Facilitating Reproducibility After the Fact

Fernando Chirigati

ViDA – Visualization and Data Analysis Lab

NYU Polytechnic School of Engineering

NEW YORK UNIVERSITY

Reproducibility may require a lot of computational effort.

Why?

Too many dependencies!

DATA

PROVENANCE

WORKFLOW

ENVIRONMENT

Too much to do, too little time!

*“authors have complained that the process **requires too much work for the benefit derived**”*

Bonnet et al., SIGMOD Record 2011

*“**Insufficient time** is the main reason why scientists do not make their data and experiment available and reproducible.”*

Carol Tenopir, Beyond the PDF 2 Conference

*“**77%** claim that they do not have **time to document and clean up the code**.”*

Victoria Stodden, Survey of the Machine Learning Community – NIPS 2010

*“It would require **huge amount of effort** to make our code work with the latest versions of these tools.”*

Collberg et al., Repeatability and Benefaction in Computer Systems Research, University of Arizona TR 14-04

From galleryhip.com

Planning for Reproducibility

Scientific Workflow Systems (VisTrails, Taverna, Kepler, …)

Virtual Machines and Containers (VirtualBox, Vagrant, Docker, …)

Configuration Management Tools (Chef, Puppet, …)

… and many others !

But what about ***reproducibility after the fact***?

Again, time-consuming and error-prone!

noWorkflow

Capturing and Analyzing Provenance of Scripts

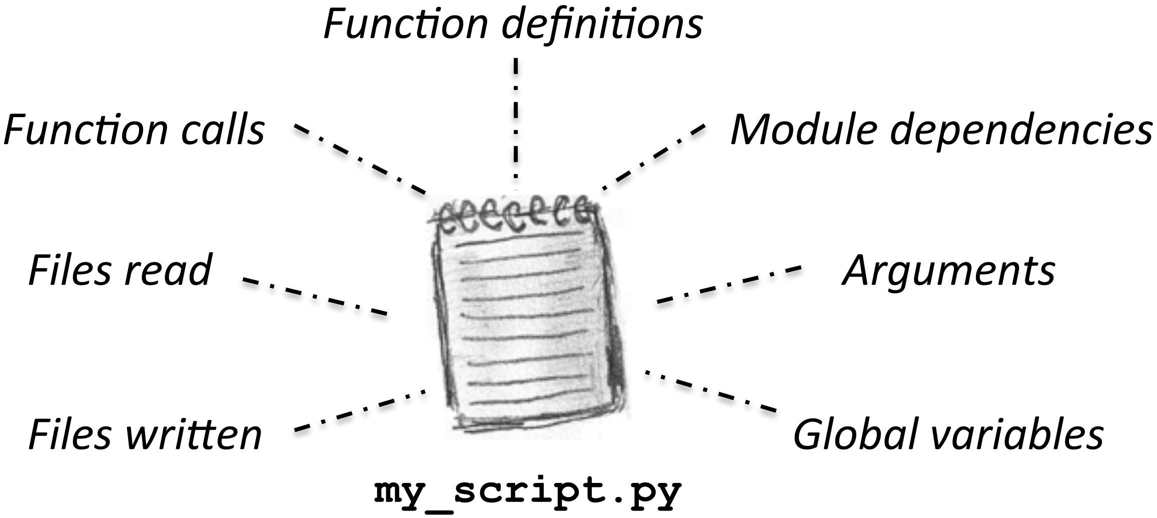
Joint work with: João Felipe Pimentel **(UFF)**
Leonardo Murta **(UFF)**
Vanessa Braganholo **(UFF)**
David Koop **(UMass-Dartmouth)**
Juliana Freire **(NYU)**

NEW YORK UNIVERSITY

Instituto de computação uff

noWorkflow

Transparently captures the *provenance* of a Python script and its various executions (trials)



Function definitions

Function calls

Module dependencies

Files read

Arguments

Files written

Global variables

`my_script.py`

Non-intrusive: no need for user-defined annotations, instrumented environment, or other requirements

Instead of running

```
$ python my_script.py
```

users run

```
$ now run my_script.py
```

That’s it.

