# Computational reproducibility in the Life Sciences and Research in Computer Science: round trip

**Sarah Cohen-Boulakia**

Université Paris-Saclay, Laboratoire de Recherche en Informatique

CNRS UMR 8623, Université Paris-Saclay, Orsay, France

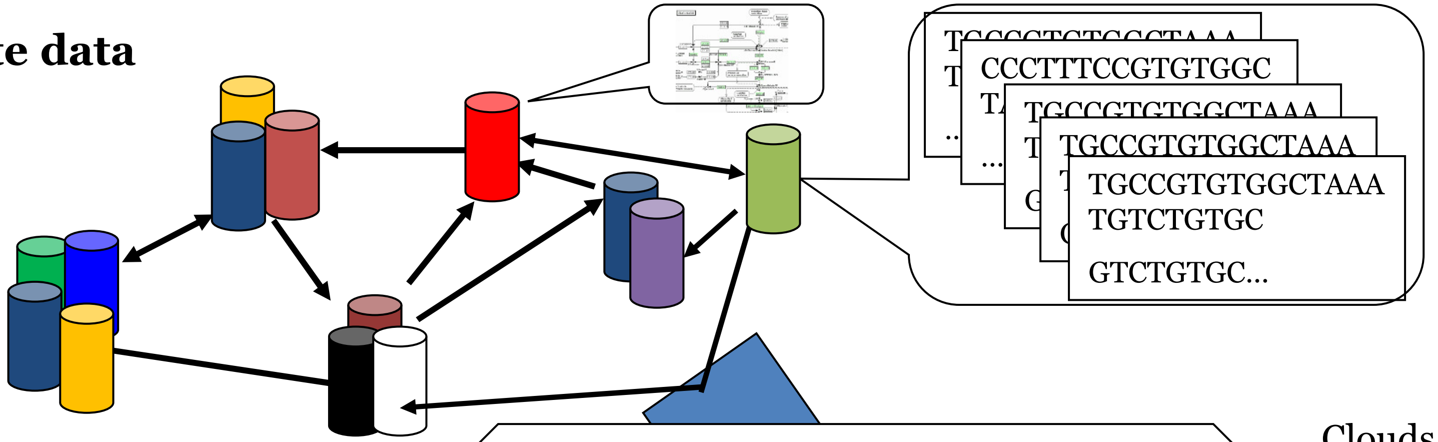
# Bioinformatics analysis

**Public and private data sources**

Distributed

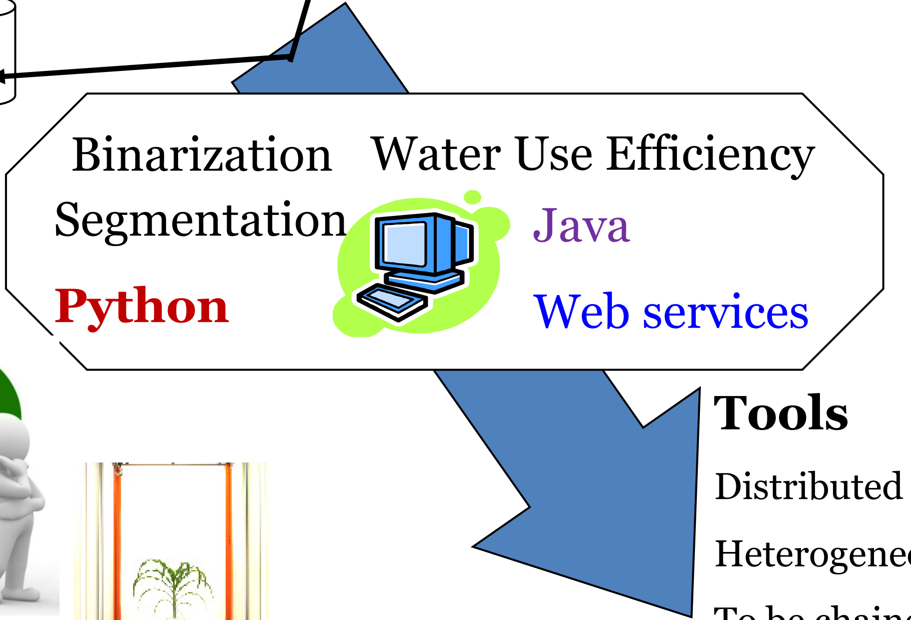
Heterogeneous

**> 1,500**



How has this plot been generated?
With which input data?
With which tools?
Parameters?

**→ Reproducibility**



Clouds

Grids

Clusters

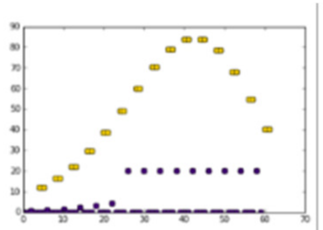
Desktop

**Tools**

Distributed **> 13,000**

Heterogeneous

To be chained



**Biologist's workspace**

## Studies on reproducibility

- Nekrutenko & Taylor, Nature Genetics (2012)
  - 50 papers published in 2011 using the Burrows-Wheeler Aligner for Mapping Illumina reads.
  - 31/50 (62%) provide no information
    - no version of the tool + no parameters used + no exact genomic reference sequence
  - 7/50 (14%) provide all the necessary details
- Alsheikh-Ali et al, PLoS one (2011)
  - 10 papers in the top-50 IF journals → 500 papers (publishers)
    - 149 (30%) were not subject to any data availability policy (0% made their data available)
    - Of the remaining 351 papers
      - 208 papers (59%) did not adhere to the data availability instructions
      - 143 make a statement of *willingness* to share
      - 47 papers (9%) deposited full primary raw data online

## Context, Challenges

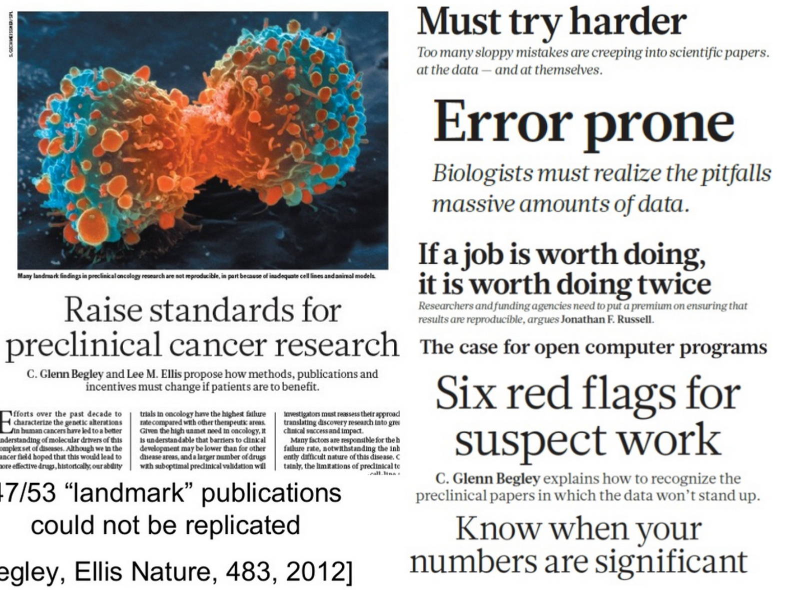
*Computational reproducibility crisis*

Increasing number of irreproducible results
- Even published in high IF venues
- Not (always) deliberately
- Computational irreproducibility increases

Various scientific domains
- Consequences may be huge (preclinical studies…)

Major challenge
- The cost of irreproducible preclinical studies have been evaluated to >$10 Billions per year (USA)

Becoming mandatory
- NSF projects, editors, ANR…


Many landmark findings in preclinical oncology research are not reproducible, in part because of inadequate cell lines and animal models.

Raise standards for preclinical cancer research

C. Glenn Begley and Lee M. Ellis propose how methods, publications and incentives must change if patients are to benefit.

Efforts over the past decade to characterize the genetic alterations in human cancers have led to a better understanding of molecular drivers of this complex set of diseases. Although we in the cancer field hoped that this would lead to more effective drugs, historically, our ability trials in oncology have the highest failure rate compared with other therapeutic areas. Given the high unmet need in oncology, it is understandable that barriers to clinical development may be lower than for other disease areas, and a larger number of drugs with suboptimal preclinical validation will investigators must reassess their approac translating discovery research into gre clinical success and impact.
Many factors are responsible for the h failure rate, notwithstanding the inh ently difficult nature of this disease. C tainly, the limitations of preclinical to

**Must try harder**

*Too many sloppy mistakes are creeping into scientific papers. at the data — and at themselves.*

**Error prone**

*Biologists must realize the pitfalls massive amounts of data.*

**If a job is worth doing, it is worth doing twice**

*Researchers and funding agencies need to put a premium on ensuring that results are reproducible, argues* **Jonathan F. Russell.**

**The case for open computer programs**

Six red flags for suspect work

**C. Glenn Begley** explains how to recognize the preclinical papers in which the data won't stand up.

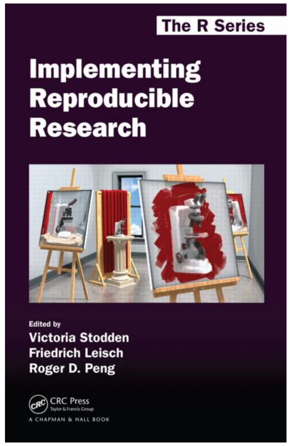
Know when your numbers are significant

47/53 "landmark" publications could not be replicated

[Begley, Ellis Nature, 483, 2012]

## Reproducibility

V. Stodden *et al.*



The R Series
Implementing Reproducible Research
Edited by Victoria Stodden Friedrich Leisch Roger D. Peng
CRC Press

*Empirical reproducibility*

- detailed information about non-computational empirical scientific experiments and observations
- In practice this is enabled by making data freely available, as well as details of how the data was collected.

