L-Vis: Visualizing Language-Level Provenance

Francis Nguyen and Joseph Wonsil

Provenance

“chronology of the ownership, custody or location of a historical object”

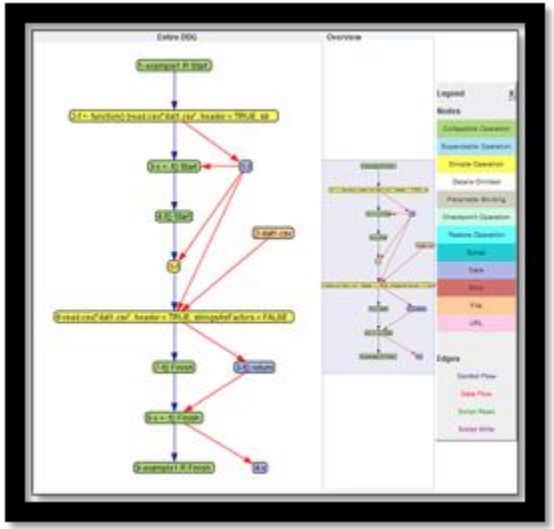
Provenance

“chronology of the ownership, custody or location of a historical object”

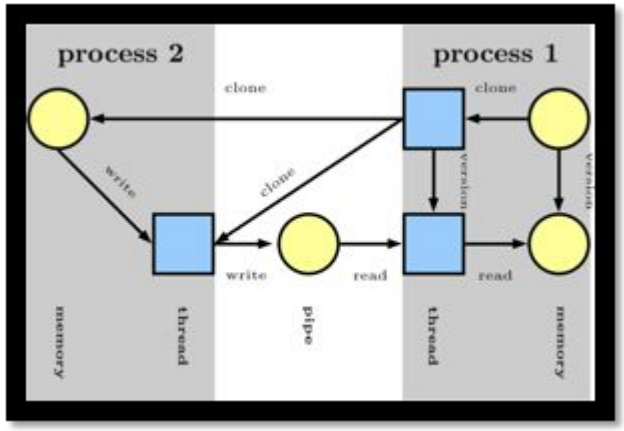
Provides: context, verification

Digital Provenance

Application



Language



System

Application-Level Provenance in Visualization

- Depicts **workflows** (a series of tasks) and **processes**
- Useful in deriving **current state** and **possible choices** or **explored options**
- Can track data-flow through a system, but tends to focus on interaction history or task history
- Useful in visual analytics!

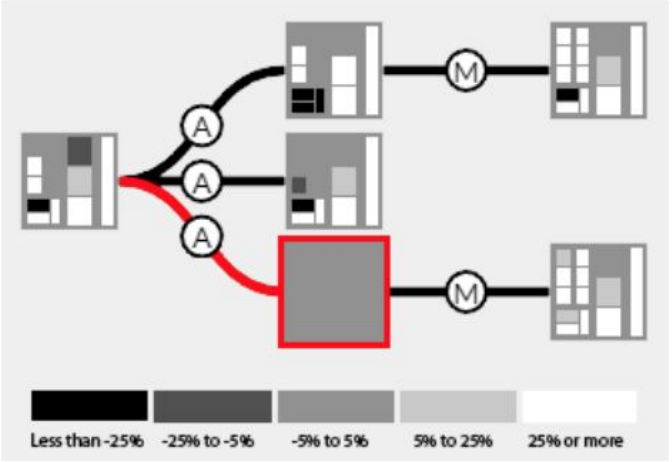
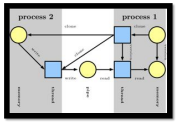


Fig 5. Provenance Tree from *Walch et al. 2018.* LightGuider: Guiding Interactive Lighting Design using Suggestions, Provenance, and Quality Visualization

System-Level Provenance



- Focus has been on **collection, not usage** in meaningful ways
- System level collects **execution traces** at the level of the operating system
- Useful for security and verification (look at deltas in provenance)

