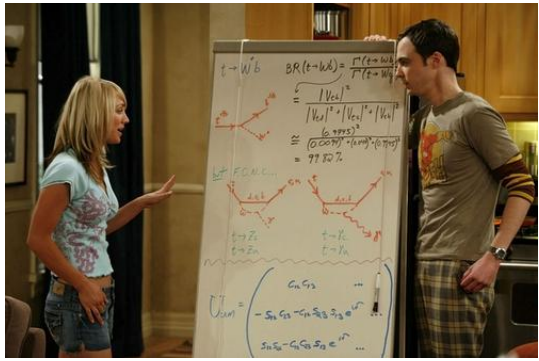
Towards Complete Tracking of Provenance in Experimental Distributed Systems Research

Tomasz Buchert
Lucas Nussbaum
Jens Gustedt

Validation in (Computer) Science



- Two classical approaches for validation:
  - Formal: equations, proofs, etc.
  - Experimental, on a scientific instrument
- Often a mix of both:
  - In Physics
  - In Computer Science
- Quite a lot of formal work in Computer Science
- But also quite a lot of experimental validation
  - Distributed computing, networking $\rightsquigarrow$ testbeds
  - Language/image processing $\rightsquigarrow$ evaluations using large corpuses

However . . .

- Experiments are often unreproducible
- It is hard to build on existing research:
  - Experimental details are not published
  - Important factors are omitted
  - Experiments are prepared and run in an *ad hoc* manner

Several techniques were created to approach these problems

Provenance

- = information about origins and/or a chain of custody of an object
- Found another meaning in computing and science as a **representation of origin and transformation of a given data object during computation**
- Provenance should enable one to answer questions such as:
  - How was that data produced?
  - When was that data produced?
  - Which nodes were involved?
- It is a successful tool in many sciences
  medicine, astrophysics, chemistry, etc.

Experimental DS research and provenance

Provenance could help improve the state of experiments in DS research:

- to document otherwise under-documented experiments
- to better understand their progress
- to track their evolution
- to make them accessible
- to justify scientific conclusions

But:

- What does *provenance* mean in the context of DS research?
- Are the existing tools (for other sciences) suitable?
- Are there specifics about experiments in DS research?

This talk

This work makes three contributions:

1. **an analysis of provenance tracking** in various domains
2. **a new classification of provenance** into three types:
   - the provenance of data
   - the provenance of description
   - the provenance of process
3. **the proposed design for a system** to collect these types of provenance

Classifications of provenance

The primary classification of provenance is into[1]:

- **prospective** provenance - obtained **before** running anything
- **retrospective** provenance - obtained by running or **after** the experiment

Others differentiate on the level where provenance operates[2]:

- **Level 0** – abstract experiment description
- **Level 1** – instantiation of a platform
- **Level 2** – instantiation of data inputs
- **Level 3** – run-time provenance

[1] **DBELM2007**; **DavFr2008**.
[2] **BarDi2008**.

Provenance in general computing

Some common approaches can provide provenance information:

- documentation (also **literate programming**)
- version tracking (via version control systems)
- software repositories with historical features
- instrumentation and monitoring
- logging (possibly non-linear)

In general, *any* information on how computation was performed contributes to provenance