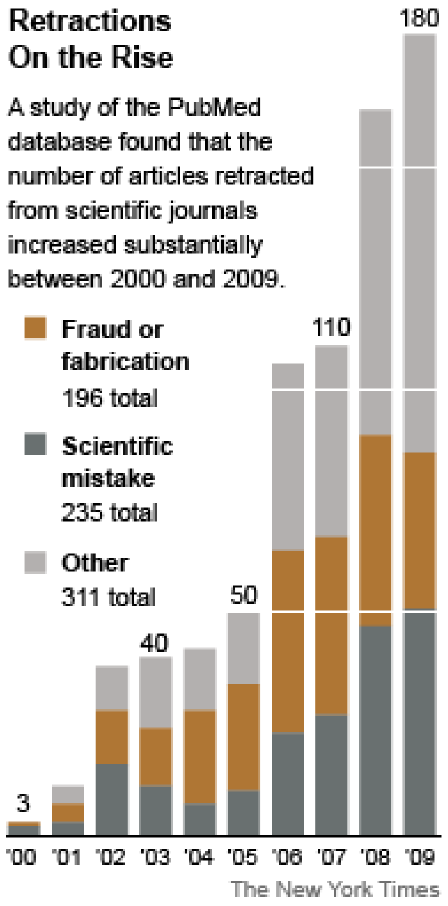
*Statistical reproducibility*

- detailed information about the choice of statistical tests, model parameters, threshold values, etc.
- This relates to pre-registration of study design to prevent p-value hacking and other manipulations.

*Computational reproducibility*

- detailed information about code, software, hardware and implementation details

  → Goal: document how data has been produced



## Scripts and reproducibility?
### Good practices

Providing scripts is an excellent first step
+ Using git/github for versioning, collaborative development

But scripts do not allow to

Distinguish between steps of the analysis
- piece of codes, methods/functions

… and execution of the analysis
- data sets used as inputs and then produced

Emphasize the major steps of the analysis
Provide solution for data management
- Naming convention for produced files, storage…

→ Scripts are difficult to share, exchange and reuse (repurpose)

## Outline

Context

Systems and tools to enhance reproducibility

- Scientific workflow systems
- Companion tools

Lessons learnt on using such systems and tools

- Reprohackathons
- Levels of reproducibility with scientific workflows
- Reproducibility-friendly features

Open Computer Science research problems

Conclusion

# Scientific workflow systems

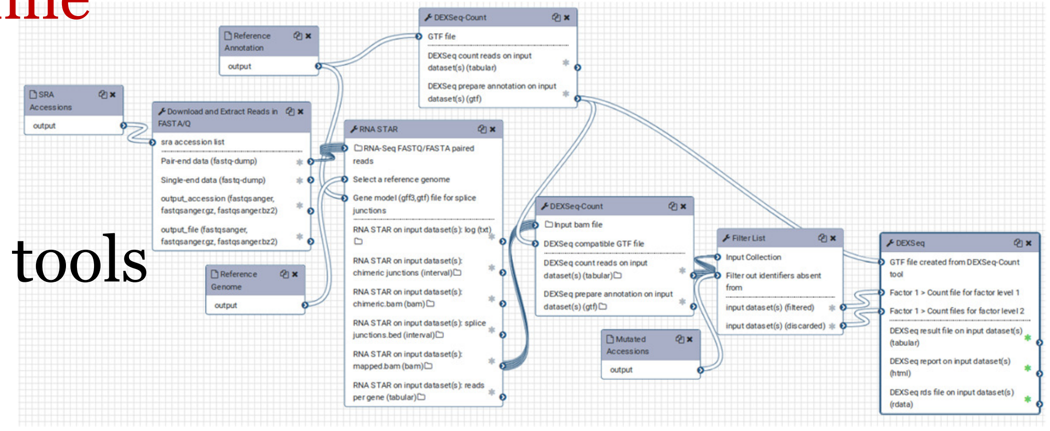
SWFS = "Data analysis pipeline"

Data flow driven

Encapsulation of scripts

WF specification: connected tools

*steps of the analysis*



WF execution: data consumed/produced

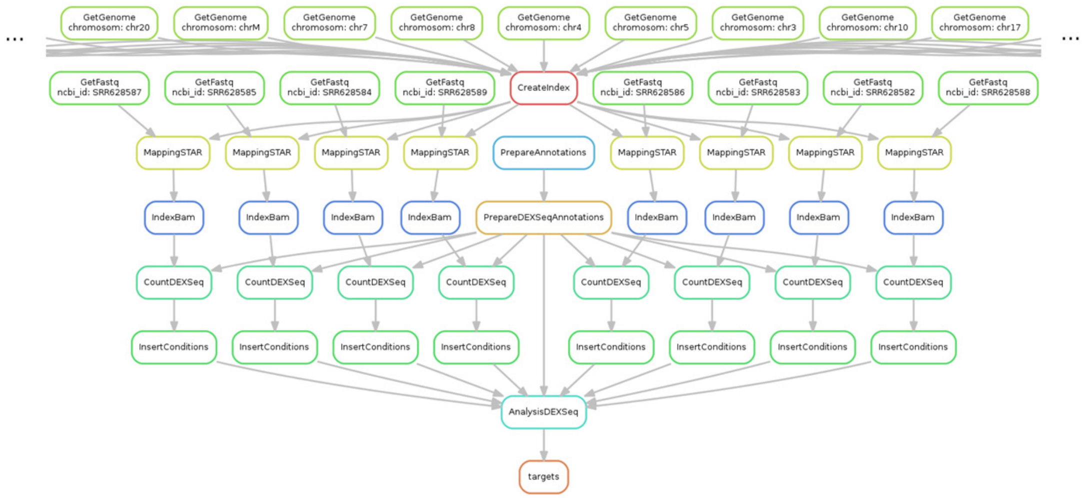
Provenance modules

*data management*

SWFS scheduling, logging,

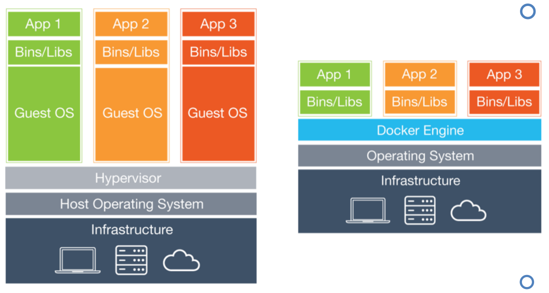
May be equipped with GUI

Galaxy, NextFlow, SnakeMake…


GetGenome chromosom: chr20
GetGenome chromosom: chrM
GetGenome chromosom: chr7
GetGenome chromosom: chr8
GetGenome chromosom: chr4
GetGenome chromosom: chr5
GetGenome chromosom: chr3
GetGenome chromosom: chr10
GetGenome chromosom: chr17
GetFastq ncbi_id: SRR628587
GetFastq ncbi_id: SRR628585
GetFastq ncbi_id: SRR628584
GetFastq ncbi_id: SRR628589
CreateIndex
GetFastq ncbi_id: SRR628586
GetFastq ncbi_id: SRR628583
GetFastq ncbi_id: SRR628582
GetFastq ncbi_id: SRR628588
MappingSTAR
PrepareAnnotations
IndexBam
PrepareDEXSeqAnnotations
CountDEXSeq
InsertConditions
AnalysisDEXSeq
targets

## Capturing the programming environment

Ensuring your workflow has everything it needs to run
Libraries, dependencies…
Virtual machines capture the programming environment
Container solutions



- package an application
  - with all of its dependencies
  - into a standardized unit for software development

include the application and its dependencies

- but share the kernel with other containers
- They
  - are not tied to any specific infrastructure;
  - run on any computer, on any infrastructure and in any cloud

Lighter solution than classical VM

➔ BioContainers: a registry of containers!

## Outline

## Our new concept: ReproHackathon

ReproHackathon

- A hackathon where
  - Given a scientific publication + input data (+ possibly contacts with authors)
  - Several (groups of) developers reimplement the methods to try to get the same result on the Cloud@IFB
- Aim: Ability of current workflow systems and companion tools to reproduce a scientific result

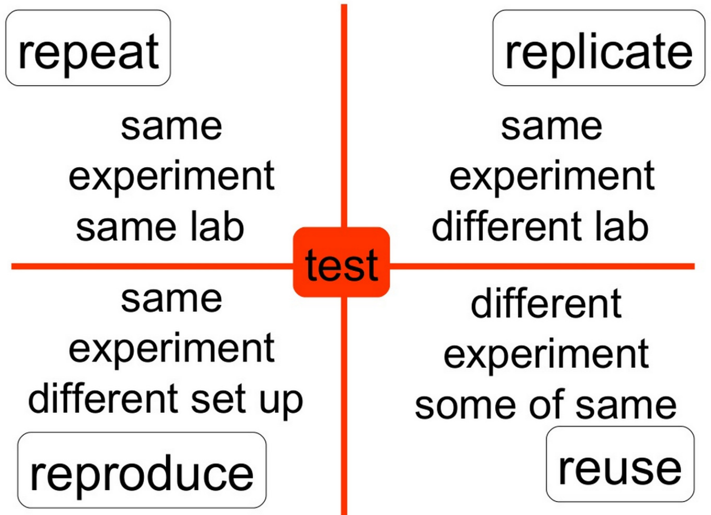
First edition

- RNA-Seq data from patients with uveal melanoma: genes involved
- Divergent published results…
- 25 participants (IGRoussy, Curie, Pasteur, Saclay,…)