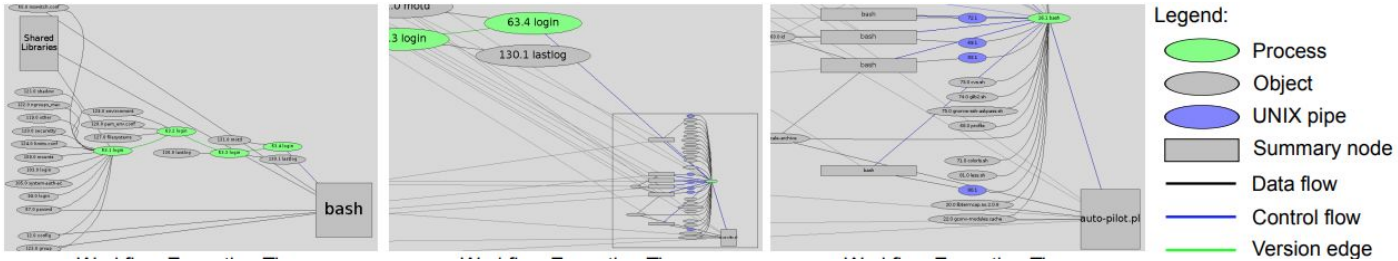
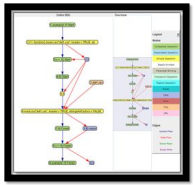


Fig 1. Macko & Seltzer. Provenance Map Orbiter: Interactive Exploration of Large Provenance Graphs.

Our focus: **Level-Language (LL) Provenance**



- Useful for reproducibility in scientific analysis
- Visualization space is relatively unexplored
- Has a *manageable eco-system to leverage (containR, RDT, Dataverse)
- We wanted to make our lives easier when inheriting “grad student code”

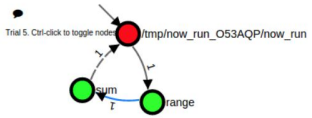
*Still painful but more accessible than system-level provenance tools

```
In [1]: %load_ext noworkflow
        %now_set_default graph.width=392 graph.height=150

In [2]: trial = %now_run script1.py --name tapp
        trial.id
Out[2]: 4

In [3]: size = 5

In [4]: %%now_run --name tapp --out=out_var $size
        import sys
        l = range(int(sys.argv[1]))
        c = sum(l)
        print(c)
Out[4]:
```




```
In [5]: out_var
Out[5]: '10\n'
```

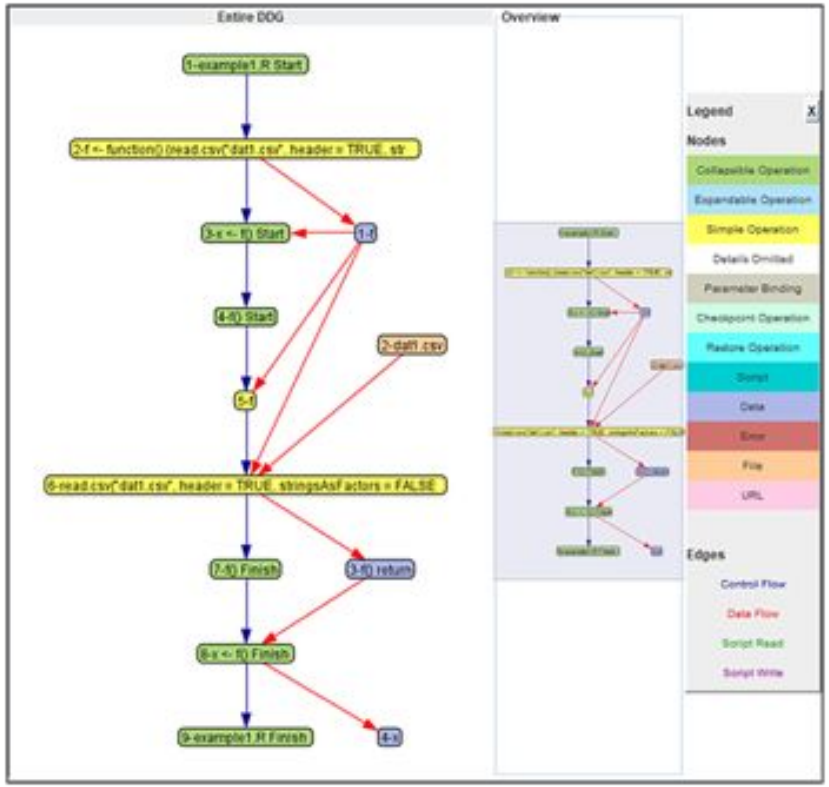
Pimental et al. 2015. Collecting and Analyzing Provenance on Interactive Notebooks: when IPython meets noWorkflow

Figure 5. Provenance collection in notebook using noWorkflow

Provenance Visualization

- If you haven’t noticed yet, the canonical representation of provenance are network graphs — node-link diagrams
- Great for showing structure and relationships
- But of course some problems follow….