Provenance Analysis

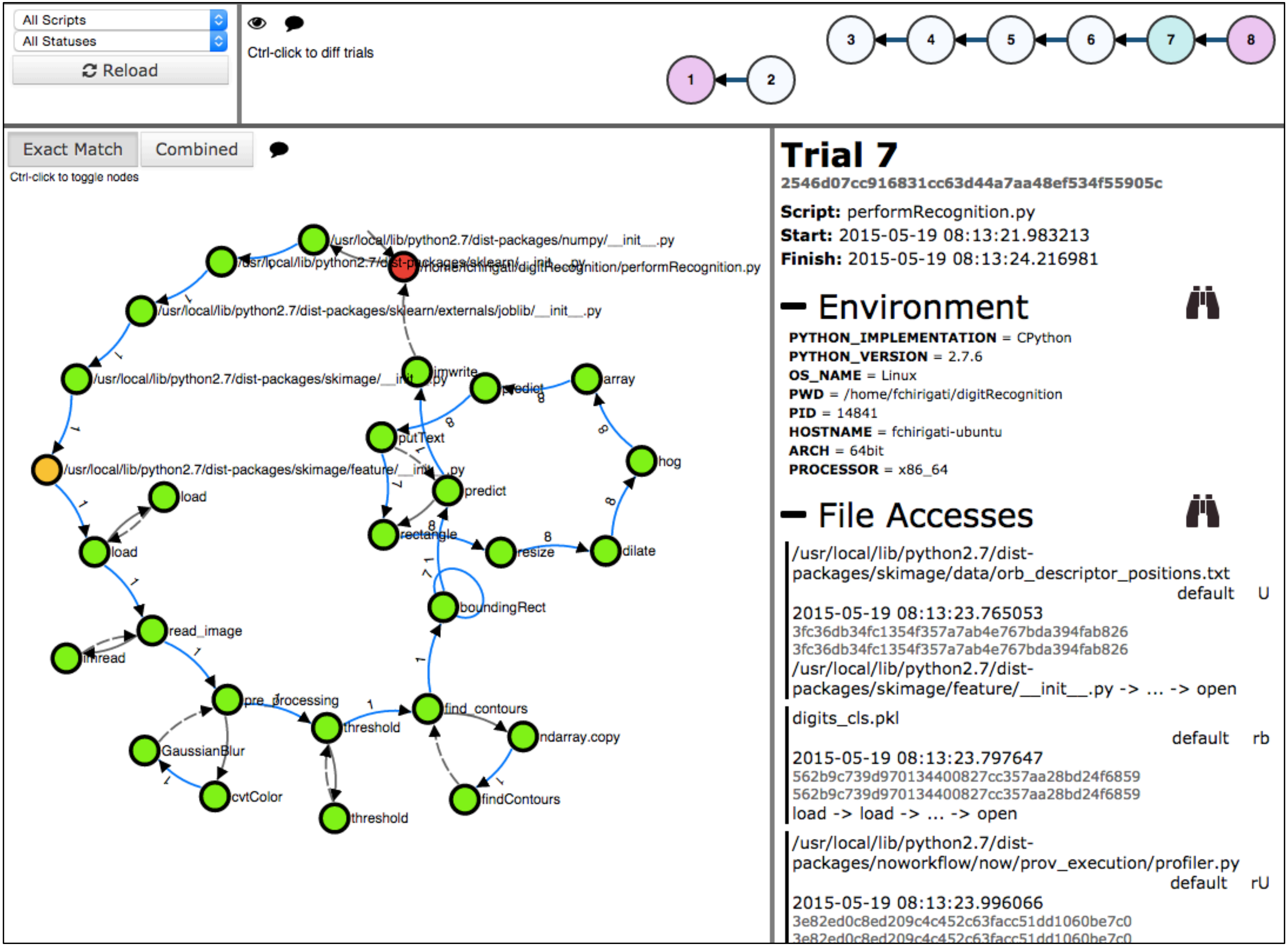
Script Evolution

Diff Analysis

Querying

Interactive Visualization

Support for Analysis in IPython Notebooks


All Scripts
All Statuses
Reload
Ctrl-click to diff trials
3
4
5
6
7
8
1
2
Exact Match
Combined
Ctrl-click to toggle nodes
/usr/local/lib/python2.7/dist-packages/numpy/__init__.py
/usr/local/lib/python2.7/dist-packages/sklearn/__init__.py
/home/fchirigati/digitRecognition/performRecognition.py
/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/__init__.py
/usr/local/lib/python2.7/dist-packages/skimage/__init__.py
imwrite
predict
array
putText
/usr/local/lib/python2.7/dist-packages/skimage/feature/__init__.py
hog
predict
load
load
rectangle
resize
dilate
boundingRect
read_image
imread
pre_processing
threshold
find_contours
ndarray.copy
GaussianBlur
cvtColor
threshold
findContours
1
8

