A SHORT INTRODUCTION TO

SnailTrail: Generalizing Critical Paths for Online Analysis of Distributed Dataflows*

strymon.systems.ethz.ch

Moritz Hoffmann, Andrea Lattuada, John Liagouris, Vasiliki Kalavri,
Desislava Dimitrova, Sebastian Wicki, Zaheer Chothia, Timothy Roscoe

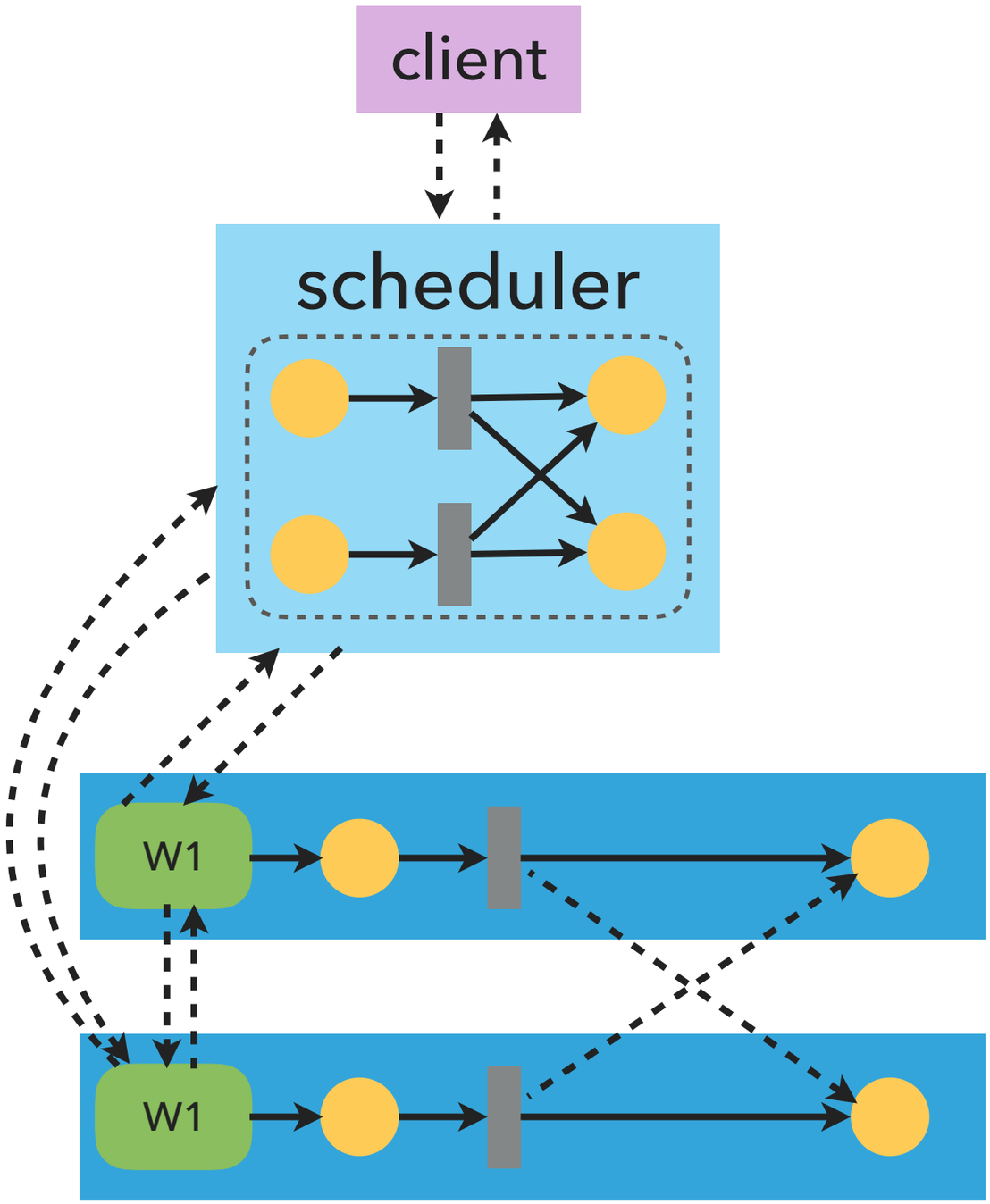
Systems Group, Department of Computer Science, ETH Zürich

firstname.lastname@inf.ethz.ch

Malte Sandstede (samalt@ethz.ch)
Slides (partly) stolen from Vasia Kalavri

Systems@ETH zürich

ANALYZING DISTRIBUTED DATAFLOWS



Hard to troubleshoot…

- long-running, dynamic workloads
- many tasks, activities, operators, dependencies
- bottleneck causes are usually not isolated but span multiple processes

ANALYZING DISTRIBUTED DATAFLOWS

Hard to troubleshoot…

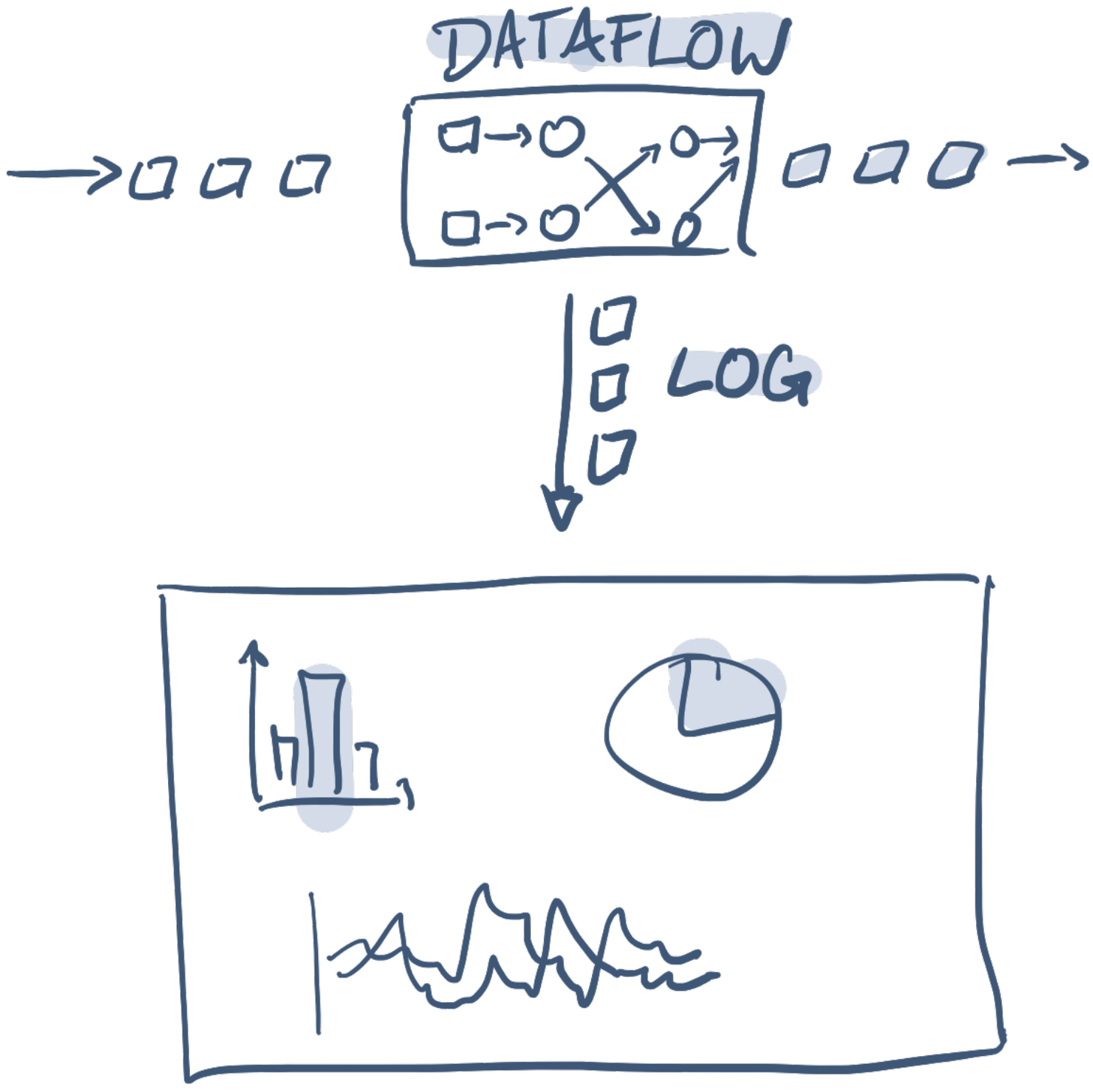
- long-running, dynamic workloads
- many tasks, activities, operators, dependencies
- bottleneck causes are usually not isolated but span multiple processes

…but very useful (duh!)

