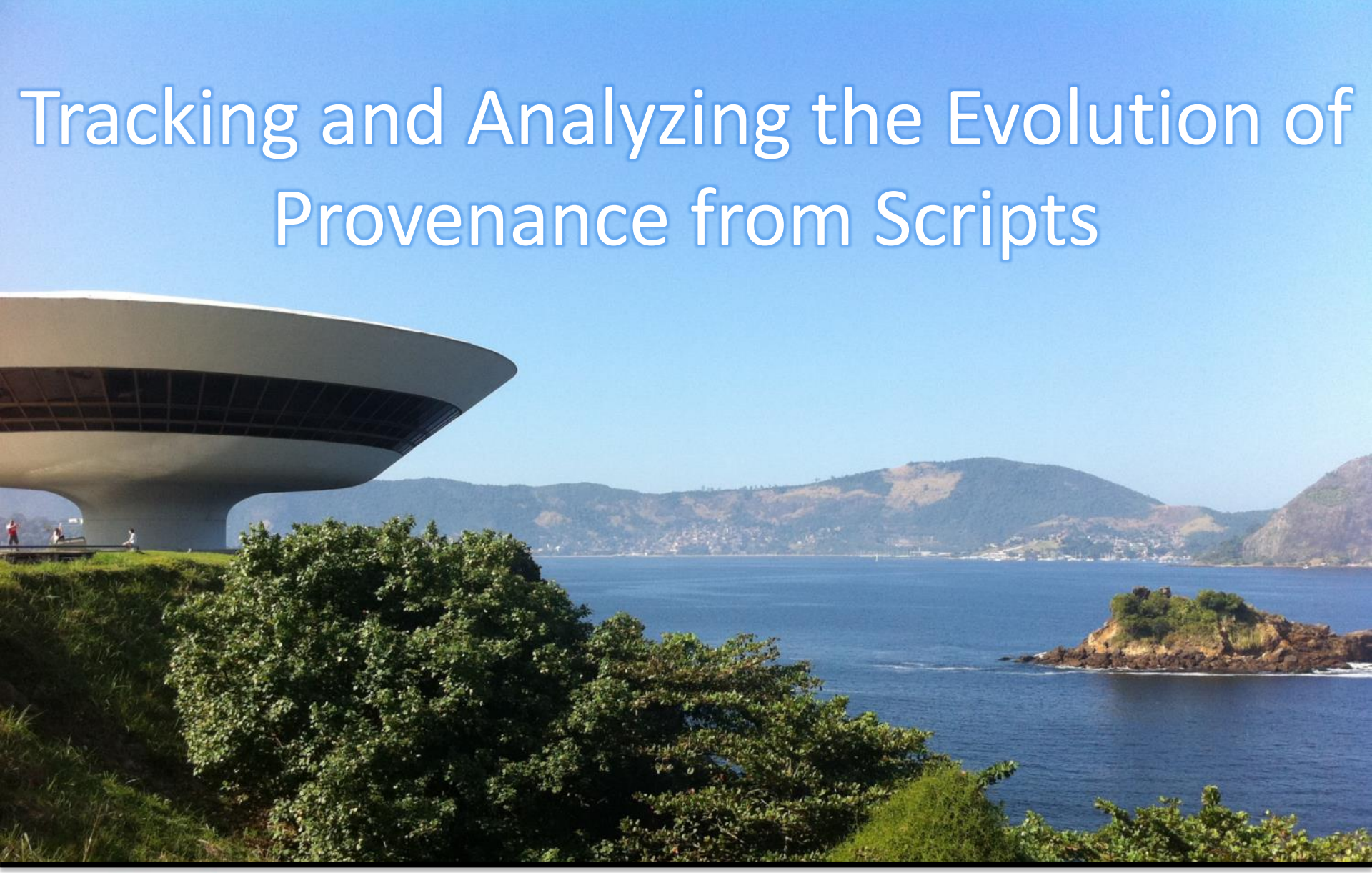
Tracking and Analyzing the Evolution of Provenance from Scripts



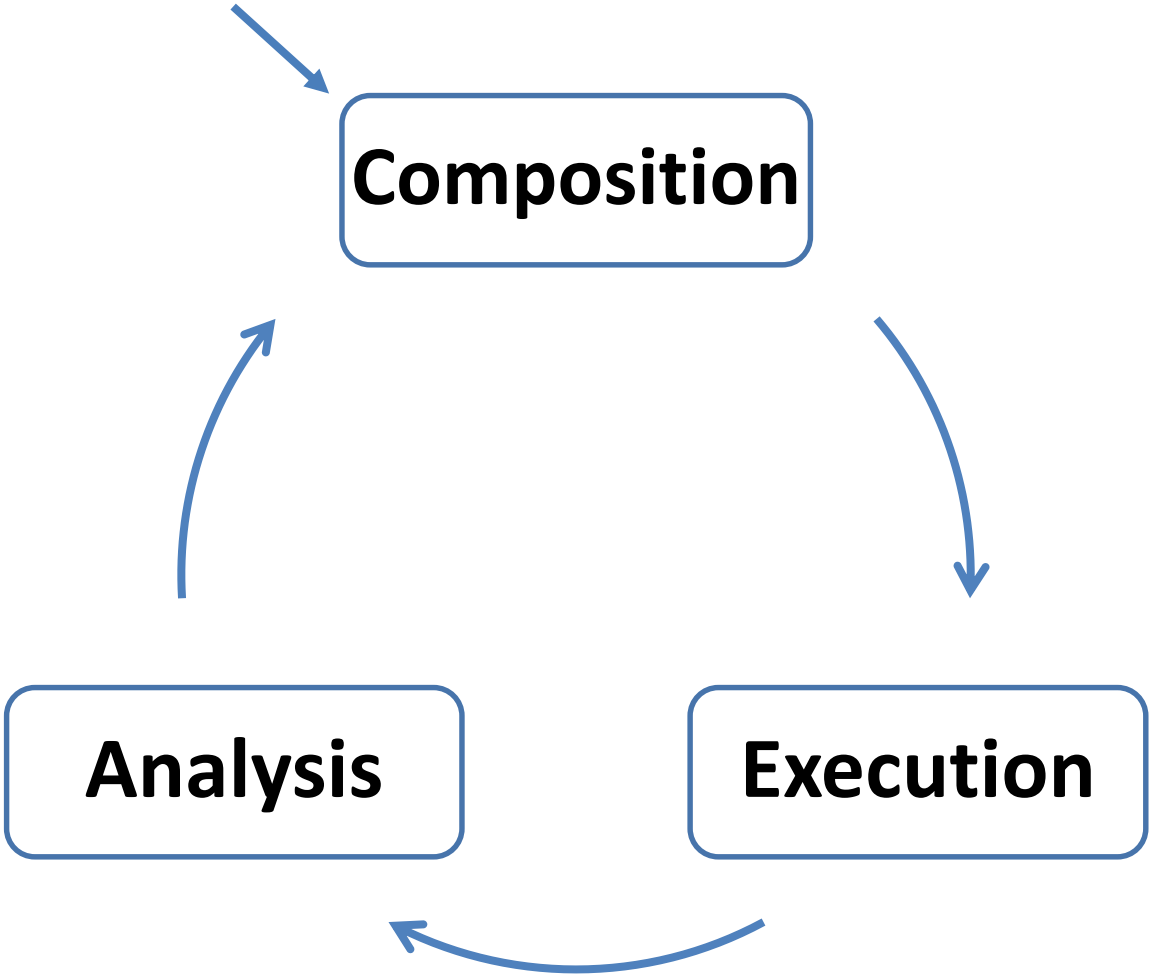
João Felipe Nicolaci Pimentel (UFF),
Juliana Freire (NYU), Vanessa Braganholo (UFF),
Leonardo Murta (UFF)

