokenize = !"tokenize.py"
function count = !"count.py"

"http://www.nytimes.com" -> "source.txt"
scrape("source.txt") -> "urls.csv"
head("urls.csv") <- "url#.txt"
&download("url#.txt") => "text#.txt"

print head("text#.txt")
Example 5: Newspaper + NER

function scrape = !"newspaper/scrape.py"
function head = !"head"
function download = !"newspaper/download.py"
function get_names = !"stanford-ner/ner.py"
function count = !"count.py"

"http://www.nytimes.com" -> "source.txt"
scrape("source.txt") -> "urls.csv"
head("urls.csv") <-> "url#.txt"
&download("url#.txt") -> "text#.txt"
&get_names("text#.txt") -> "names#.txt"
count("names#.txt") -> "counts.csv"

print head("counts.csv")
Lessons Learned

- Maintain constant communication
- Plan ahead in detail
- Weekly group meeting are important
- Focus on test driven approach
- Always test basics and push the code
Further Development

- Improve error-checking
- Add support for additional data formats and algorithm languages
- Add additional plugin algorithms
- Implement plugin manager
- Create online platform
THANK YOU