- Provenance & Explainability
- Performance Tuning
- Debugging
- SLAs

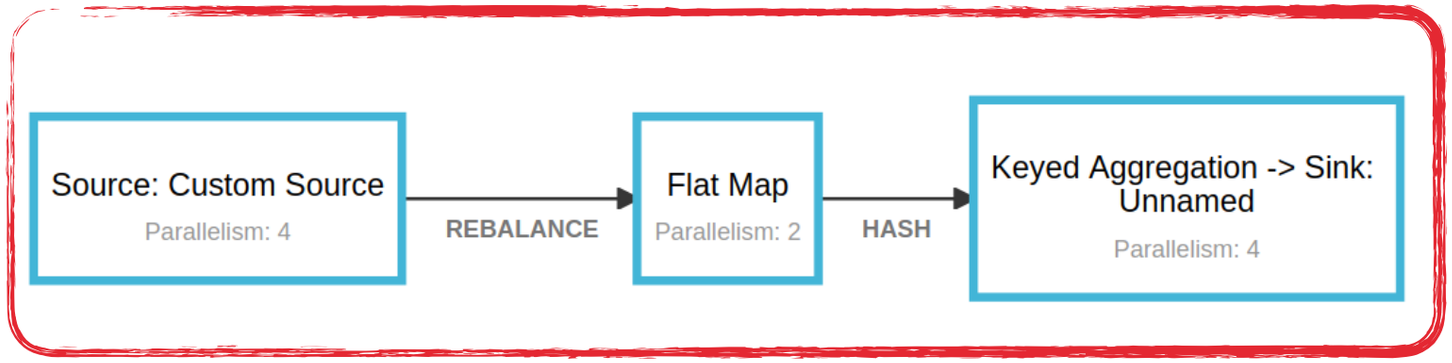
CONVENTIONAL PERFORMANCE METRICS

CONVENTIONAL PERFORMANCE METRICS



CONVENTIONAL PERFORMANCE METRICS

Dataflow graph



Custom aggregate metrics

Aggregate data exchange

Subtasks | TaskManagers | Metrics | Accumulators | Checkpoints | Back Pressure

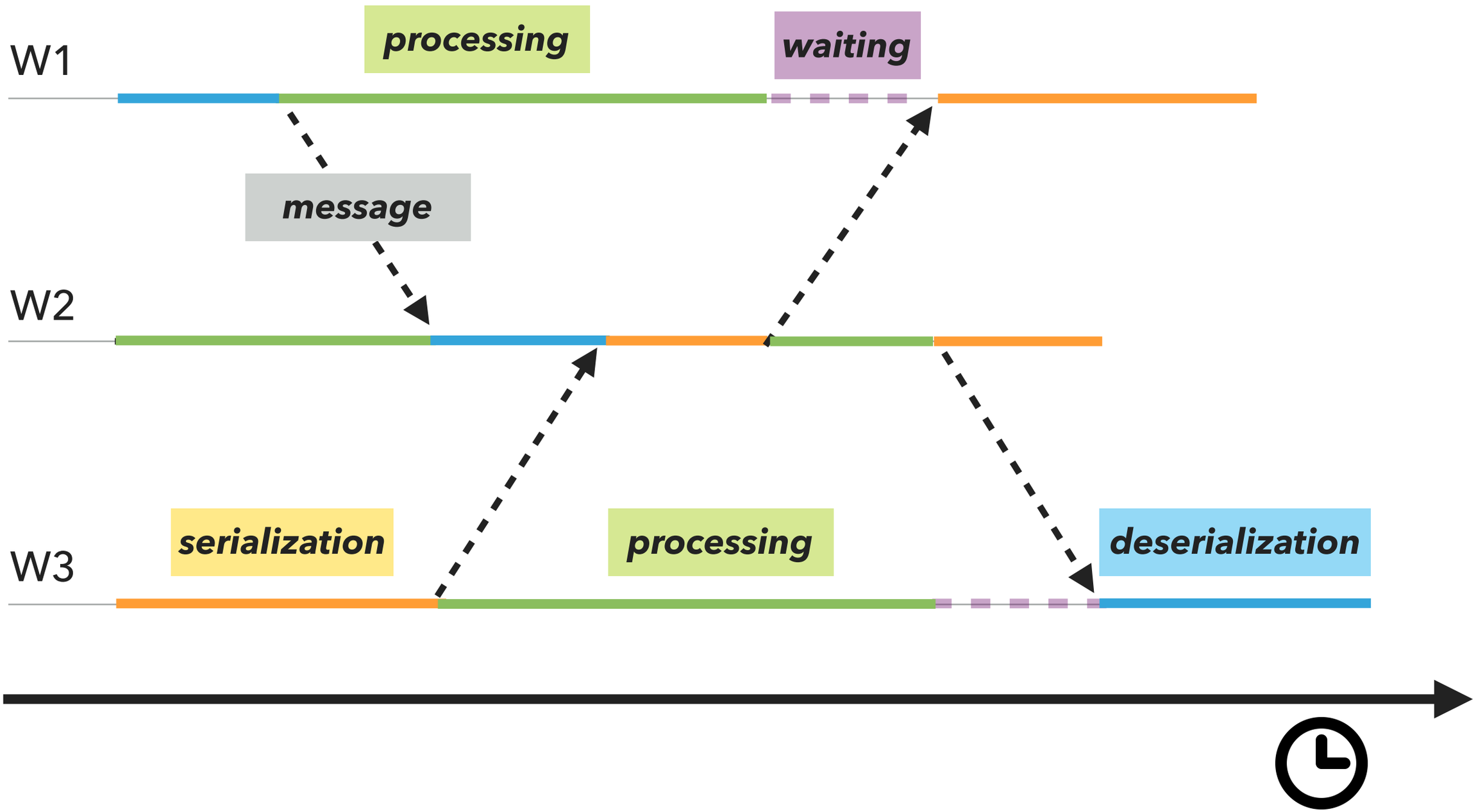
Start Time	End Time	Duration	Name	Bytes received	Records received	Bytes sent	Records sent	Parallelism	Tasks	Status
2017-10-19, 17:45:43	2017-10-19, 17:47:42	1m 58s	Source: Custom Source	0 B	0	2.56 GB	12,942,429	4	0 0 4 0 0 0 0	RUNNING
2017-10-19, 17:45:43	2017-10-19, 17:47:42	1m 58s	Flat Map	2.54 GB	12,844,494	4.59 GB	254,656,192	2	0 0 2 0 0 0 0	RUNNING

Start Time	End Time	Duration	Bytes received	Records received	Bytes sent	Records sent	Attempt	Host	Status
2017-10-19, 17:45:43		1m 58s	1.26 GB	6,401,851	2.29 GB	126,924,569	1	kalamari:43402	RUNNING
2017-10-19, 17:45:43		1m 58s	1.27 GB	6,442,643	2.30 GB	127,731,623	1	kalamari:43402	RUNNING