Reprohackathon 2 Lyon, July 2018
Phylogenetics

Reprohackathon 3 Montpellier Nov 2019
Plant phenotyping

# Levels of computational reproducibility


repeat
same experiment same lab
replicate
same experiment different lab
test
same experiment different set up
reproduce
different experiment some of same
reuse

Drummond C Replicability is not Reproducibility: Nor is it Good Science, online
Peng RD, Reproducible Research in Computational Science *Science 2 Dec 2011: 1226-1227.*

## 3 ingredients

Workflow Specification
- Chained Tools

Workflow Execution
- Input data and parameters

Environment
- OS/librairies …

## Repeat

- *Redo*: exact same context
- Same workflow, execution setting, environement
- Identical *output*

→Aim = proof for reviewers ☺

## Replicate

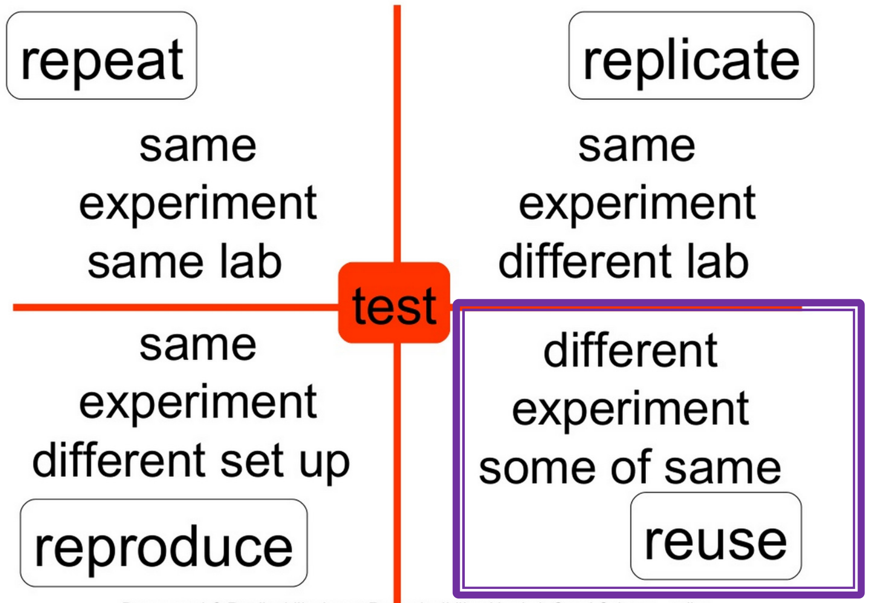
- Variation allowed in the workflows, execution setting, environement
- Similar *output*

→ Aim = robustness

# A continuum of possibilities

## Reproduce

- Same *scientific result*
- But the means used may be changed
- Different workflows, execution setting, environment
- Different output but in accordance with the result



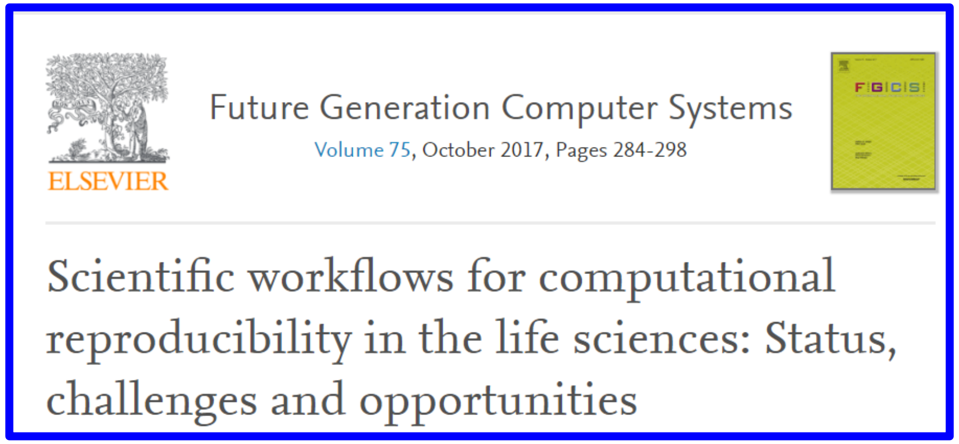
repeat
same experiment same lab
replicate
same experiment different lab
test
same experiment different set up
reproduce
different experiment some of same
reuse

Drummond C Replicability is not Reproducibility: Nor is it Good Science, online
Peng RD, Reproducible Research in Computational Science *Science 2 Dec 2011: 1226-1227.*

## Reuse

- Different scientific result
- Use of tools/… designed in another context

## Reproducibility-friendly features


ELSEVIER

Future Generation Computer Systems

Volume 75, October 2017, Pages 284-298

Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities

6 Systems: Galaxy, Nextflow, SnakeMake, VisTrails, OpenAlea, Taverna

**Specification**

- Language (XML, Python…)
- Interoperability (CWL…)
- Description of steps
  - Remote services
  - Command line
  - Access to source code
- Modularity (nested workflows?)
- Annotation (tags, ontologies, myexperiment…)

**Execution**

- Language and standard (PROV…,) → repeat … reuse
- Presentation (interactivity with the results/provenance, notebooks) → replicate … reuse
- Annotations → reuse

**Environment**

Ability to run workflows within a given environment

- Virtual machines
  - VMWare, KVM, VirtualBox, Vagran,…
- Lighter solutions (containers)
  - Docker, Rocket, OpenVZ, LXC, Conda
- Capturing the command-line history, input/output, specification: CDE, ReproZip

## Outline

Context

Systems and tools to enhance reproducibility

Lessons learnt on using such systems and tools

Open Computer Science research problems
Research topics
Focus on 3 graph-oriented kind of problems

Conclusion

## Developing workflows

Bridge the gap between scripts and workflows

Supporting several programming languages in the same environment of development

Tests in workflows

- Unit tests, integration tests...
- Providing samples may be an issue (privacy...)

Workflow Maintenance: set of compatible libraries?

- Docker (containers), VM allows to freeze the environment

→ Need to liquefy!

- Given a program P that can be repeated in an environment E... ... Find an environment E' (E' uses more recent versions of libraries than E) where P still *works*

# Discovering workflows [Reuse]

Query languages for repositories?

*Given a workflow – find similar workflows*

Detecting patterns within workflows

Indexing workflows

Reconstruct their histories

Core of the problem:

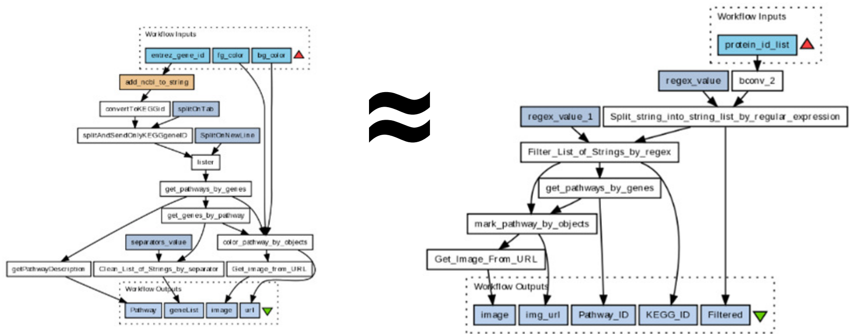
Workflow similarity

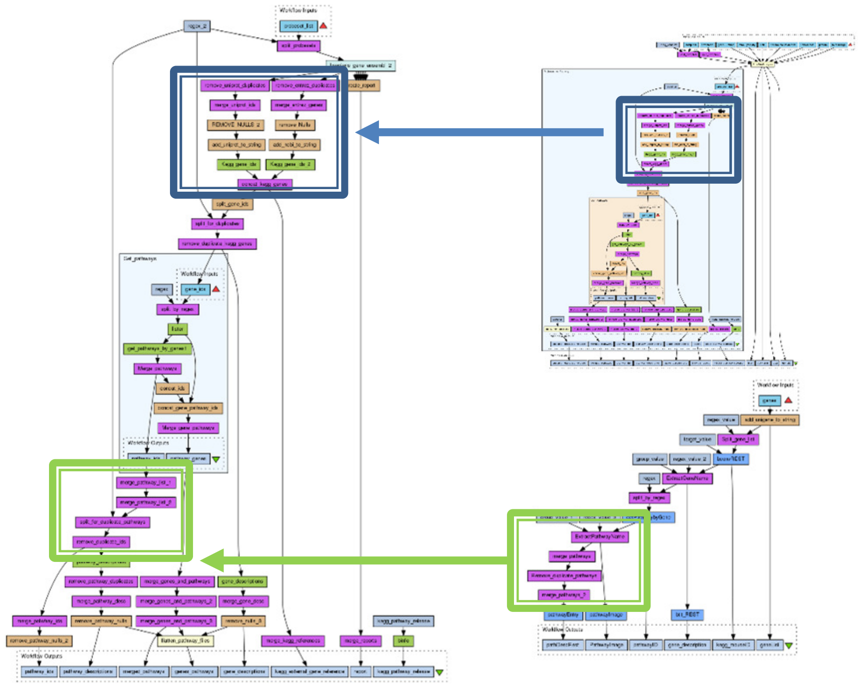
State-of-the-art [SCB+14]

Based on the graph structures or annotations (ontologies)

