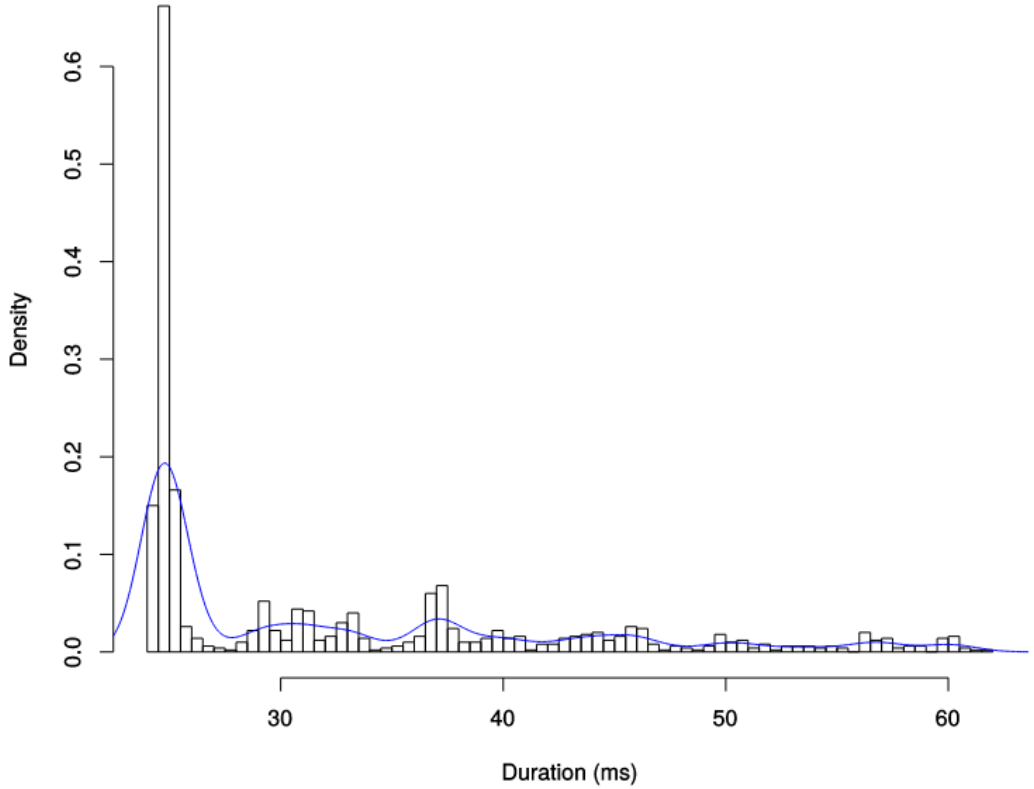
Comparison of Traces to Diagnose Performance Variations