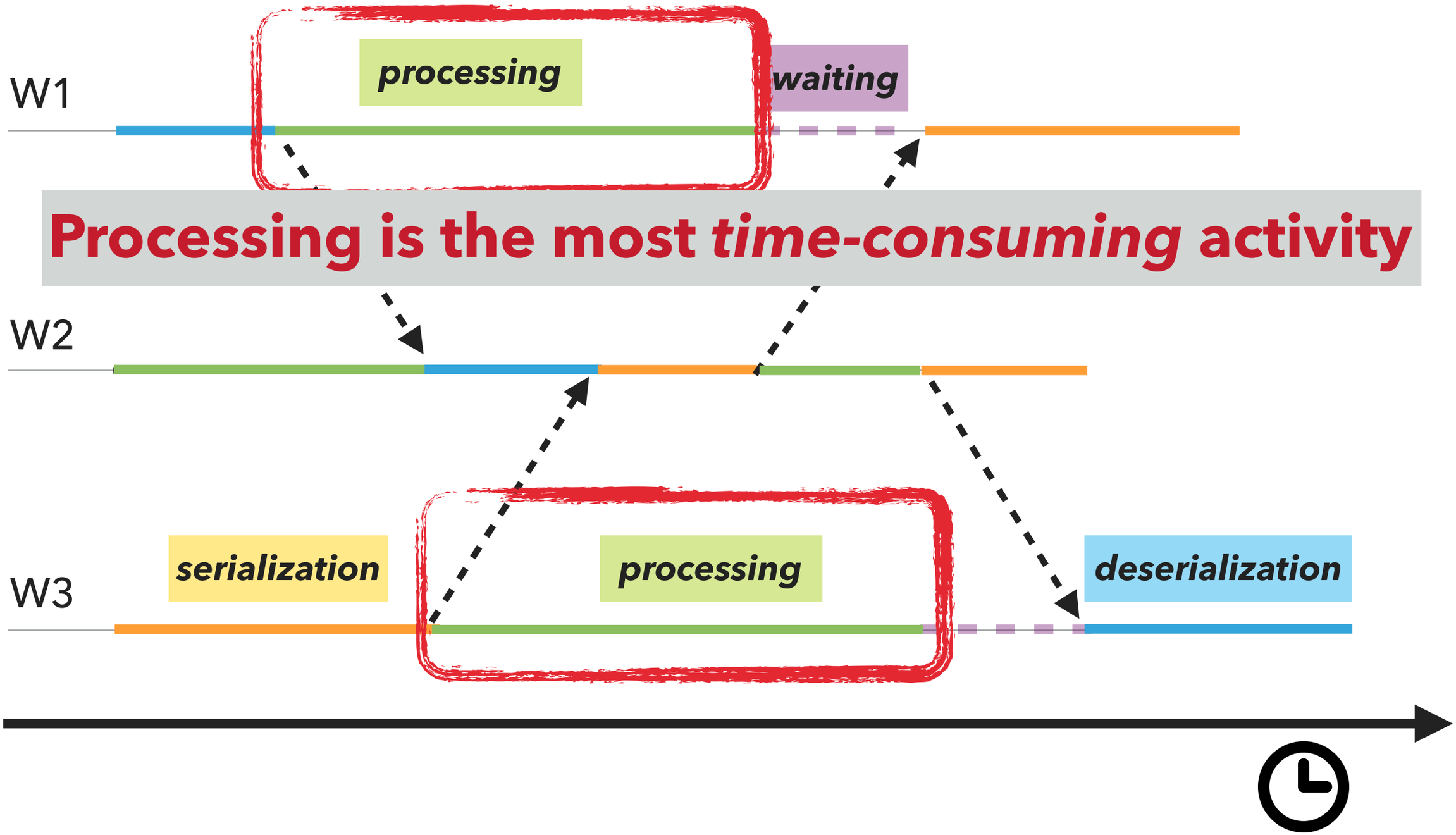
2017-10-19, 17:45:43	2017-10-19, 17:47:42	1m 58s	Keyed Aggregation -> Sink: Unnamed	4.59 GB	254,646,032	0 B	0	4	0 0 4 0 0 0 0 RUNNING

Duration

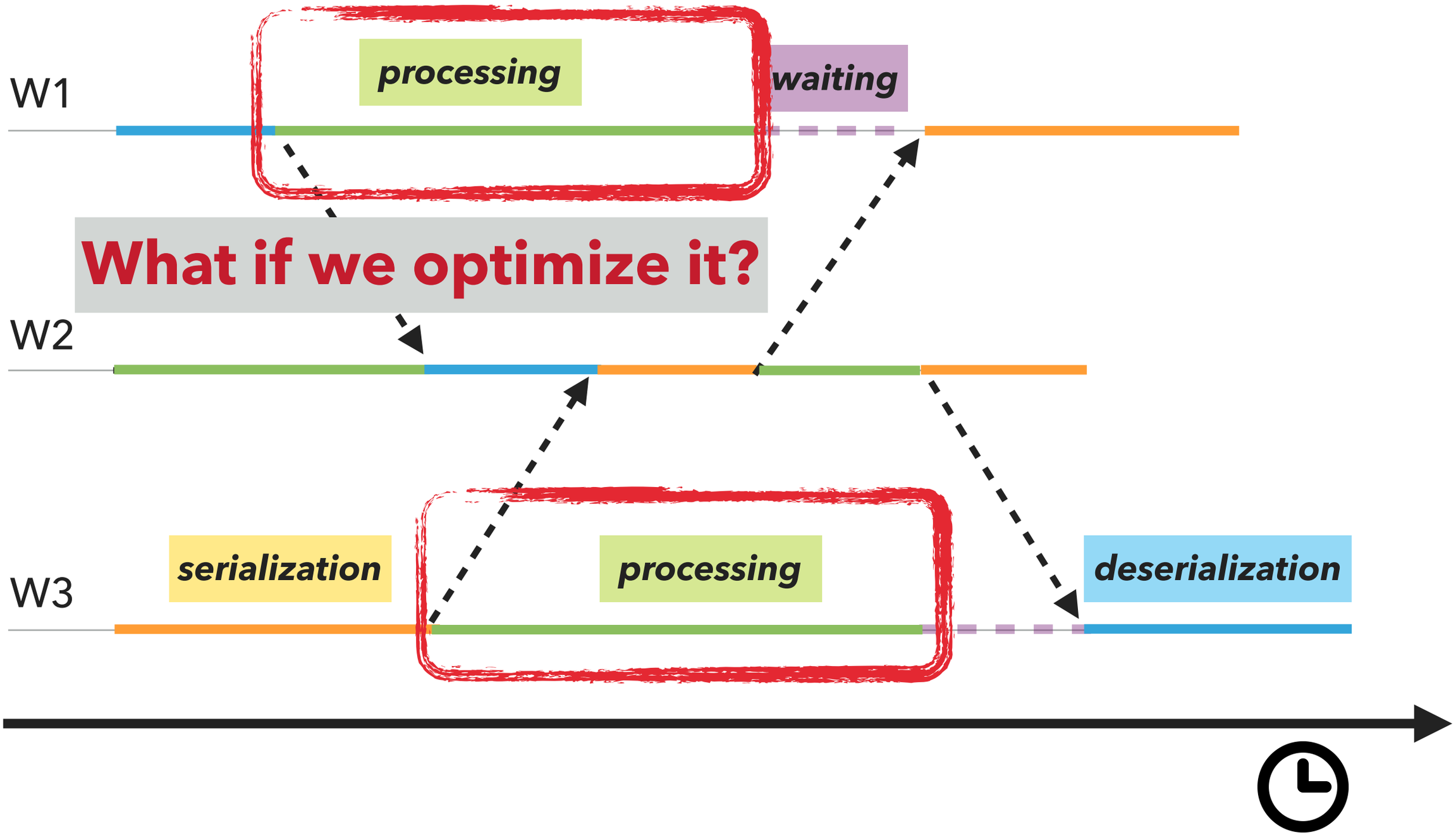
CONVENTIONAL: A BOTTLENECK IN A PARALLEL EXECUTION?



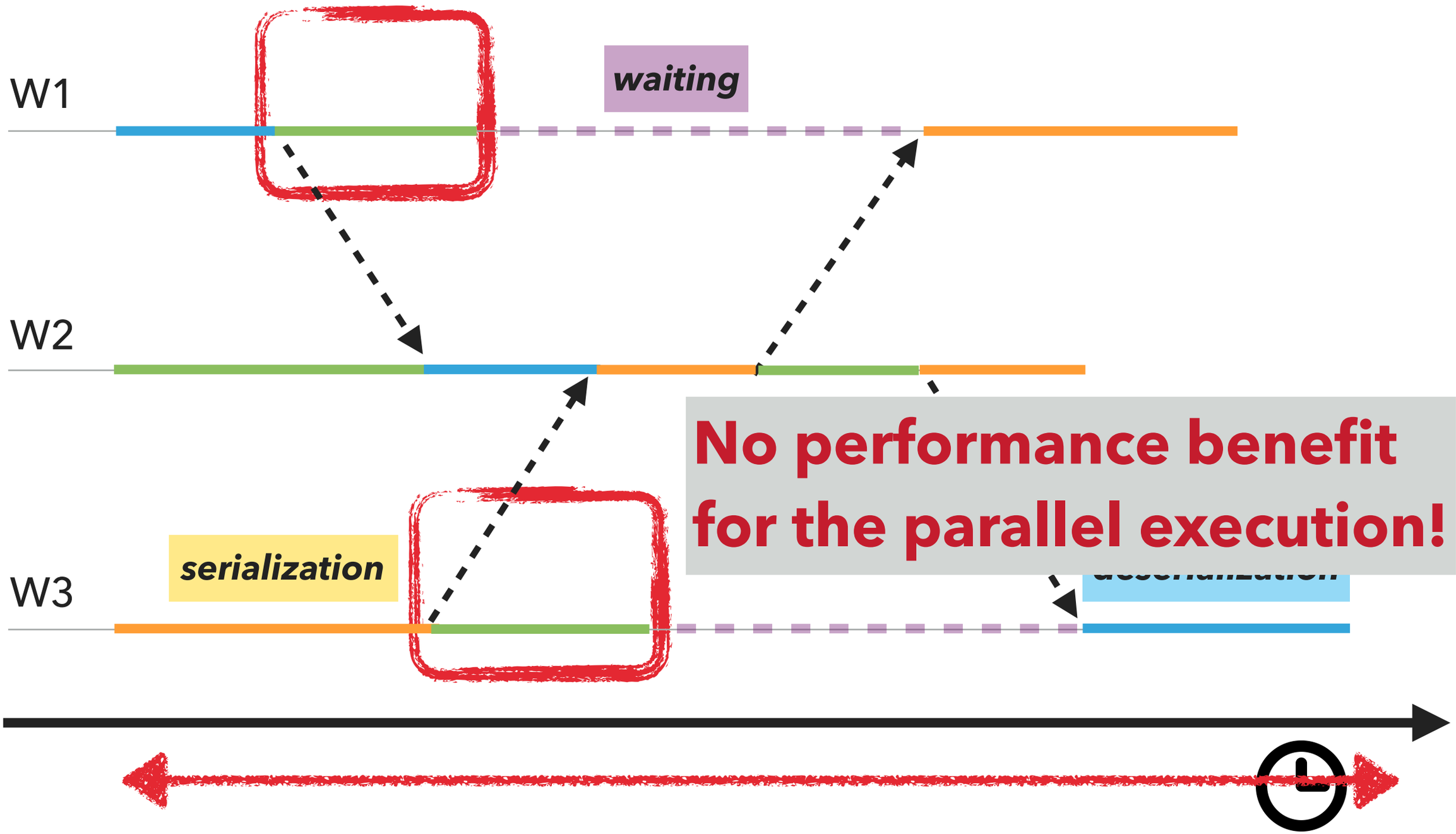
CONVENTIONAL: A BOTTLENECK IN A PARALLEL EXECUTION?



