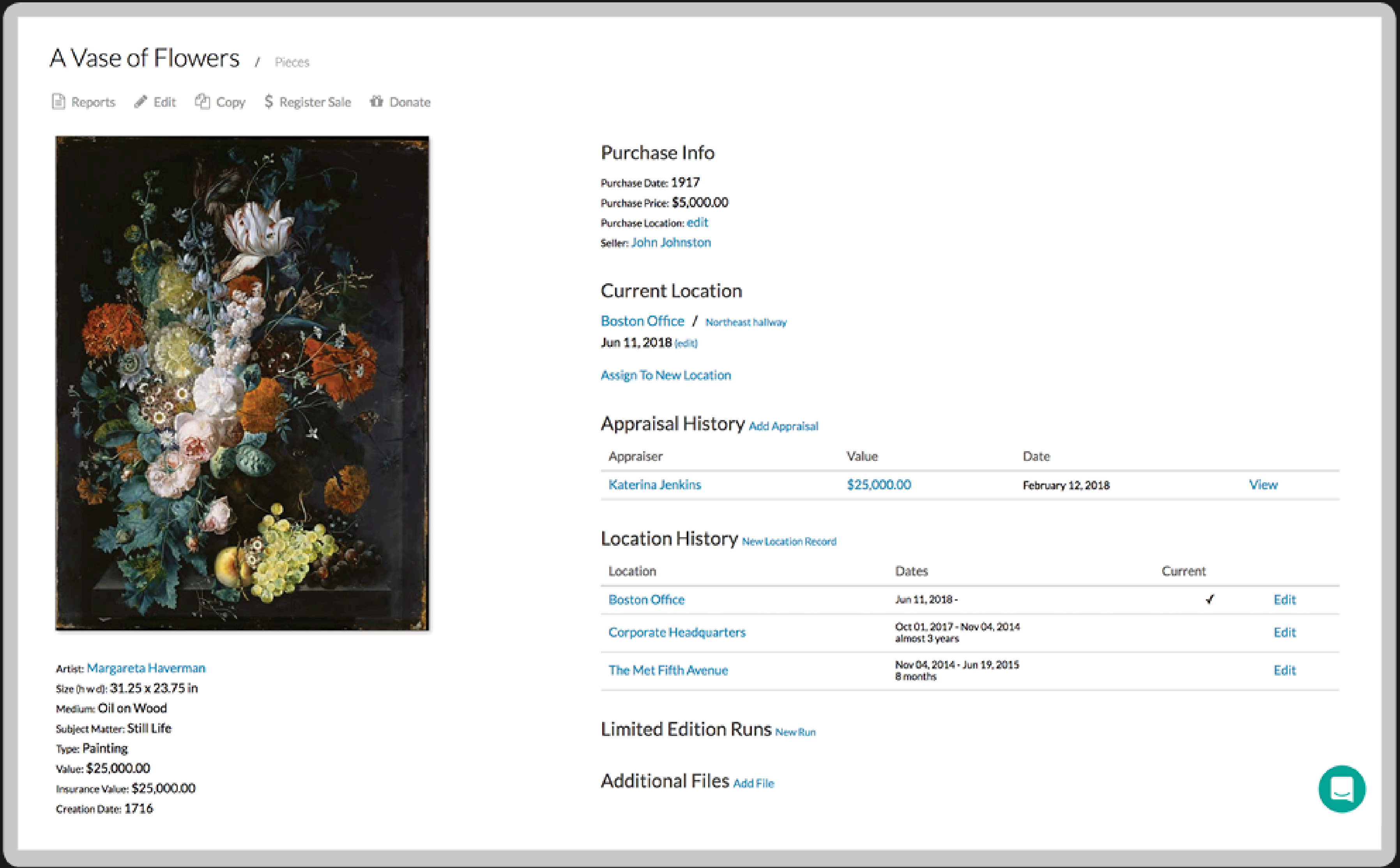
System-level provenance tracers

by Samuel Grayson, Reed Milewicz, Daniel S. Katz, and Darko Marinov

Takeaways

1. Provenance is useful for computational scientists
2. Current state-of-the-art not practical use
3. But there is hope

What is provenance?