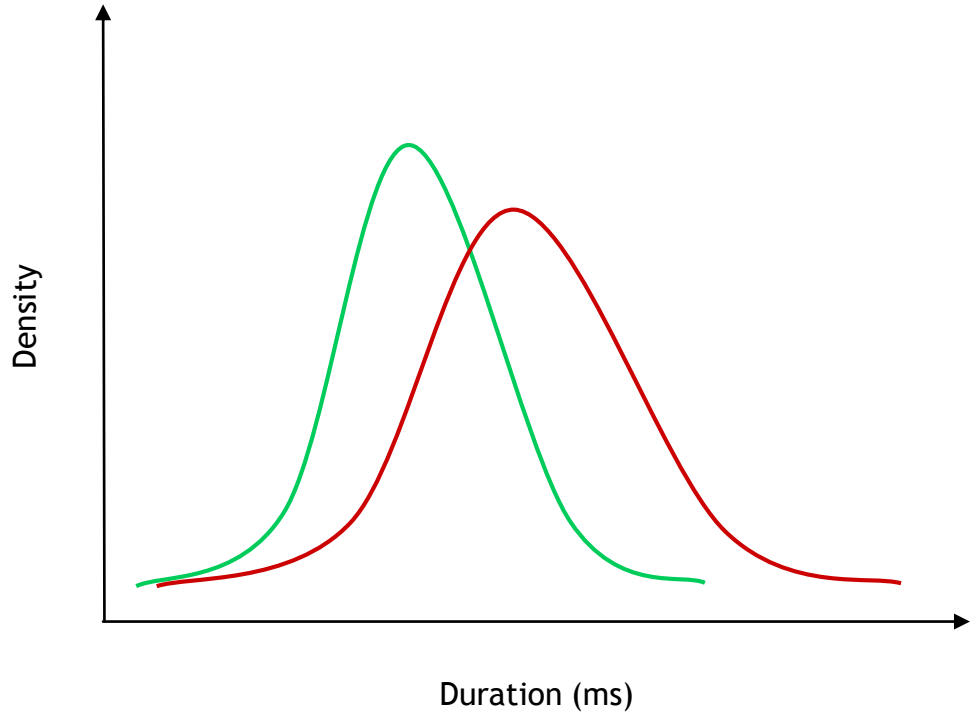
Presented by François Doray

Introduction | Motivation

- **Distribution of the duration of remote procedure calls**



- **Distribution of request time for 2 different versions of a Web service**

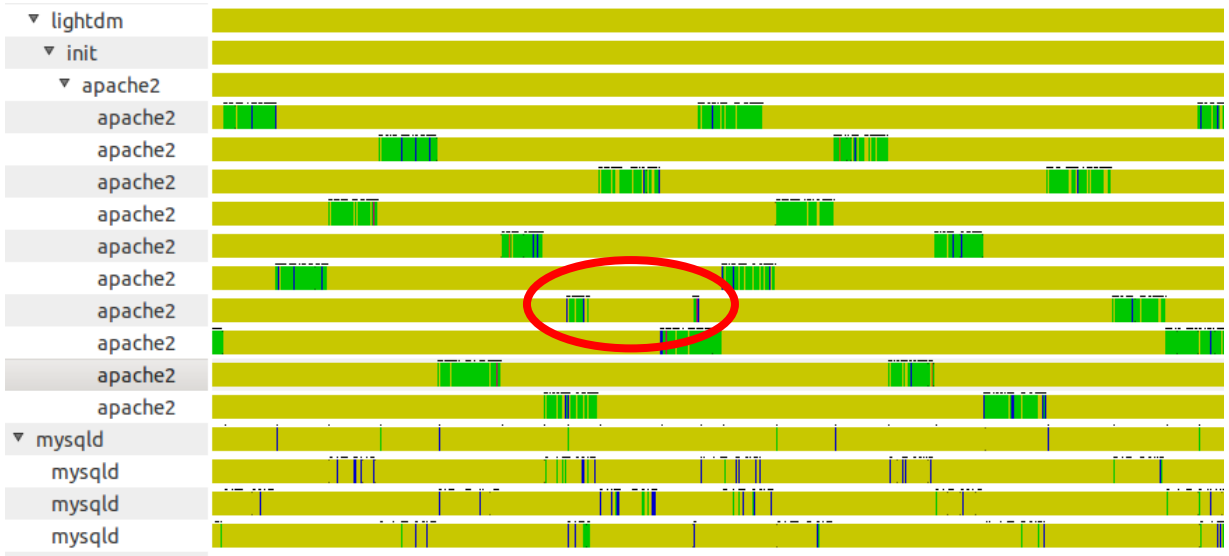


Introduction | Motivation

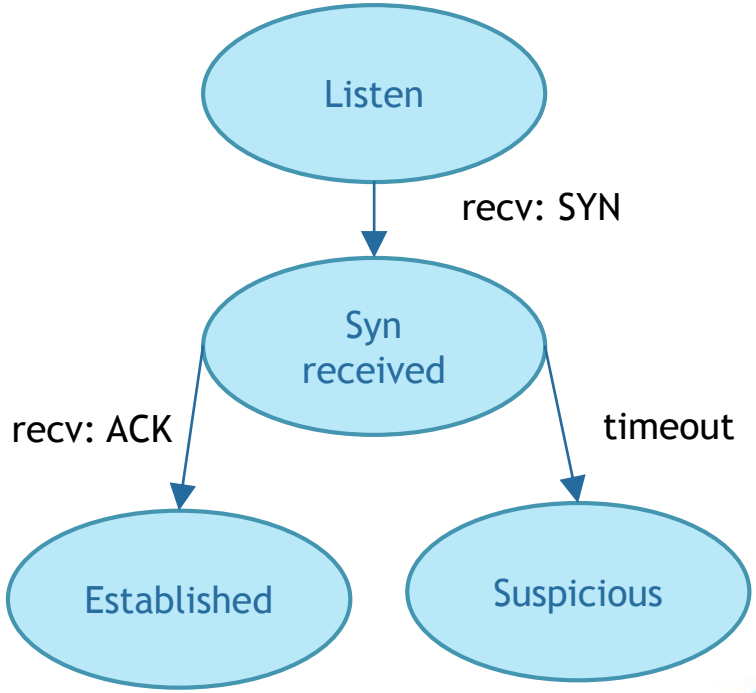
- Possible causes of performance variation:
  - Different requests.
  - Different binary (application or external library).
  - Different code path (e.g. garbage collection).
  - Different system load (CPU, memory, disk, network).
  - Different system configuration.
  - Different resource sharing.
  - …

Introduction | Motivation

- Trace Viewer:
  - Huge amount of information not related to the observed problem.



- Trace Filters:
  - Rules written for a specific problem.