Issue: Scale

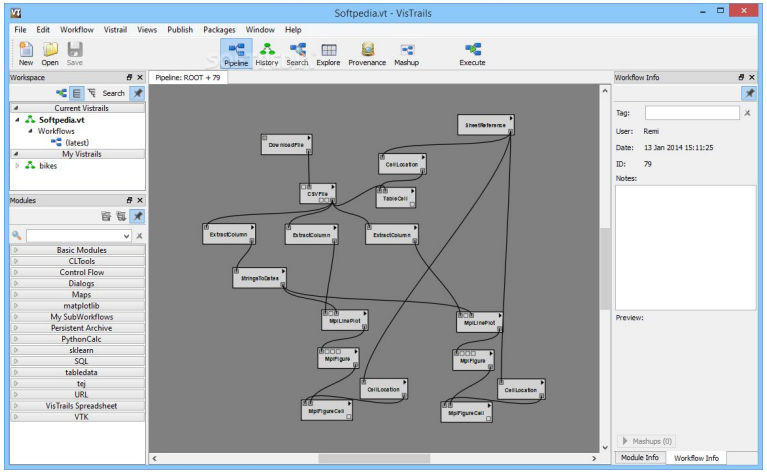


Two Line R Script Provenance Graph

12 Nodes
8 Edges

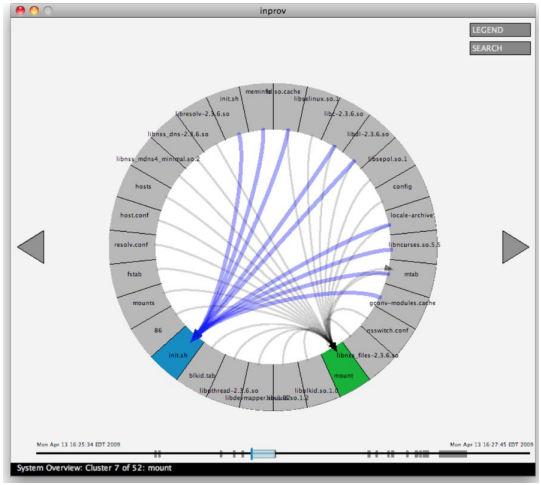
Issue: Robustness of Assessment

- Only a handful of studies quantitatively measure the usefulness of provenance visualization
  - Corlissen et al. 2010, Macko et al. 2011, Borkin et al. 2013
- How do we know network graphs are the best idiom?



VisTrails

https://www.softpedia.com/get/Programming/Other-Programming-Files/VisTrails.shtml



InProv

Borkin et al. 2013

Issue: Novel Usage

- LL-Prov tools visualize provenance to look at differences or bugs
- Often allow users to explore the provenance graph
- Visualize “for the sake” of it

- We are interested in the lens of **program comprehension**
  - How do provenance visualizations aid cognition of programs?
  - How can it facilitate the development of mental models of code?

Past Solutions/Existing Approaches

- Semantic Zoom in network-graphs
- Traditional approach is to only select “relevant nodes”
- Graph summarization nodes — bigger nodes represent clusters
- New graph layout algorithms based on differing metrics
    - Time-based layouts, unsupervised clustering

Data Abstraction - rdtLite (nodes and edges)

- Procedure nodes
- Data nodes
- The agent node
- The environment node
- Library nodes
- Function nodes

- Procedure to Procedure edges
- Procedure to Data edges
- Data to Procedure edges
- Function to Procedure edges
- Library to Function edges

Task Abstraction

We conducted brief informal interviews for what our vis should support in addition to corroborating information from related work.