Need to design hybrid and efficient solutions

NB : Reusing (and searching for) Notebooks is another open point





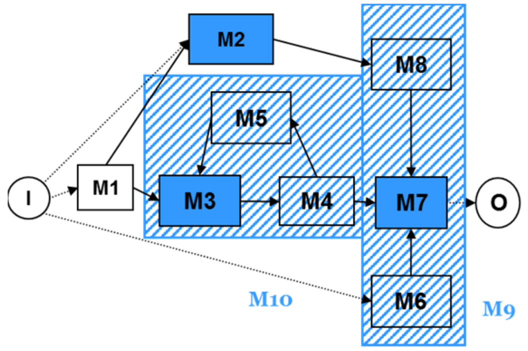
# Simplifying workflows [Reuse]

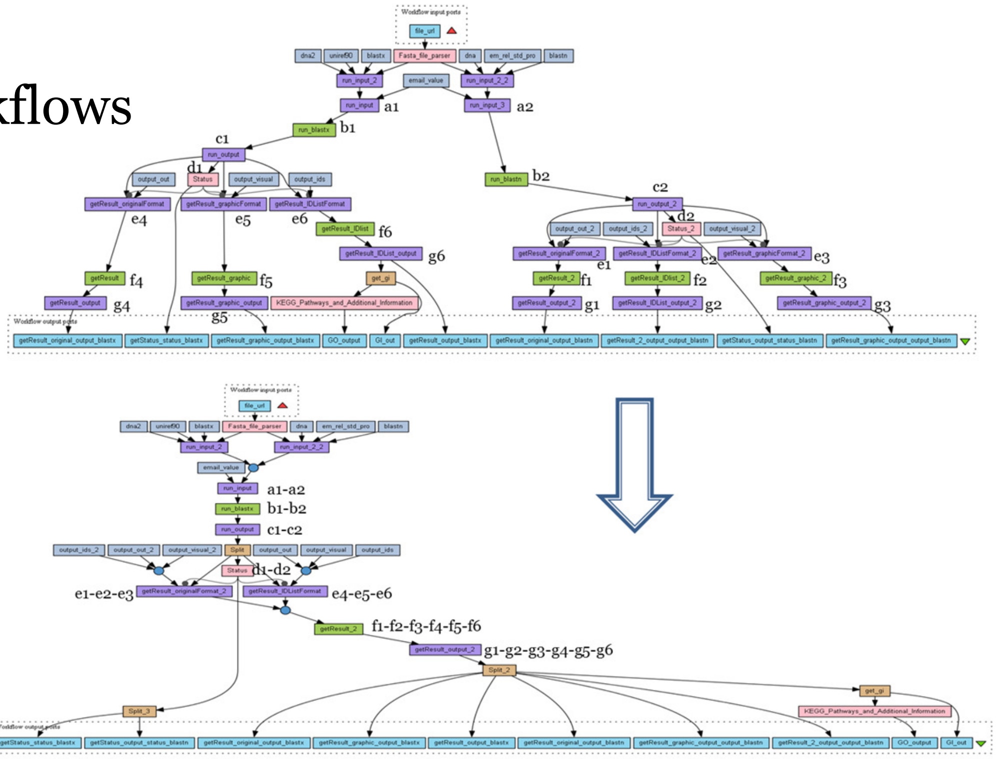
Designing more coarse-grained workflows

- Automatic Design of subworkflows (graph-based)
- Abstraction of provenance traces
- Summarization (Web Semantics)

Refactoring workflows

- Remove redundancies in workflows
- Rewritting, Anti-patterns





## Outline

Context

Systems and tools to enhance reproducibility

Lessons learnt on using such systems and tools

Open Computer Science research problems

Research topics

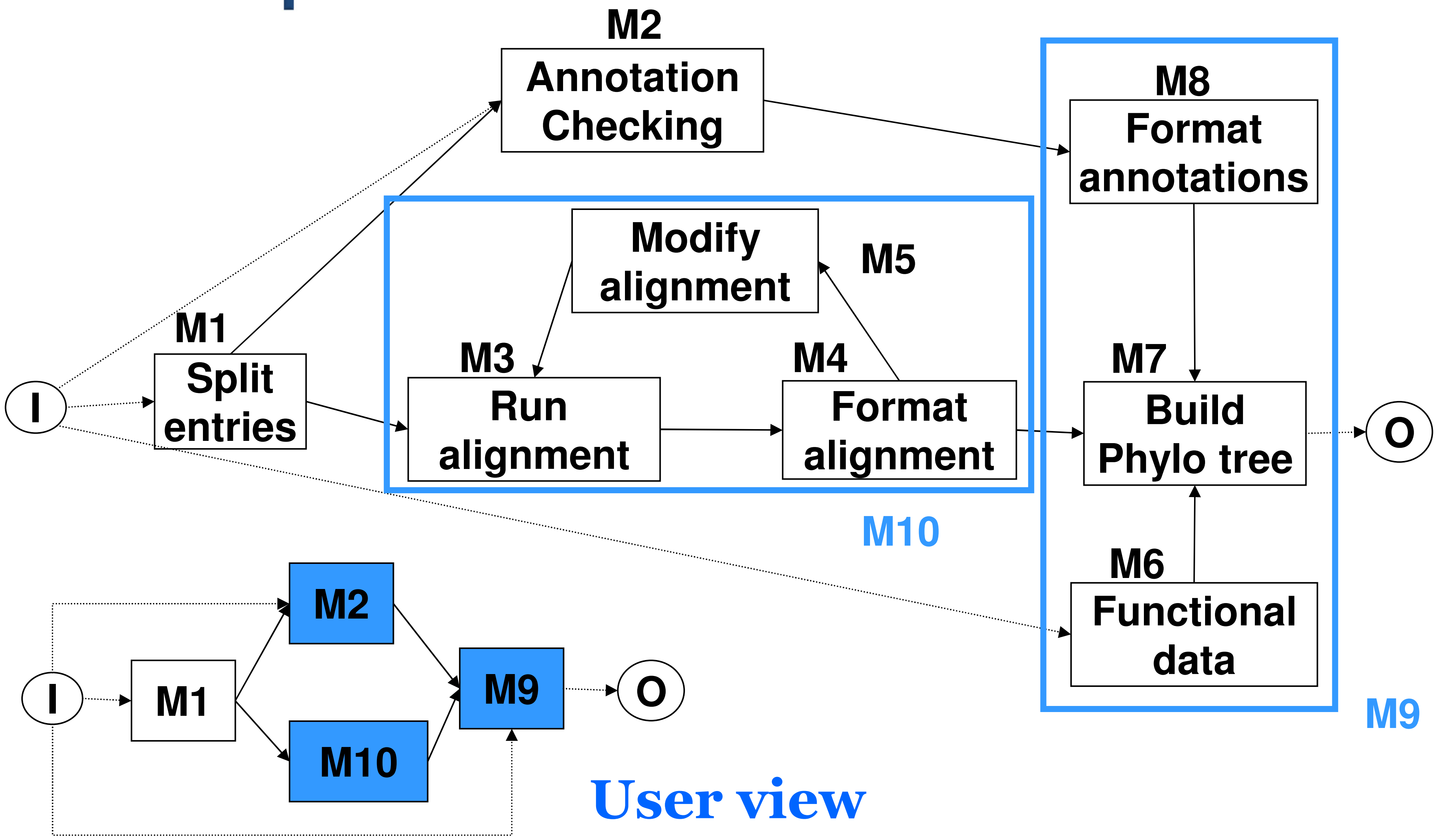
Focus on 3 graph-oriented kinds of problems

Conclusion

## Problem 1: Abstracting workflows (composition)

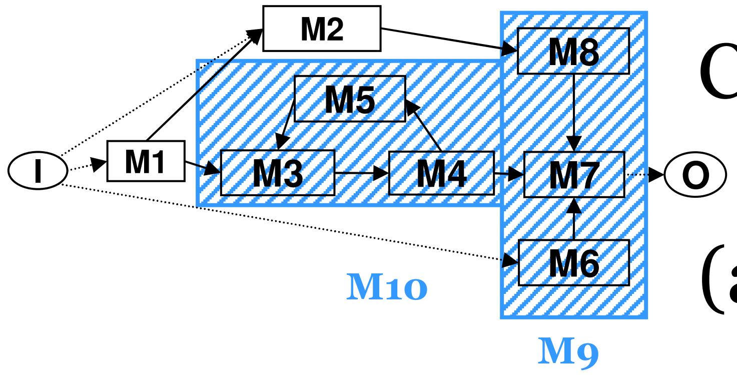
Reducing the complexity of workflows
making them easier to share