Instituto de Computação UFF

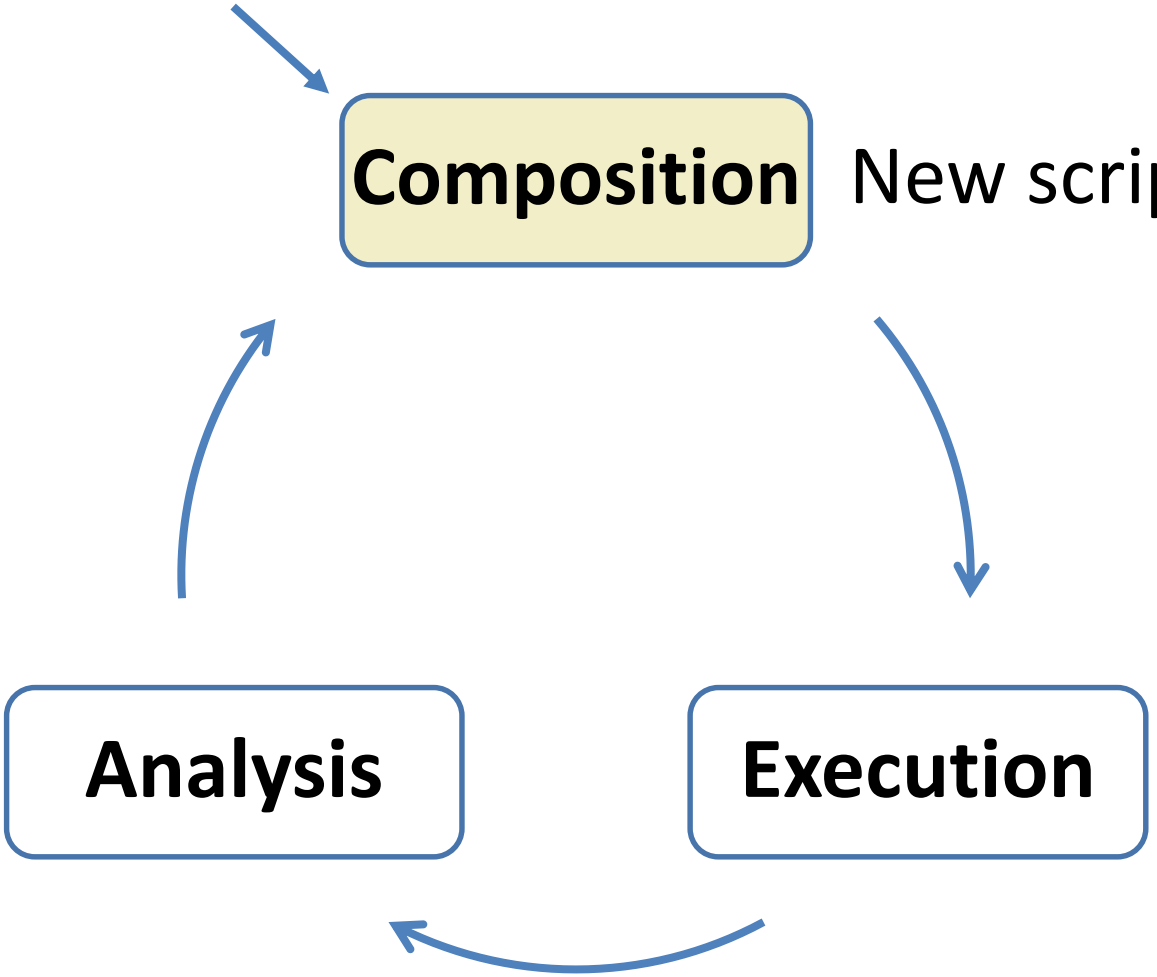
NYU TANDON SCHOOL OF ENGINEERING

Motivation

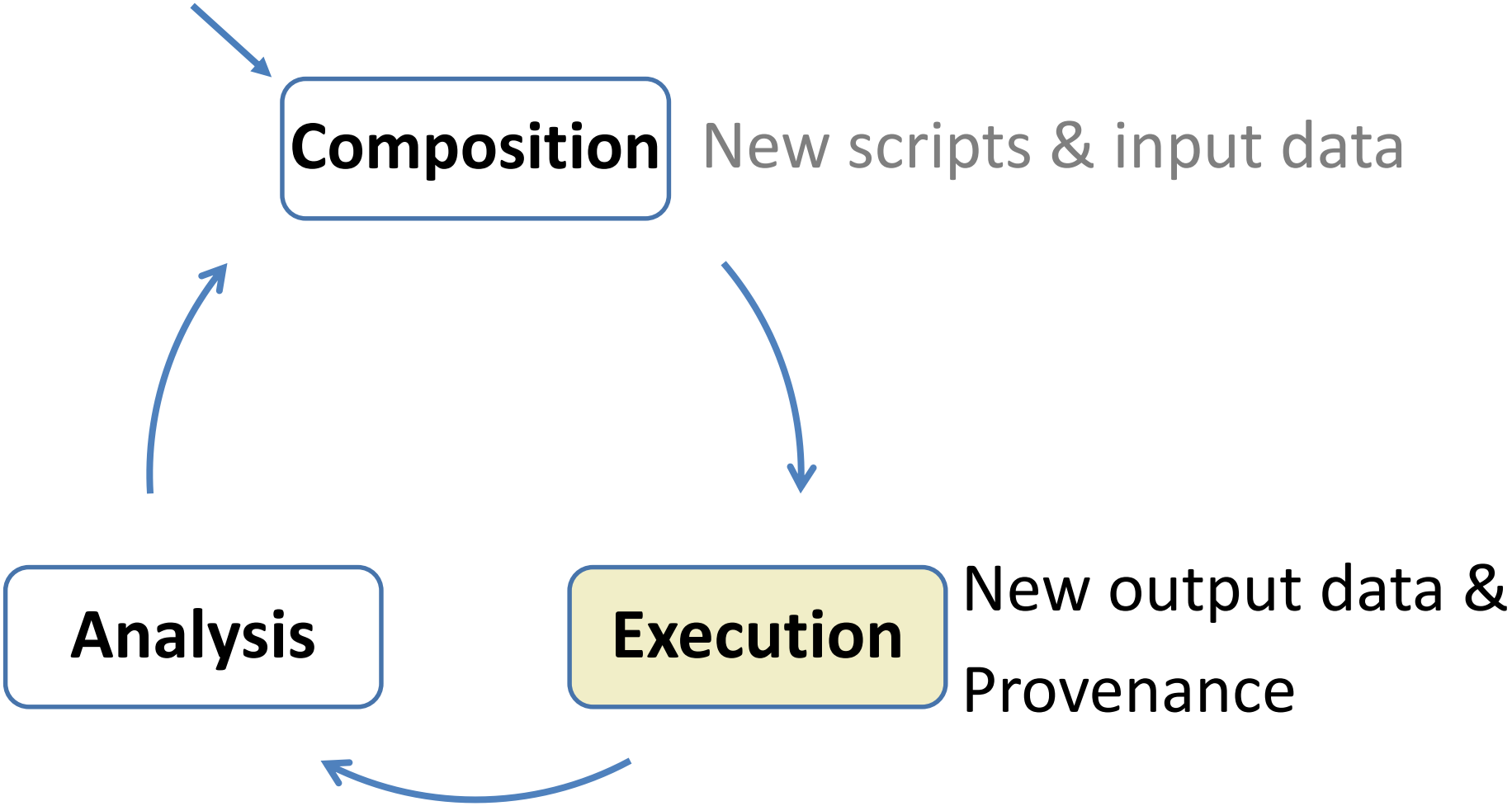
Life Cycle of Script Experiments



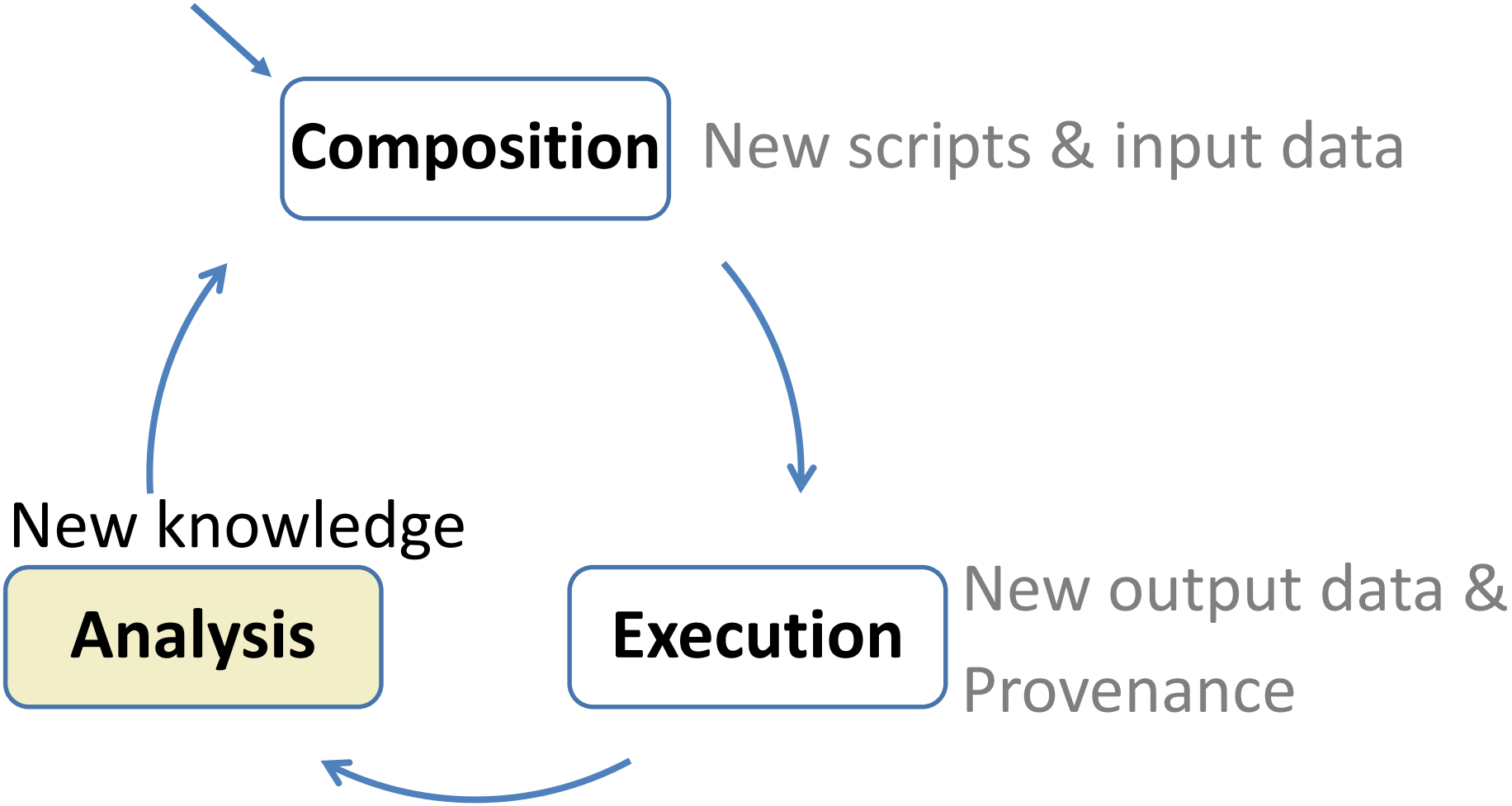
Life Cycle of Script Experiments



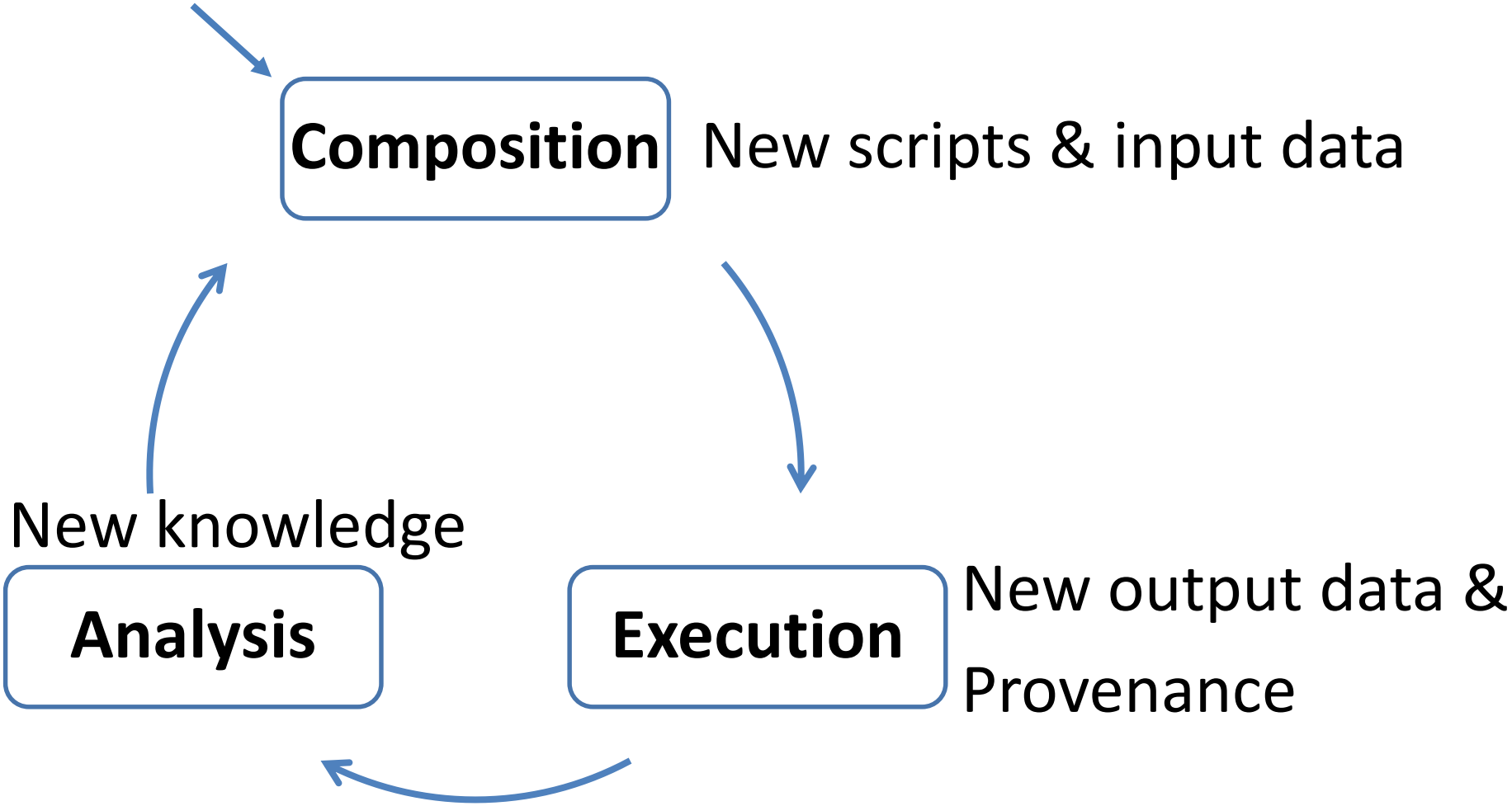
Life Cycle of Script Experiments



Life Cycle of Script Experiments



Trial and Error



Evolution

- Scripts evolve
- Input and Output Data evolve
- Provenance evolve

- It might be necessary to go back to old versions

Evolution

- Scripts evolve
- Input and Output Data evolve
- Provenance evolve

- It might be necessary to go back to old versions

- **How can we support this scenario of evolution and provide provenance for scientists without imposing extra effort?**

Provenance of Scripts

	Require Changes	Transparent
Coarse-grained	• YesWorkflow – McPhillips et al., 2015	• Sumatra – Davison, 2012
Fine-grained	• API – Bochner; Gude; Schreiber, 2008 • StarFlow – Angelino; Yamins; Seltzer, 2010 • RDataTracker – Lerner and Boose, 2014	• noWorkflow – Murta et al., 2014 • LLVM compiler – Tariq et al., 2012 • DataTracker – Stamatogiannakis et al., 2014

Provenance of Scripts

Coarse-grained

Fine-grained

Require Changes

- YesWorkflow
  - McPhillips et al., 2015
- API
  - Bochner; Gude; Schreiber, 2008
- StarFlow
  - Angelino; Yamins; Seltzer, 2010
- RDataTracker
  - Lerner and Boose, 2014

Transparent

- Sumatra
  - Davison, 2012
- noWorkflow
  - Murta et al., 2014
- LLVM compiler
  - Tariq et al., 2012
- DataTracker
  - Stamatogiannakis et al., 2014

Provenance of Scripts

Coarse-grained

Fine-grained

Require Changes

- YesWorkflow
  - McPhillips et al., 2015
- API
  - Bochner; Gude; Schreiber, 2008
- StarFlow
  - Angelino; Yamins; Seltzer, 2010
- RDataTracker
  - Lerner and Boose, 2014

Transparent

- Sumatra
  - Davison, 2012
- noWorkflow
  - Murta et al., 2014
- LLVM compiler
  - Tariq et al., 2012
- DataTracker
  - Stamatogiannakis et al., 2014

Provenance of Scripts

Coarse-grained

Fine-grained

Require Changes

- YesWorkflow
  - McPhillips et al., 2015
- API
  - Bochner; Gude; Schreiber, 2008
- StarFlow
  - Angelino; Yamins; Seltzer, 2010
- RDataTracker
  - Lerner and Boose, 2014

Transparent

- Sumatra
  - Davison, 2012
- noWorkflow
  - Murta et al., 2014
- LLVM compiler
  - Tariq et al., 2012
- DataTracker
  - Stamatogiannakis et al., 2014

Provenance of Scripts

Coarse-grained

Fine-grained

Require Changes

- YesWorkflow
  - McPhillips et al., 2015
- API
  - Bochner; Gude; Schreiber, 2008
- StarFlow
  - Angelino; Yamins; Seltzer, 2010
- RDataTracker
  - Lerner and Boose, 2014

Transparent

- Sumatra
  - Davison, 2012
- noWorkflow
  - Murta et al., 2014
- LLVM compiler
  - Tariq et al., 2012
- DataTracker
  - Stamatogiannakis et al., 2014

Approach

Versioning

- Evolution-aware provenance-capturing tool
  - Visualize and navigate on the evolution history
  - Compare trials
- Fine-grained collection
  - Track intermediate files
  - Execution trace (optionally)

Proof of concept

Coarse-grained

Fine-grained

Require Changes

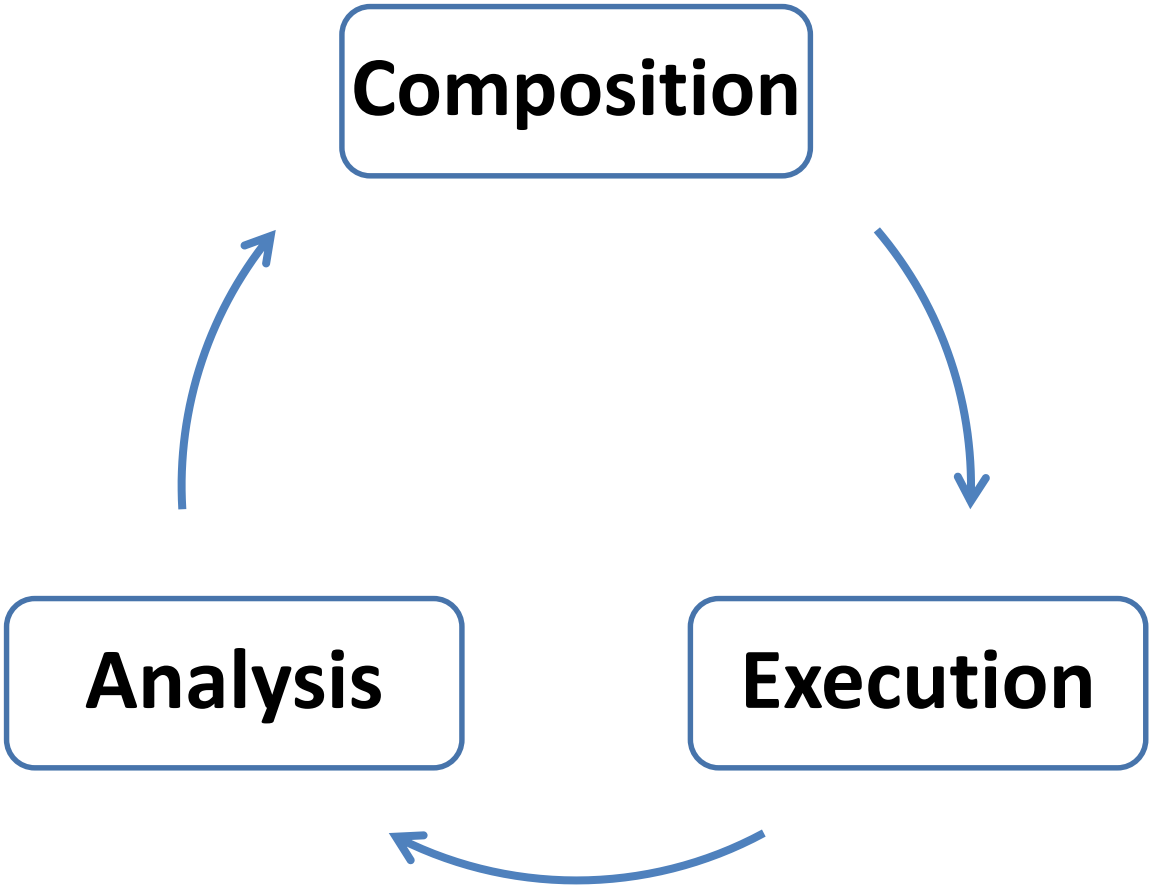
- YesWorkflow
  - McPhillips et al., 2015
- API
  - Bochner; Gude; Schreiber, 2008
- StarFlow
  - Angelino; Yamins; Seltzer, 2010
- RDataTracker
  - Lerner and Boose, 2014

Transparent

- Sumatra
  - Davison, 2012
- noWorkflow
  - Murta et al., 2014
- LLVM compiler
  - Tariq et al., 2012
- DataTracker
  - Stamatogiannakis et al., 2014

Guiding Example

Life Cycle of Experiments

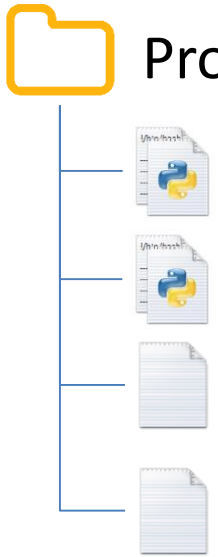


New experiment!

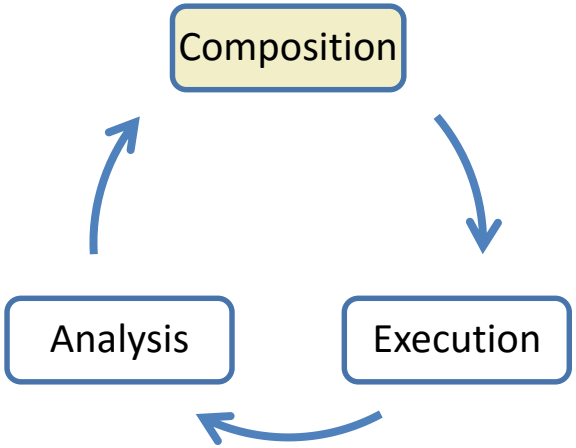
Could you check if the precipitation of Rio de Janeiro remains constant across years?

1st Iteration

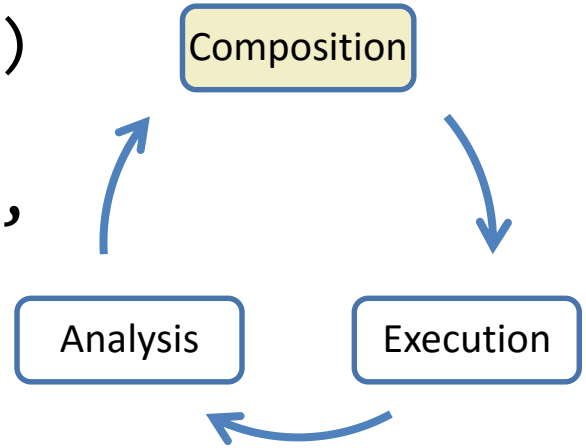
- H_1: "The precipitation for each month remains constant across years"



Data: 2013, 2014 [BDMEP]

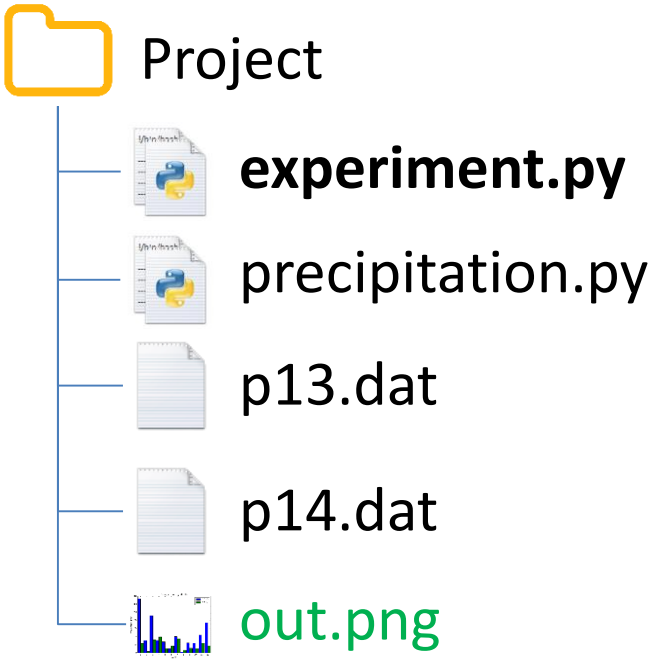


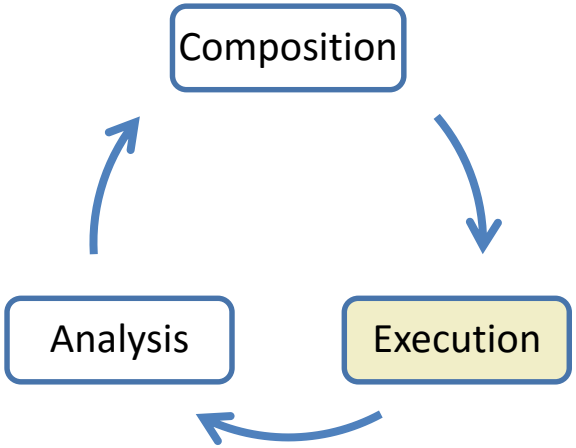
```
1| import numpy as np
2| from precipitation import read, sum_by_month
3| from precipitation import create_bargraph
4|
5| months = np.arange(12) + 1
6|
7| d13, d14 = read("p13.dat"), read("p14.dat")
8|
9| prec13 = sum_by_month(d13, months)
10| prec14 = sum_by_month(d14, months)
11|
12| create_bargraph("out.png", months,
13|    ["2013", "2014"],
14|    prec13, prec14)
```



Trial

$ python experiment.py

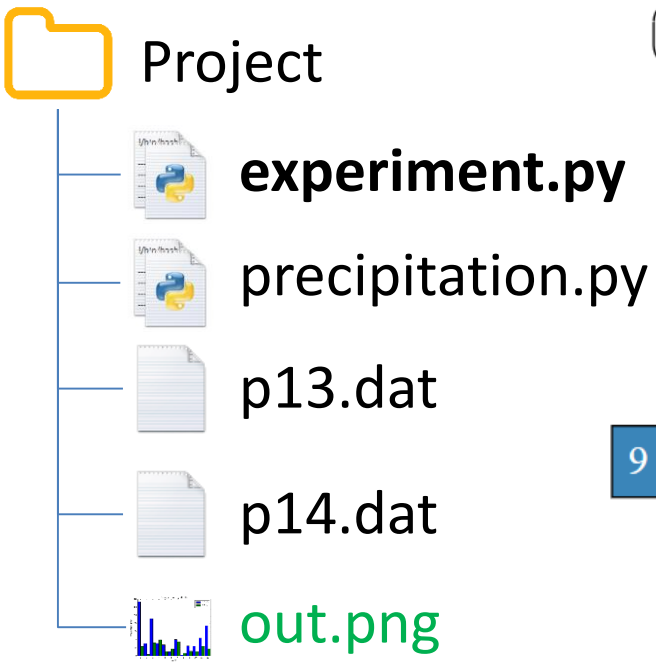


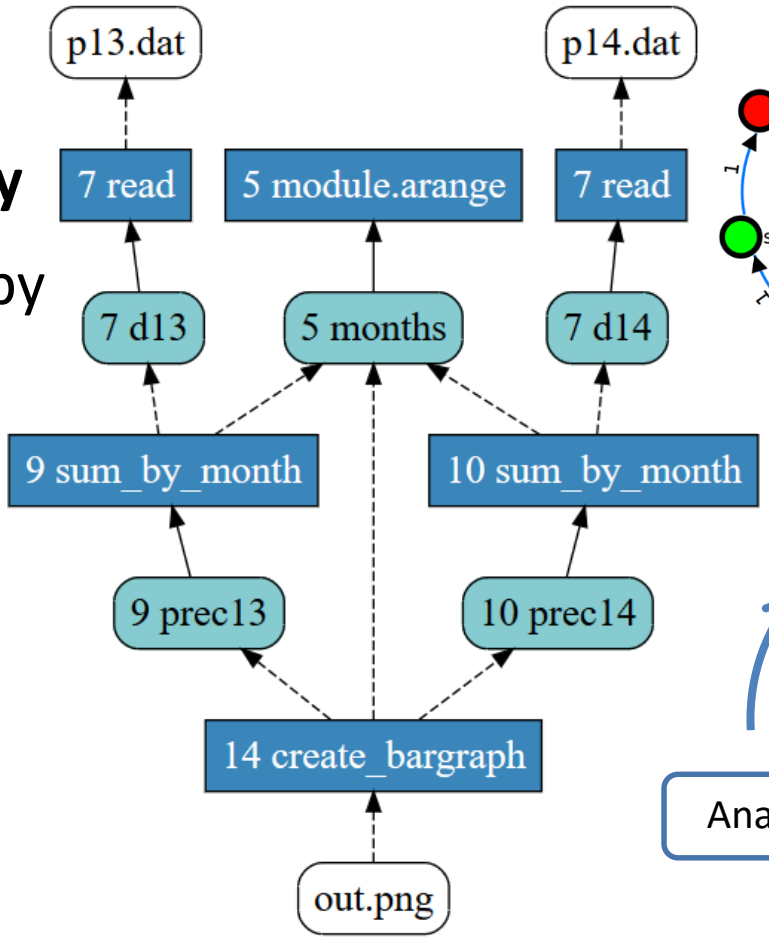


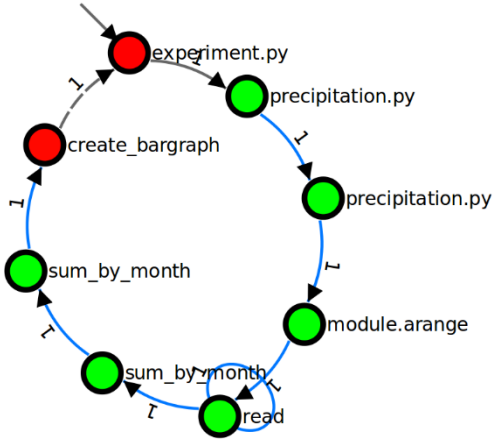
What happened during the execution?