Trial 7

2546d07cc916831cc63d44a7aa48ef534f55905c

Script: performRecognition.py
Start: 2015-05-19 08:13:21.983213
Finish: 2015-05-19 08:13:24.216981

− Environment

PYTHON_IMPLEMENTATION = CPython
PYTHON_VERSION = 2.7.6
OS_NAME = Linux
PWD = /home/fchirigati/digitRecognition
PID = 14841
HOSTNAME = fchirigati-ubuntu
ARCH = 64bit
PROCESSOR = x86_64

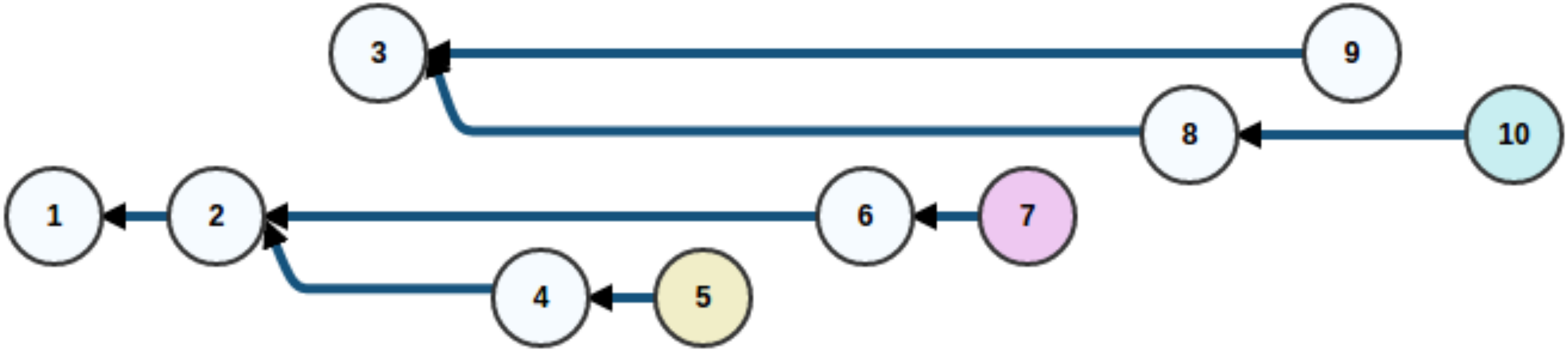
− File Accesses

/usr/local/lib/python2.7/dist-packages/skimage/data/orb_descriptor_positions.txt
default U
2015-05-19 08:13:23.765053
3fc36db34fc1354f357a7ab4e767bda394fab826
3fc36db34fc1354f357a7ab4e767bda394fab826
/usr/local/lib/python2.7/dist-packages/skimage/feature/__init__.py -> ... -> open