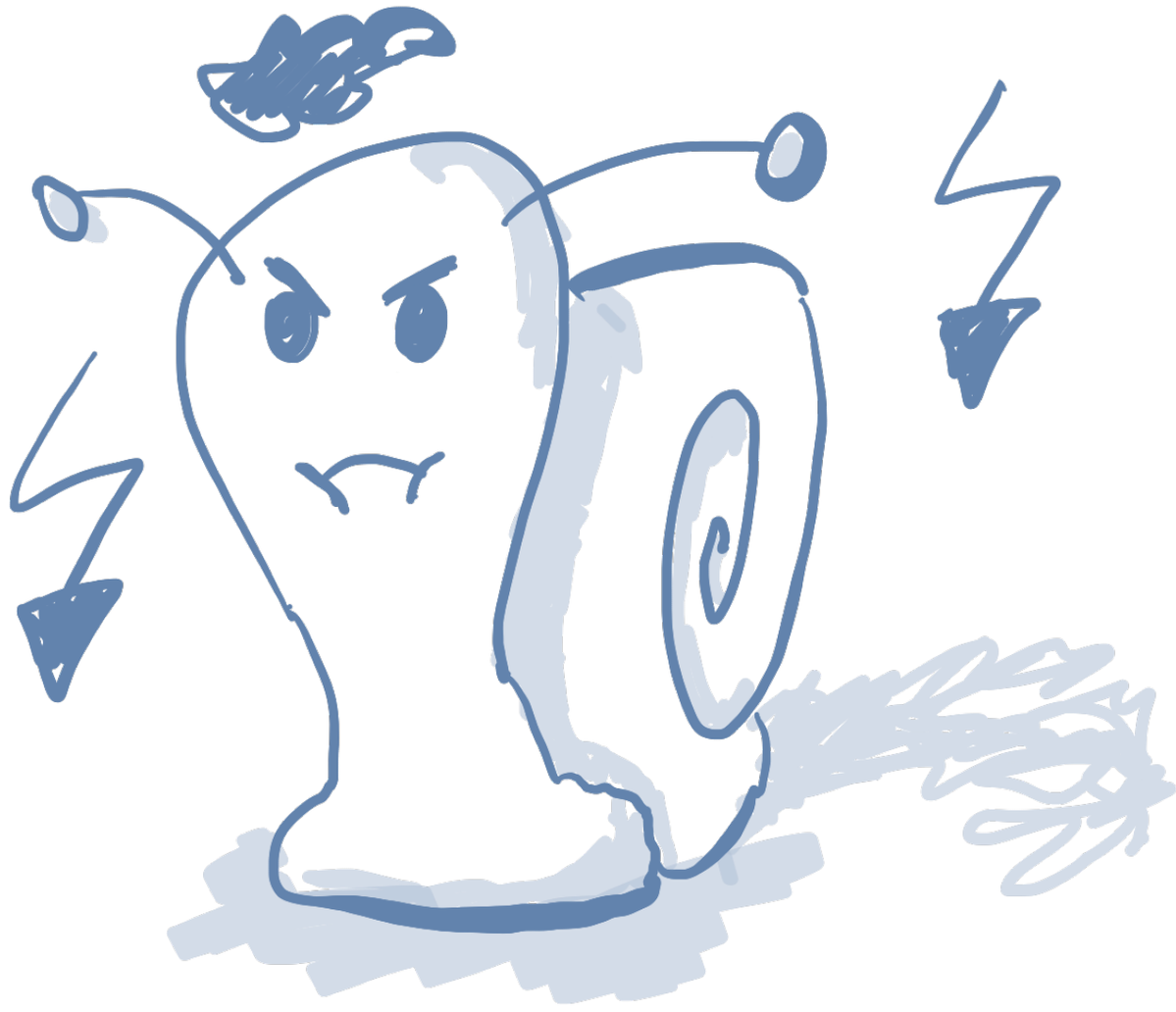
CONVENTIONAL: A BOTTLENECK IN A PARALLEL EXECUTION?



CONVENTIONAL: A BOTTLENECK IN A PARALLEL EXECUTION?