Provenance in scientific workflows

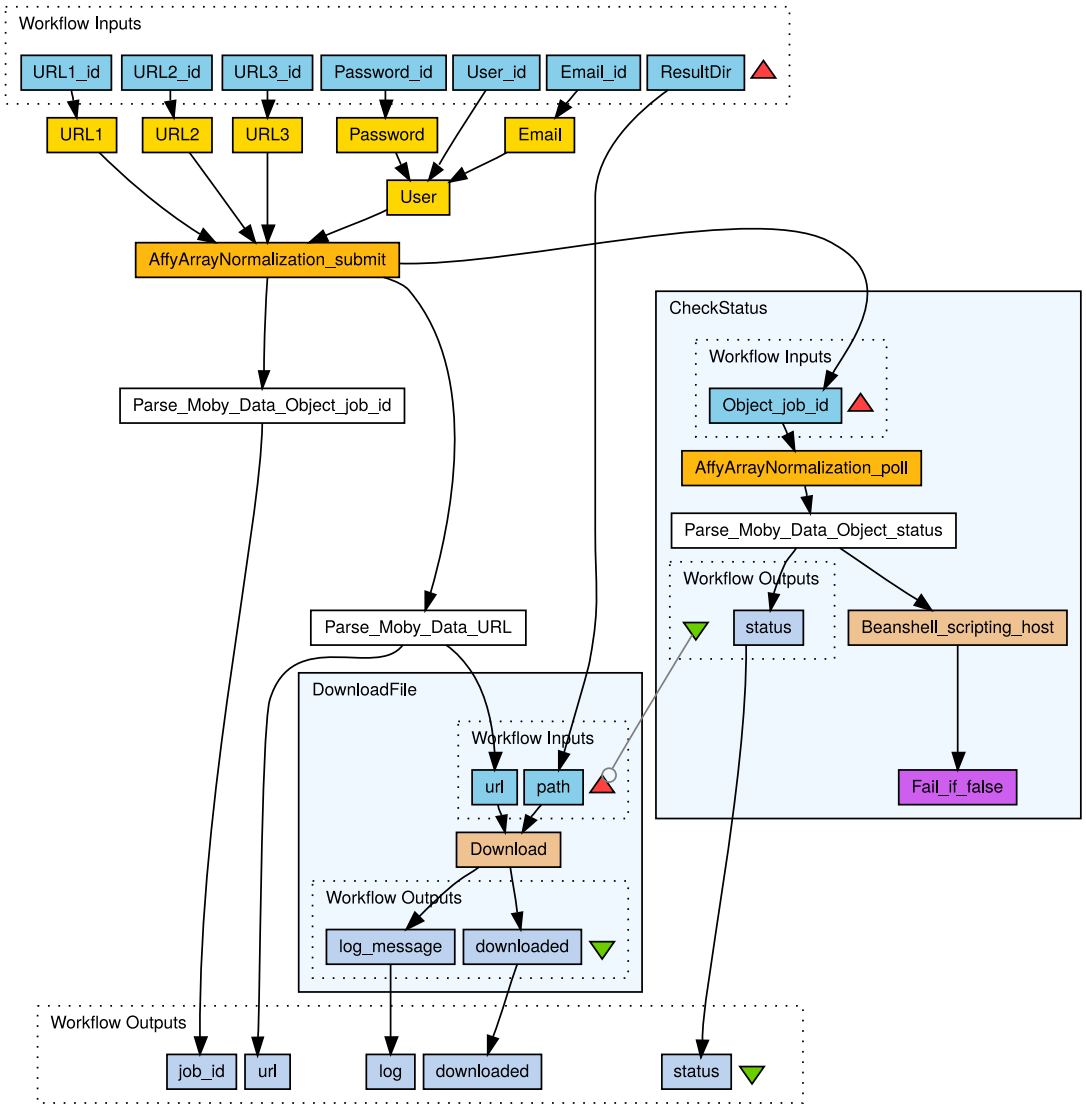
Data provenance:

- the most common type of provenance
- nearly a synonym for provenance, in practice

Scientific workflows are well-known tools to collect and store it:

- they describe the set of tasks needed to carry out a computational process (usually as a DAG)
- they try to hide platform details
- can be used (to some extent) without technical expertise

Scientific workflows (example)



Provenance in experimental DS research

The recommended way to run complex experiments is to use **experiment management tools** which:

- provide easier abstractions to work with
- offload difficult and tedious tasks
- monitor the experiment
- distribute files and collect data

However, **provenance tracking** is an almost non-existing feature[3]

[3] **BuRNR2014**.

Why a new classification?

- The existing classifications miss important aspects:
  - the runtime behavior in time and space
  - the evolution of experiment description (over the development of the experiment)
  - the questions involving them and data provenance
- Experiments in DS research[4] are less data-centric than control-centric:
  - most of the time is spent controlling the platform
  - data collection can be often postponed
  - data is analyzed later in bulk

[4] we focus here on *in-situ* experiments – which excludes simulations

Three types of provenance

We propose a new classification of provenance into:

- **provenance of data** (as in scientific workflows)
- **provenance of description** (platform specification, the textual (possibly source code) or graphical representation of the experiment)
- **provenance of process** (runtime and causal information)

Three types of provenance

We propose a new classification of provenance into:

- **provenance of data** (as in scientific workflows)
- **provenance of description** (platform specification, the textual (possibly source code) or graphical representation of the experiment)
- **provenance of process** (runtime and causal information)

FAQ: *They are all data. So aren't they all addressed by provenance of data?*

- *Provenance of description* operates at another level, and does not require the execution of the experiment
- Each type of provenance has a different (ideal) representation
  $\rightsquigarrow$ more appropriate representation and visualization ; more efficient storage and access