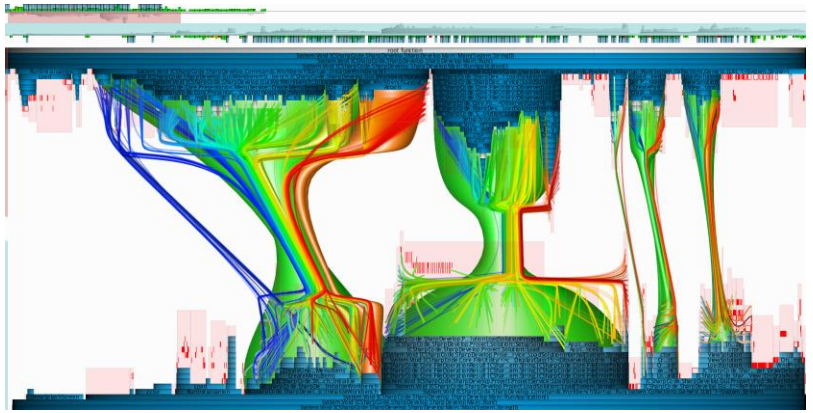
Introduction | Objective

- Automatically identify the **root cause** of a performance variation between multiple executions of the same task by **comparing** traces.

Literature

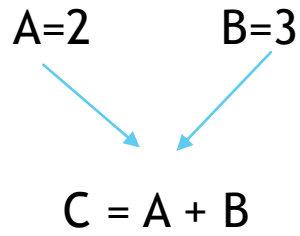
- TraceDiff: **Visualization** of a comparison between function call traces.

  (Trümper and al., 2013)



- Matching instructions between different versions of the same code using the **dynamic data dependence graph.**

  (X. Zhang and R. Gupta, 2005)

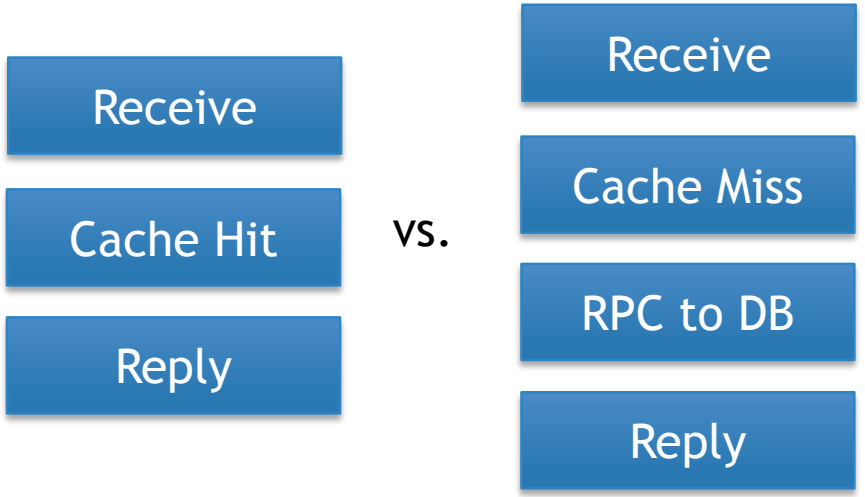


TraceDiff screenshot from the article:
Trümper, Jonas, Jurgen Döllner, and Alexandru Telea. "Multiscale visual comparison of execution traces." *2013 IEEE 21st International Conference on Program Comprehension (ICPC)*. IEEE, 2013.

Literature

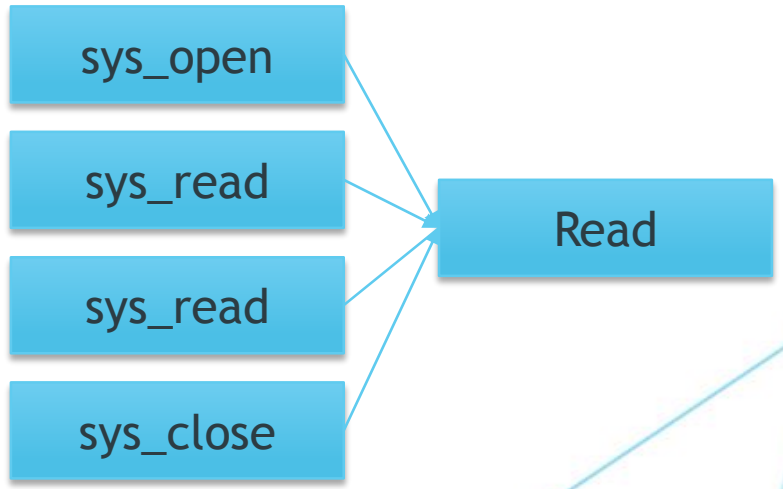
- Spectroscope / Magpie / Pinpoint: Extract and **cluster distributed request flow** graphs.