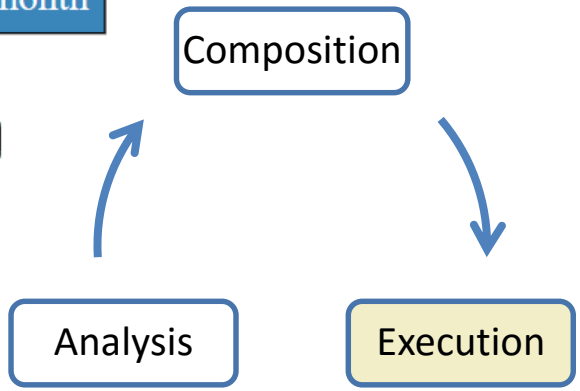
Provenance

$ **now run -e Tracker** experiment.py









Version Model

Trial History

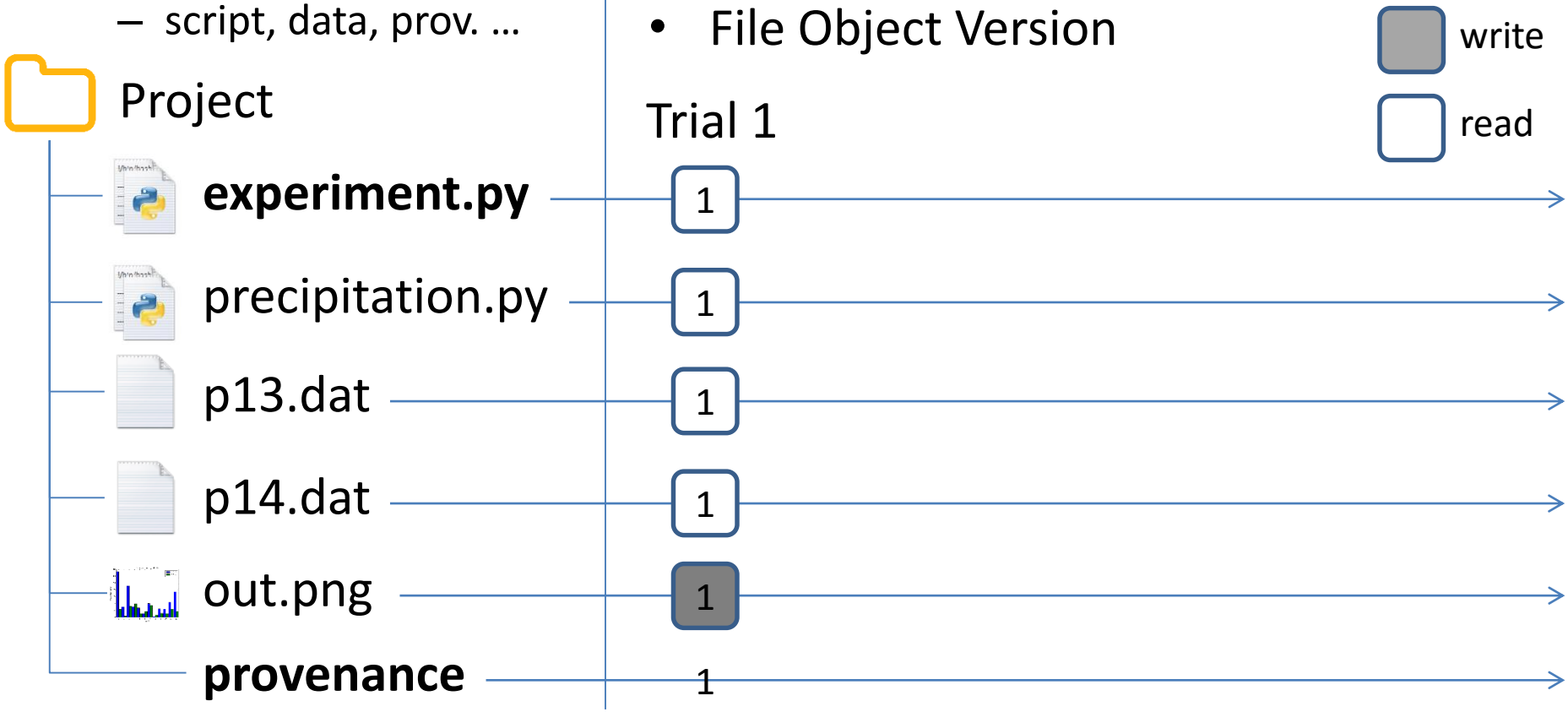
1

Product Space

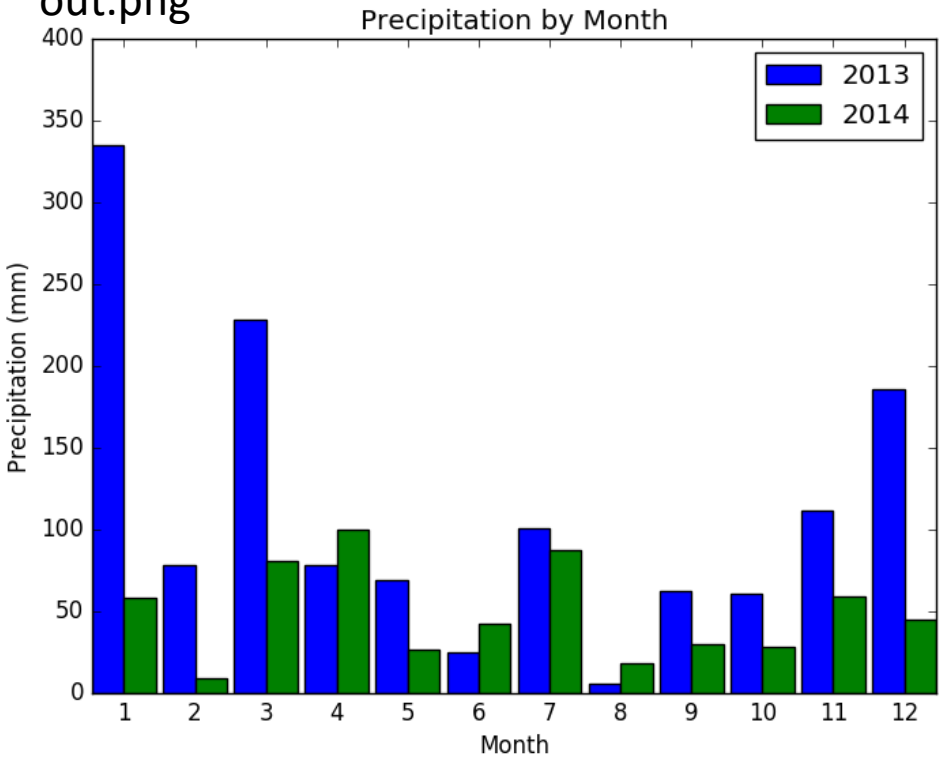
- Experiment itself
  - script, data, prov. …

Version Space

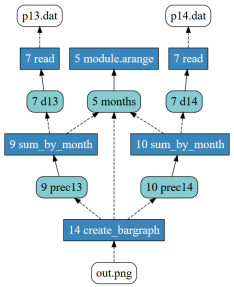
- Trial Version
- File Object Version

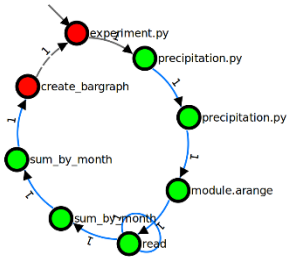


out.png

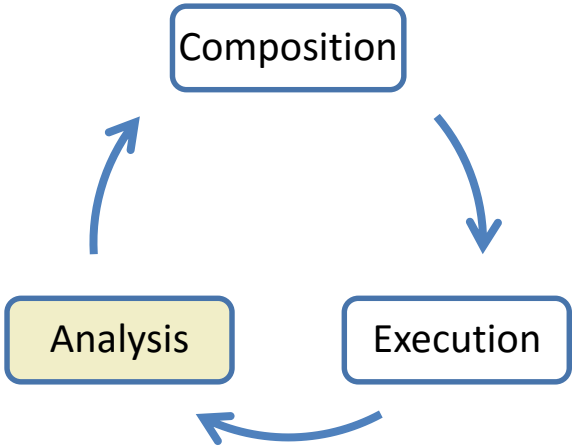


SELECT …



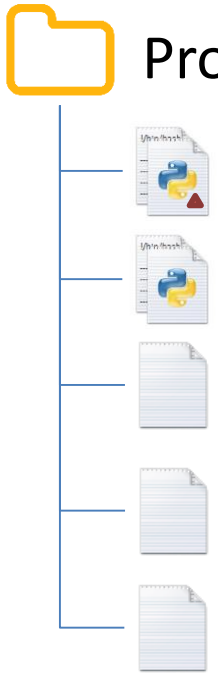


Conclusion:
“Drought in 2014”

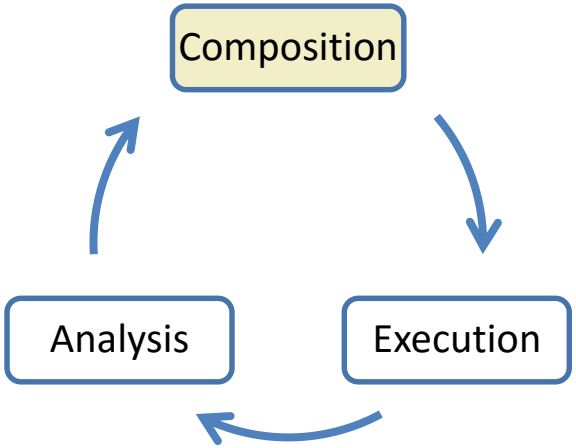


2nd Iteration

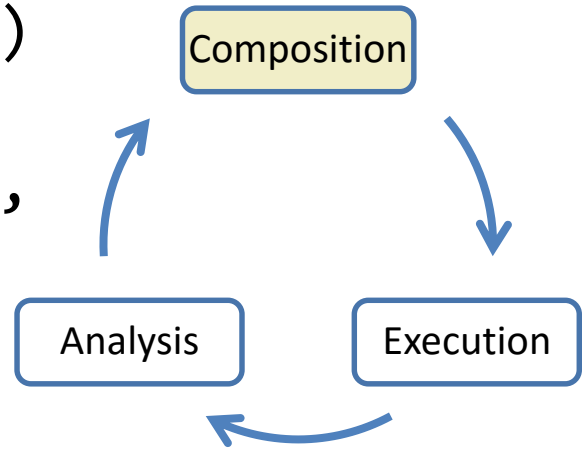
- H_2: "The precipitation for each month remains constant across years if there is no drought"



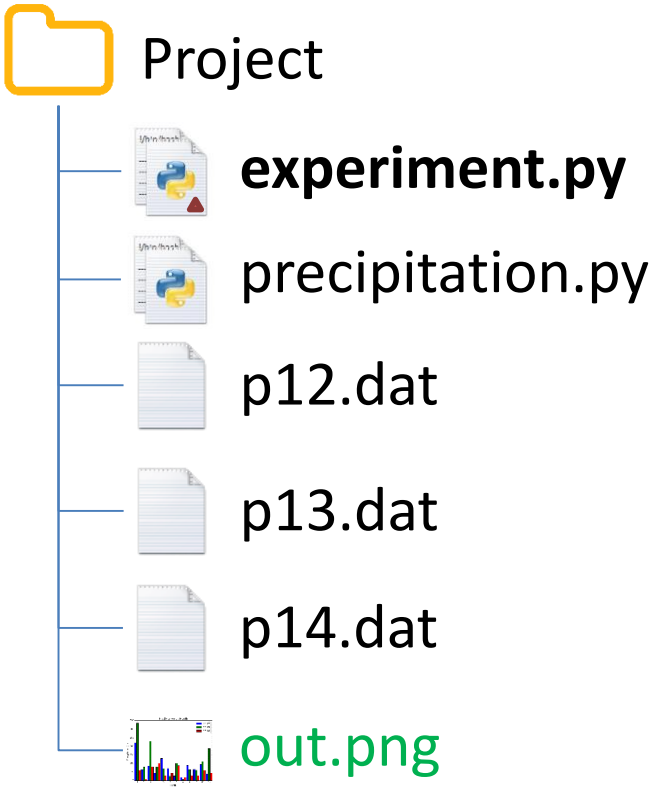
Data: 2012, 2013, 2014 [BDMEP]

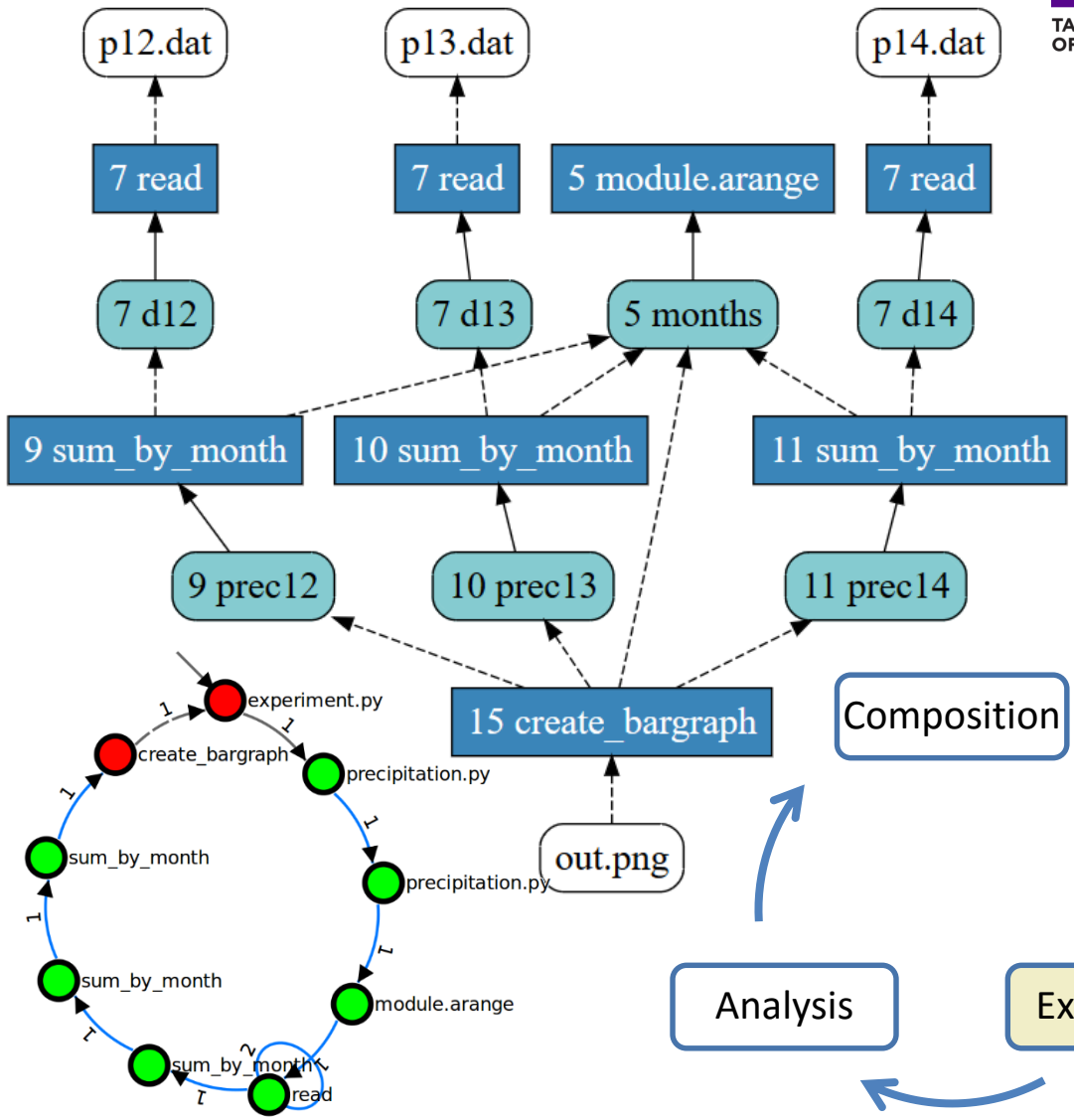


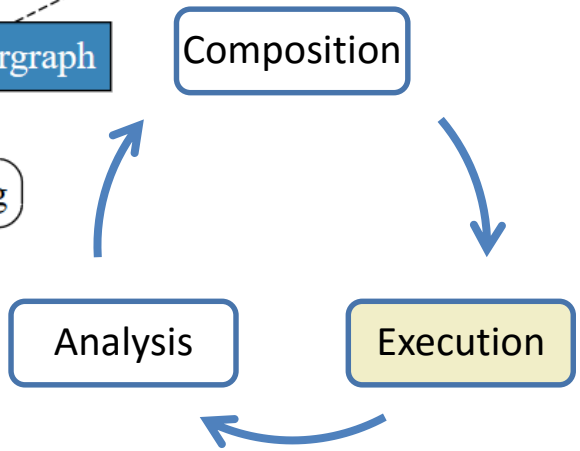
```
1| import numpy as np
2| from precipitation import read, sum_by_month
3| from precipitation import create_bargraph
4|
5| months = np.arange(12) + 1
6| d12 = read("p12.dat")
7| d13, d14 = read("p13.dat"), read("p14.dat")
8| prec12 = sum_by_month(d12, months)
9| prec13 = sum_by_month(d13, months)
10| prec14 = sum_by_month(d14, months)
11|
12| create_bargraph("out.png", months,
13|    ["2012", "2013", "2014"],
14|    prec12, prec13, prec14)
```