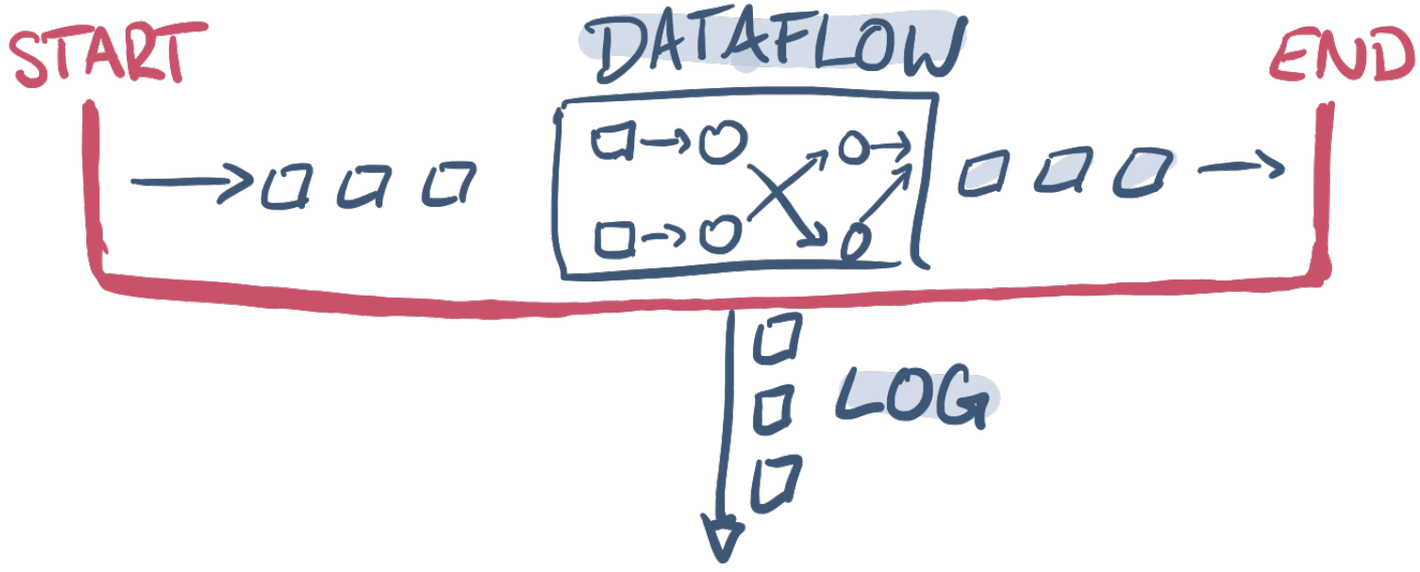


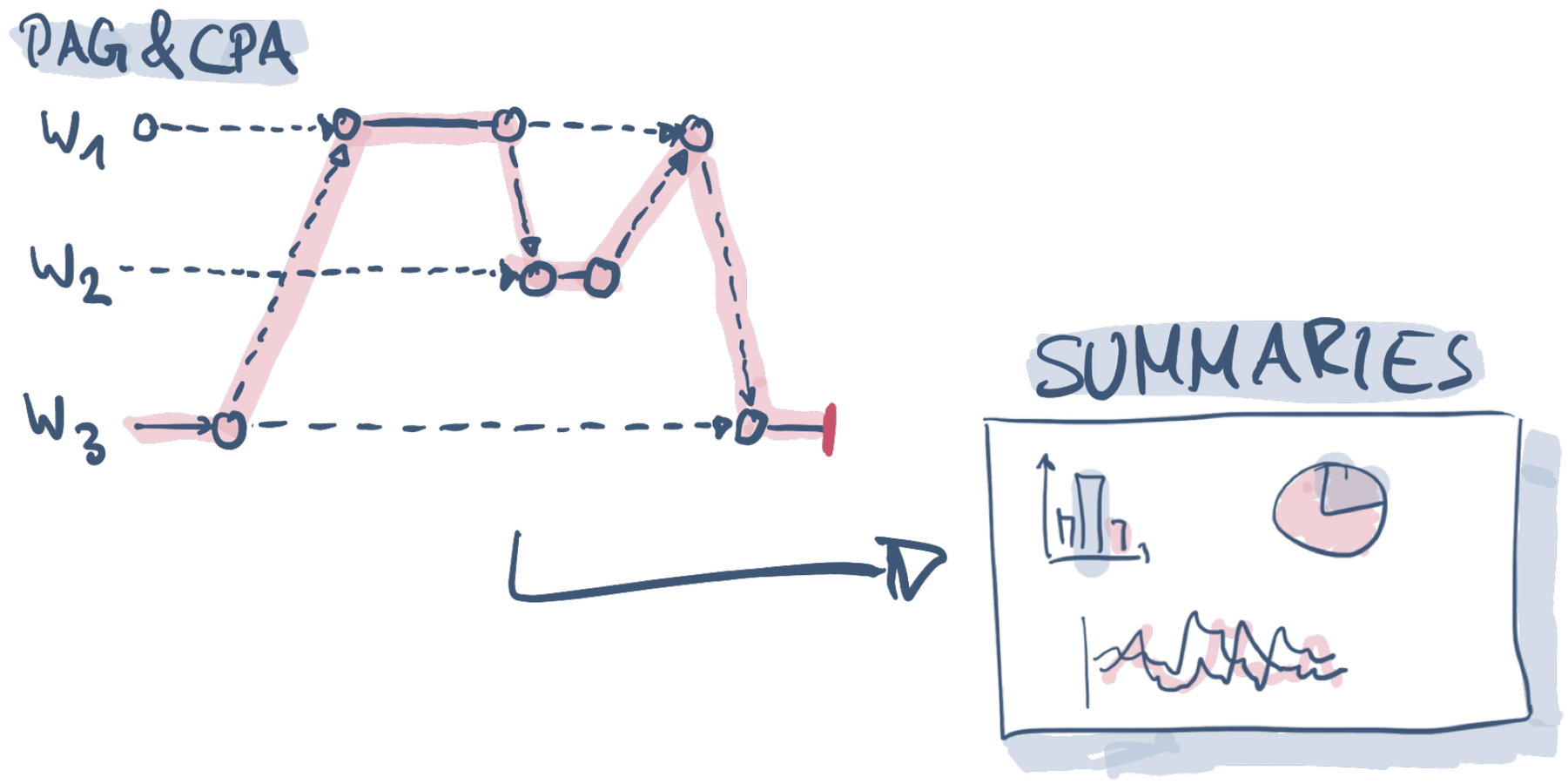
CONVENTIONAL PROFILING CAN BE MISLEADING