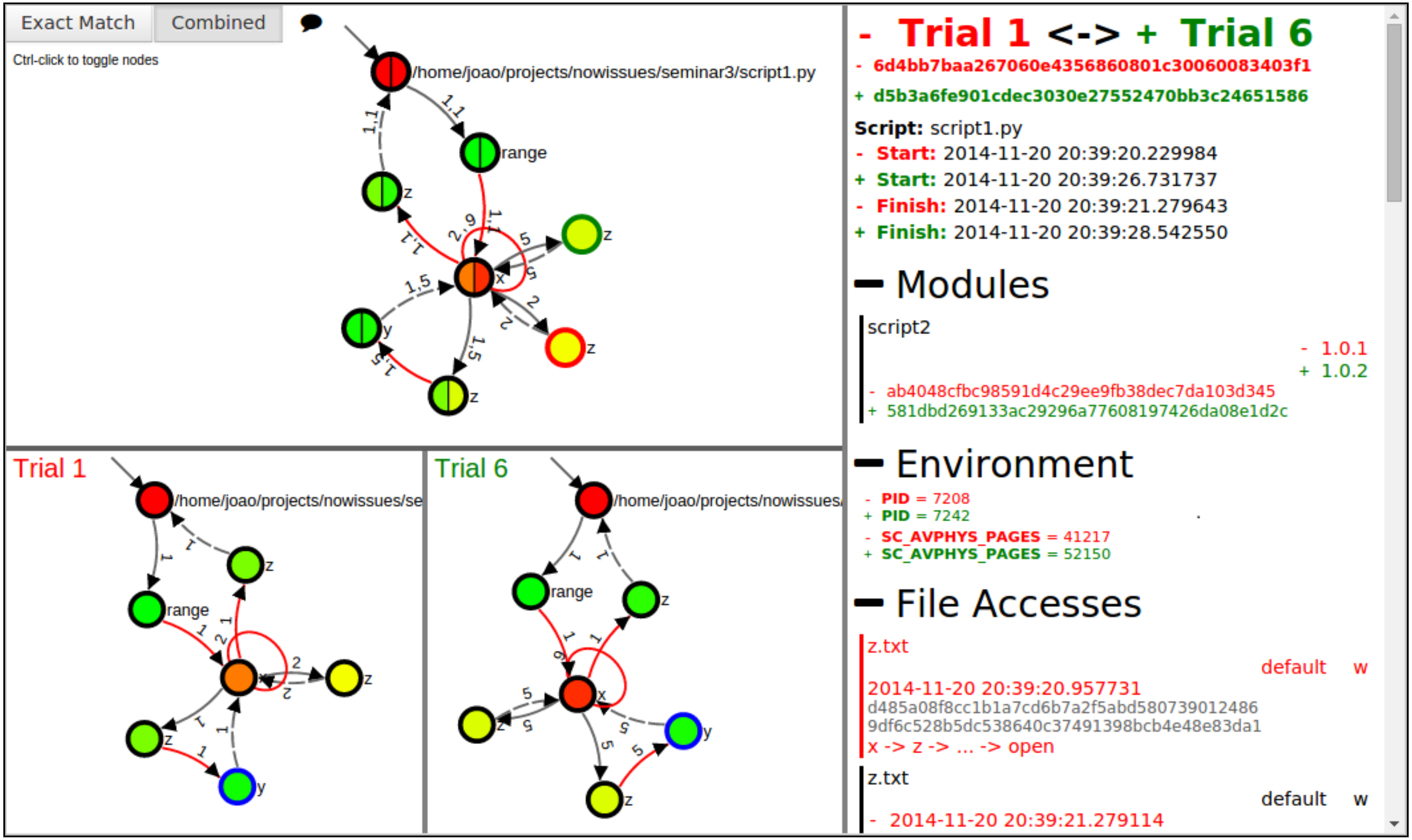
digits_cls.pkl
default rb
2015-05-19 08:13:23.797647
562b9c739d970134400827cc357aa28bd24f6859
562b9c739d970134400827cc357aa28bd24f6859
load -> load -> ... -> open

/usr/local/lib/python2.7/dist-packages/noworkflow/now/prov_execution/profiler.py
default rU
2015-05-19 08:13:23.996066
3e82ed0c8ed209c4c452c63facc51dd1060be7c0
3e82ed0c8ed209c4c452c63facc51dd1060be7c0

Script Evolution



Diff Analysis


Exact Match
Combined
Ctrl-click to toggle nodes
/home/joao/projects/nowissues/seminar3/script1.py
range
z
x
y
Trial 1
/home/joao/projects/nowissues/se
Trial 6
/home/joao/projects/nowissues/

- Trial 1 <-> + Trial 6

- 6d4bb7baa267060e4356860801c30060083403f1

+ d5b3a6fe901cdec3030e27552470bb3c24651586

Script: script1.py
- **Start:** 2014-11-20 20:39:20.229984
+ **Start:** 2014-11-20 20:39:26.731737
- **Finish:** 2014-11-20 20:39:21.279643
+ **Finish:** 2014-11-20 20:39:28.542550

– Modules

script2
- 1.0.1
+ 1.0.2
- ab4048cfbc98591d4c29ee9fb38dec7da103d345
+ 581dbd269133ac29296a77608197426da08e1d2c