# Composite modules



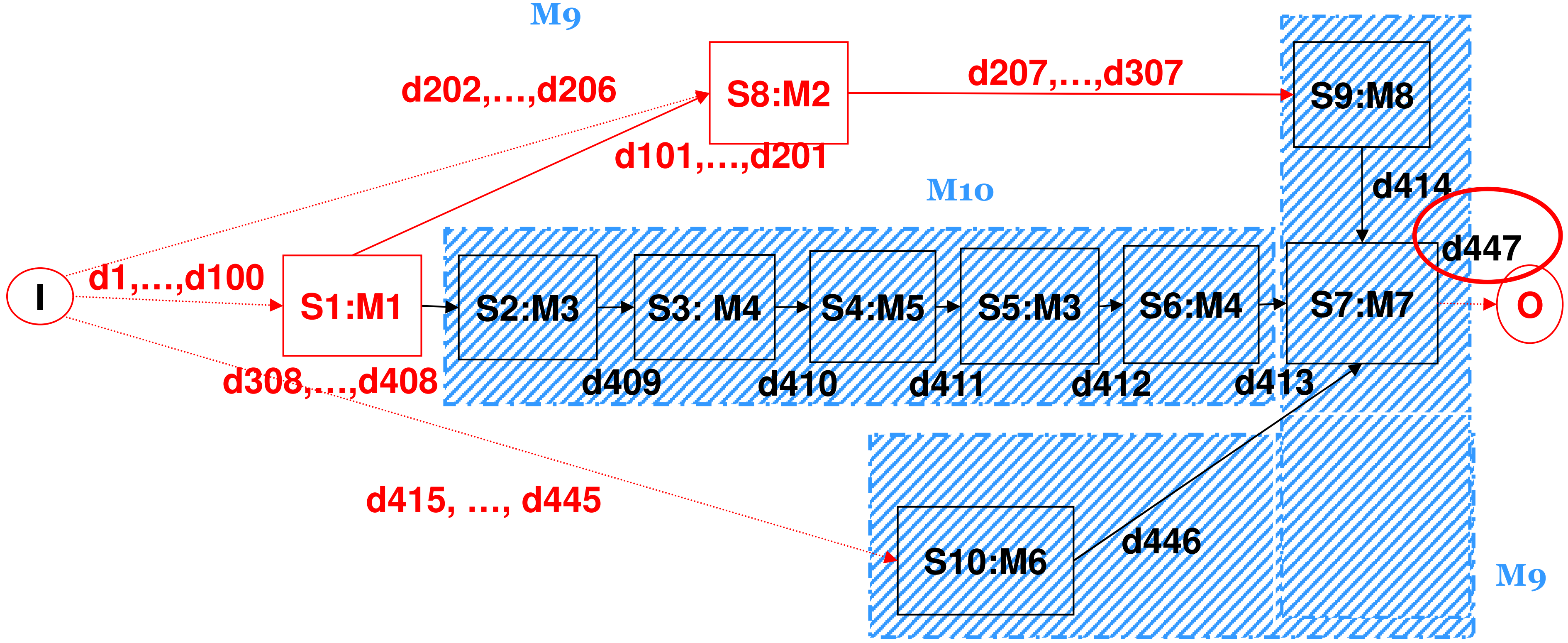
User view

# Composite modules



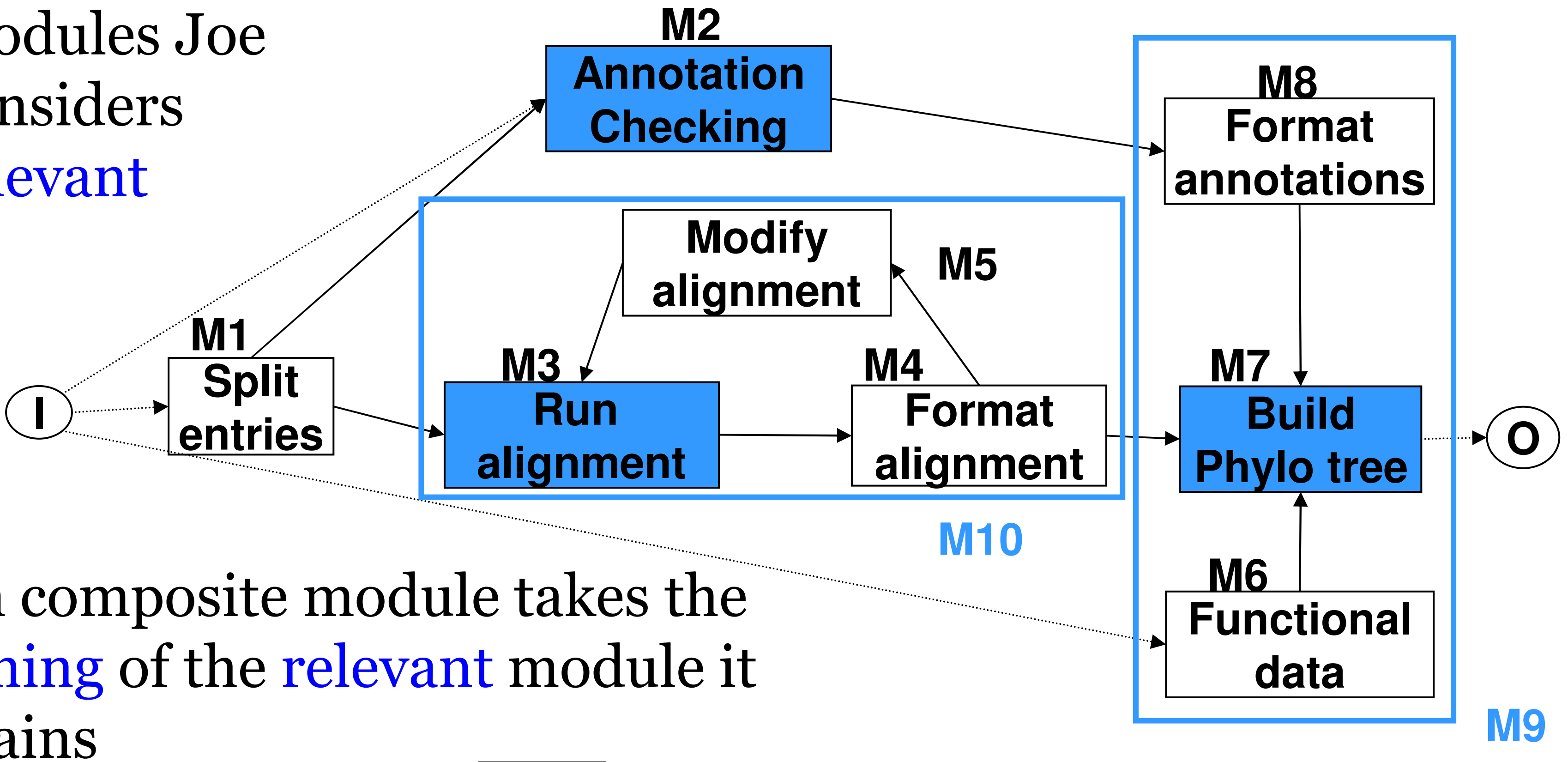
Composition simplifies provenance

(and the wf specification)

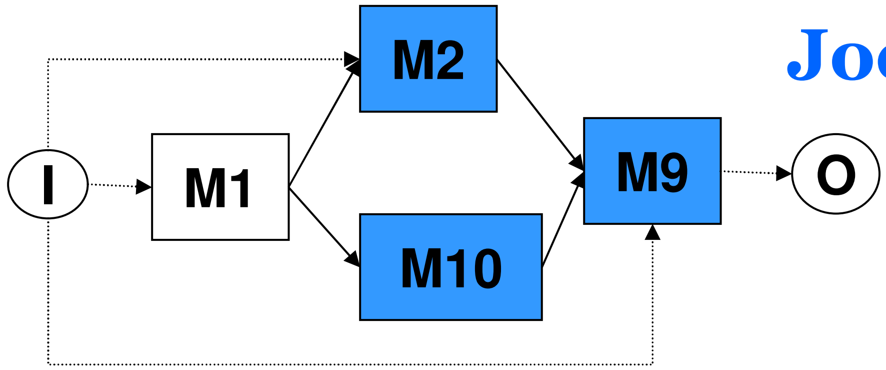


# Relevant user view

Modules Joe considers relevant



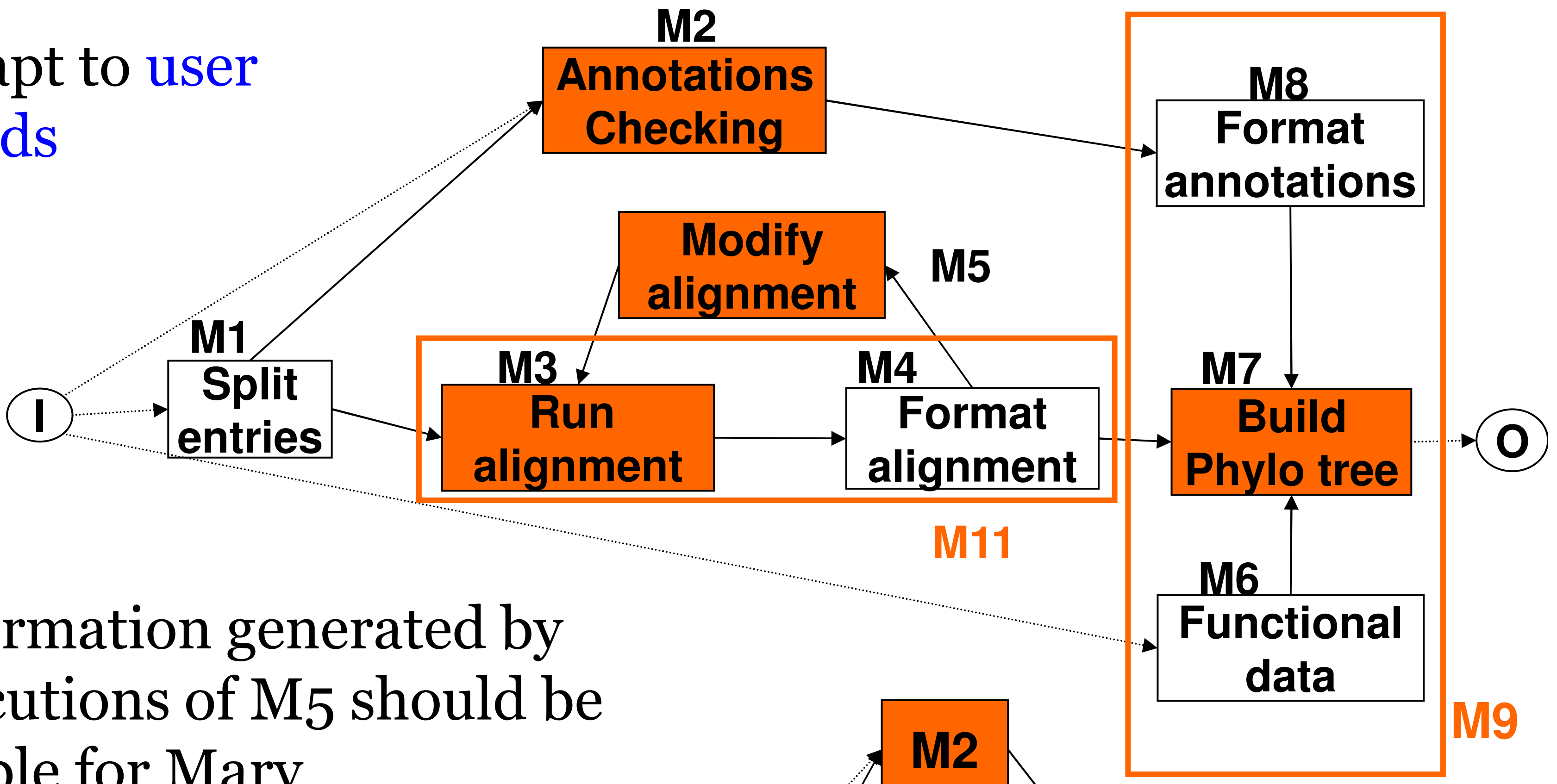
Each composite module takes the meaning of the relevant module it contains