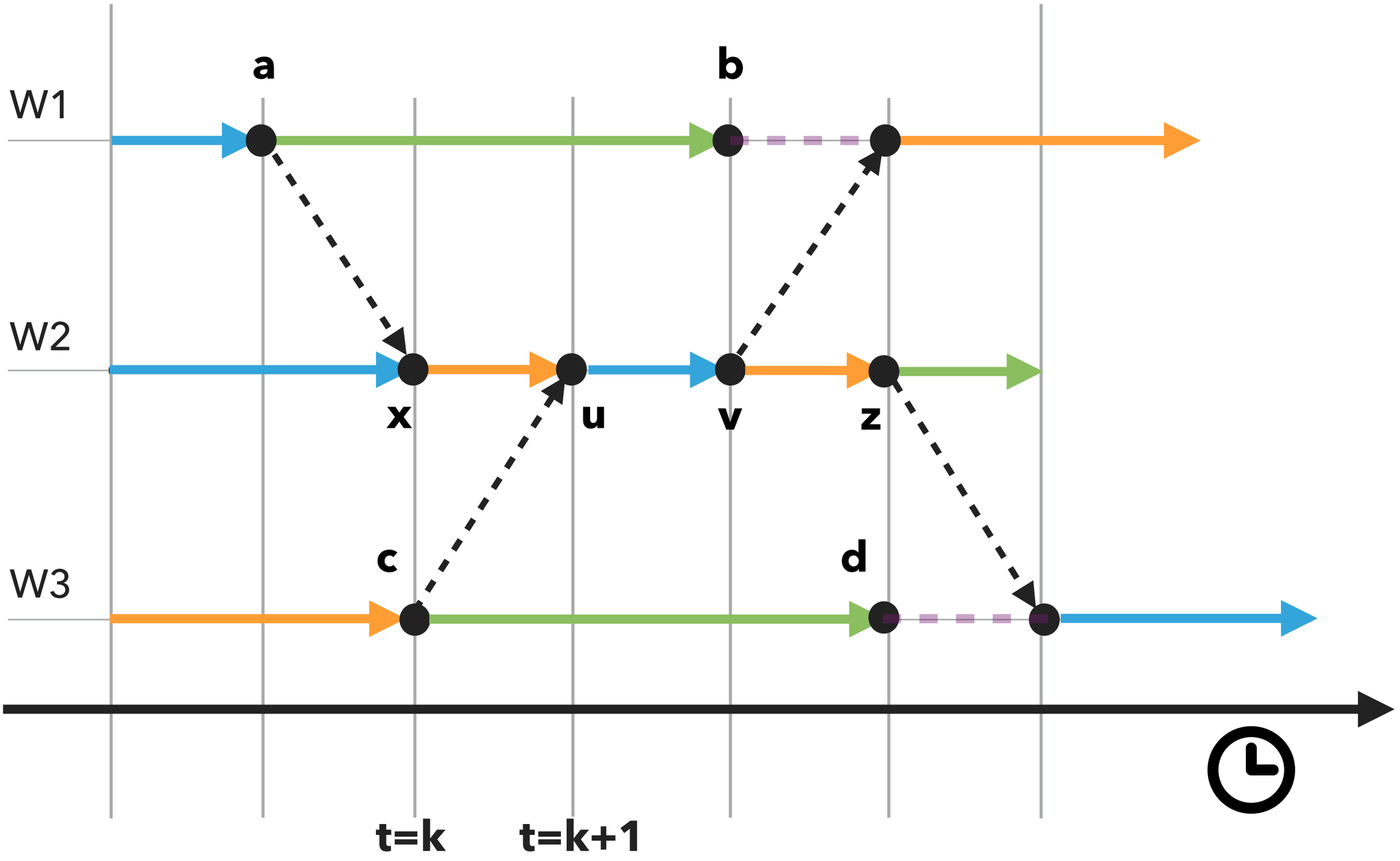
CRITICAL PATH ANALYSIS

CRITICAL PATH ANALYSIS

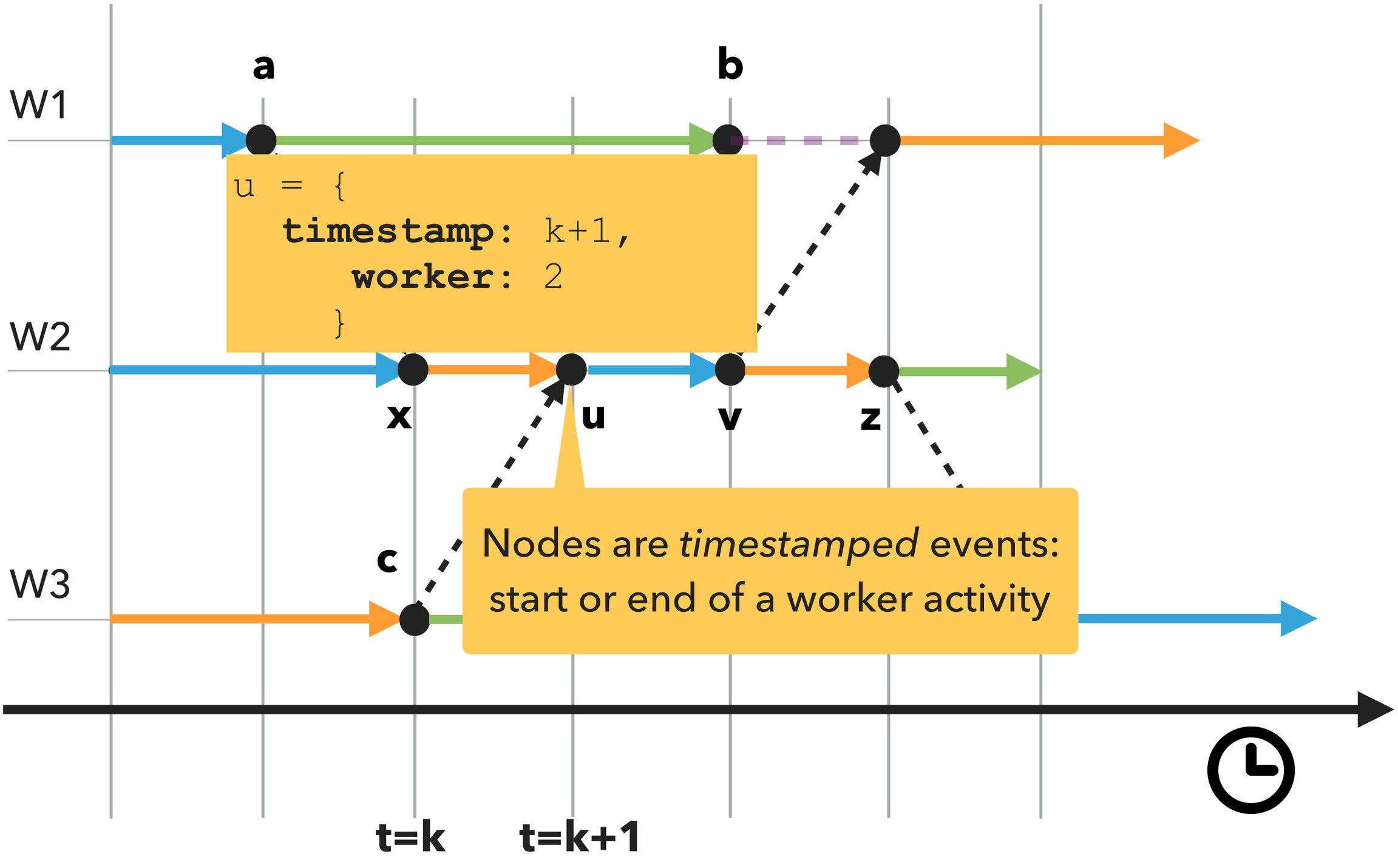




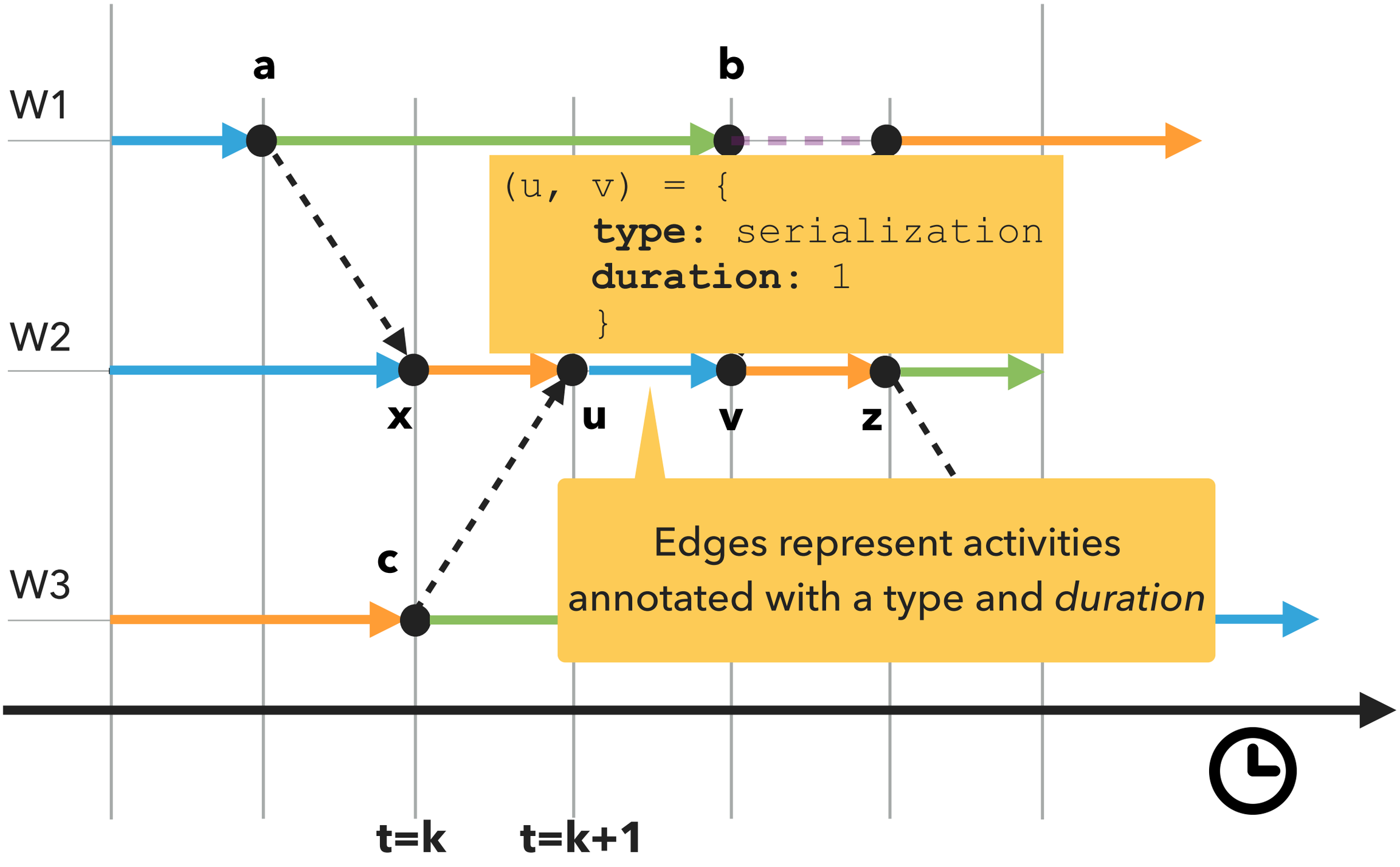
THE PROGRAM ACTIVITY GRAPH (PAG)



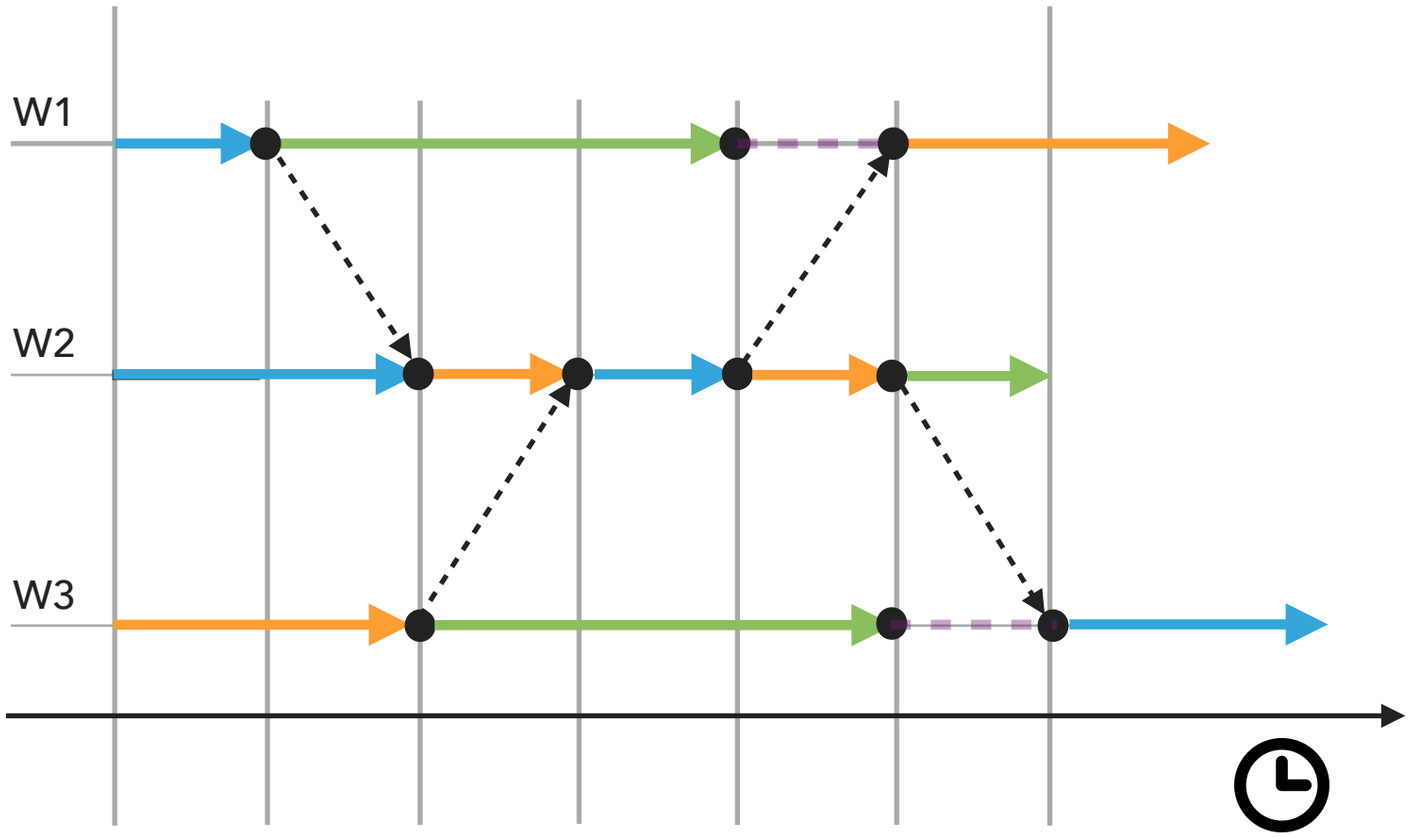
THE PROGRAM ACTIVITY GRAPH (PAG)