  (R. R. Sambasivan and al., 2011)



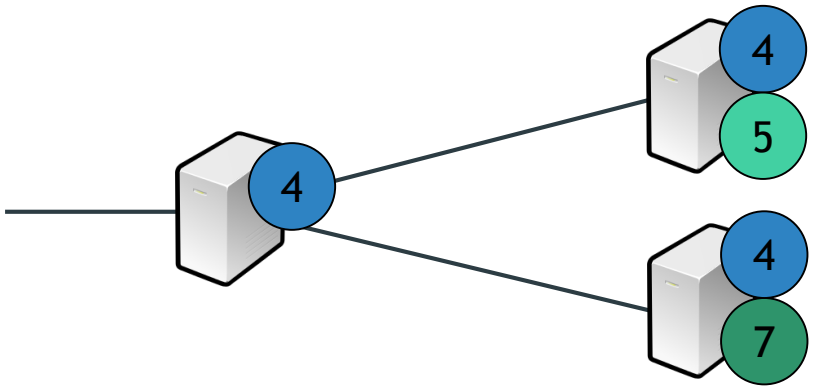
- Compare executions on different OS using **high-level concepts** to detect attacks.

  (A. Hamou-Lhadj and al, 2013)

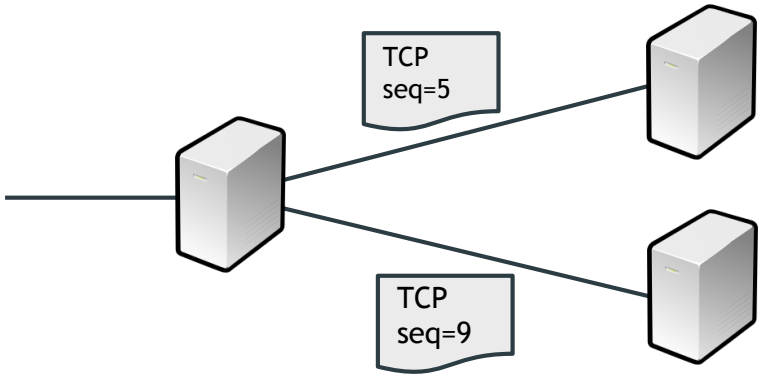


Associate Events to Tasks | Literature

- **Dapper**: Query identifier recorded with each event.
  (B.H. Sigelman and al., 2010)



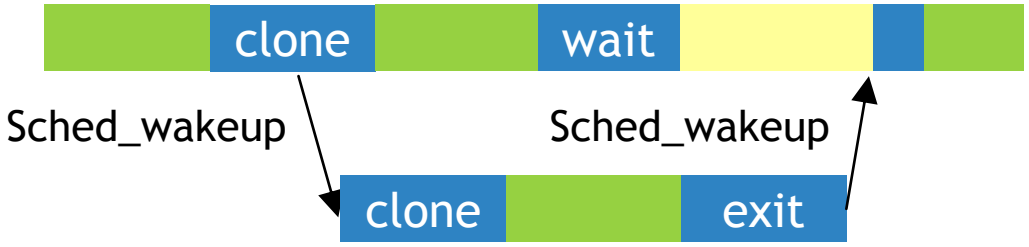
- **Magpie** and **TraceCompass critical path**: Control flow retrieved using a model based on system events.
  (R. Isaacs and al., 2004) (F. Giraldeau, 2014)



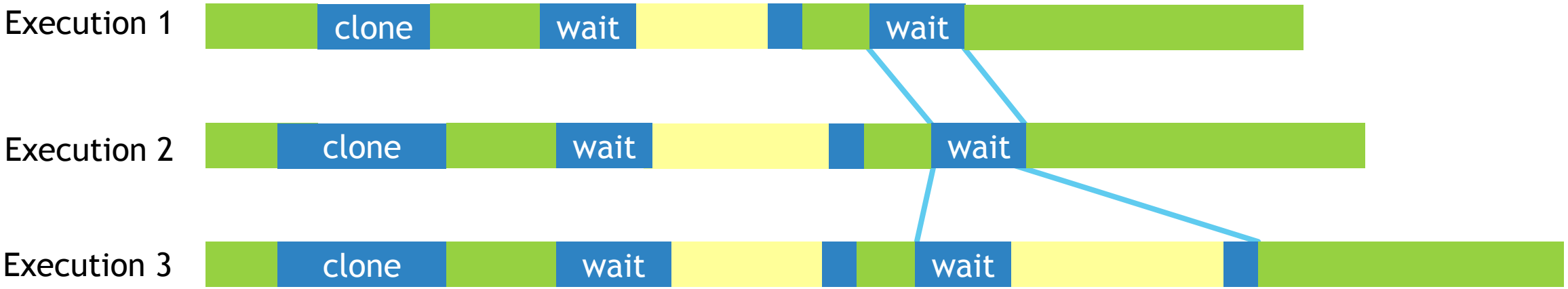
Server Image: http://commons.wikimedia.org/wiki/File:Gorilla-server.svg

Associate Events to Tasks | Solution

- Abstract control flow builder that can find dependencies between thread segments using:
  - Low-level kernel events.
  - User-space events.



Events Matching



- The second “wait” system call in the third execution is abnormal.