1. Reverse engineering of design patterns. (Follow crash nodes)
2. Navigate multiple overviews of the system architecture at various levels of abstraction. (Multiple views & data abstraction)
3. Investigate specific contexts. (Semantic zoom)
4. Support goal-directed, hypothesis-driven comprehension. For example, the cause of the bug is **x**.
5. View paths or relationships that led to the current focus. (graph layout)
6. Understand syntactic and semantic relationships between variables and functions. (graph layout)

Scenario

- Final-year PhD student has scripts/analyses on a dataset — upload data/scripts to containR
- New graduate student joins the lab & needs to edit code!
- ContainR and provenance work!
  - The old scripts still run exactly as they did before because they are running in a container.
- L-Vis helps the student learn **how** the old plots were created
  - Allows them to cleanly insert new code into the analysis
  - Uses the **detail view** of the plot node in order to view all the code-snippets and variables related to generating the plot
- Dynamic what-if analysis
  - To begin building a mental model of the plot generation code, the new student uses the **detail view** and **changes variables** in order to see how the plot changes dynamically

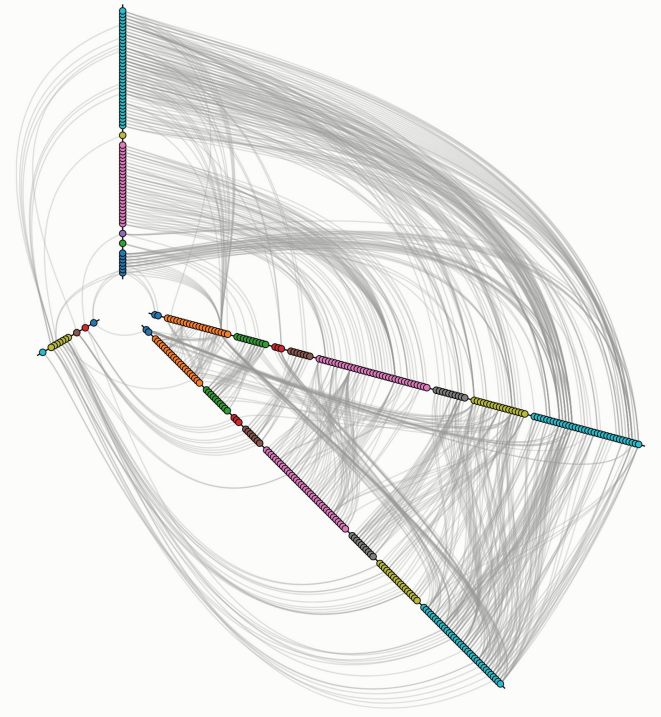
Our solution

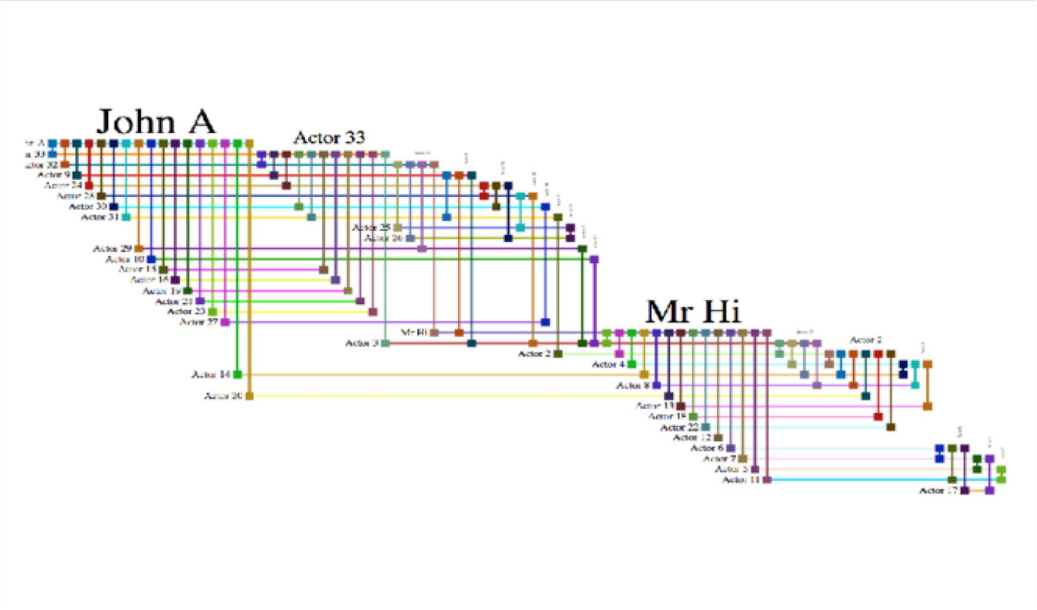
Data abstractions —

- We choose to filter all nodes and explicitly show **crash nodes** - these are procedures where two (or more) data nodes are both used and a new data node is created.
- Through visualizing crash nodes, the focus of the visualization becomes the path of data through the scripts and how different inputs may interact / depend on each other.