THE PROGRAM ACTIVITY GRAPH (PAG)

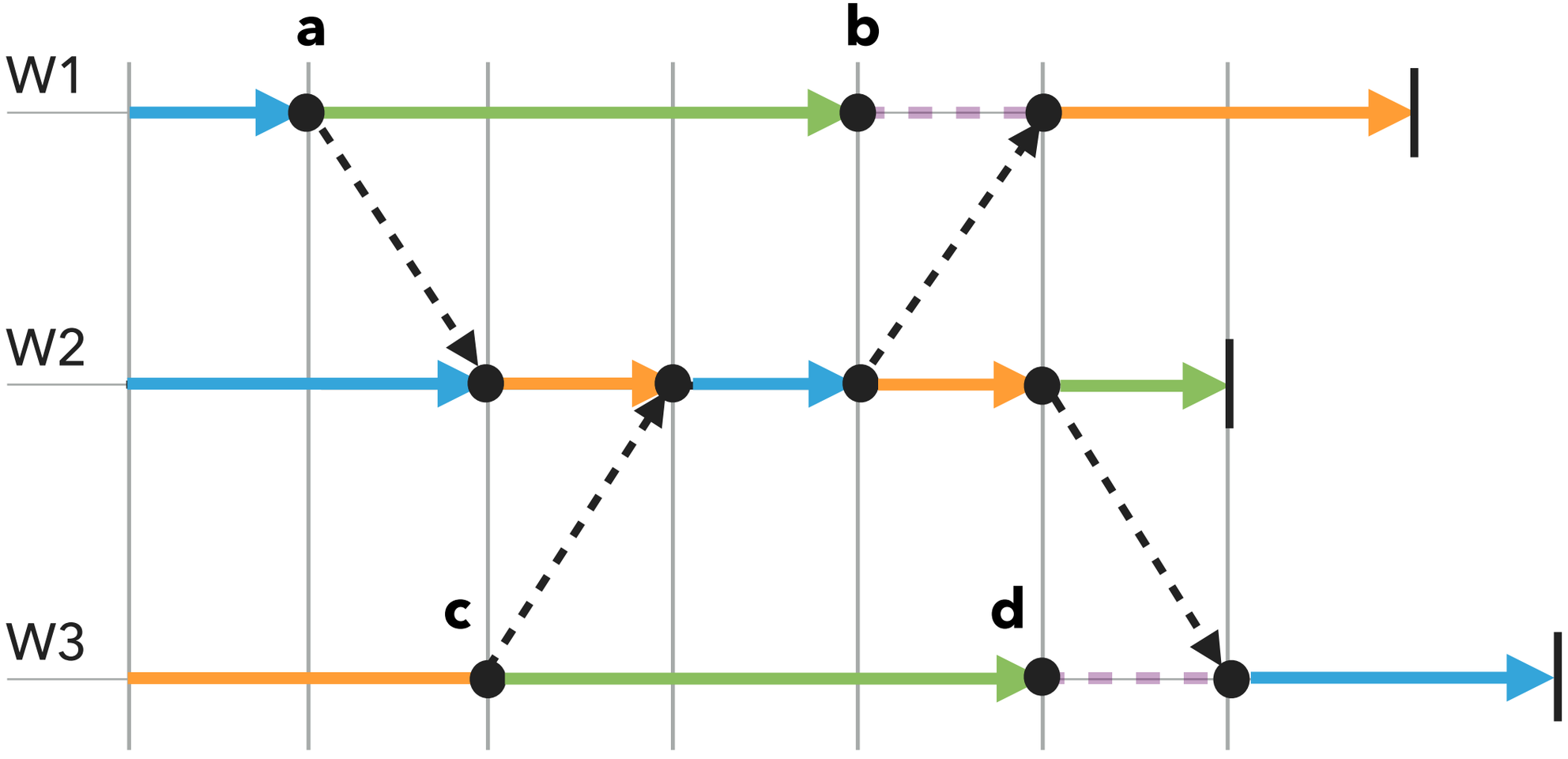


The Program Activity Graph captures **computational dependencies** among parallel workers



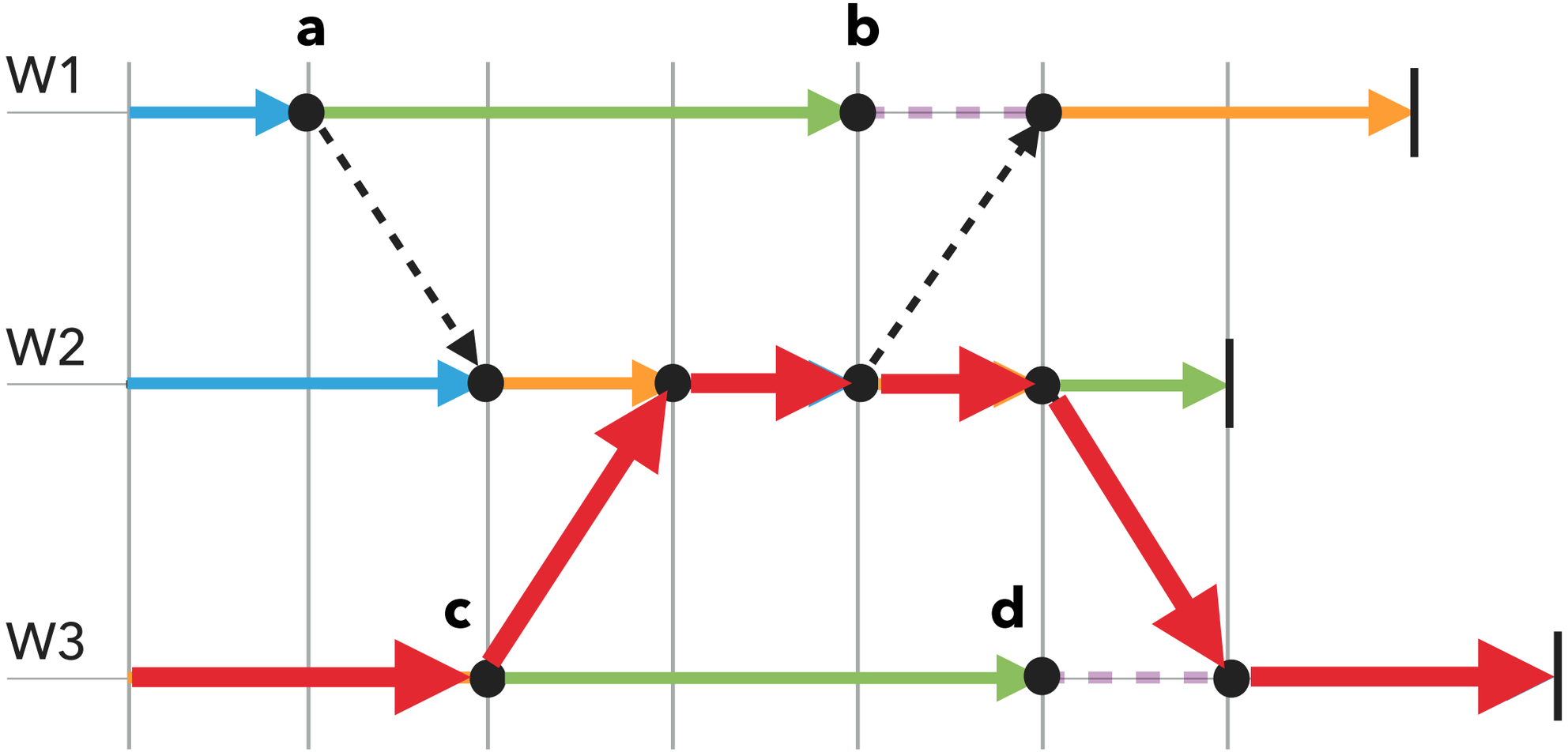
- Which activities **delay** the overall execution?
  - i.e. which activities lie on the **critical path** of execution?

CRITICAL PATH: A BOTTLENECK IN A PARALLEL EXECUTION!