Events Matching | Literature

- Dynamic Programming

  $O(n^2)$

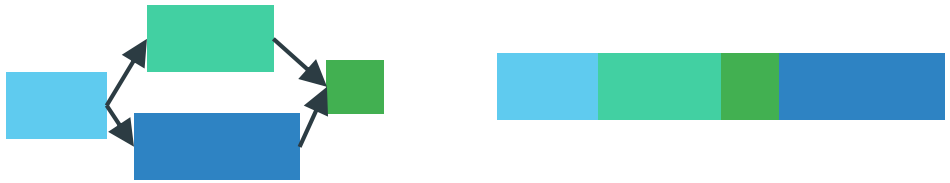
	A	B	C
A	0	1	2
D	1	1	2
B	2	1	2

- Dynamic Programming, by level of call stack

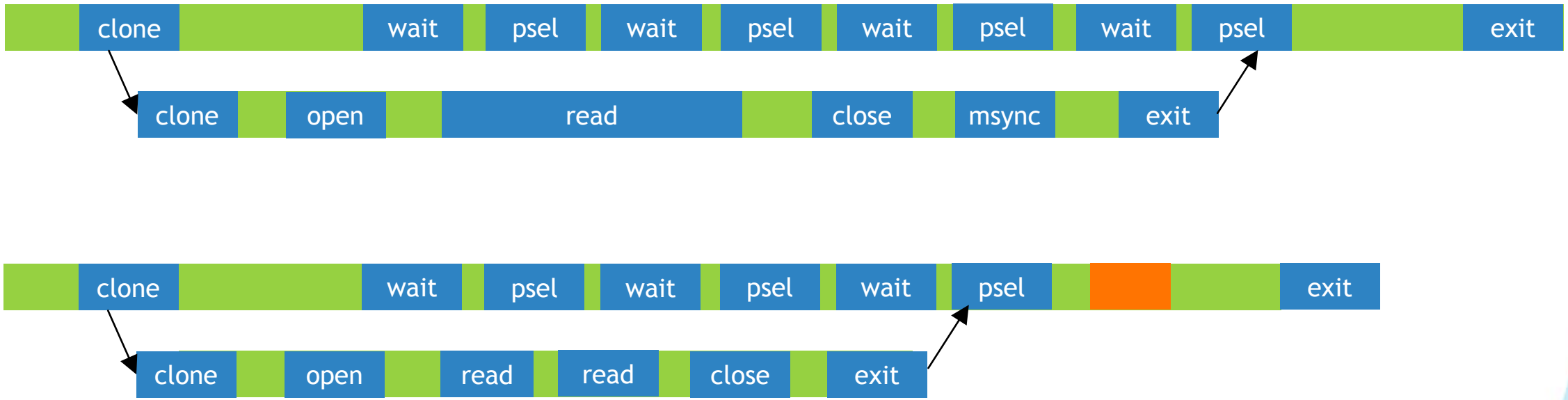
  (M. Weber, 2012)

- Depth-First Search and Dynamic Programming

  (R. R. Sambasivan, 2011)