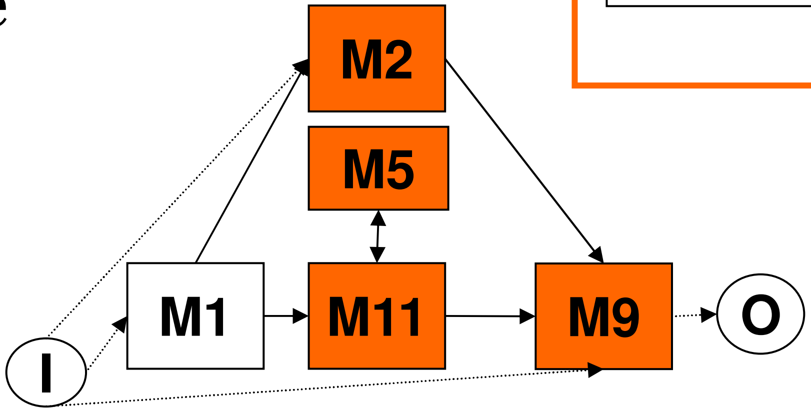
Joe's view

# User views may differ

Adapt to user needs



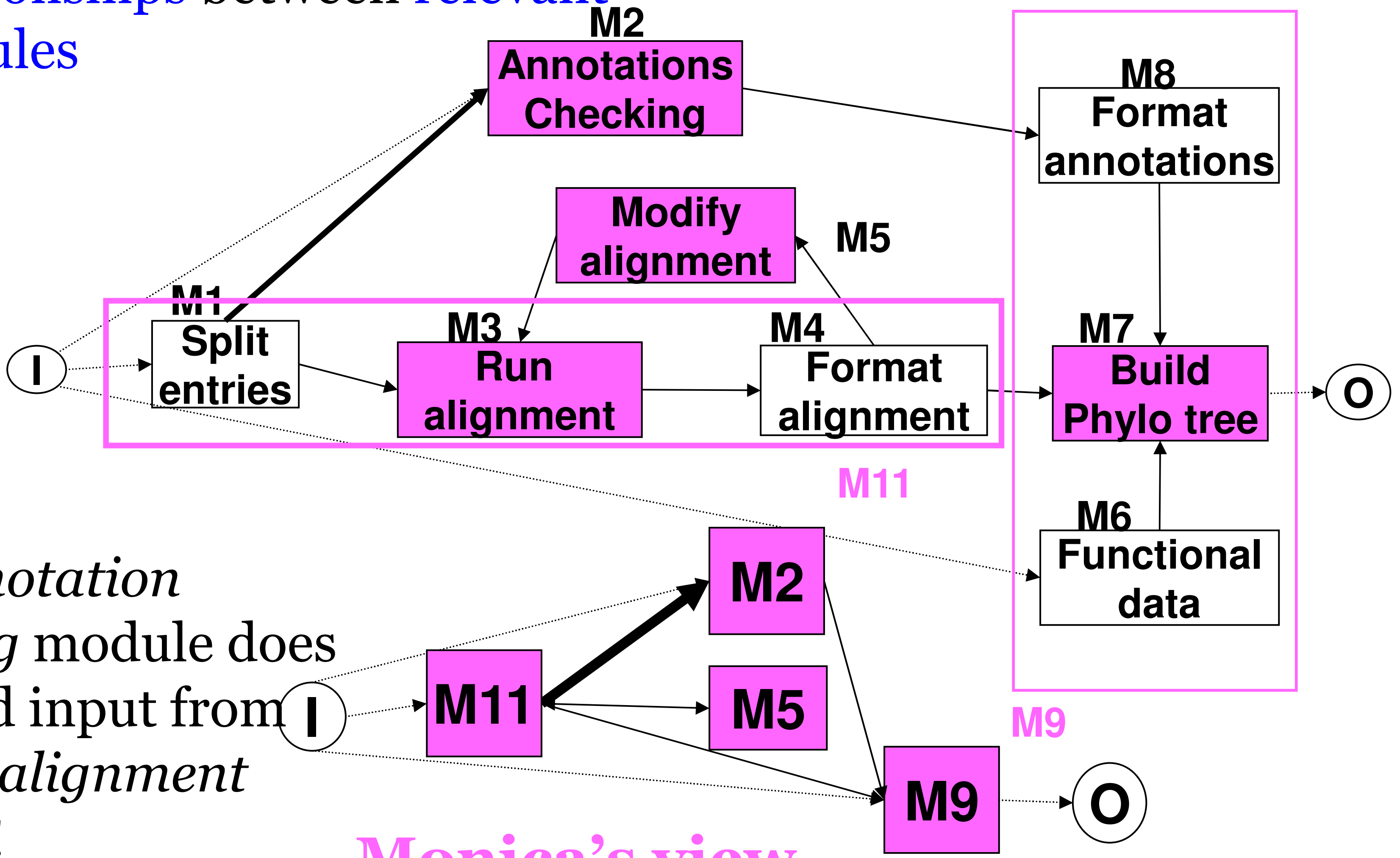
Information generated by executions of M5 should be visible for Mary



Mary's view

# Grouping may be error-prone!

Grouping should preserve the relationships between relevant modules



The *annotation checking* module does not need input from the *run alignment* module!

Monica's view

## Hope and next challenges

- Hope
  - *ZOOM* provides a polynomial-time algorithm to automatically construct user views
    - which preserve the dataflow (no missing path and no new path between two relevant tasks)
    - and produces a minimal user view

In collaboration with Penn UNIVERSITY *of* PENNSYLVANIA

- Next challenges

  Repairing user views badly designed

  Providing such functionalities in real systems

  Using it on provenance information in provenance systems

## Problem 2: rewritting workflows

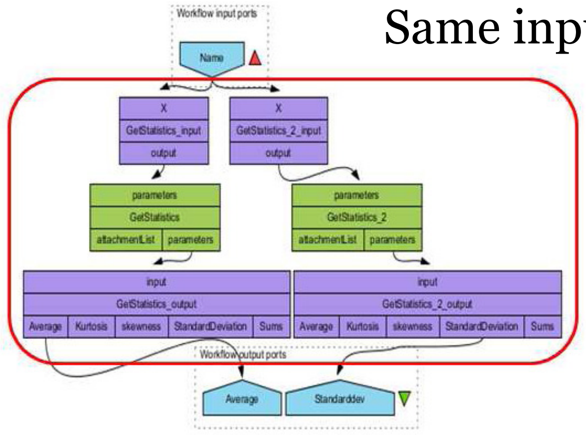
Repairing workflows,
making them easier to share

# Redundancy in Workflows

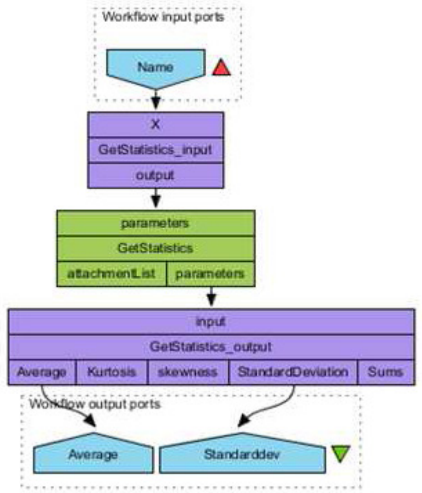
3 processors duplicated!