A Vase of Flowers / Pieces

Reports Edit Copy $ Register Sale Donate

Artist: Margareta Haverman
Size (h w d): 31.25 x 23.75 in
Medium: Oil on Wood
Subject Matter: Still Life
Type: Painting
Value: $25,000.00
Insurance Value: $25,000.00
Creation Date: 1716

Purchase Info

Purchase Date: 1917
Purchase Price: $5,000.00
Purchase Location: edit
Seller: John Johnston

Current Location

Boston Office / Northeast hallway
Jun 11, 2018 (edit)

Assign To New Location

Appraisal History Add Appraisal

Appraiser	Value	Date	
Katerina Jenkins	$25,000.00	February 12, 2018	View

Location History New Location Record

Location	Dates	Current	
Boston Office	Jun 11, 2018 -	✓	Edit
Corporate Headquarters	Oct 01, 2017 - Nov 04, 2014 almost 3 years		Edit
The Met Fifth Avenue	Nov 04, 2014 - Jun 19, 2015 8 months		Edit

Limited Edition Runs New Run

Additional Files Add File

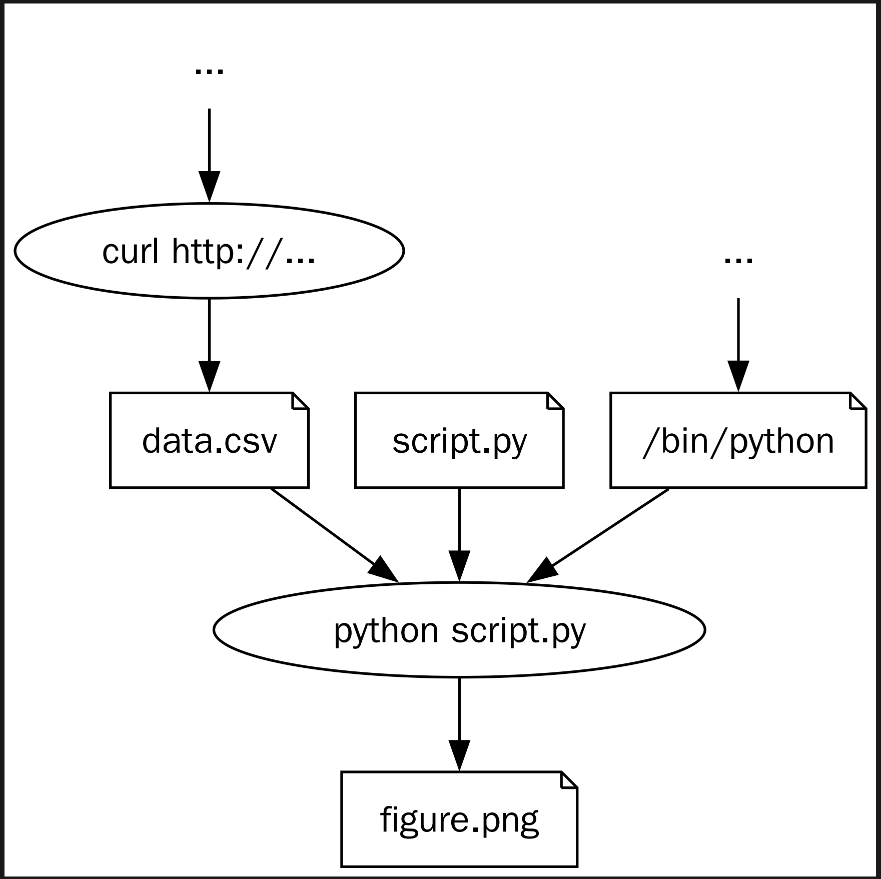
What is computational provenance?

1. Process by which a comp. artifact (often a file) was generated
2. The inputs to that process
3. The provenance of those inputs (recursively)

Expressible as a graph whose nodes are processes or files

Comp. Provenance Example

```
curl http://...
python script.py
```



What is it good for?

- Record/replay reproducibility (*)
- What parameter of X did this output use? Digital notebook
- How does the ancestry of these two objects differ?
- Create a Spack/Nix/Guix/Makeflow automatically (*)

What is the state of the art?

How to collect provenance data?

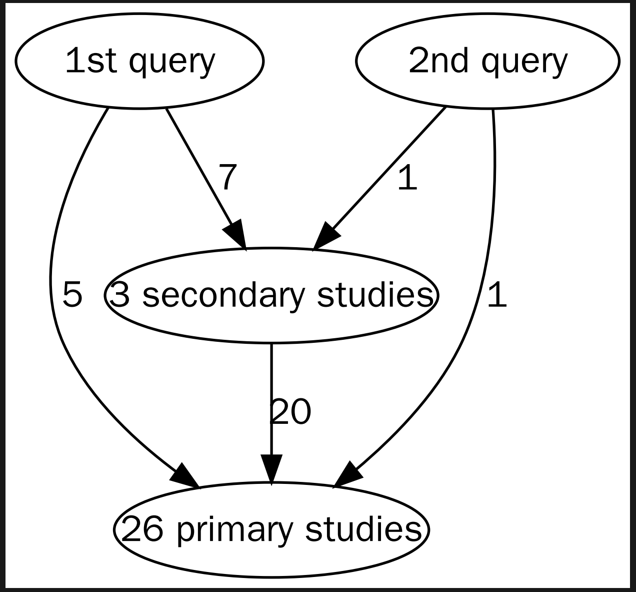
- Language-level (rdtLite)
- Switch to workflows
- System-level (*)

How to collect provenance data?