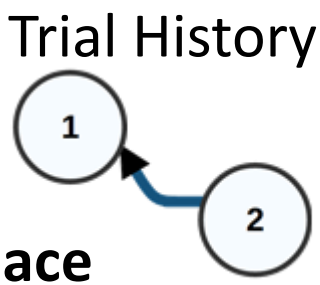
$ now run -e Tracker experiment.py

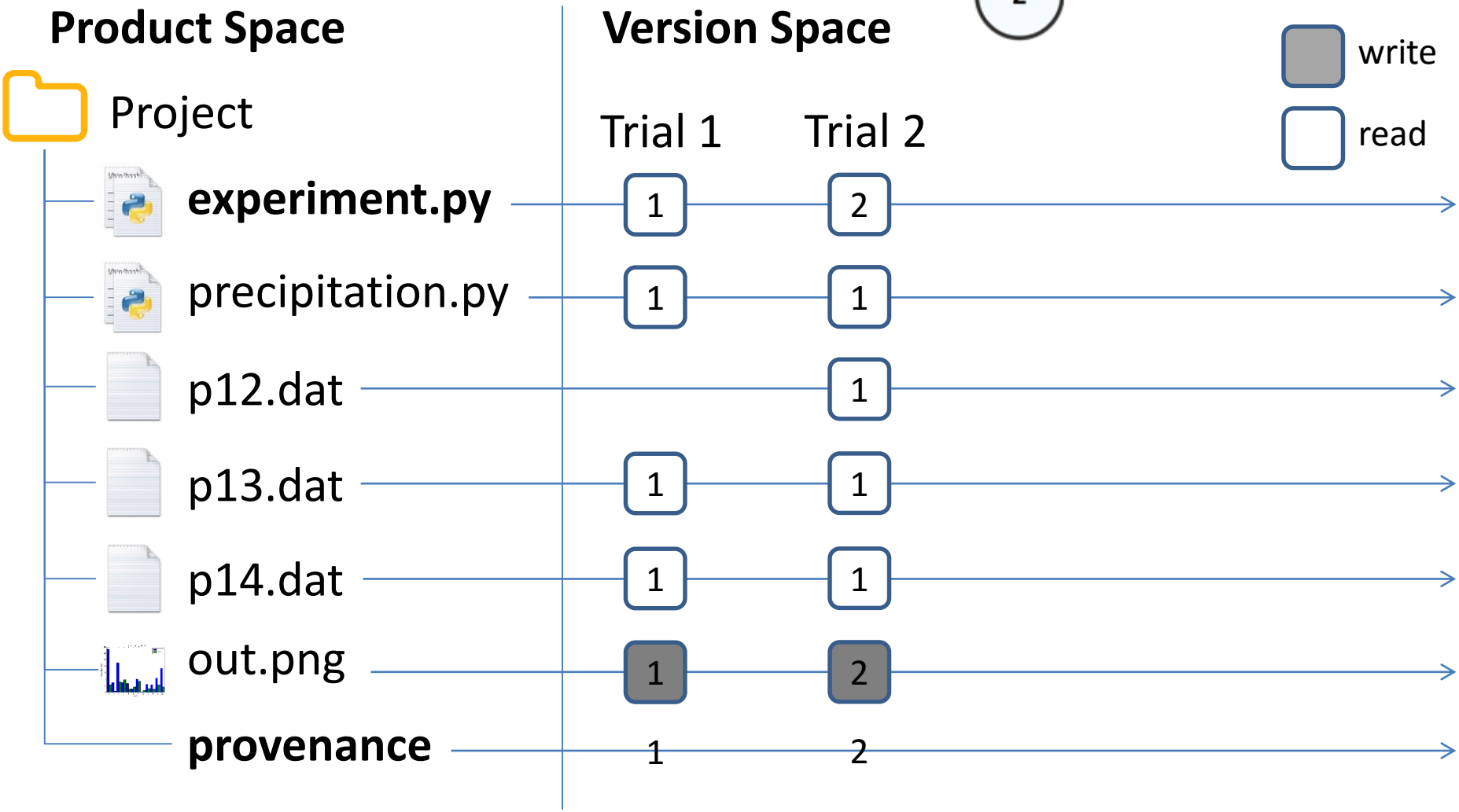




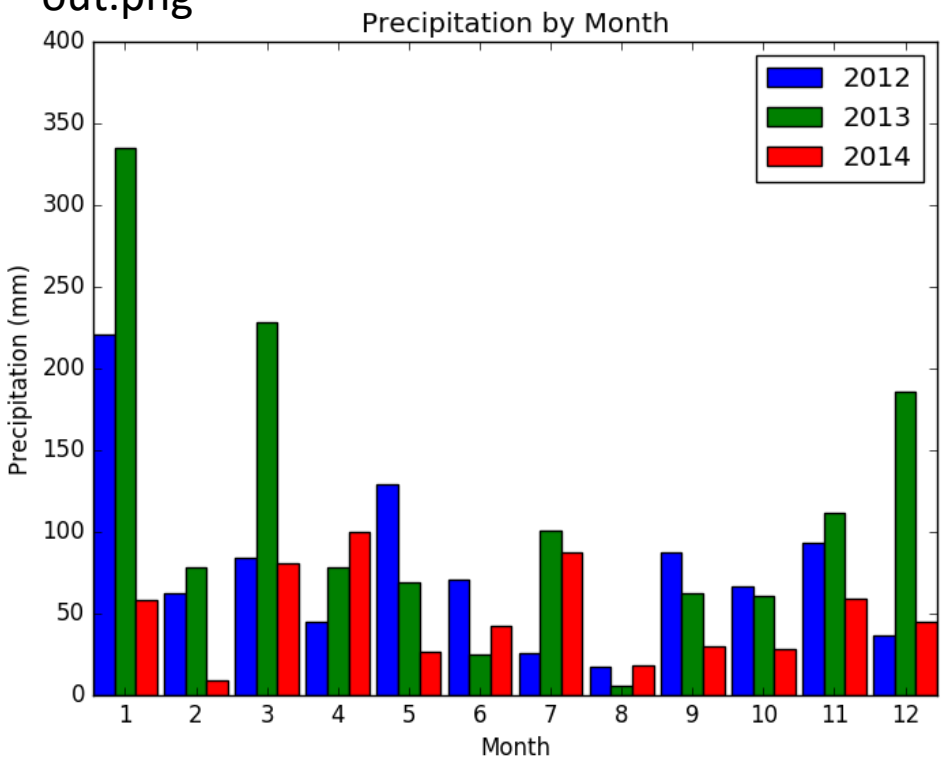


Version Model

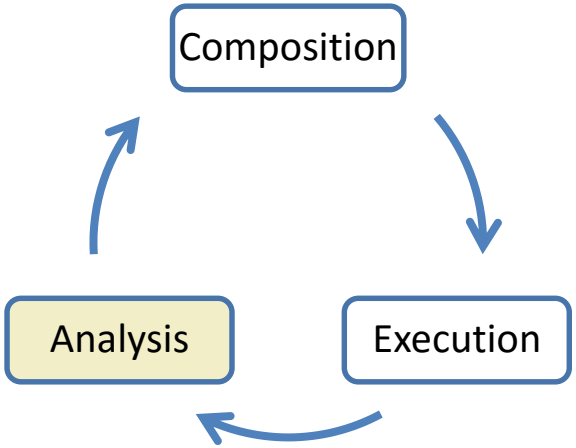




out.png

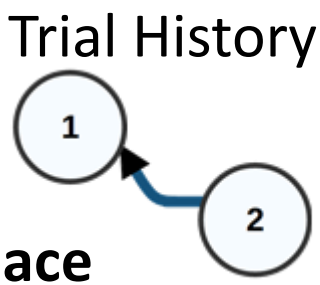


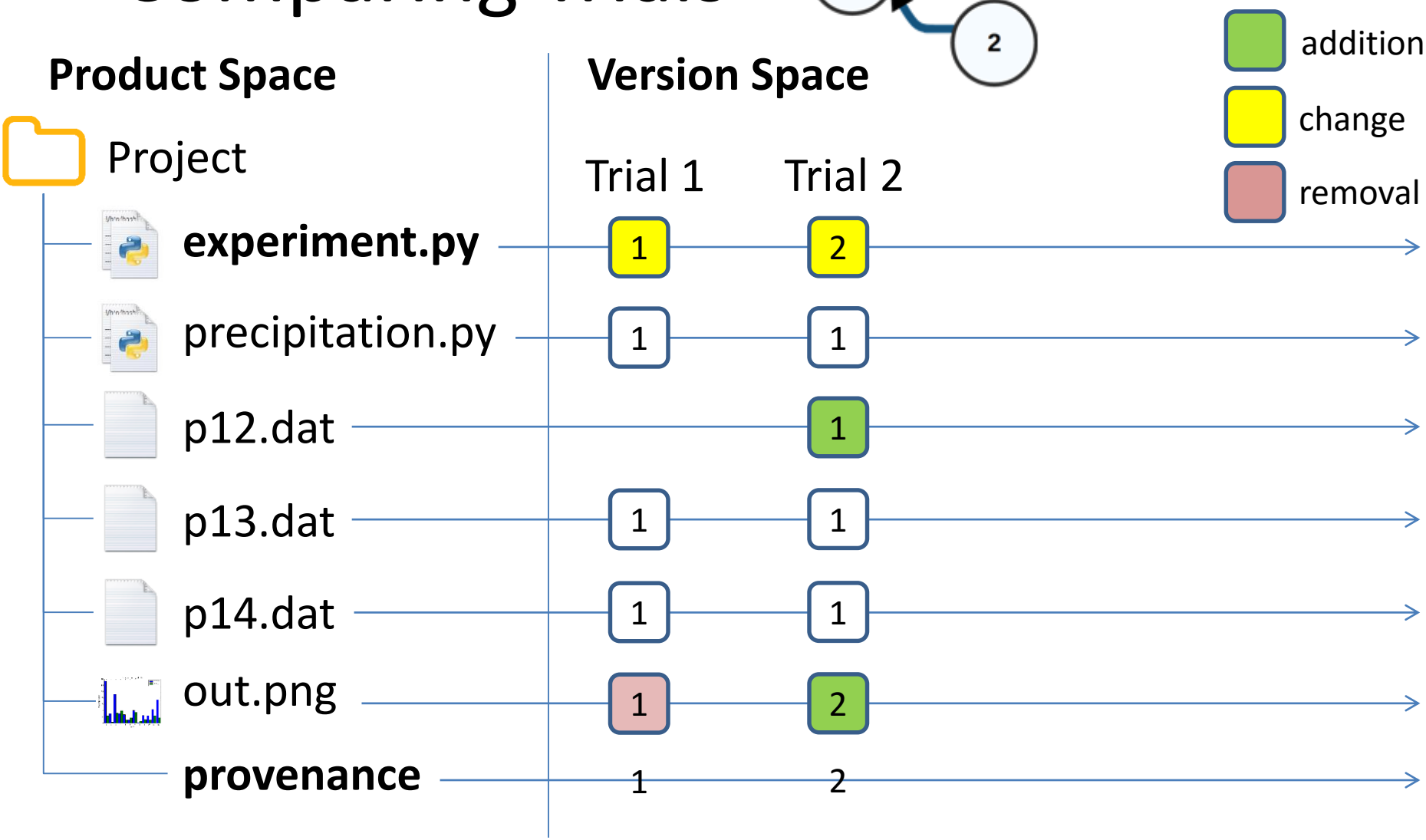
Conclusion:
"2012 was similar to 2013"



What were the differences between Trial 1 and Trial 2?

Comparing Trials



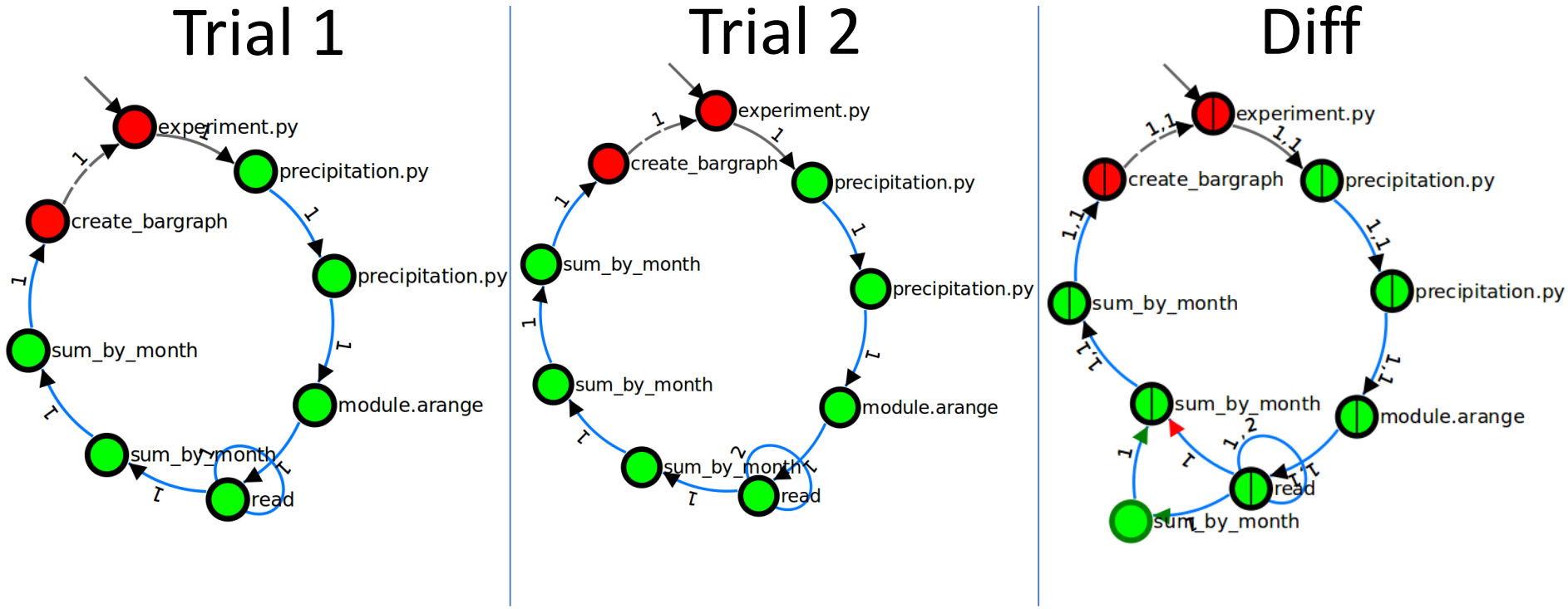


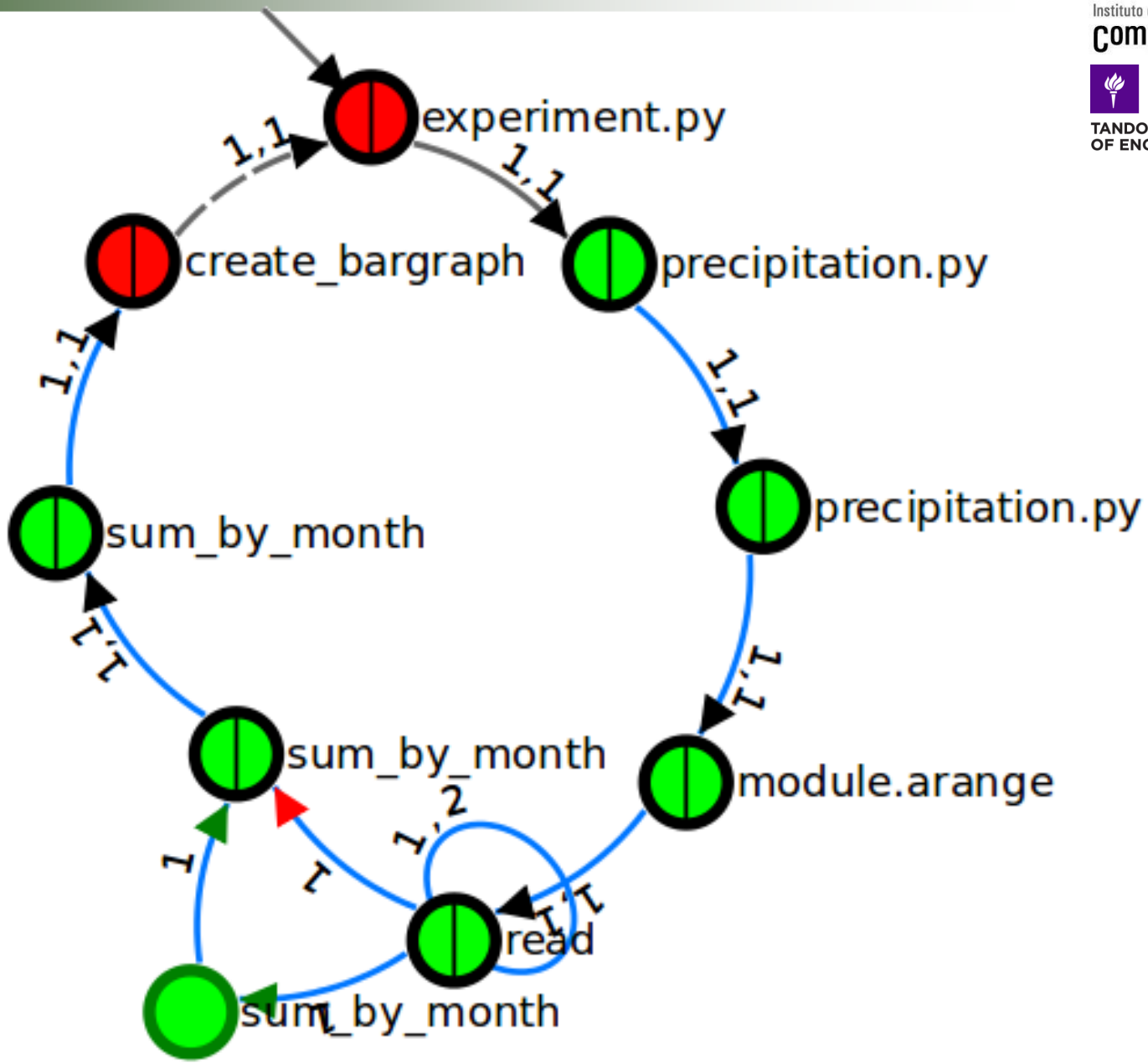
noWorkflow Diff

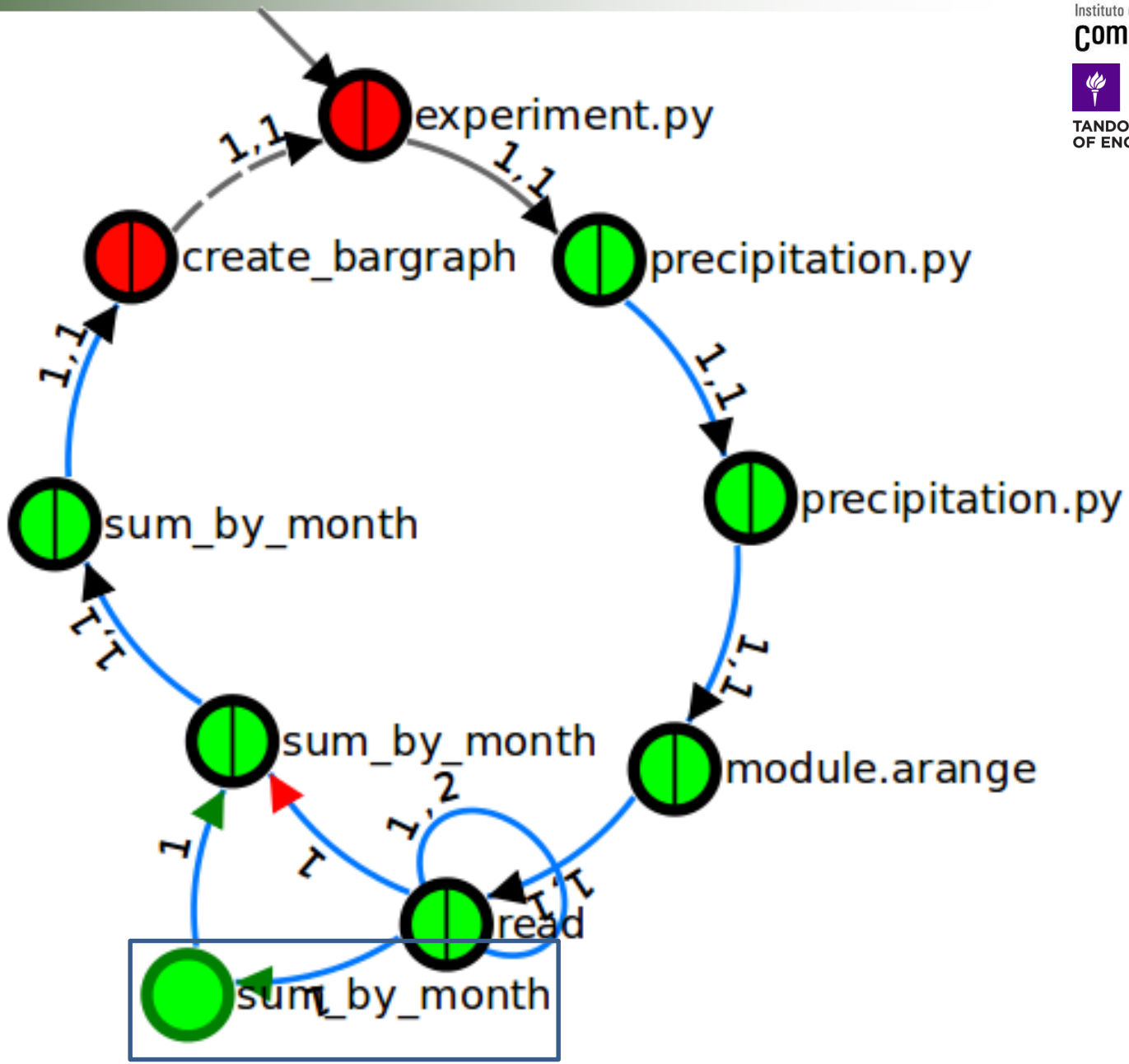
```
$ now diff 1 2 -f --brief
[now] trial diff:
  Start changed from 2016-05-30 19:03:26.105716
                  to 2016-05-30 19:05:26.276369
  Finish changed from 2016-05-30 19:04:27.729060
                   to 2016-05-30 19:06:24.863268
  Duration text changed from 0:01:01.623344 to 0:00:58.586899
  Code hash changed from a66f3052414673feed5e49812e6940a92bba7679
                     to ff62d0f369315fbc209c39379ccf93437725fa31
  Parent id changed from <None> to 1

[now] Brief file access diff
[Additions]   | [Removals]          | [Changes]
(r) p12.dat   | (wb) out.png (new)  |
(wb) out.png  |                     |
```