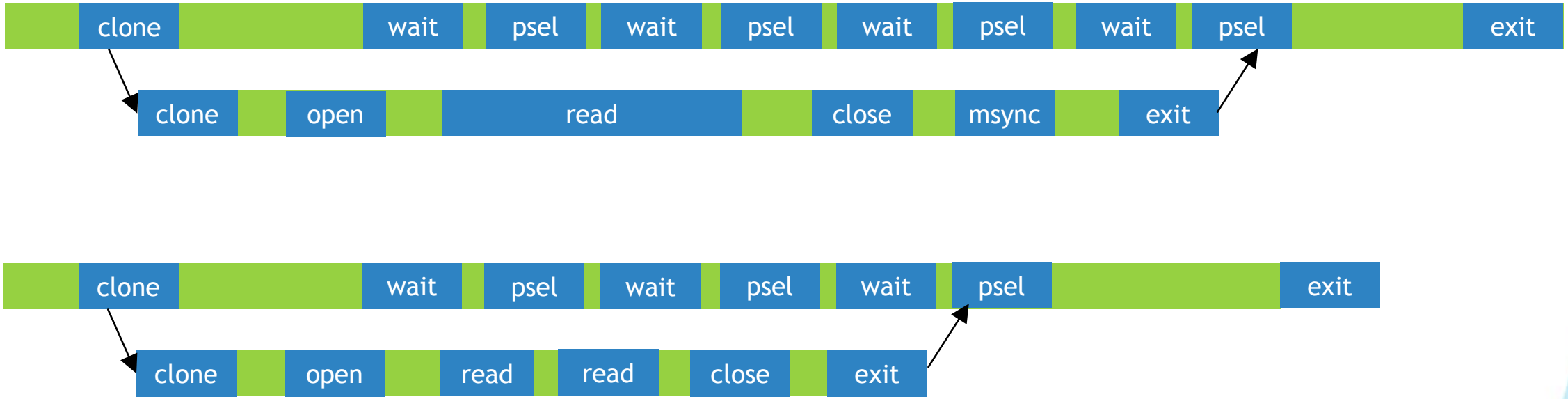


Events Matching| Solution



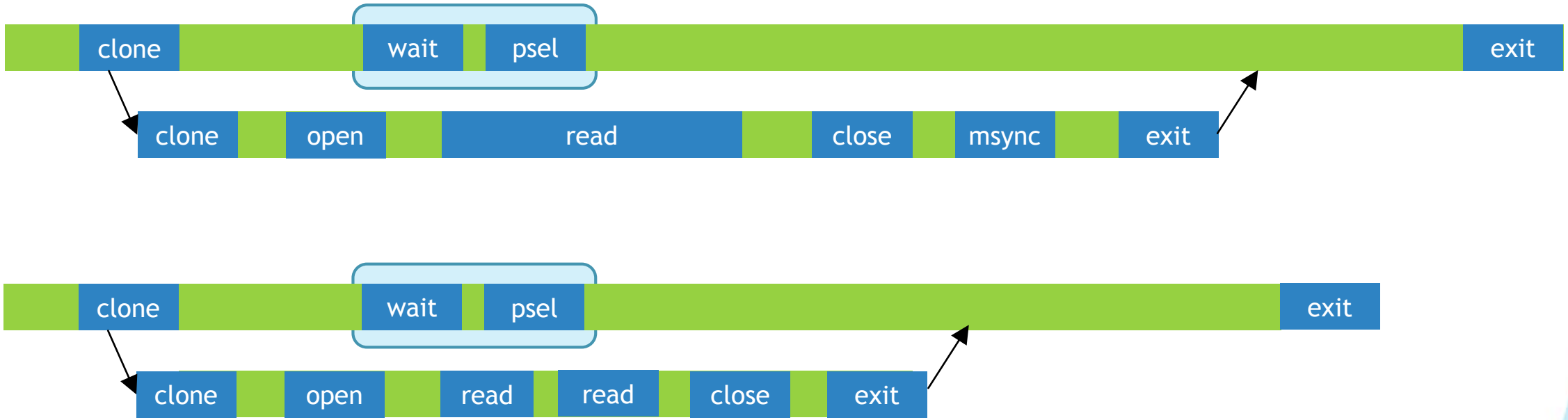
Events Matching | Solution

- Remove irrelevant states: pre-empted, interrupted.



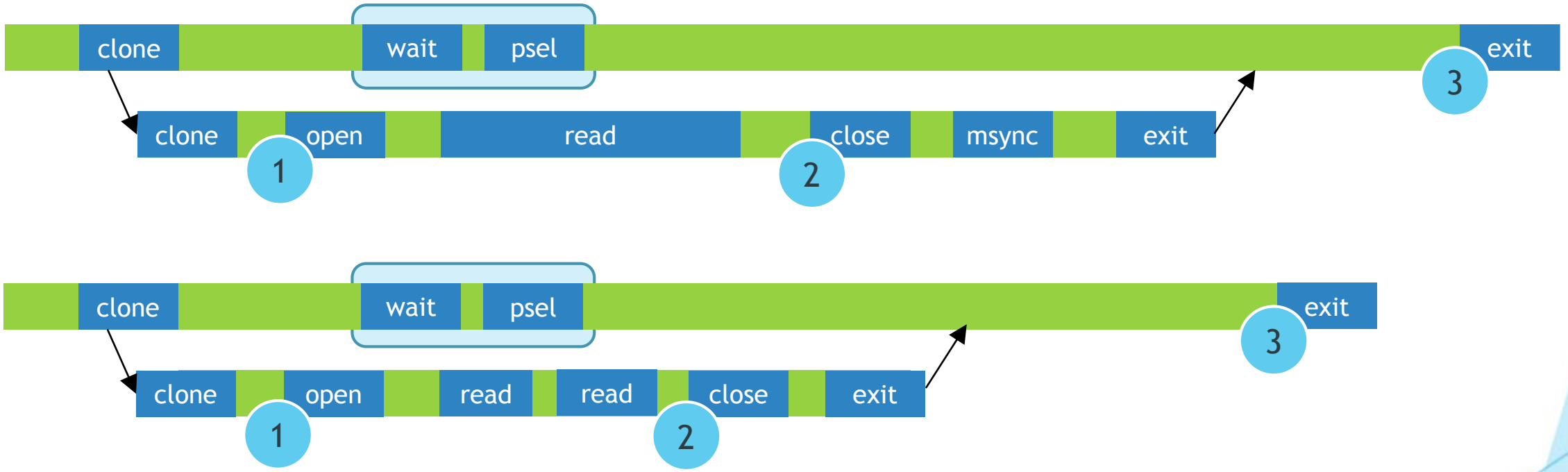
Events Matching | Solution

- Collapse loops using a sliding window.