- Language-level (rdtLite)
- Switch to workflows
- System-level (*)

Rapid review

Search queries: "Computational provenance" & "System-level provenance"



Feasibility study

Linux sys-level prov tracers

Provenance tracer	Strategy	Modified kernel?	Privileges required?
PASSv2, CamFlow, …	Modify kernel	yes	yes
SPADE, eBPF, …	Built-in auditing	no	yes
Sciunit, ReproZip, …	User tracing	no	no

Linux sys-level prov tracers

Provenance tracer	Strategy	Modified kernel?	Privileges required?
PASSv2, CamFlow, …	Modify kernel	yes	yes
SPADE, eBPF, …	Built-in auditing	no	yes
Sciunit, ReproZip, …	User tracing	no	no

Performance analysis

Selected provenance tracers

- RecProv: Could not locate source code or authors
- OPUS: Could not reproduce at all (Py 2.7)
- CDE: Sometimes crashes
- PTU/Sciunit: Sometimes out-of-mem, false negatives
- ReproZip: Works
- CARE: Works

Related tools

- Strace: ptrace syscall tracking
- RR-debugger: ptrace record-replay
- fsatrace: LD_PRELOAD file tracking

Benchmarks used by prior works

Prior pubs.	This work	Benchmark
12	Yes	HTTP servers/clients
9	No	Web browsers
6	Yes	FTP servers/clients
5	Yes	Un/archive
5	Yes	BLAST
4	Yes	Postmark
3	Yes	lmbench

Performance

Benchmark	Native	fsatrace	CARE	strace	RR	ReproZip
	None	Lib. interp.	Ptrace	Ptrace	Ptrace	Ptrace
BLAST	0	0	2	2	93	8
Tar Unarchive	0	4	44	114	195	149
Python import	0	5	85	84	150	346
VCS checkout	0	5	71	160	177	428
Compile w/Spack	0	-1	119	111	562	359
Postmark	0	2	231	650	259	1733
cp	0	37	641	380	232	5791
Others not shown	…	…	…	…	…	…
Geometric mean	**0**	**0**	**45**	**66**	**146**	**193**

Discussion

Security	Comp. sci.
Un-circumventable at all costs	Can assume the codes are not intentionally deceptive
Assume we have root on baremetal	Don't have root on large-scale computers
Use servers (like HTTP) as benchmarks	Use workflows and comp experiments as benchmarks

Why aren't prov tracers used

- Require root
- Fragile/crashy
- Hard to install
- Too slow
- Happy with existing system

But there is hope!

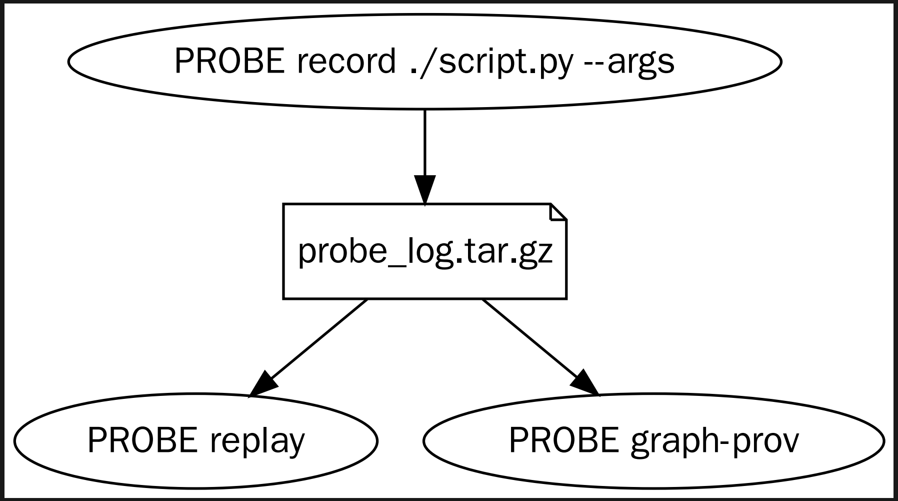
- Computational is niche
- Lib. interposition understudied in prior work
- Common set of benchmarks