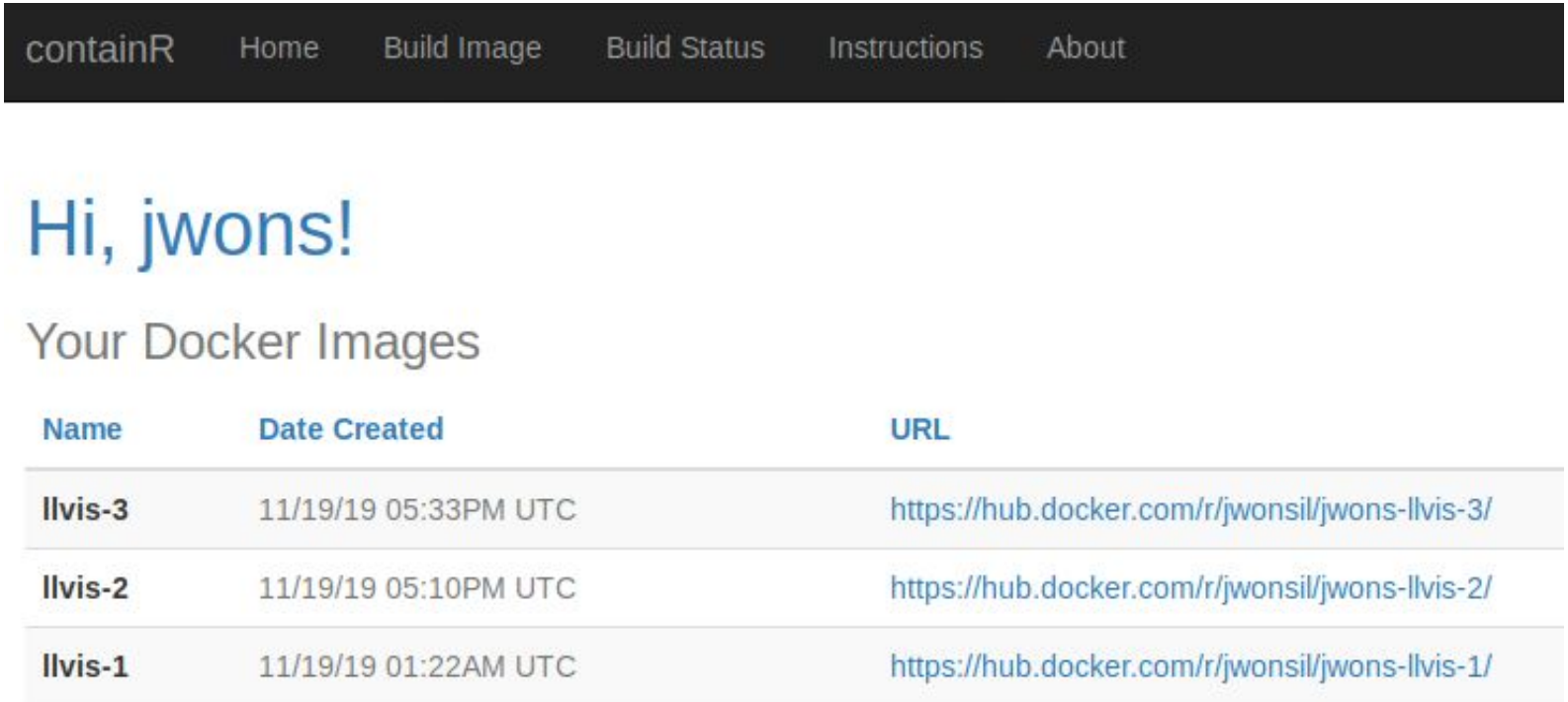
Our solution

Exploratory visualizations — Graph Layouts





Platform: containR



In the future…

- Finish the prototype and add it to the containR workflow
- Optional filters of data to toggle node-link diagrams on/off
- Design study to derive more formal tasks from users and get iterative feedback
- (informal) Quantitative study to compare L-Vis to other LL-prov tools

L-Vis: Visualizing Language-Level Provenance