No redundancy

Same input

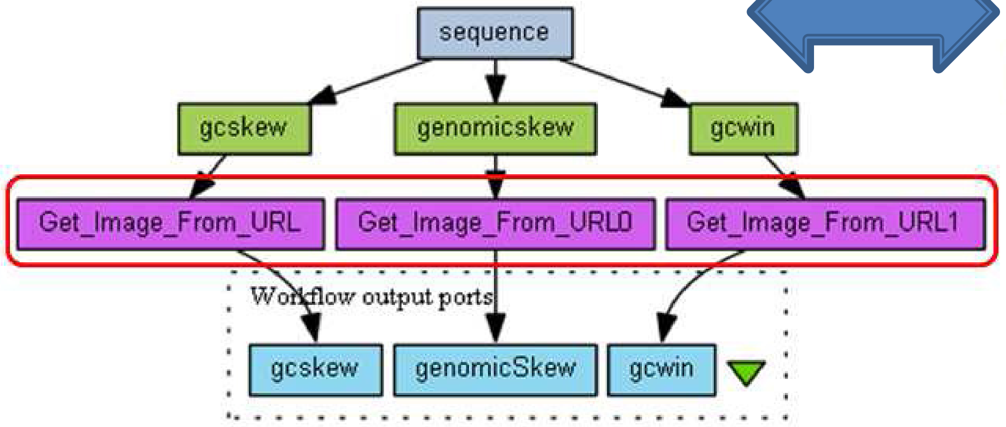


Equivalent

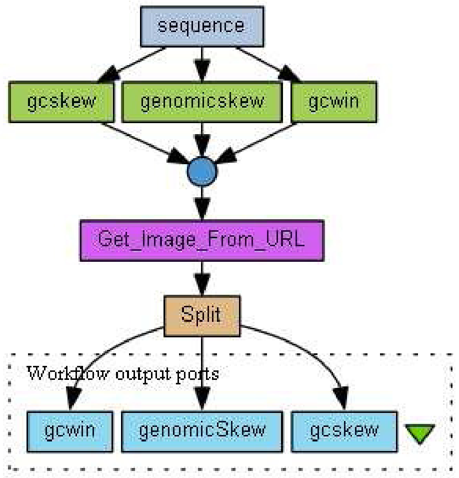


Equivalent

(i)



(ii)



## Rewriting workflows

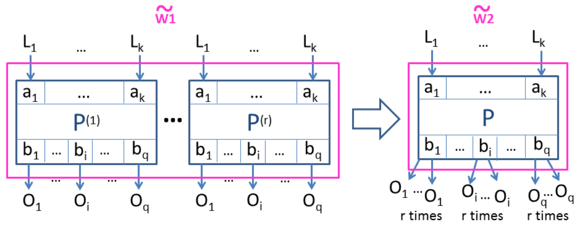
- Assumptions before merging several copies of a processor
  - Only copies with the exact same code
  - Only copies that do not depend on each other
  - Only deterministic processors (same input → same output)

- Need to understand the semantics of the system
  - Determining the anti-patterns and designing their corresponding rewritings

## Hope and next challenges

- Hope
  - *DistillFlow* detects anti-patterns and rewrites Taverna workflows

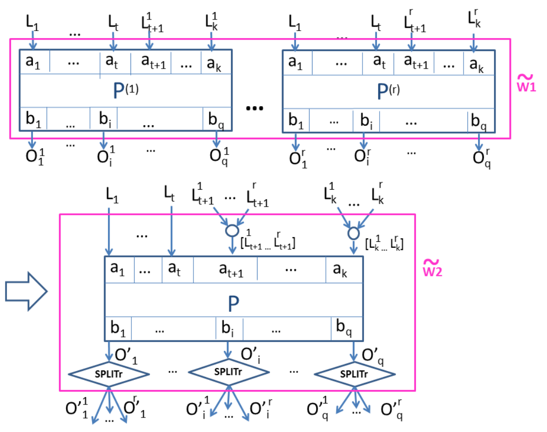
In collaboration with MANCHESTER 1824 The University of Manchester



$L_i$ can be one single value or a list of values

- Next Challenges
  - Larger sets of anti-patterns
  - Rewritting on in-use systems (Galaxy, NextFlow, SnakeMake)

Processor P applies cross product to values on ports $a_1$ to $a_t$ and dot product to values on ports $a_{t+1}$ to $a_k$



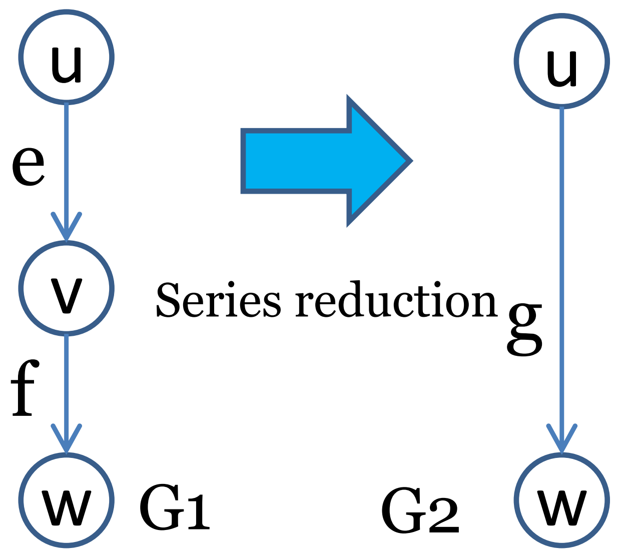
## Problem 3: exploiting specific graph structures

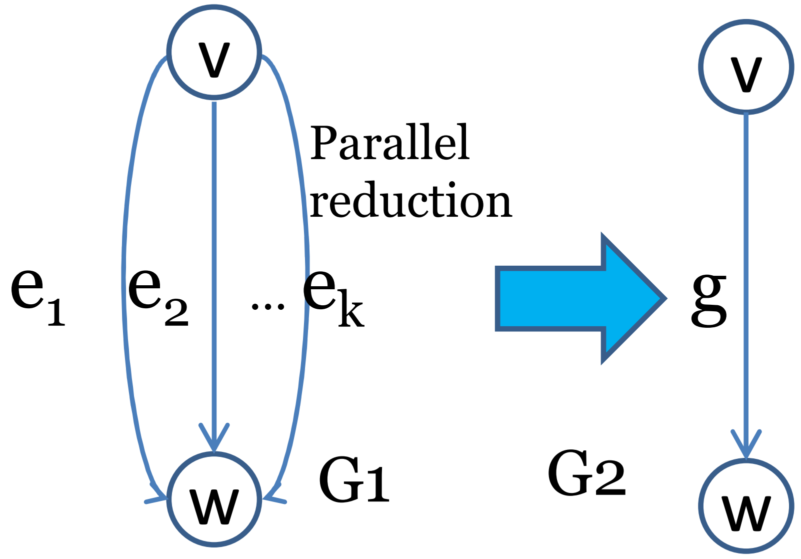
Comparing graphs is a difficult problem that can be simplier on specific graph structures

# Definition of SP-graphs

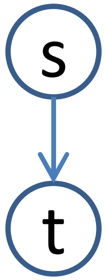
G is SP iff MaxRed(G) = BSP

- MaxRed(G): iteratively performs series and parallel reductions on a given graph G

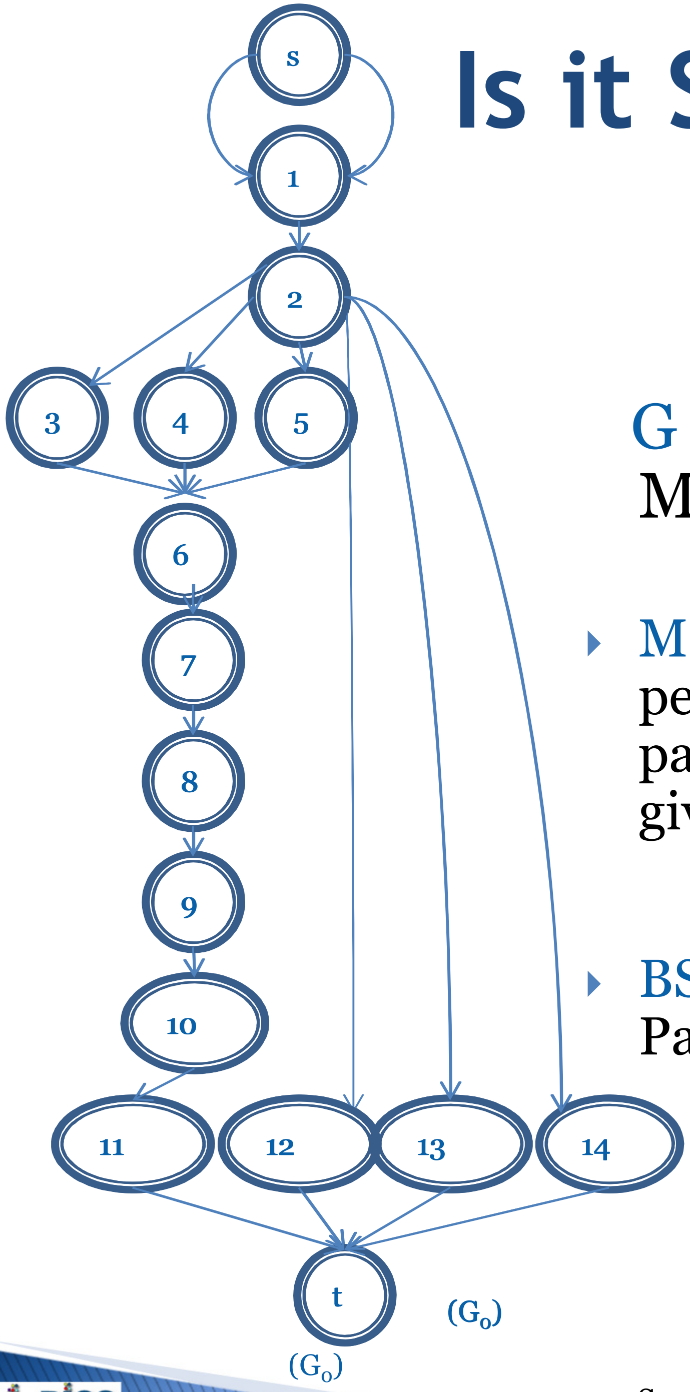




- BSP: Basic Series-Parallel



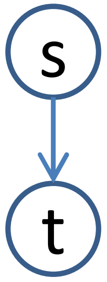
# Is it Series-Parallel?