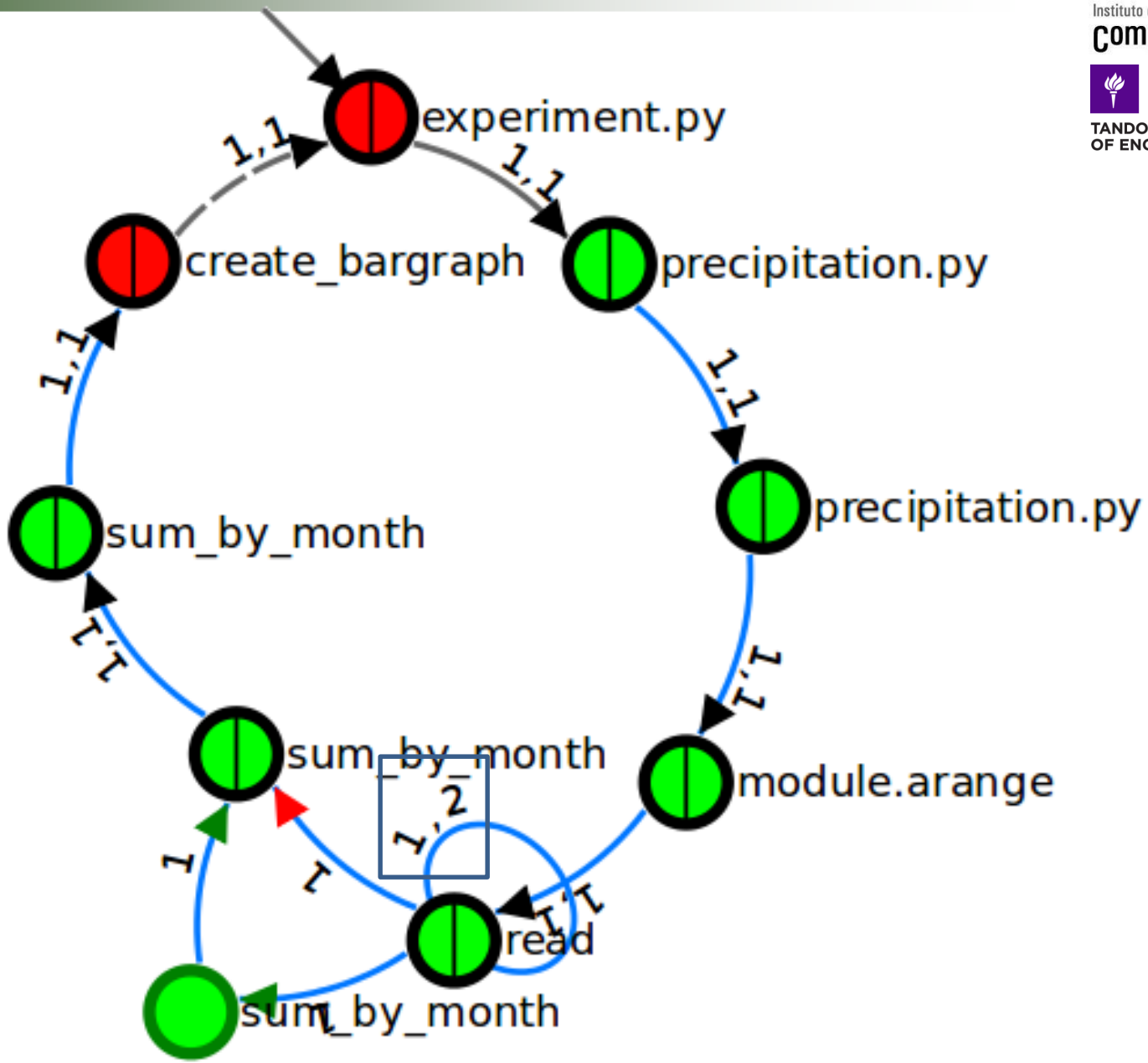
Activations Diff

- Visualization tool: `now vis`




experiment.py
1,1
1,1
create_bargraph
precipitation.py
1,1
1,1
sum_by_month
precipitation.py
1,1
1,1
sum_by_month
module.arange
2
1,1
1
1
1,1
read
sum_by_month

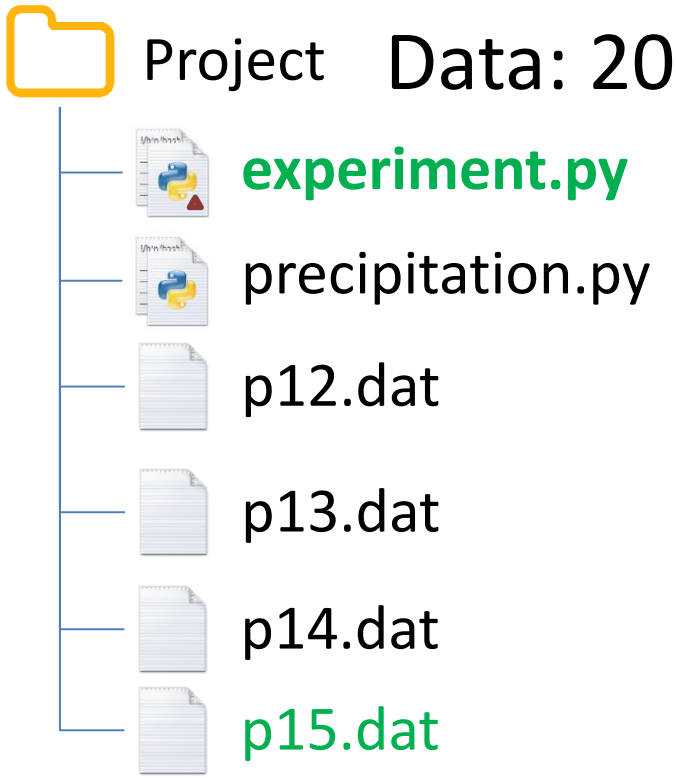

experiment.py
1,1
1,1
create_bargraph
precipitation.py
1,1
1,1
sum_by_month
precipitation.py
1,1
1,1
sum_by_month
module.arange
2
1,1
1
1
1,1
read
sum_by_month

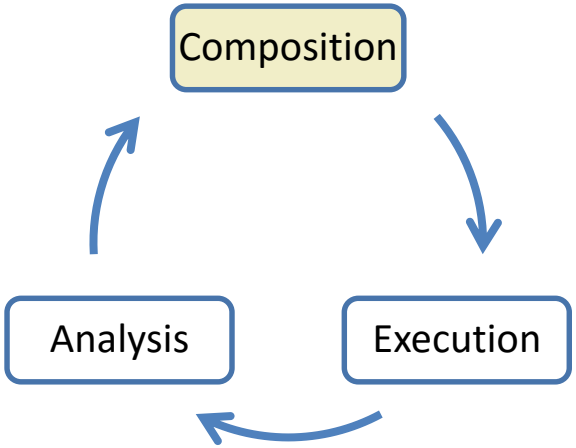

experiment.py
1,1
1,1
create_bargraph
precipitation.py
1,1
1,1
sum_by_month
precipitation.py
1,1
1,1
sum_by_month
module.arange
1,2
1
1
1,1
read
sum_by_month

I don’t think its enough to compare just these years. Could you add data from 2015?

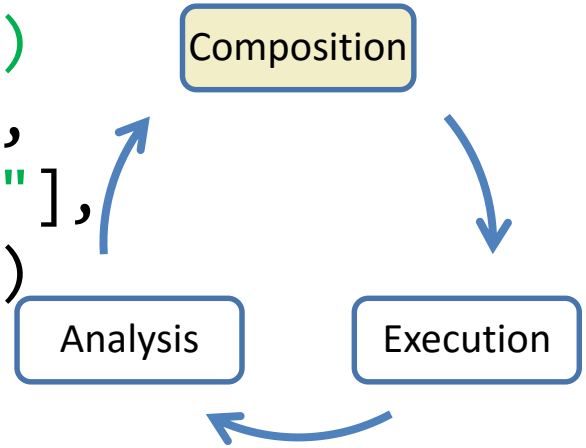
3rd Iteration

- H_2: "The precipitation for each month remains constant across years if there is no drought"





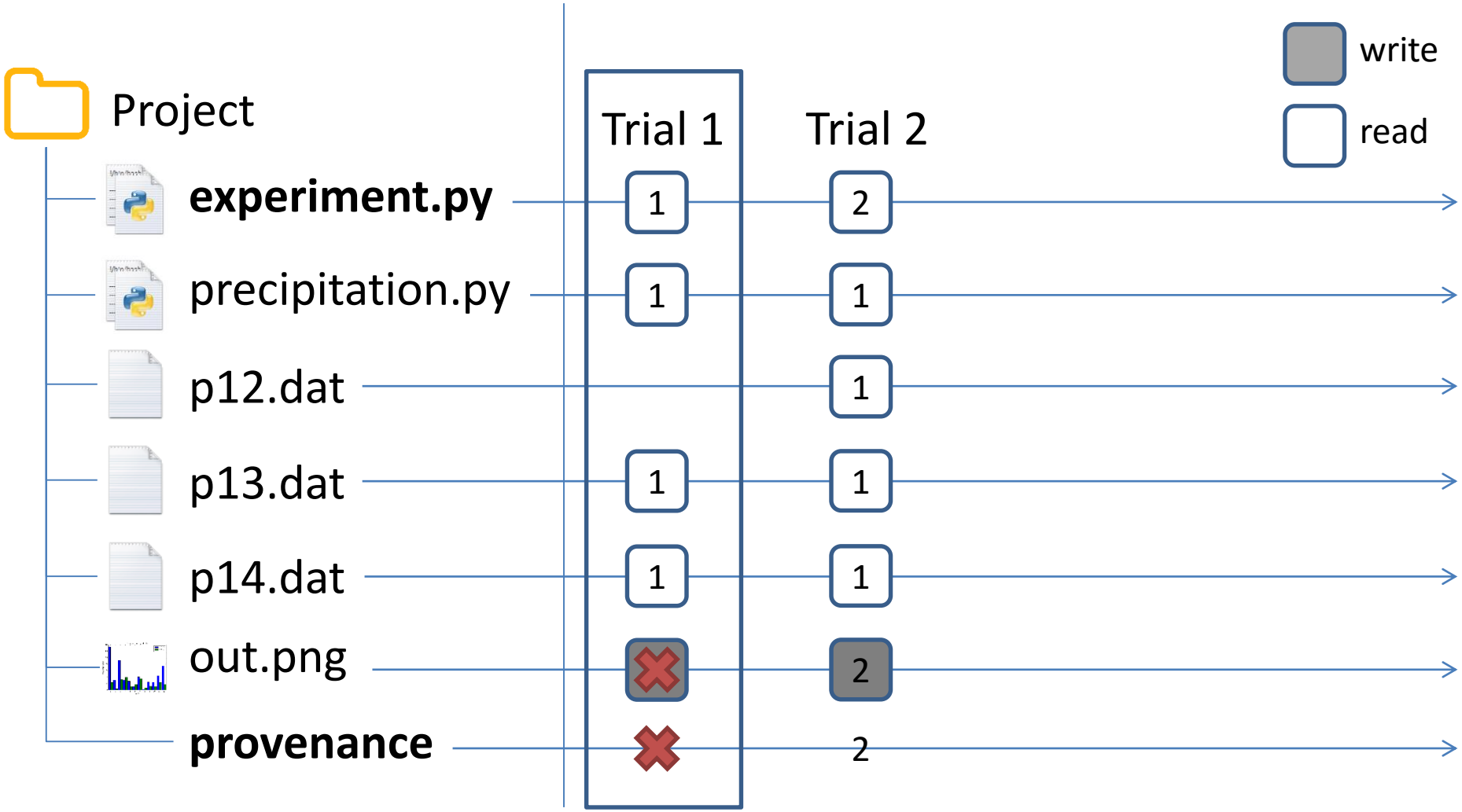
```
1| import numpy as np
2| from precipitation import read, sum_by_month
3| from precipitation import create_bargraph
4|
5| months = np.arange(12) + 1
6| d12, d15 = read("p12.dat"), read("p15.dat")
7| d13, d14 = read("p13.dat"), read("p14.dat")
8| prec12 = sum_by_month(d12, months)
9| prec13 = sum_by_month(d13, months)
10| prec14 = sum_by_month(d14, months)
11| prec15 = sum_by_month(d15, months)
12| create_bargraph("out.png", months,
13|    ["2012", "2013", "2014", "2015"],
14|    prec12, prec13, prec14, prec15)
```



In the meantime

Forget what I said. There are unusual rainy days. Could you repeat the first trial without them?

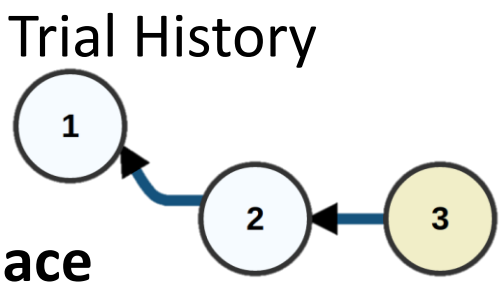
$ now restore 1



What about the 3rd iteration?

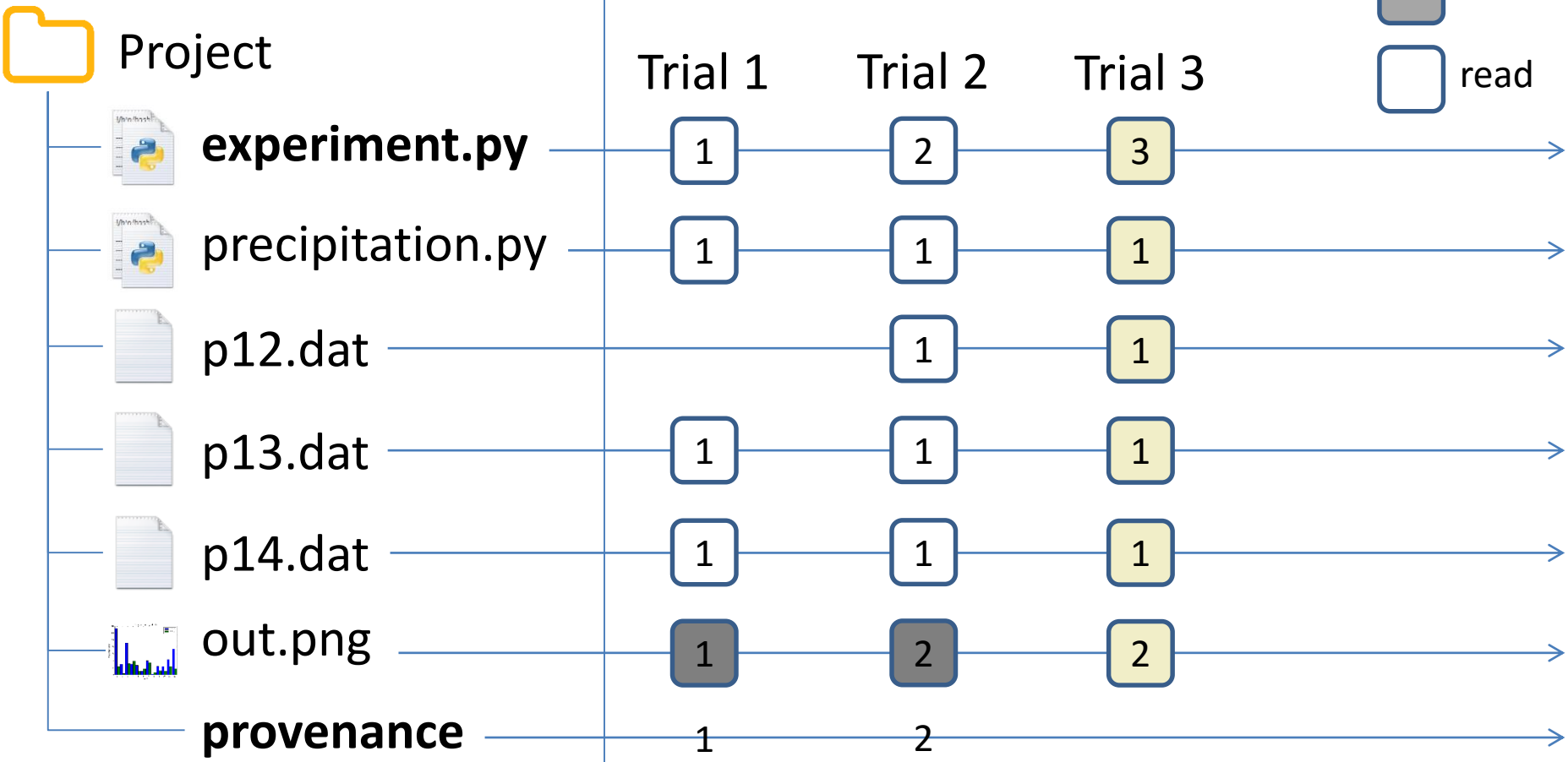
~~3rd Iteration~~

Backup Trial



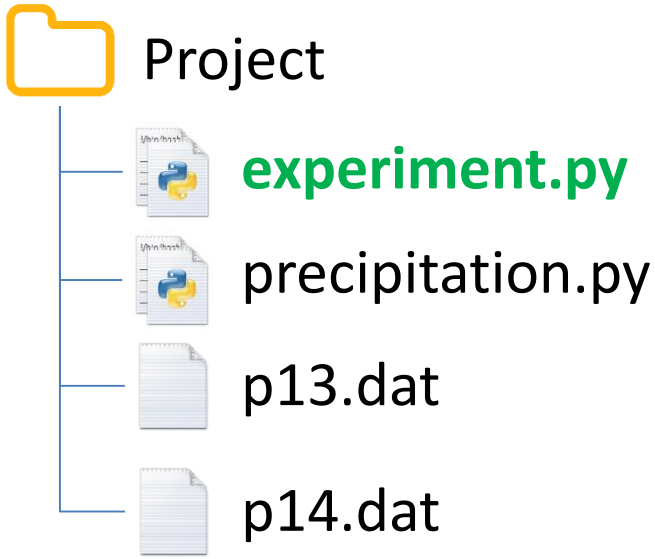
Product Space

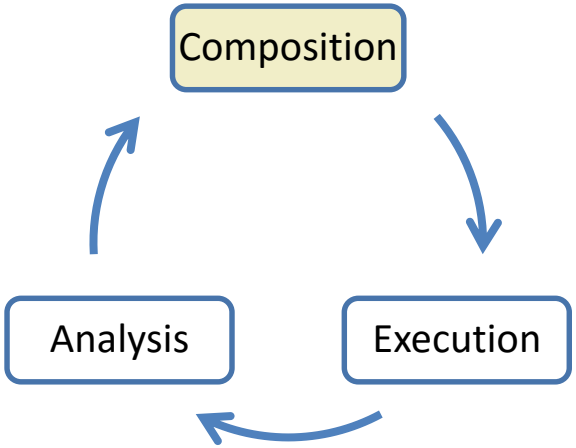
Version Space



4th Iteration

- H_3: "The precipitation for each month remains constant across years if we disregard unusual days"
- Data: 2013, 2014 [BDMEP]