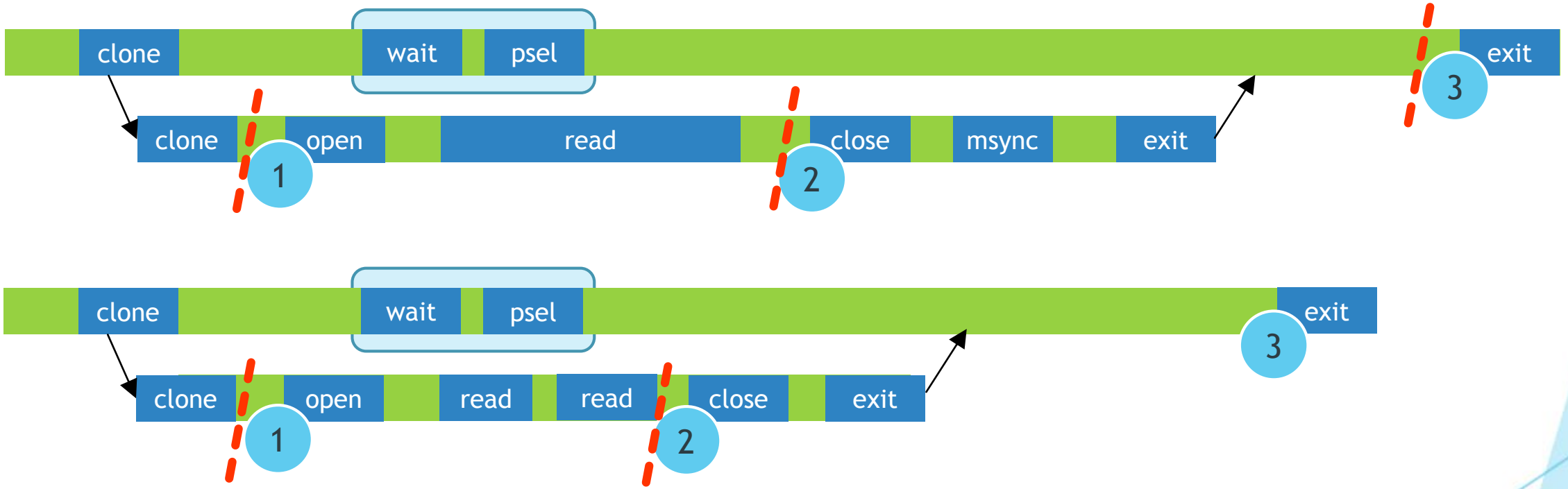
Events Matching | Solution

- Automatically match unambiguous events.



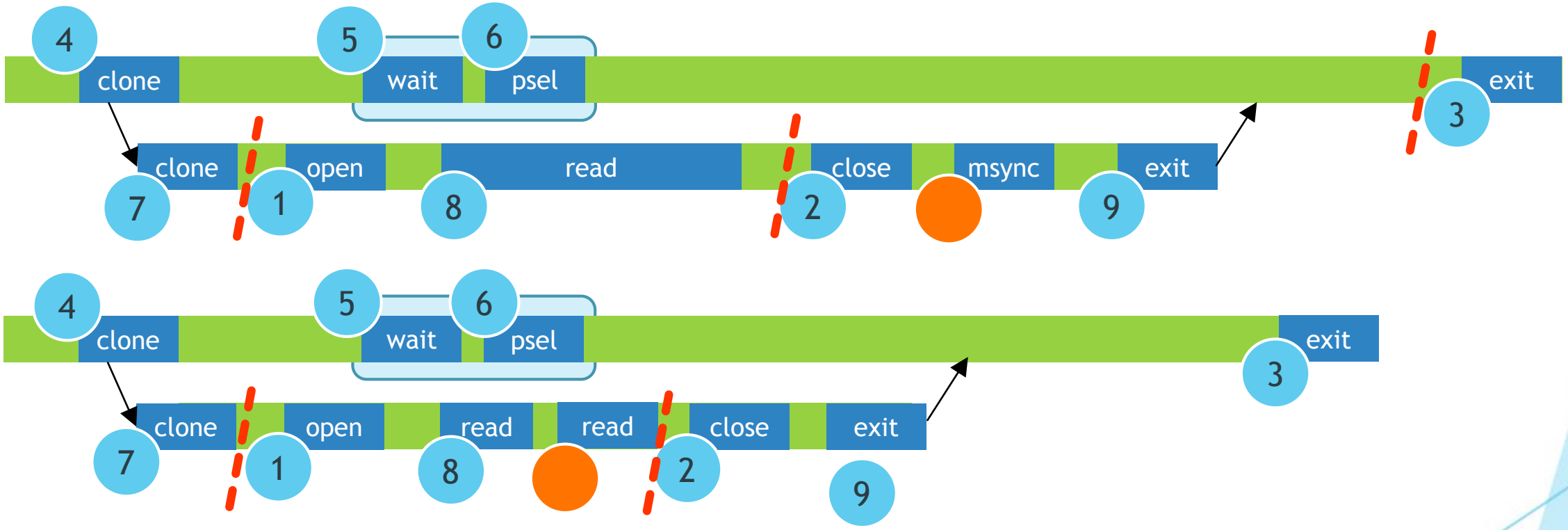
Events Matching | Solution

- Slice the graph at the position of matched events.