– Environment

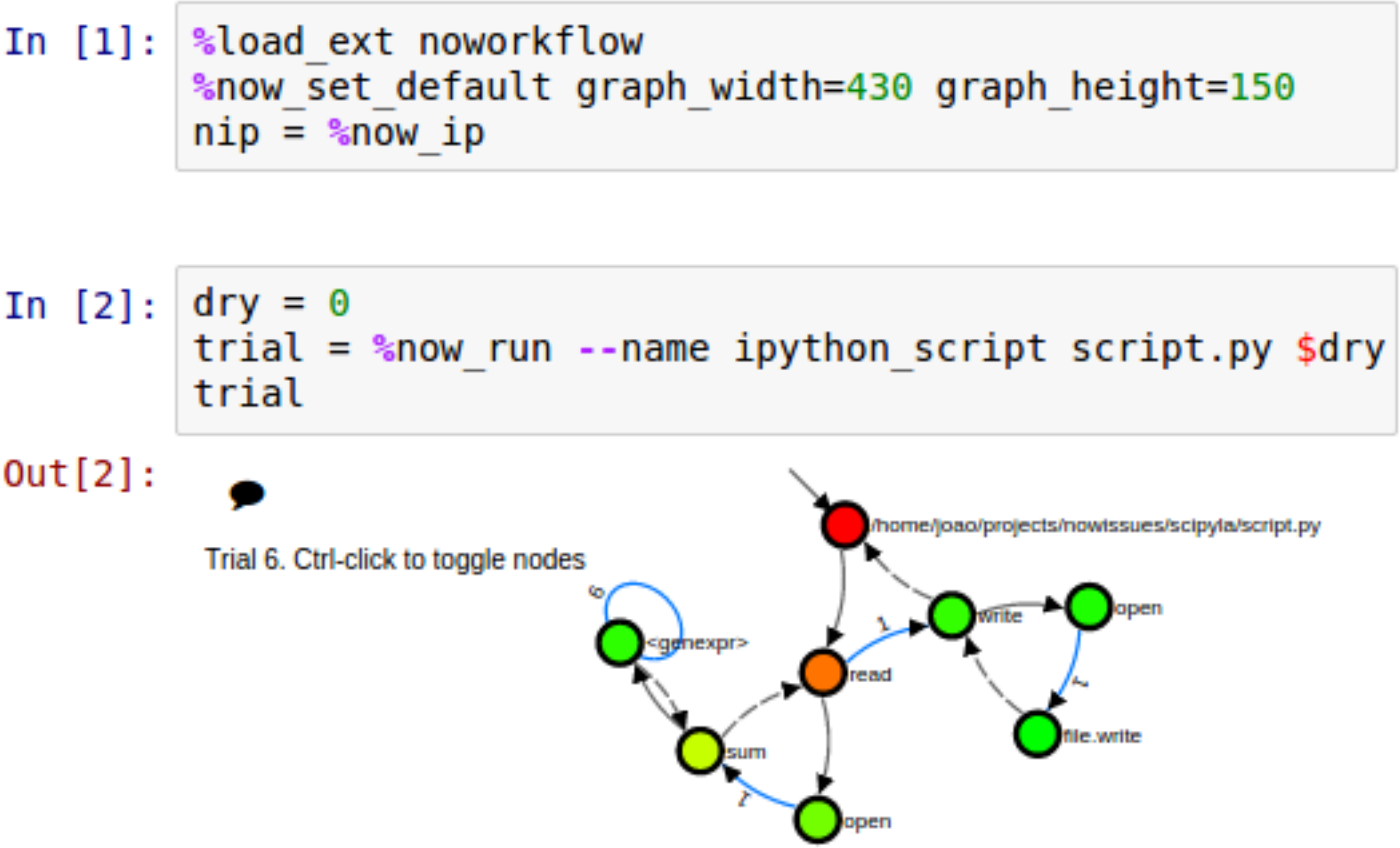
- **PID** = 7208
+ **PID** = 7242
- **SC_AVPHYS_PAGES** = 41217
+ **SC_AVPHYS_PAGES** = 52150

– File Accesses

z.txt
default w
2014-11-20 20:39:20.957731
d485a08f8cc1b1a7cd6b7a2f5abd580739012486
9df6c528b5dc538640c37491398bcb4e48e83da1
x -> z -> ... -> open

z.txt
default w
- 2014-11-20 20:39:21.279114

Provenance Analysis



```
In [1]: %load_ext noworkflow
        %now_set_default graph_width=430 graph_height=150
        nip = %now_ip
```

```
In [2]: dry = 0
        trial = %now_run --name ipython_script script.py $dry
        trial
```

Out[2]:

Trial 6. Ctrl-click to toggle nodes

REPROZIP

PACKING EXPERIMENTS FOR REPRODUCIBILITY

Joint work with: Rémi Rampin
Dennis Shasha
Juliana Freire

NEW YORK UNIVERSITY

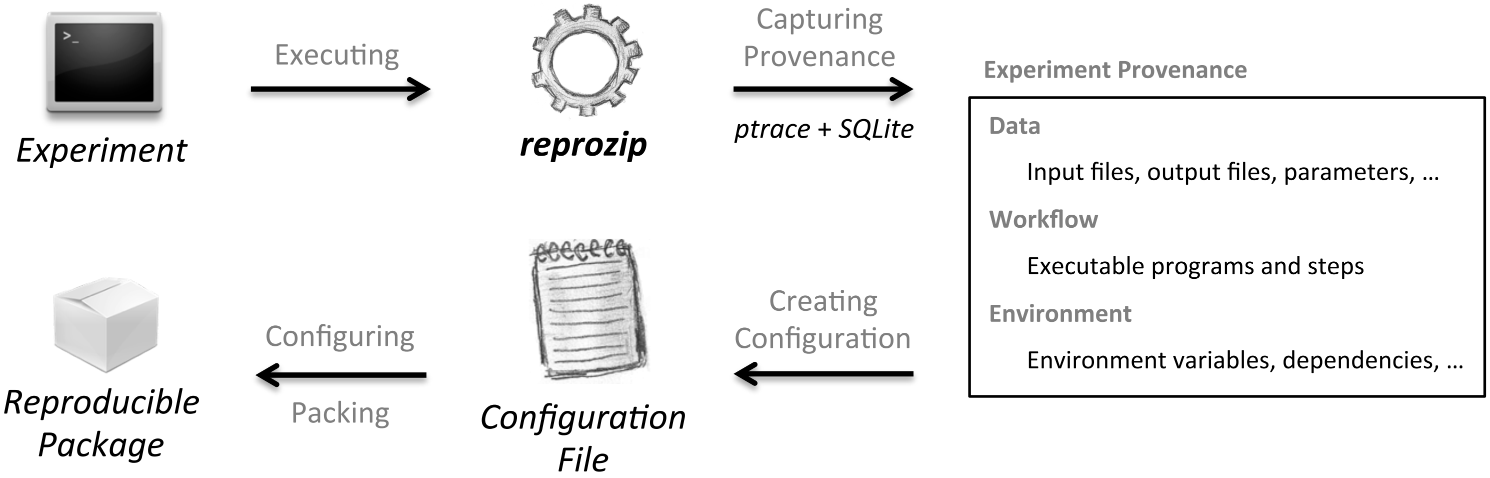
ReproZip is a packaging tool

From reputablemoving.com

From wykop.pl

Packing Experiments

Computational Environment ***E*** (Linux)