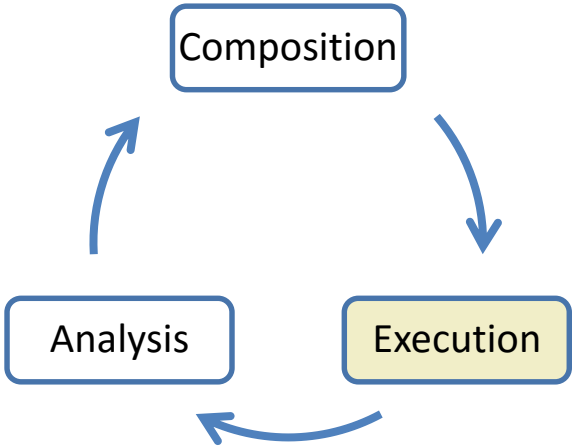


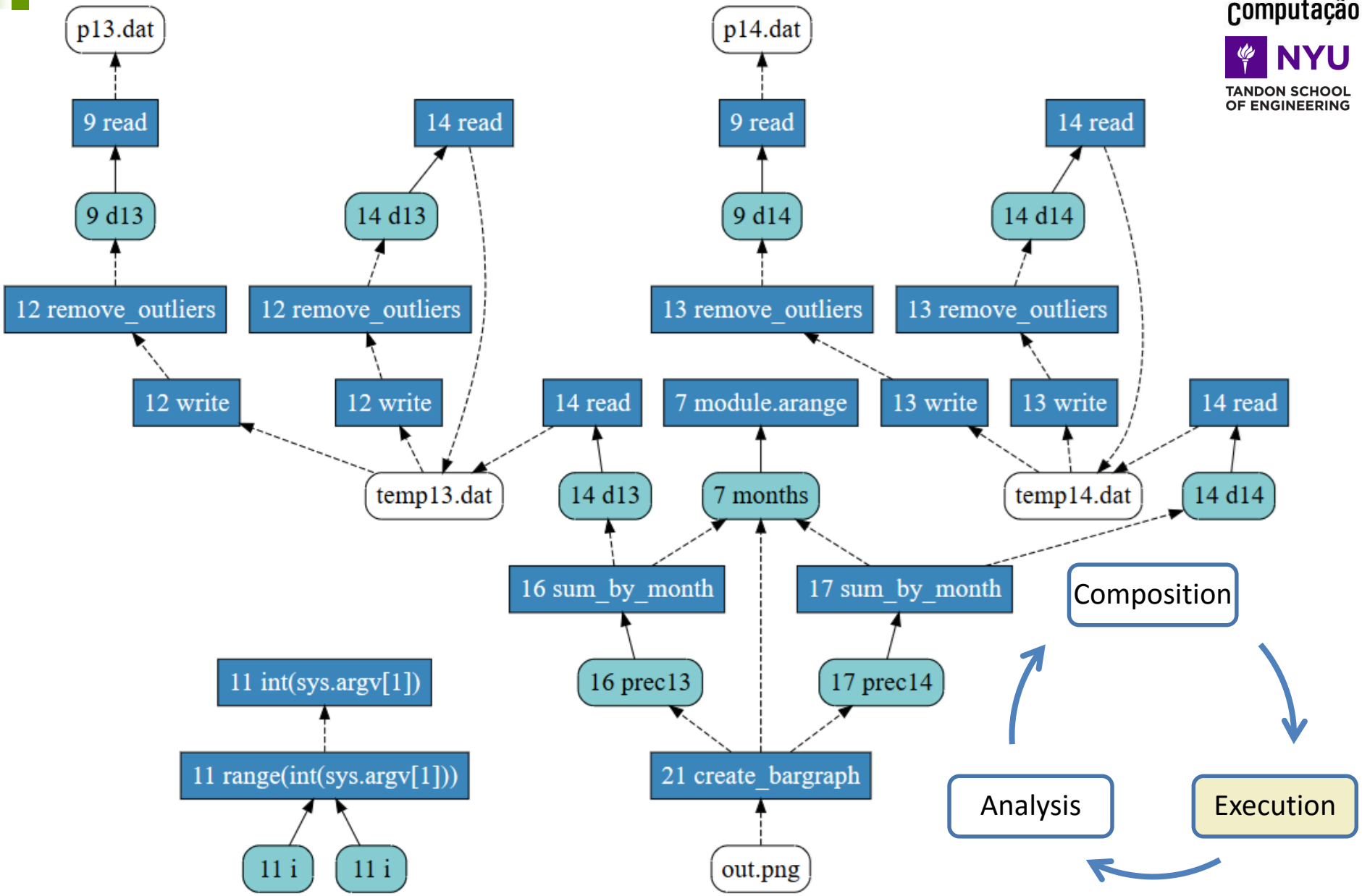
```
1|  import sys
5|  from precipitation import write, remove_outliers
7|  months = np.arange(12) + 1
9|  d13, d14 = read("p13.dat"), read("p14.dat")
10|
11|  for i in range(int(sys.argv[1])):
12|     write("temp13.dat",remove_outliers(d13), 2013)
13|     write("temp14.dat",remove_outliers(d14), 2014)
14|     d13,d14=read("temp13.dat"), read("temp14.dat")
15|
16|  prec13 = sum_by_month(d13, months)
17|  prec14 = sum_by_month(d14, months)
19|  create_bargraph("out.png", months,
20|                  ["2013", "2014"],
21|                  prec13, prec14)
```

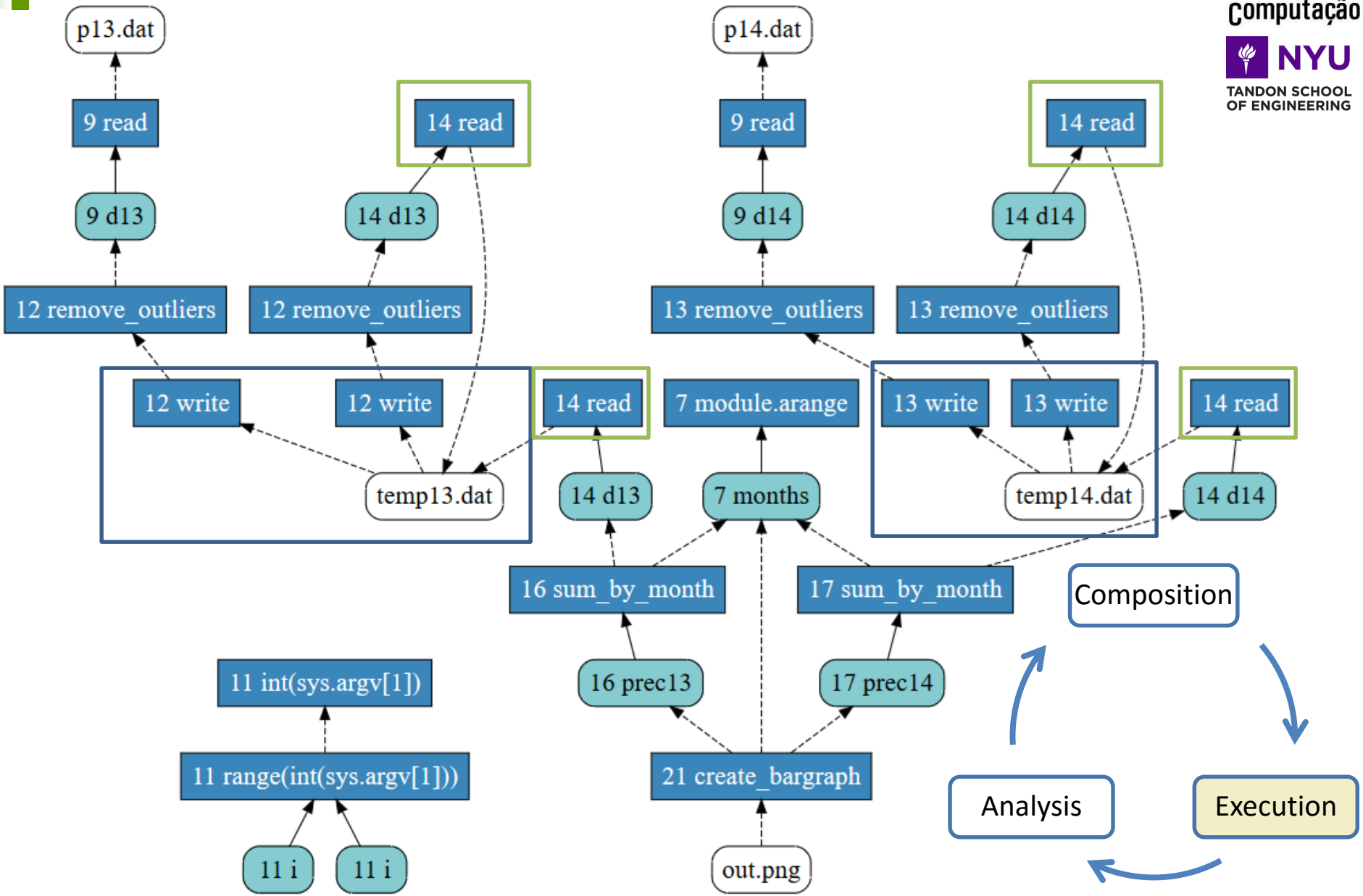
$ now run -e Tracker experiment.py **2**

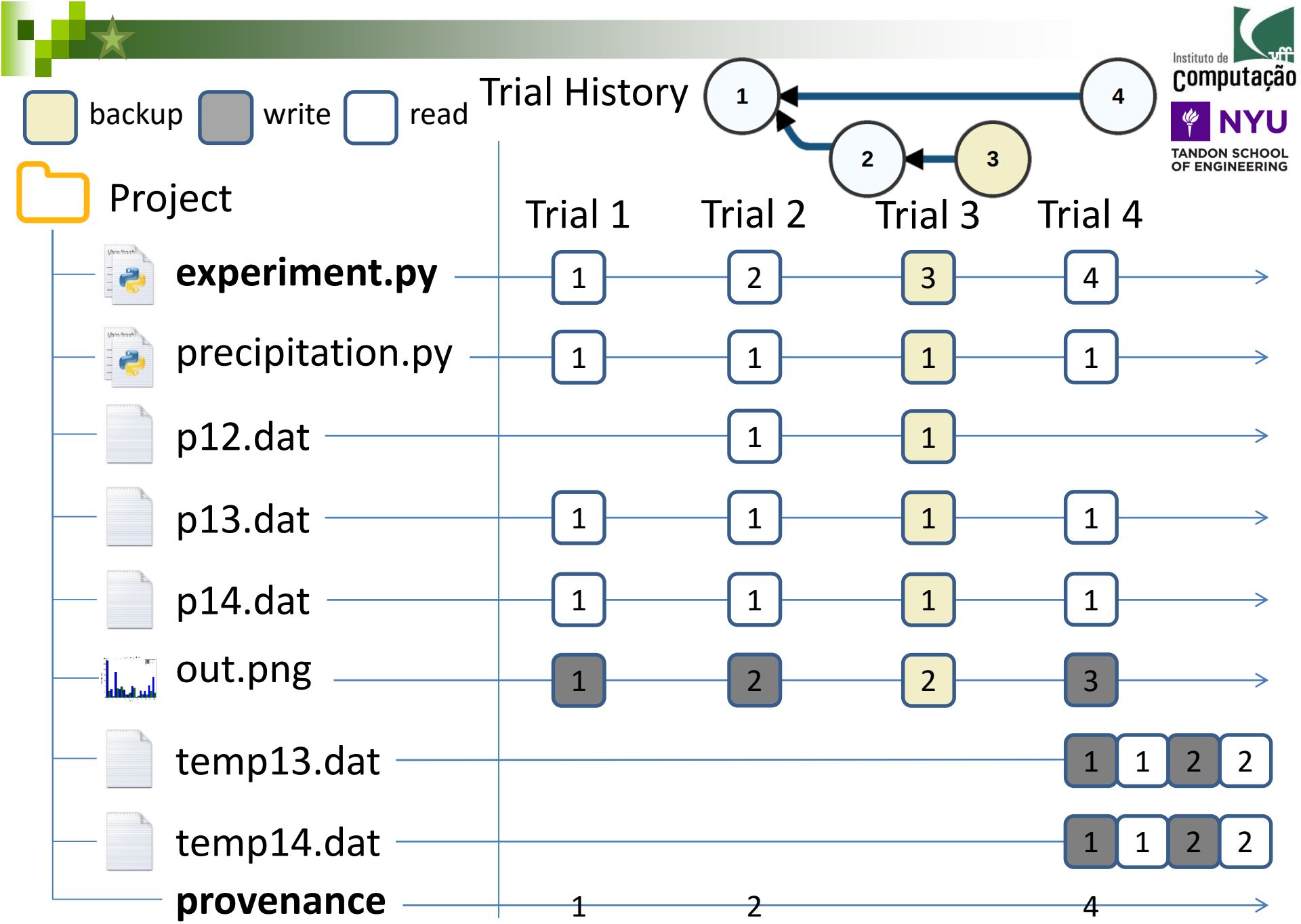
Project

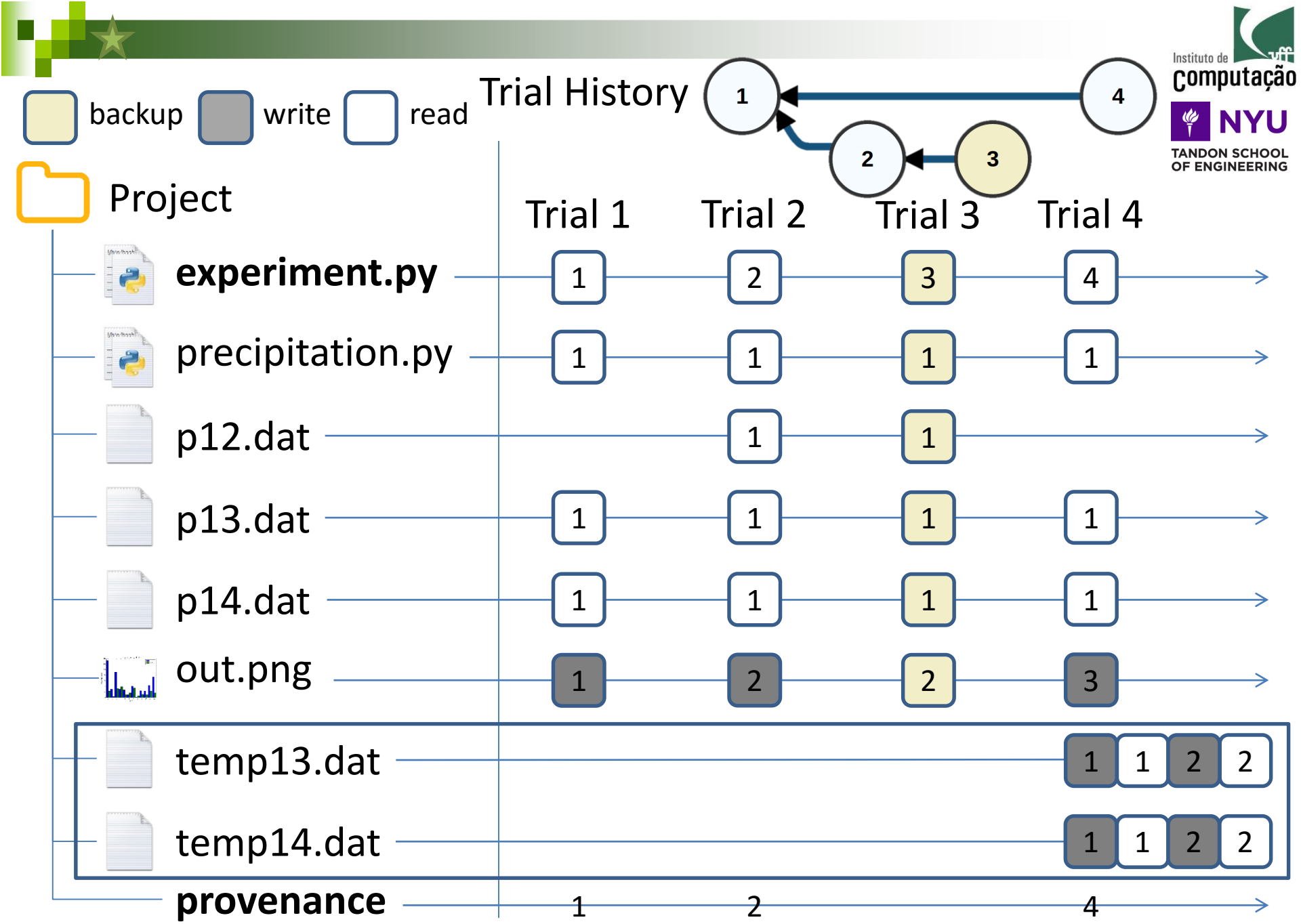
- **experiment.py**
- precipitation.py
- p12.dat
- p13.dat
- p14.dat
- out.png
- temp13.dat
- temp14.dat