Events Matching | Solution

- Apply the dynamic programming algorithm on depth-first search traversals of the sub-graphs.



Compare Metrics| Literature

- χ^2 hypothesis test to check whether invariants are respected.

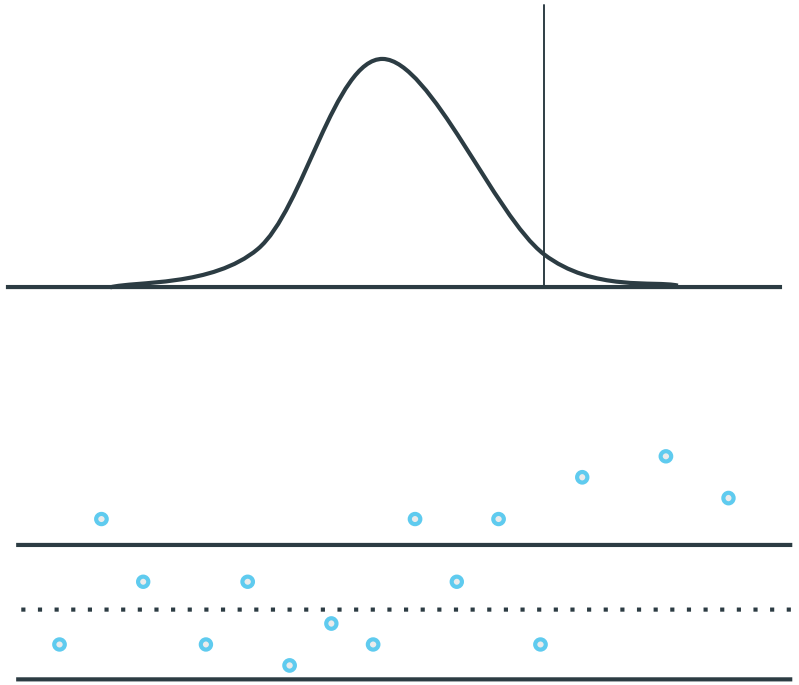
  (M.A. Munawar, 2008)

- **Control chart** (Shewhart) to detect machines whose performance deteriorates.

  (T.H. Nguyen, 2012)

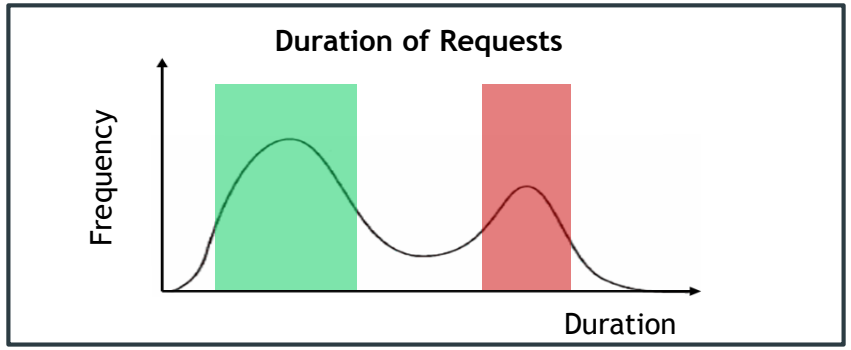
- **Mogul at Netflix** Automatic correlation of metrics.

  (B. Gregg, 2014)

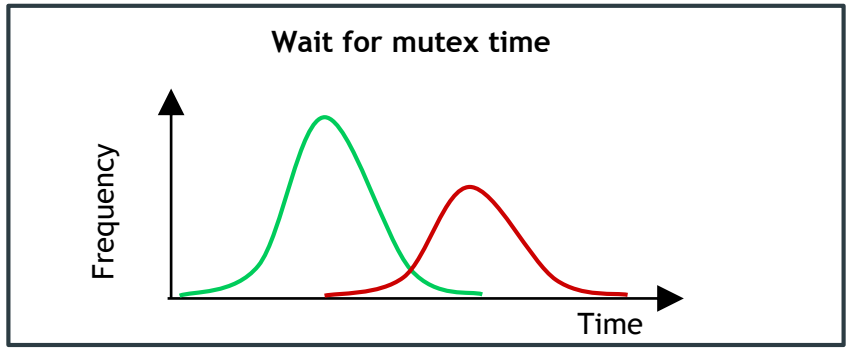


Compare Metrics | Solution

- Distribution View



- Distribution Comparison View



Demo

- *Demo*

Road Ahead | Automatic Segmentation

- Automatically find **recurrent patterns.**

- Compare the **frequency of occurrence** of recurrent patterns in different traces (different machines, different days).

Conclusion

- Objective:
  - Automatically identify the **root cause** of a performance variation between multiple executions of the same task by **comparing** traces.