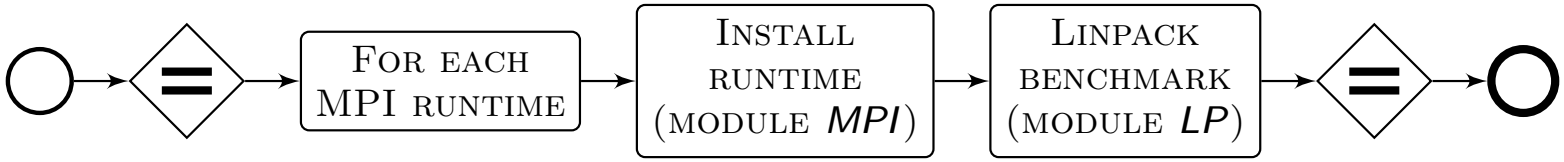
New classification and the existing ones

Our classification intersects with the previous ones:

Name	Moment of collection	Level
Data	Retrospective	L3 (also L2, if present)
Description	Prospective	L0 & L1
Process	Retrospective	L3

Example of an experiment

The experiment compares different MPI runtimes using the Linpack benchmark.



Provenance questions

A common way to evaluate and design provenance systems is to test which questions can be answered by them.

The examples of questions are:

- **data** – *What were the results of a benchmark?*
- **description** – *Who is the author of module X?*
- **process** – *What is the Gantt diagram of the experiment?*

Examples of questions involving many types are:

- **data & description** – *Did the system specification reflect reality?*
- **data & process** – *What modules executed at node X?*
- **description & process** – *Who authored a change that caused X to fail?*

Design of a provenance system

In what follows, we make the following assumptions:

1. the experiment is *in-situ* and in the domain of DS research
2. the experiment description is a control-flow based
3. data processing does not constitute a large fraction of the experiment execution

The following design is proposed as an extension of our experiment management tool, XPFlow[5,6].

[5] **BuNuG2014**.
[6] `http://xpflow.gforge.inria.fr/`

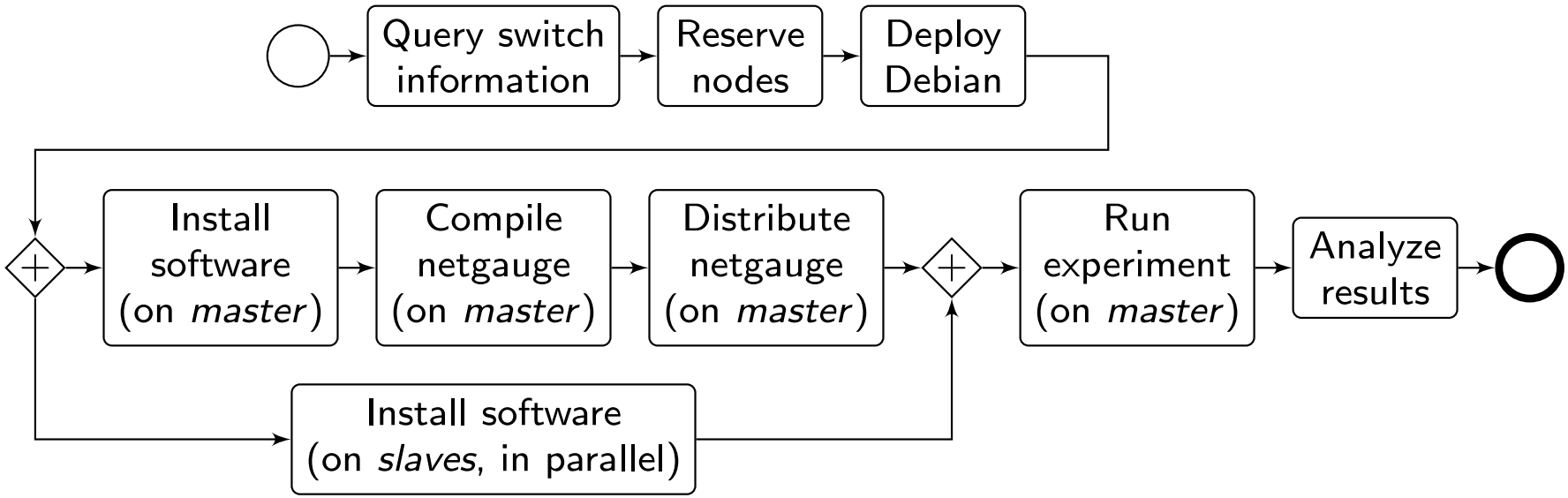
Experiments as control-flows

There are 2 main concepts in our control-flow based approach:

- activities – low-level building blocks of experiments such as:
  - command execution
  - software installation
  - data collection
- workflow patterns – aggregating other activities and patterns:
  - sequential execution
  - parallel execution
  - efficient command execution on multiple nodes

Contrary to scientific workflows, we use a **plain-text DSL**.