Experiment

Executing

reprozip

Capturing Provenance

ptrace + SQLite

Experiment Provenance

Data

Input files, output files, parameters, …

Workflow

Executable programs and steps

Environment

Environment variables, dependencies, …

Creating Configuration

Configuration File

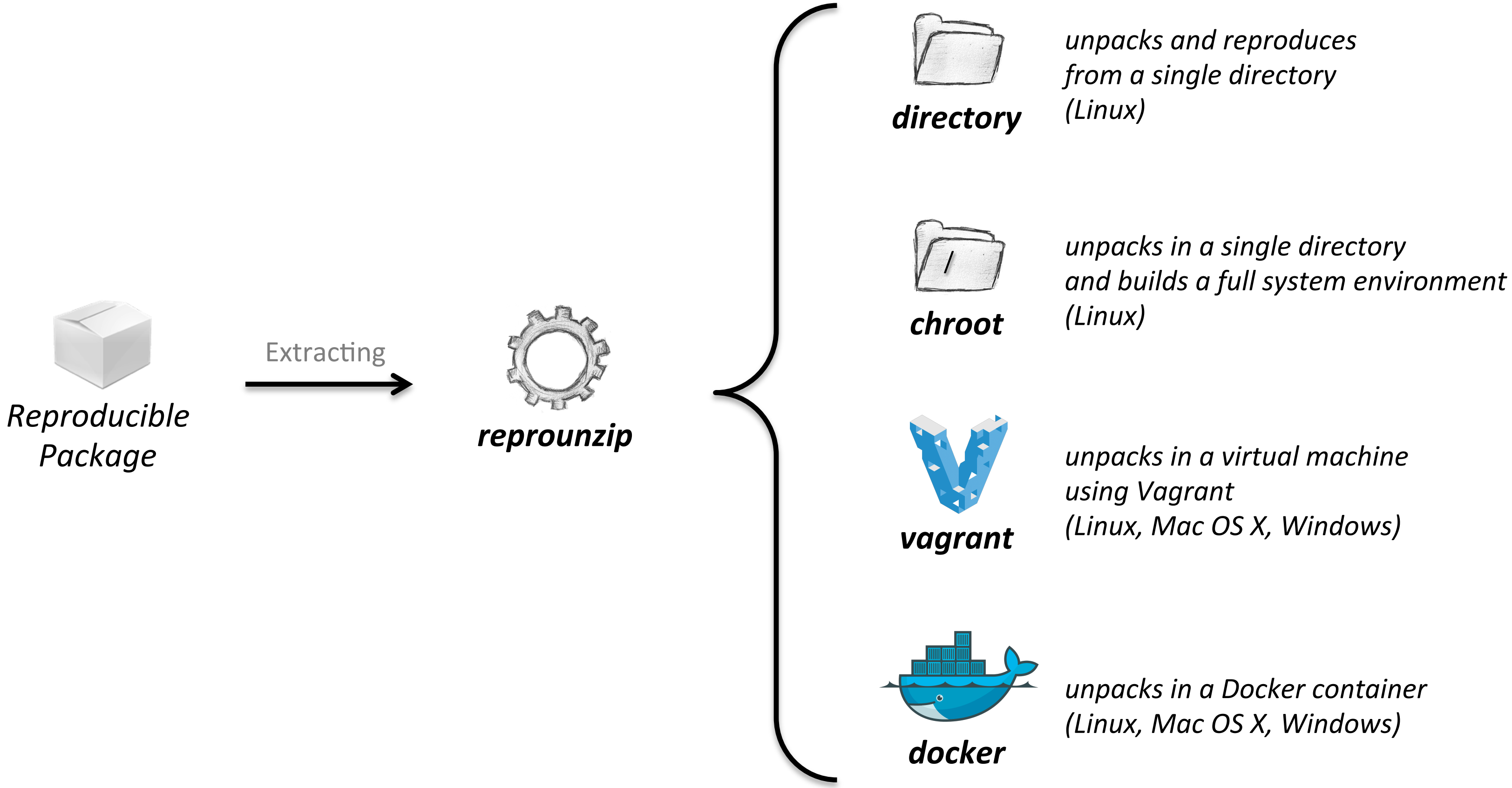
Configuring

Packing

Reproducible Package

Unpacking Experiments

Computational Environment ***E'*** (potentially different than ***E***)



Reproducible Package

Extracting

reprounzip

directory — *unpacks and reproduces from a single directory (Linux)*

chroot — *unpacks in a single directory and builds a full system environment (Linux)*

vagrant — *unpacks in a virtual machine using Vagrant (Linux, Mac OS X, Windows)*

docker — *unpacks in a Docker container (Linux, Mac OS X, Windows)*

Unpacking Experiments

Running an unpacker:

Setting up: *setup*

Replicating results / changing command line parameters: *run*

Changing input files: *upload*

Getting output files: *download*

Natively installing required software dependencies:

installpkgs

News!

ReproZip …

… has been adopted in the Bonneau Lab (NYU)

http://bonneaulab.bio.nyu.edu/

… will be used by the ACM SIGMOD 2015 Reproducibility Review

http://db-reproducibility.seas.harvard.edu/

… will be used by the Information Systems journal http://www.journals.elsevier.com/information-systems/

Try!

noWorkflow Website: *https://github.com/gems-uff/noworkflow*

L. Murta, V. Braganholo, F. Chirigati, D. Koop, and J. Freire: *noWorkflow: Capturing and Analyzing Provenance of Scripts*. In Provenance and Annotation of Data and Processes, vol. 8628, Lecture Notes in Computer Science (LNCS), pp. 71-83, Springer International Publishing, 2015

ReproZip Website: : *http://vida-nyu.github.io/reprozip/*

F. Chirigati, D. Shasha, and J. Freire: *Packing Experiments for Sharing and Publication*. In Proceedings of the 2013 International Conference on Management of Data (SIGMOD), pp. 977-980, 2013

F. Chirigati, D. Shasha, and J. Freire: *ReproZip: Using Provenance to Support Computational Reproducibility*. In Proceedings of the 5th USENIX conference on Theory and Practice of Provenance (TaPP), 2013

Send your feedback and interesting use cases!