Conclusion

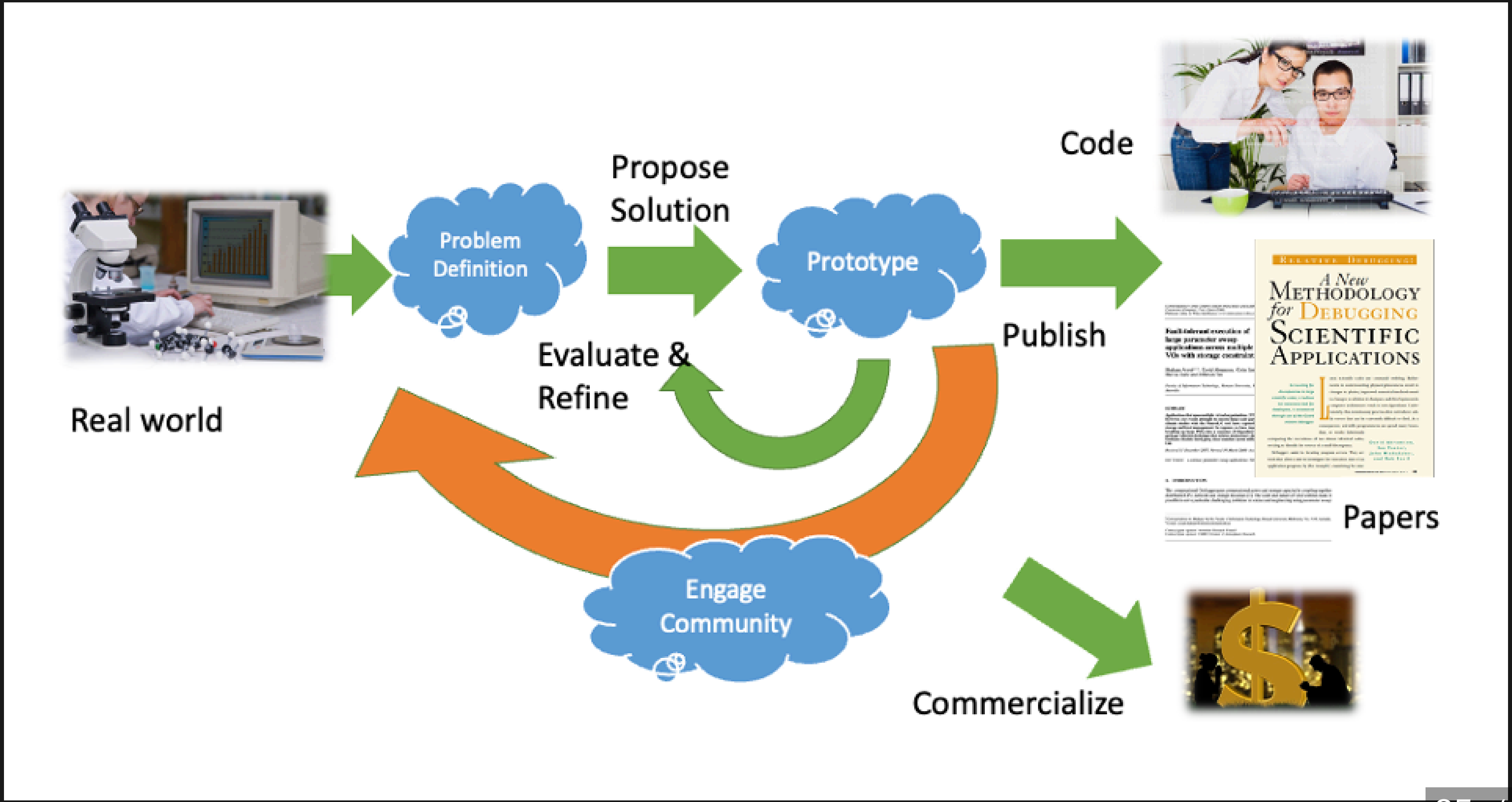
1. Provenance is useful for computational scientists
    - Record/replay, digital notebook, prov diff, create Nix package
2. Current state-of-the-art not practical use
    - must not modify kernel; must be rootless; must not crash; must be fast
3. But there is hope
    - lib. interposition

PROBE: Provenance for Replay OBservation Engine

https://github.com/charmoniumQ/PROBE



Difficulty of translational CS



Invitation for collaboration

- Using record/replay (*)
- Using provenance to generate Guix/Nix/Spack/Makeflow (*)
- Consuming provenance in other ways (WRROC *, *)
- Using the benchmark set
- sam@samgrayson.me
- https://github.com/charmoniumQ/PROBE