Experiments as control-flows



Provenance of experiment data

In our design, the **data provenance**:

- can be stored in a mostly unstructured way
- for example, using a key-value store
- with meta-data (timestamp, etc.)
- and named links to other related data

For example, in our MPI experiment, it includes:

- benchmark results
- runtime configuration of nodes
- other raw collected data (such as monitoring)

Provenance of experiment description

Our approach to control of experiments is modular via reusable workflows

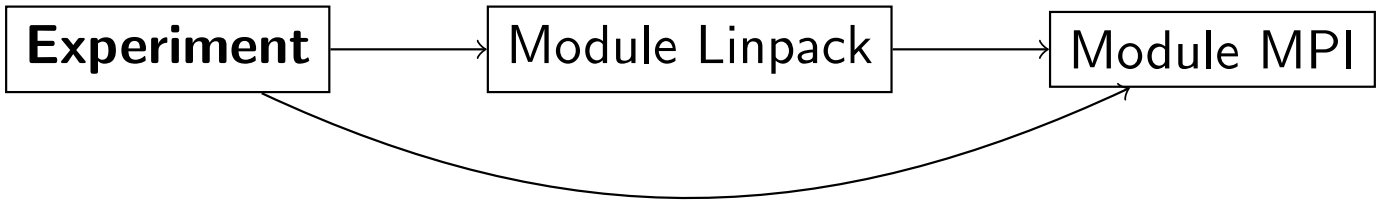
We extend this to **versioned workflows** with the **Git** version control system

Versioned workflows are:

- represented as a consistent *tag* in a repository
- referenced by tag name by other workflows
- stored in generally distributed repositories

A Full dependency tree can be obtained by traversing all dependencies

It implies that **provenance of description** is a **directed, acyclic graph**



Provenance of experiment process

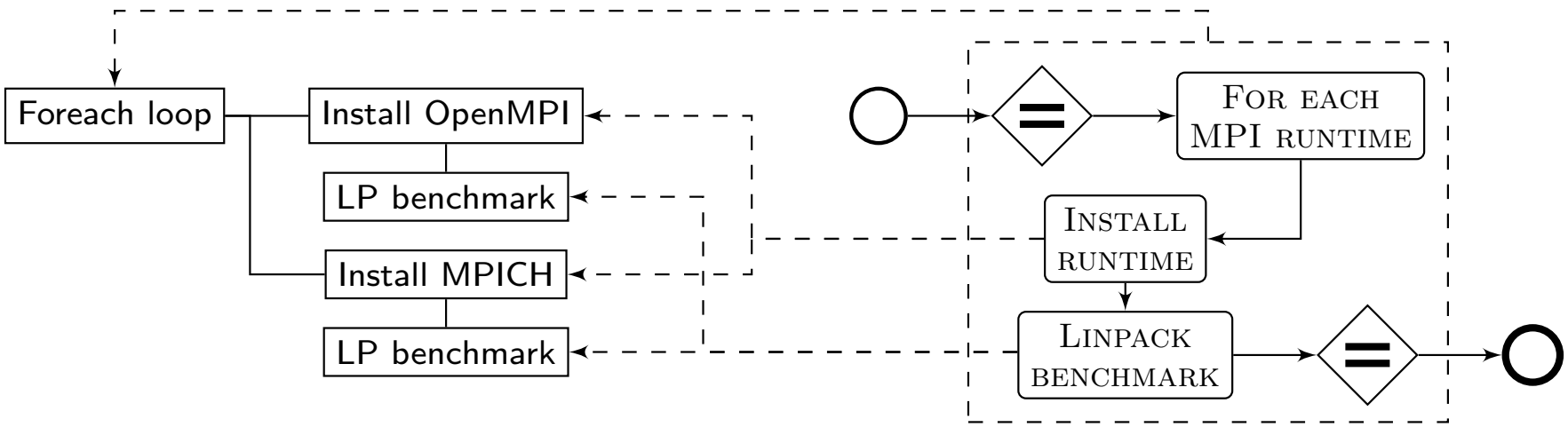
Provenance of process is based on the workflow structure of experiments.

The **hierarchical structure** of workflows implies a tree-like log.

Each entry in the tree records:

- timing information (start, finish, etc.)
- metadata associated with activities or workflow patterns, for example:
  - failure rates
  - node that executed a command
  - files that were produced

The log could be stored in a key-value store used by data provenance.



Conclusions

In this work, we presented:

- the analysis of provenance use in various domains of computer science with a focus on experimental distributed systems research
- the new classification of provenance into 3 interacting types that particularly suit control-centric workflows
- the design of a system collecting provenance and respecting this classification

We plan to:

- implement the design in our experiment control engine
- enhance the design, in particular introduce a formal model

Thank you

Questions?