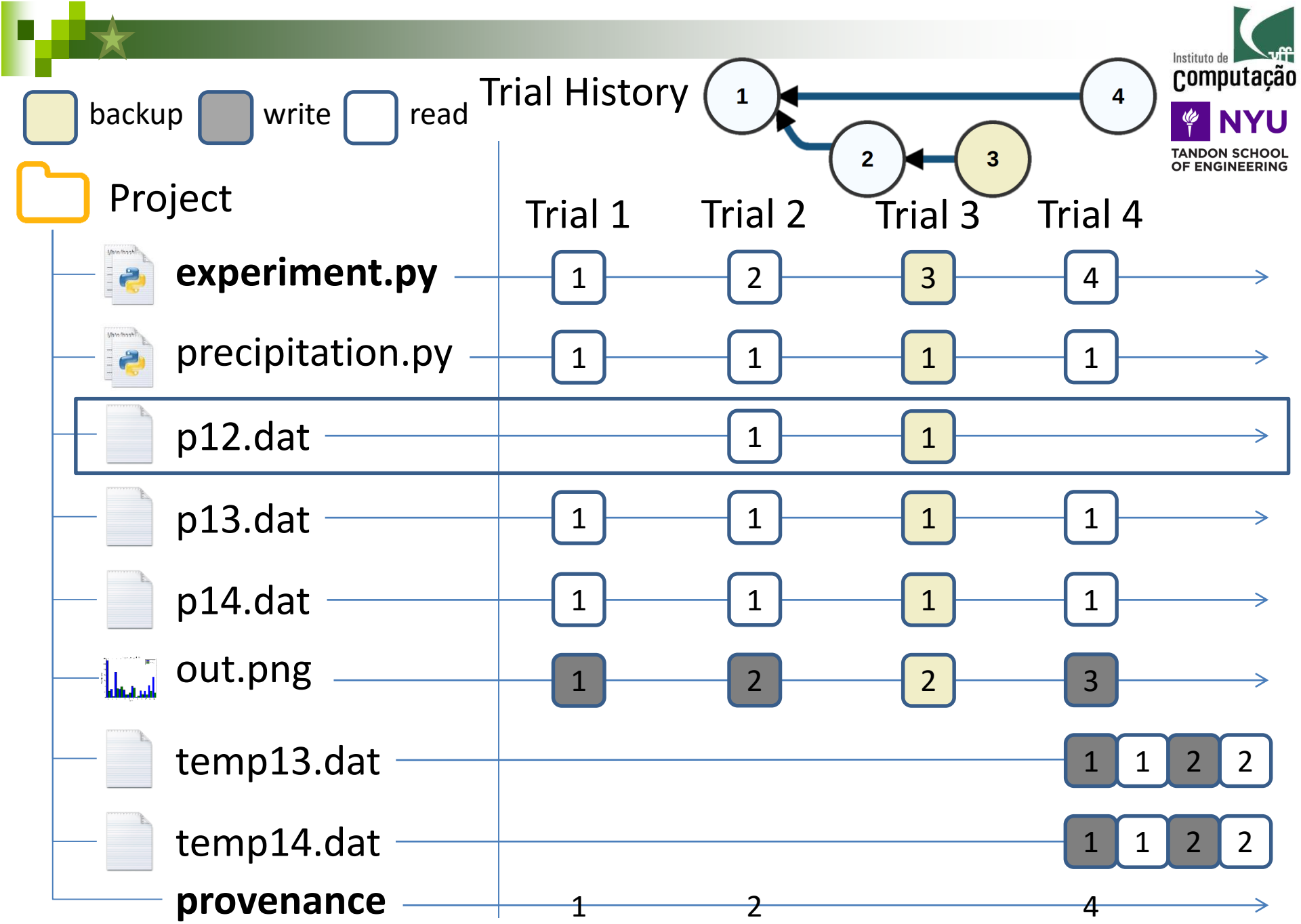


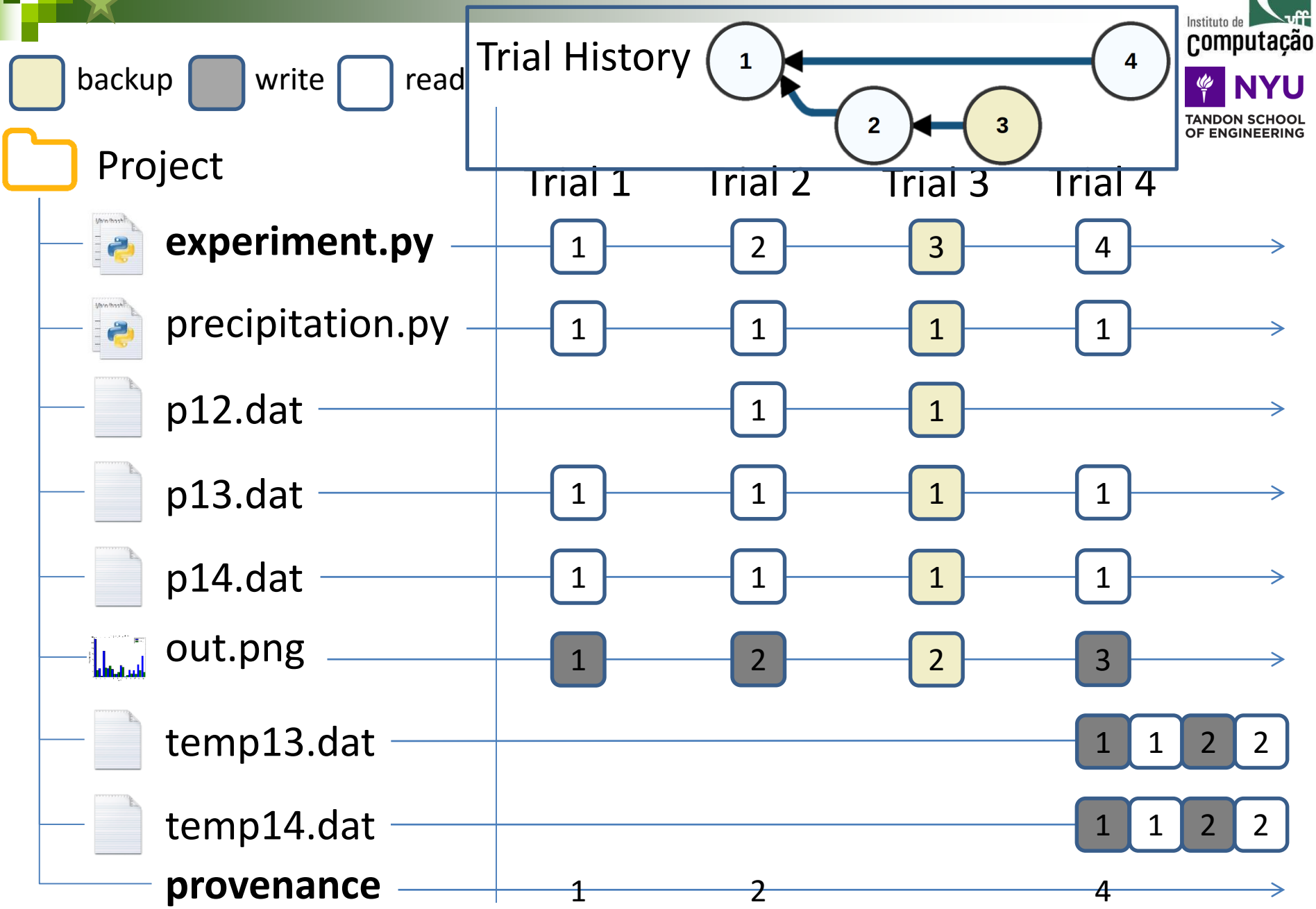

p13.dat
9 read
9 d13
12 remove_outliers
12 write
14 read
14 d13
12 remove_outliers
12 write
temp13.dat
14 read
14 d13
p14.dat
9 read
9 d14
13 remove_outliers
13 write
7 module.arange
7 months
temp14.dat
14 read
14 d14
13 remove_outliers
13 write
16 sum_by_month
17 sum_by_month
16 prec13
17 prec14
21 create_bargraph
out.png
11 int(sys.argv[1])
11 range(int(sys.argv[1]))
11 i
11 i
Composition
Execution
Analysis

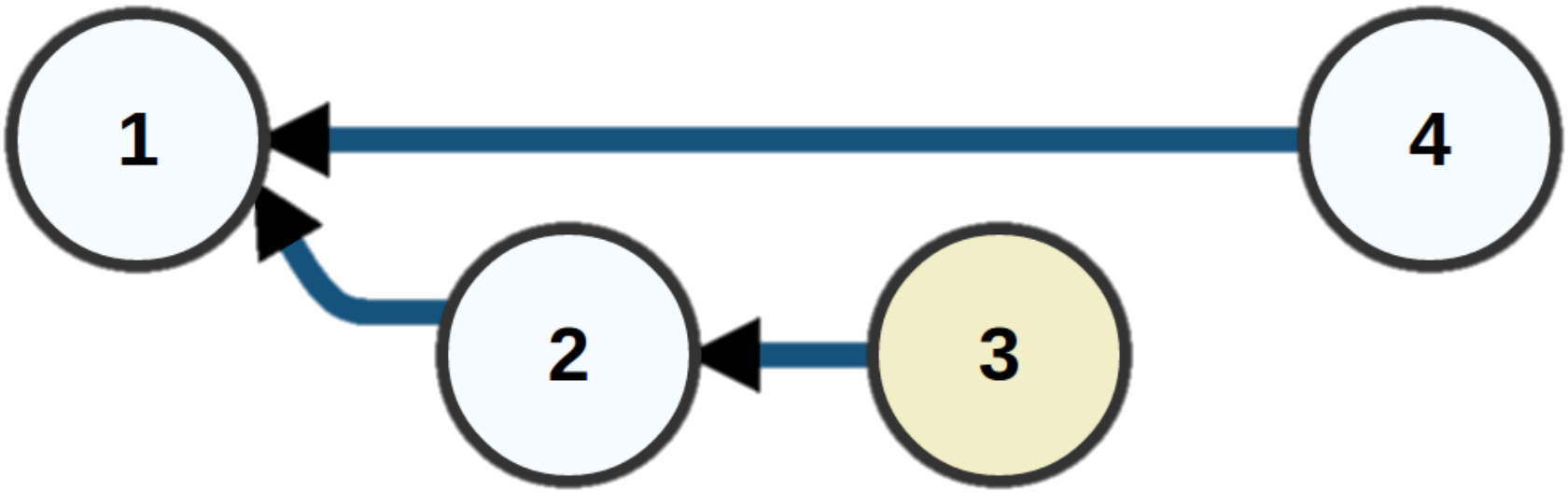

p13.dat
9 read
9 d13
12 remove_outliers
14 read
14 d13
12 remove_outliers
12 write
12 write
temp13.dat
14 read
14 d13
16 sum_by_month
16 prec13
p14.dat
9 read
9 d14
13 remove_outliers
14 read
14 d14
13 remove_outliers
13 write
13 write
temp14.dat
14 read
14 d14
17 sum_by_month
17 prec14
7 module.arange
7 months
11 int(sys.argv[1])
11 range(int(sys.argv[1]))
11 i
11 i
21 create_bargraph
out.png
Composition
Execution
Analysis


backup
write
read
Trial History
1
2
3
4
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
temp13.dat
temp14.dat
provenance

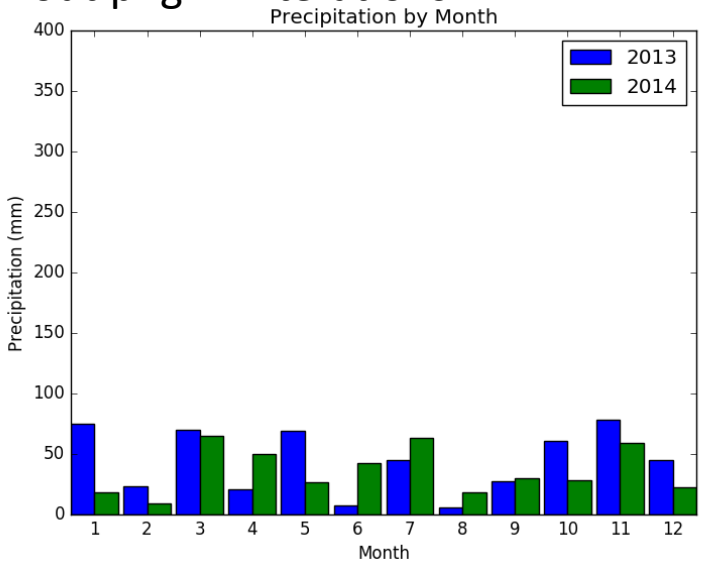

backup
write
read
Trial History
1
2
3
4
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
temp13.dat
temp14.dat
provenance

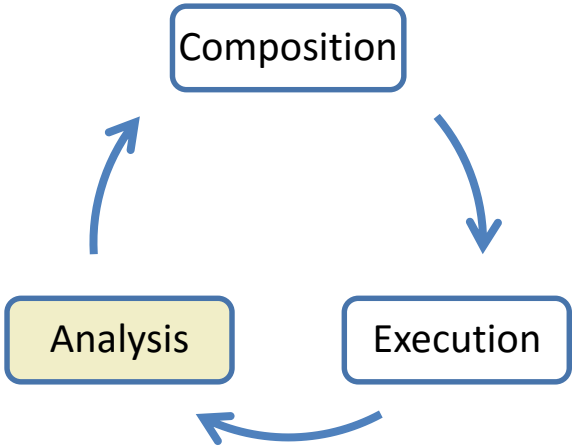

backup
write
read
Trial History
1
2
3
4
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
temp13.dat
temp14.dat
provenance


backup
write
read
Trial History
1
2
3
4
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
temp13.dat
temp14.dat
provenance