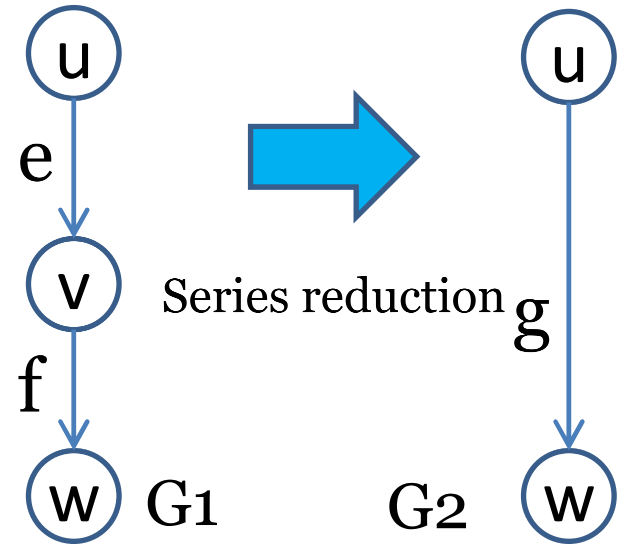


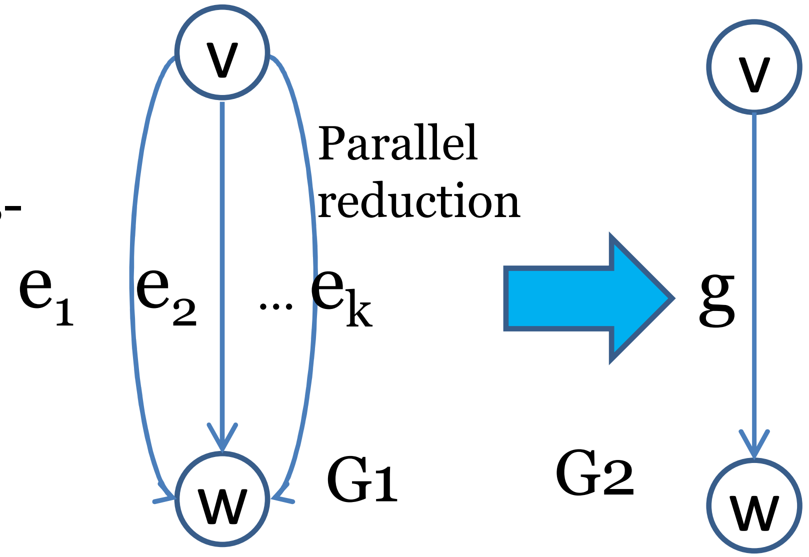
YES!

G is SP iff
MaxRed(G) = BSP

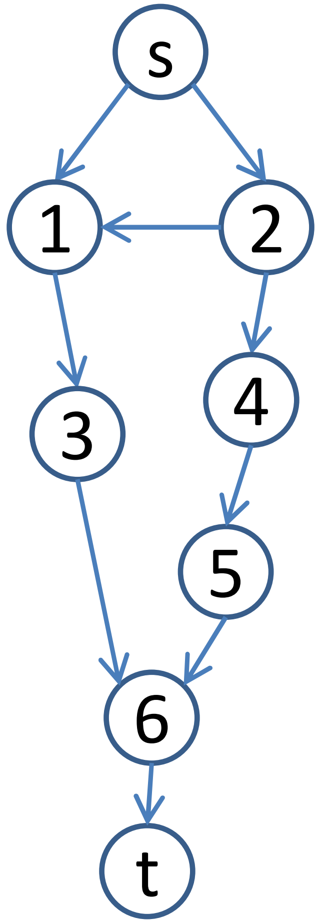
- MaxRed(G): iteratively performs series and parallel reductions on a given graph G
- BSP: Basic Series-Parallel







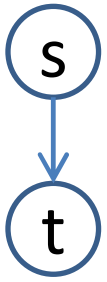
# Is it Series-Parallel?

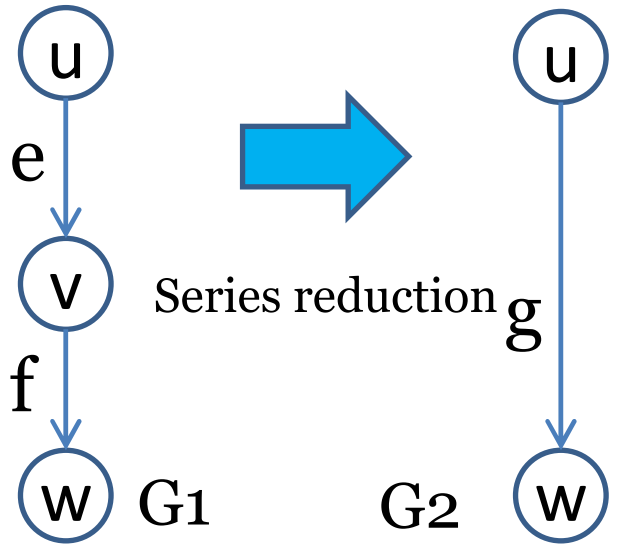


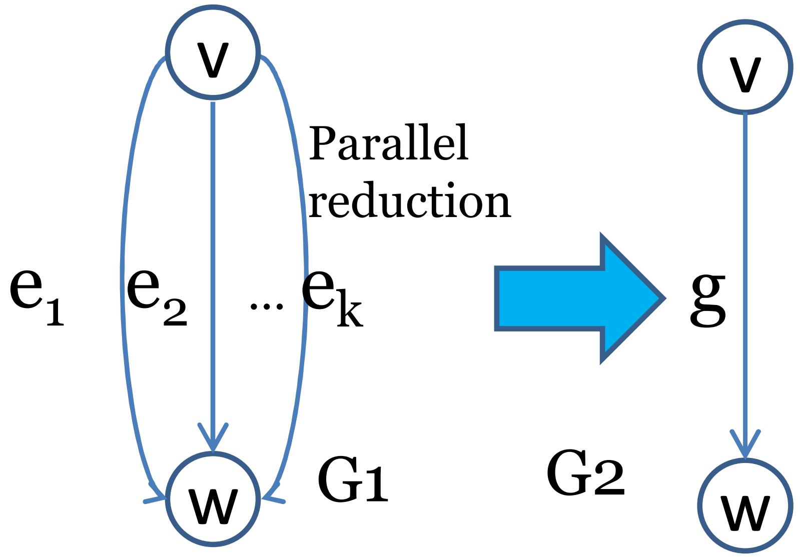
NO!

G is SP iff
MaxRed(G) = BSP

- MaxRed(G): iteratively performs series and parallel reductions on a given graph G
- BSP: Basic Series-Parallel



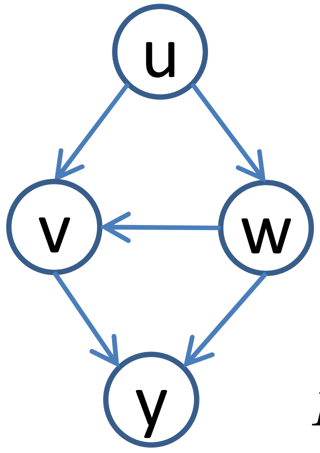




## Another definition (Non SP-graphs)

G is non-SP iff MaxRed(G) contains $G_{\text{forbidden}}$

$G_{\text{forbidden}}$



v and w are called reduction nodes

*Intuitively, such graphs cannot be synchronized*

Subgraph isomorphism is polynomial for SP graphs

## Some hope and next Challenges

- Hope
  - In *PDiffView* SP-graphs have been used to solve the problem of providing a polynomial algorithm to compare two executions of the same workflow

In collaboration with Penn UNIVERSITY of PENNSYLVANIA HUMBOLDT-UNIVERSITÄT ZU BERLIN

- Next challenges
  - Guiding users in designing (close to) SP-graph structures
  - Extending SP structures to cover more expressive workflow structures while making sure that polynomial-time algorithms can be found to compare workflows

→ Impact on querying workflow repositories, reconstructing workflow history, …

# Conclusion

Many scientific results are not computationally reproducible

Providing scripts is an excellent start

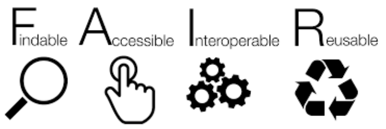
Scientific workflows are increasingly mature solutions

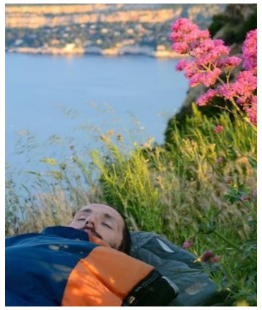
- Tracking the exact connected tools used
- Track the exact data used, produced and tool parameters setting
  - →Provenance modules
- Coarse-grain version of the analysis to better capture the analysis steps

Several open challenges are directly related to improvement in research in computer science (graphs, algorithmics…)

Workflows play key role to produce FAIR data

FAIR metrics for workflows have to be defined too!





Join us!

cohen@lri.fr

**MaDICS CNRS GDR**