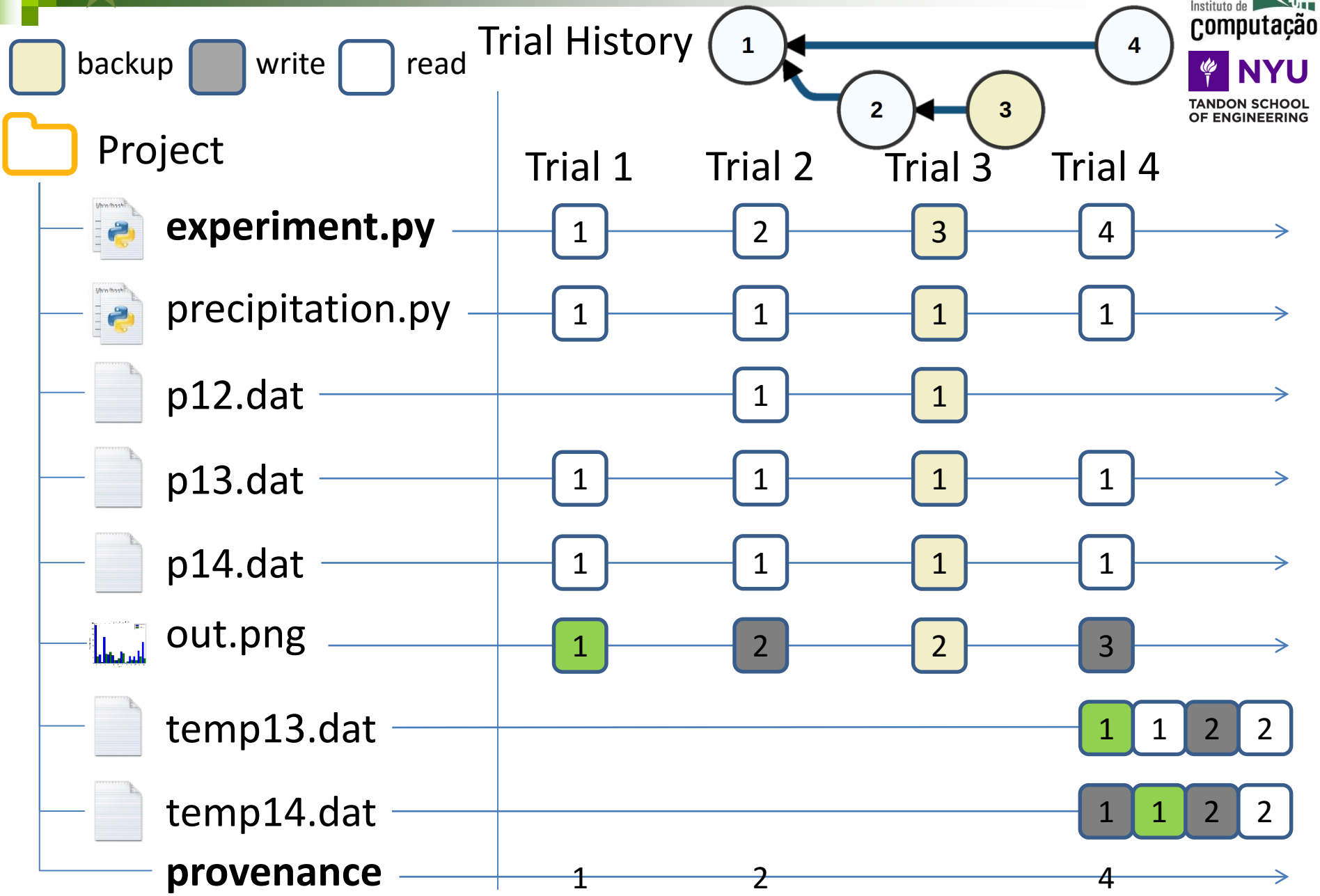

1
4
2
3

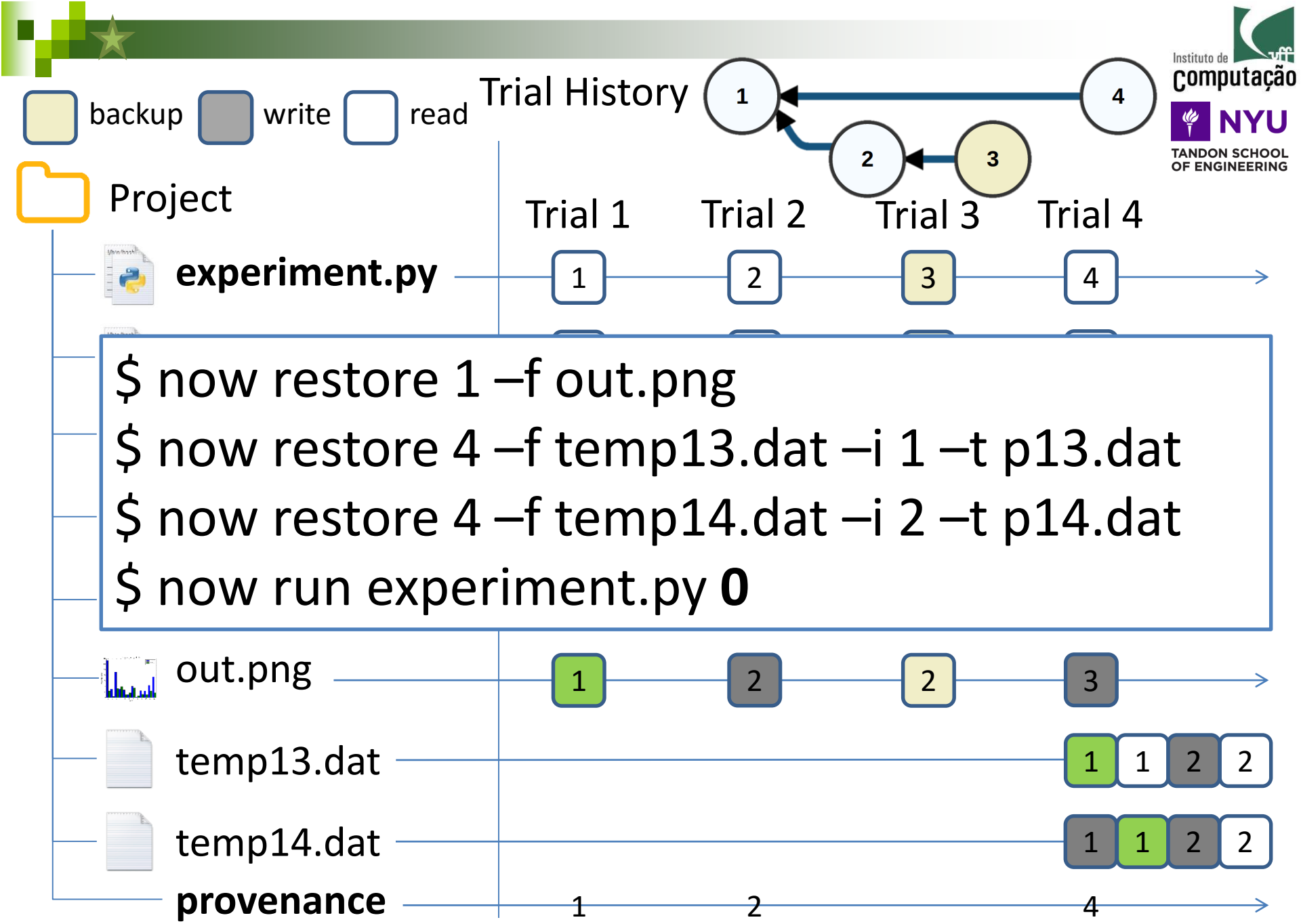
out.png – 2 iterations



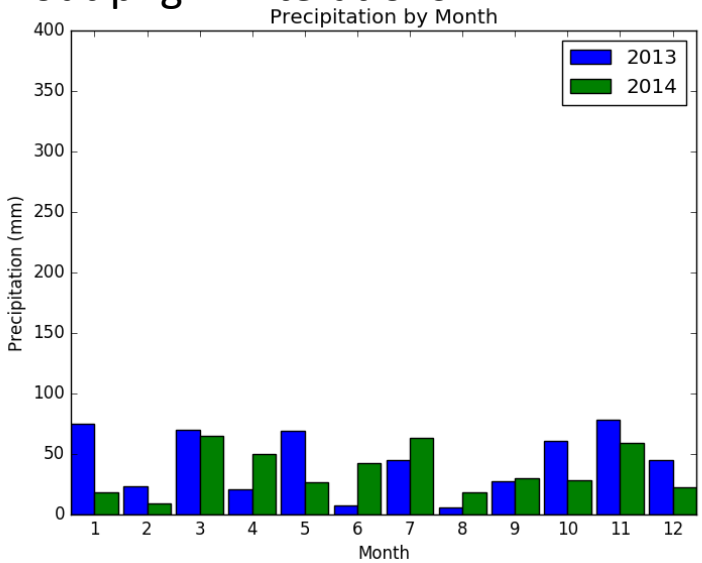


What would be the results if we had only one iteration of outlier removal? Could you compare to the result without removals (Trial 1)?

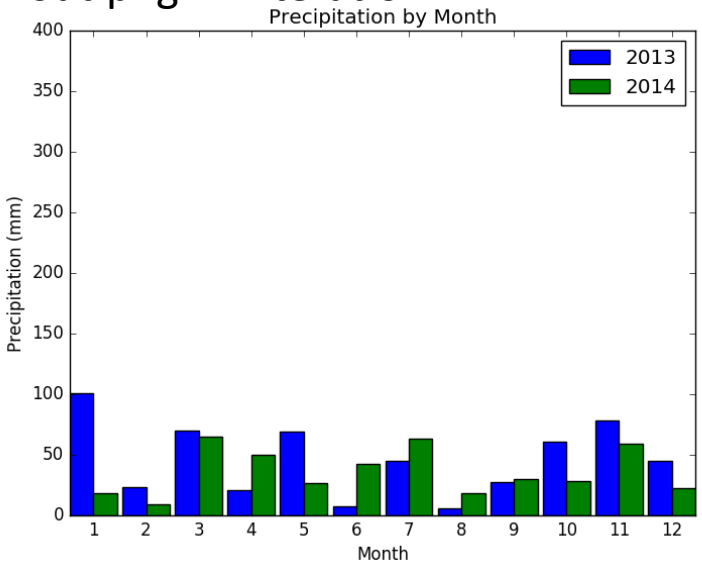

backup
write
read
Trial History
1
2
3
4
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
temp13.dat
temp14.dat
provenance


backup
write
read
Trial History
1
2
3
4
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
1
2
3
4
$ now restore 1 –f out.png
$ now restore 4 –f temp13.dat –i 1 –t p13.dat
$ now restore 4 –f temp14.dat –i 2 –t p14.dat
$ now run experiment.py 0
out.png
1
2
2
3
temp13.dat
1
1
2
2
temp14.dat
1
1
2
2
provenance
1
2
4

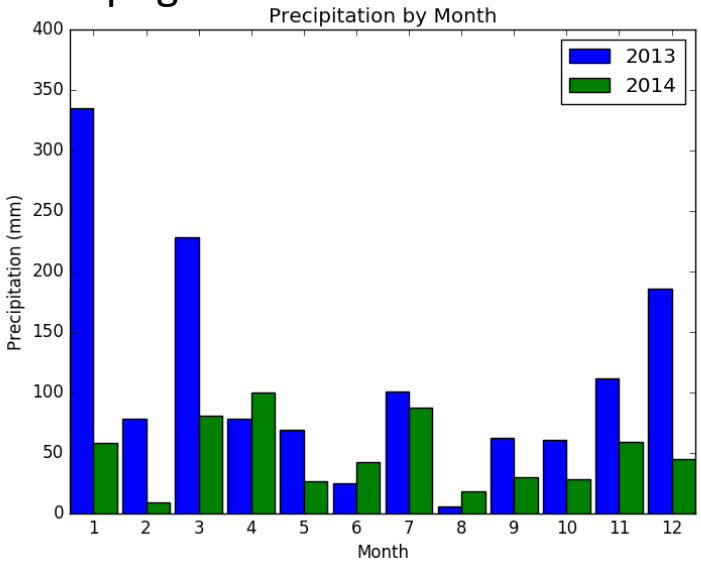
out.png – 2 iterations

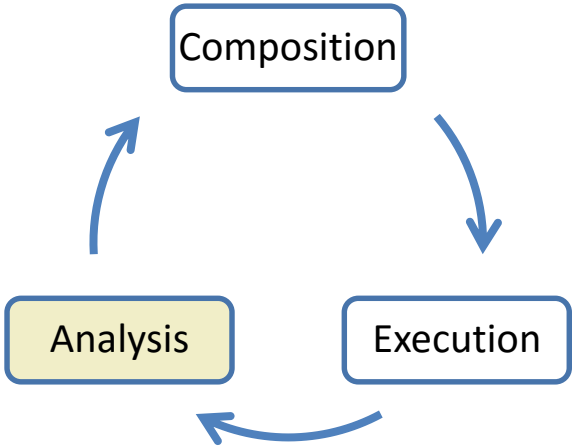


out.png – 1 iteration

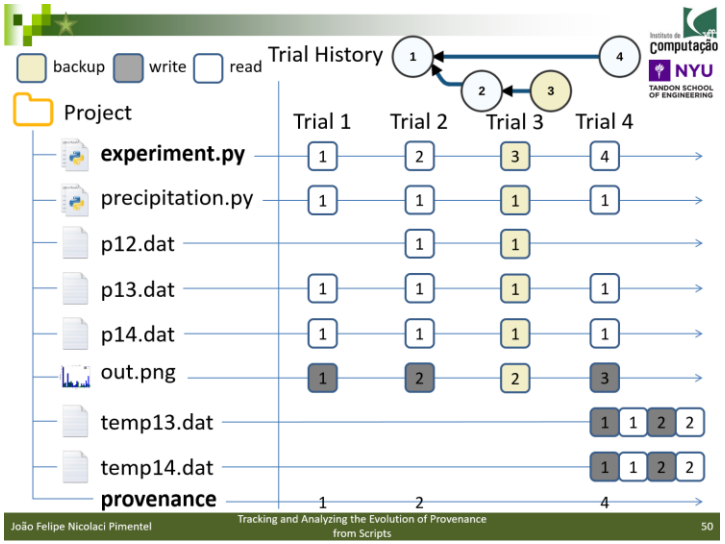


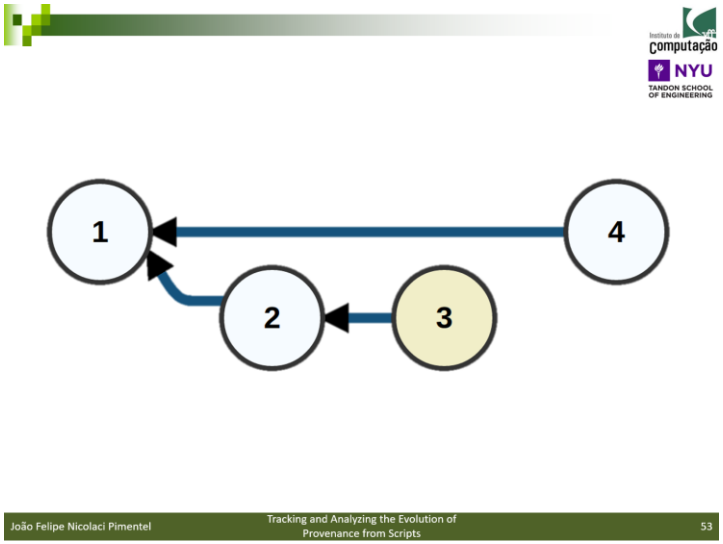
out.png – Trial 1

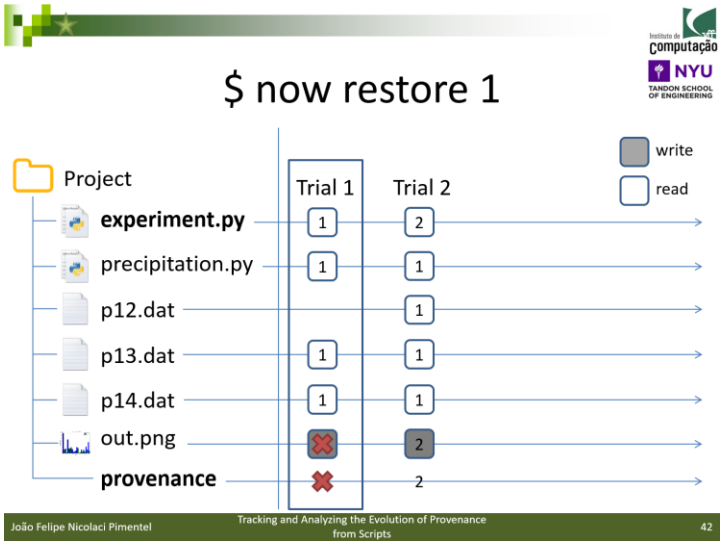


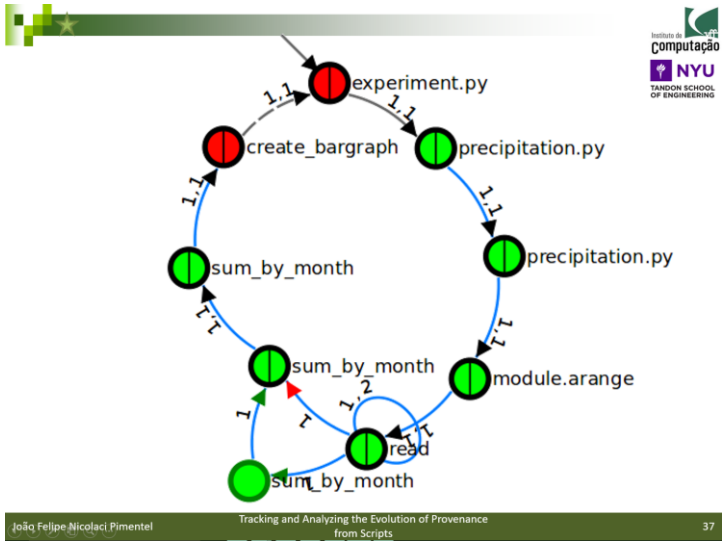


Conclusion


backup
write
read
Trial History
Project
Trial 1
Trial 2
Trial 3
Trial 4
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
temp13.dat
temp14.dat
provenance
João Felipe Nicolaci Pimentel
Tracking and Analyzing the Evolution of Provenance from Scripts
50


1
2
3
4
João Felipe Nicolaci Pimentel
Tracking and Analyzing the Evolution of Provenance from Scripts
53


$ now restore 1
write
read
Project
Trial 1
Trial 2
experiment.py
precipitation.py
p12.dat
p13.dat
p14.dat
out.png
provenance
João Felipe Nicolaci Pimentel
Tracking and Analyzing the Evolution of Provenance from Scripts
42


experiment.py
create_bargraph
precipitation.py
sum_by_month
precipitation.py
sum_by_month
module.arange
read
sum_by_month
João Felipe Nicolaci Pimentel
Tracking and Analyzing the Evolution of Provenance from Scripts
37

Future Work

- Optimization techniques
  - balance storage and re-computation costs
- Alternatives on detecting file object changes
- Semantic versioning for trials
  - Express intention of evolution
- Improve activation graph matching and add other provenance comparison to noWorkflow

Tracking and Analyzing the Evolution of Provenance from Scripts

jpimentel@ic.uff.br
https://github.com/gems-uff/noworkflow

João Felipe Nicolaci Pimentel (UFF),
Juliana Freire (NYU), Vanessa Braganholo (UFF),
Leonardo Murta (UFF)

Instituto de Computação UFF

NYU TANDON SCHOOL OF ENGINEERING

Limitations

- Overhead to collect all accesses
- Implementation stores at fine-grain
  - Wasteful
- Implementation restores only local modules

Related Work

Version Control System

- Most approaches that capture provenance from scripts focus on a single trial
  - Scientists need to keep track of the experiment evolution by themselves
- Version control systems
  - Coarse-grained provenance
    - files
  - May not handle provenance

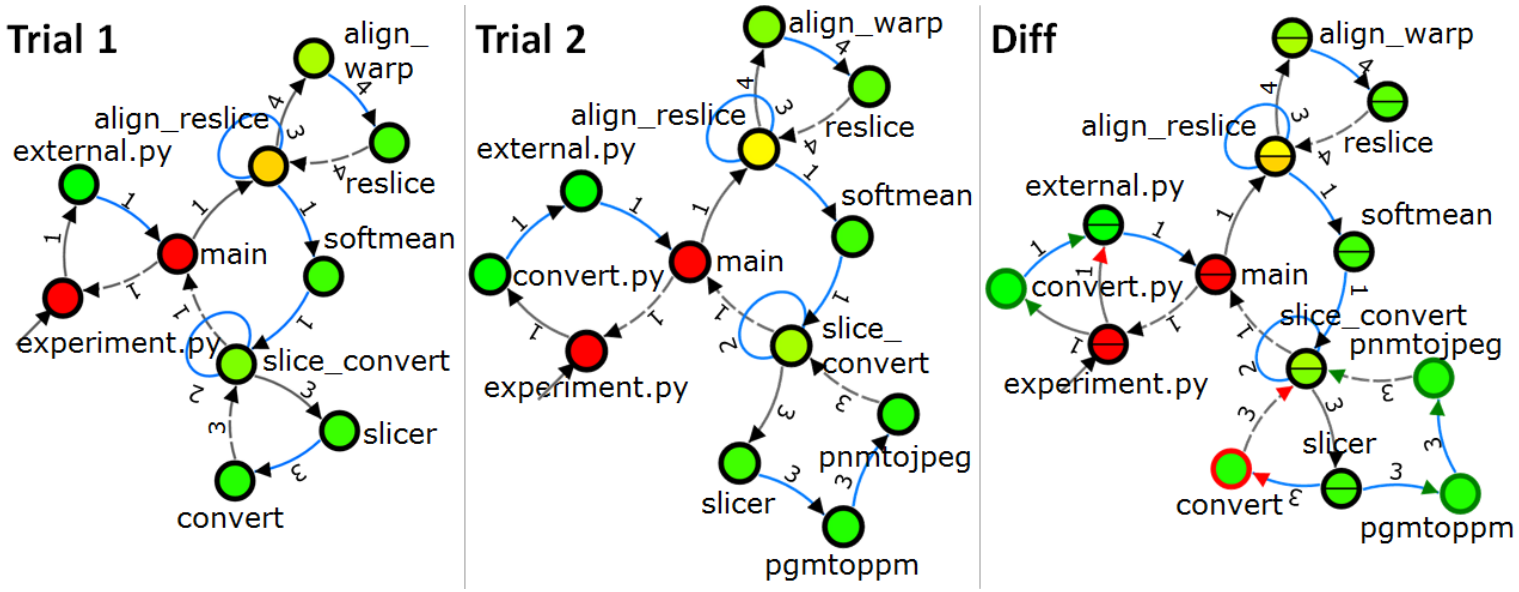
Sumatra

- Use version control systems to capture files before and after execution
  - Handle the evolution
- Coarse-grained collection
  - Do not capture intermediate files

Evaluation

- A set of 9 questions were obtained and adapted from the First Provenance Challenge (FPC) and ProvBench Workshops
- We evaluate our work by answering them

- Q1: if a scientist has executed an experiment twice, but has replaced some procedures in the second trial, what are the trial differences? [FPC]
- Q2: comparing multiple executions according to their parameters, what are the differences on execution behavior? [Swift-PROV]



- Q3: how differences in the input data relate to differences in the values? [CSIRO-PROV]

```
$ now diff 1 2 -f --brief
[now] trial diff:
  Start changed from 2016-02-11 04:49:09.008354
                  to 2016-02-11 04:49:09.898675
  Finish changed from 2016-02-11 04:49:09.536409
                   to 2016-02-11 04:49:10.276422
  Duration text changed from 0:00:00.528055 to 0:00:00.377747
  Code hash changed from cd1be11a2308ab217327a7d361138cb7f6c25106
                     to 2f637ec102961a7677e3f629ab88612d8875f04f
  Parent id changed from <None> to 1

[now] Brief file access diff
[Additions]            | [Removals]              | [Changes]
(rb) atlax-x.ppm       | (w) atlax-x.gif (new)   |
(w) atlax-x.jpg (new)  | (w) atlax-x.pgm (new)   |
(w) atlax-x.pgm        | (w) reslice1.img (new)  |
(w) atlax-x.ppm (new)  | (wb) warp.warp (new)    |
(w) reslice1.hdr       | ...                     |
(wb) warp.warp         |                         |
...                    |                         |
```

- Q4: using historical provenance, which parts of the execution fail frequently? [CSIRO-PROV]

```
SELECT name, count(name) AS c
FROM function_activation
WHERE return_value = "-1"
GROUP BY name
ORDER BY c DESC;
```

- Q5: which trials are related to a given trial?
- Q6. a given trial was derived from which trial? [VisTrails-PROV]
- Q7. what are the available trials, and what are their durations?
- Q8. how many trials are associated to a given source code?
- Q9. how many trials present failures? [Wf4Ever-PROV]