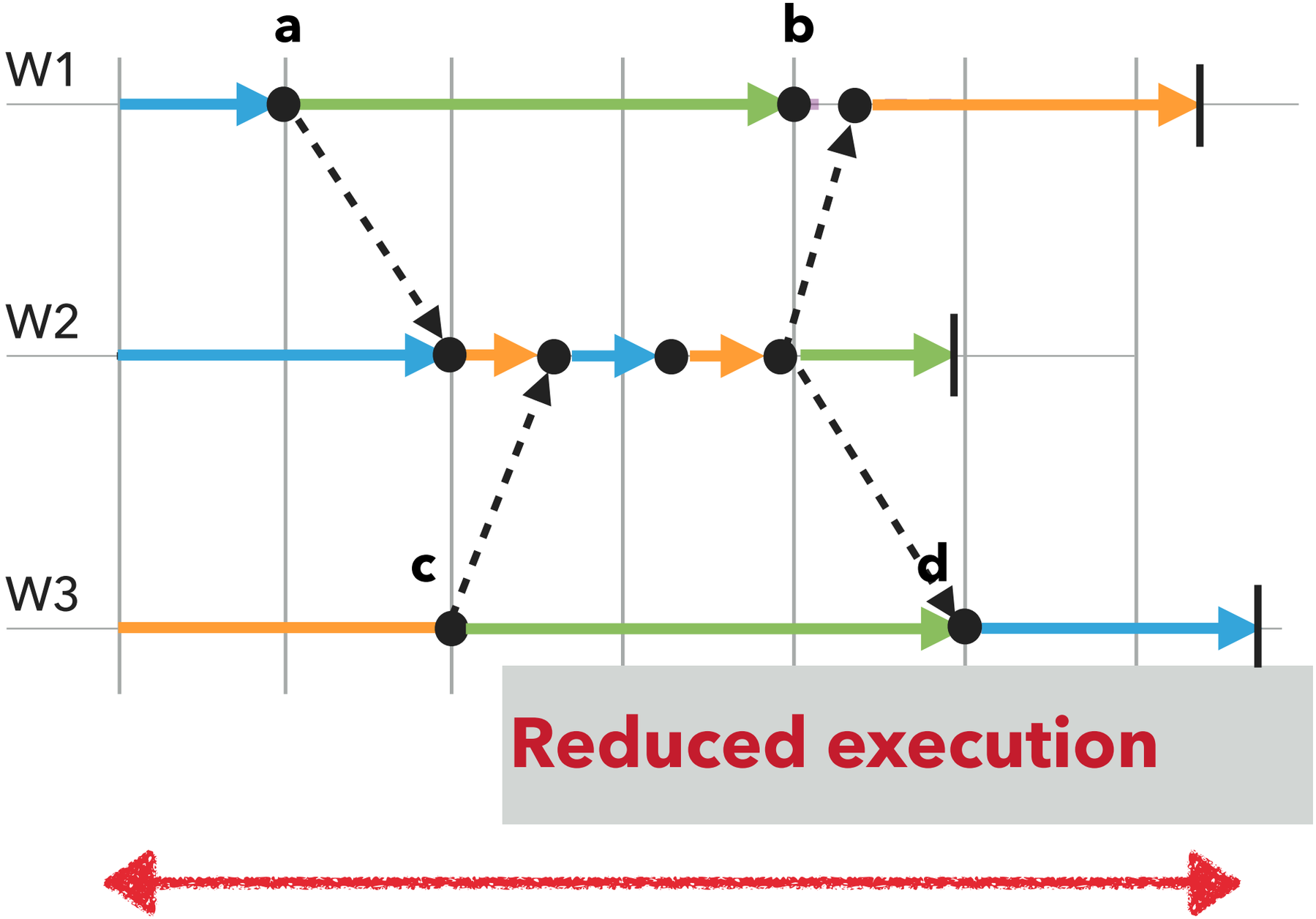
The **longest path** in the execution history
(not considering waiting activities)

CRITICAL PATH: A BOTTLENECK IN A PARALLEL EXECUTION!



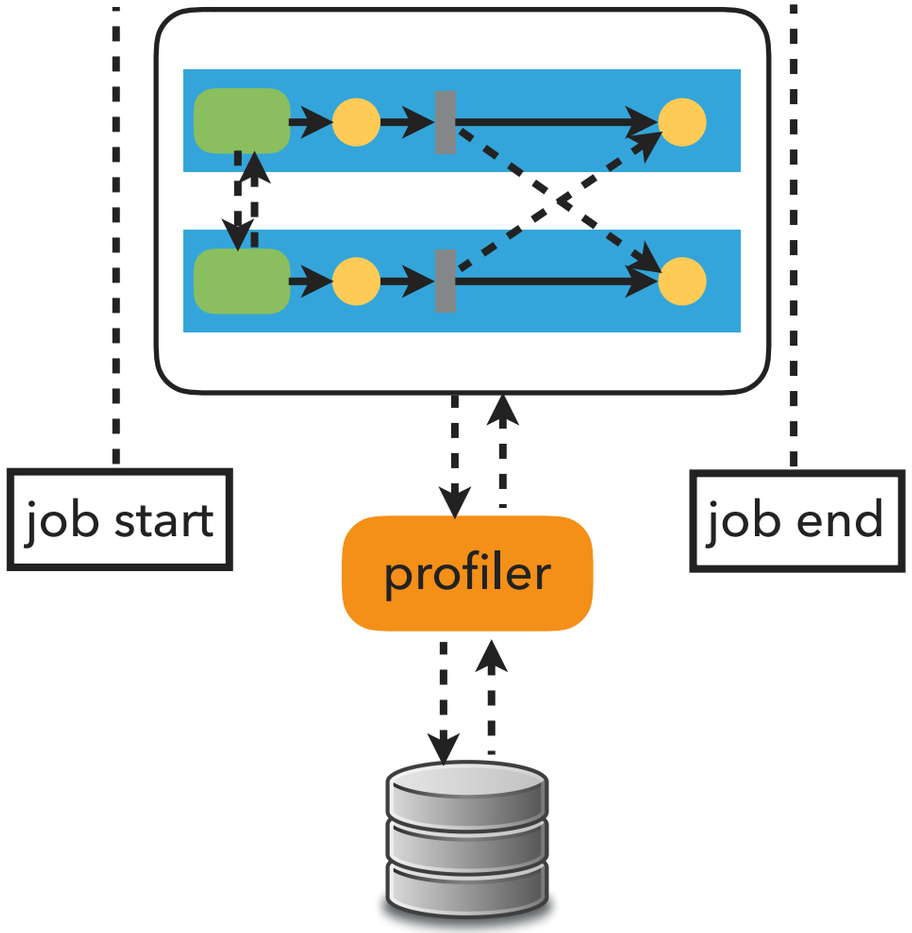
The **longest path** in the execution history
(not considering waiting activities)

CRITICAL PATH: A BOTTLENECK IN A PARALLEL EXECUTION!

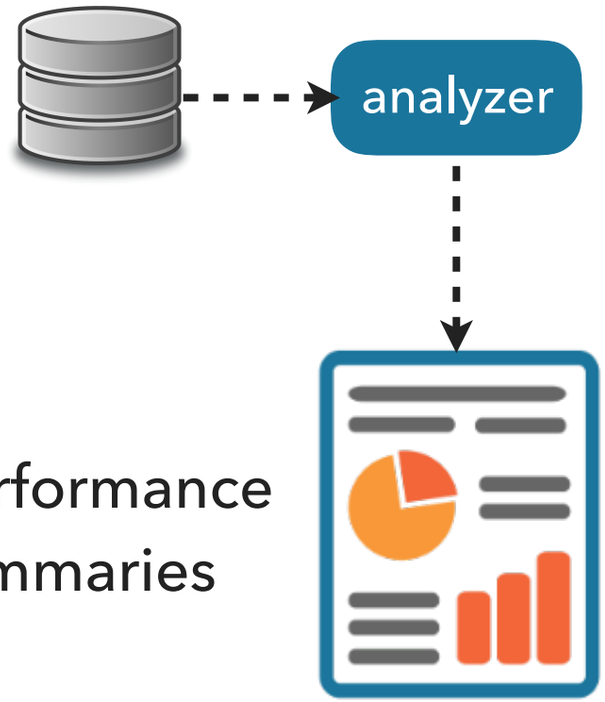


POST-MORTEM CRITICAL PATH ANALYSIS IS EASY

1. Collect traces during execution



2. Analyze traces offline



Streaming applications run continuously with potentially unbounded input!

- There might be no "job end"
- The PAG and critical path are continuously evolving
- Stale profiling information is not useful

How do we compute the critical path online?

SnailTrail: Generalizing Critical Paths for Online Analysis of Distributed Dataflows*

strymon.systems.ethz.ch

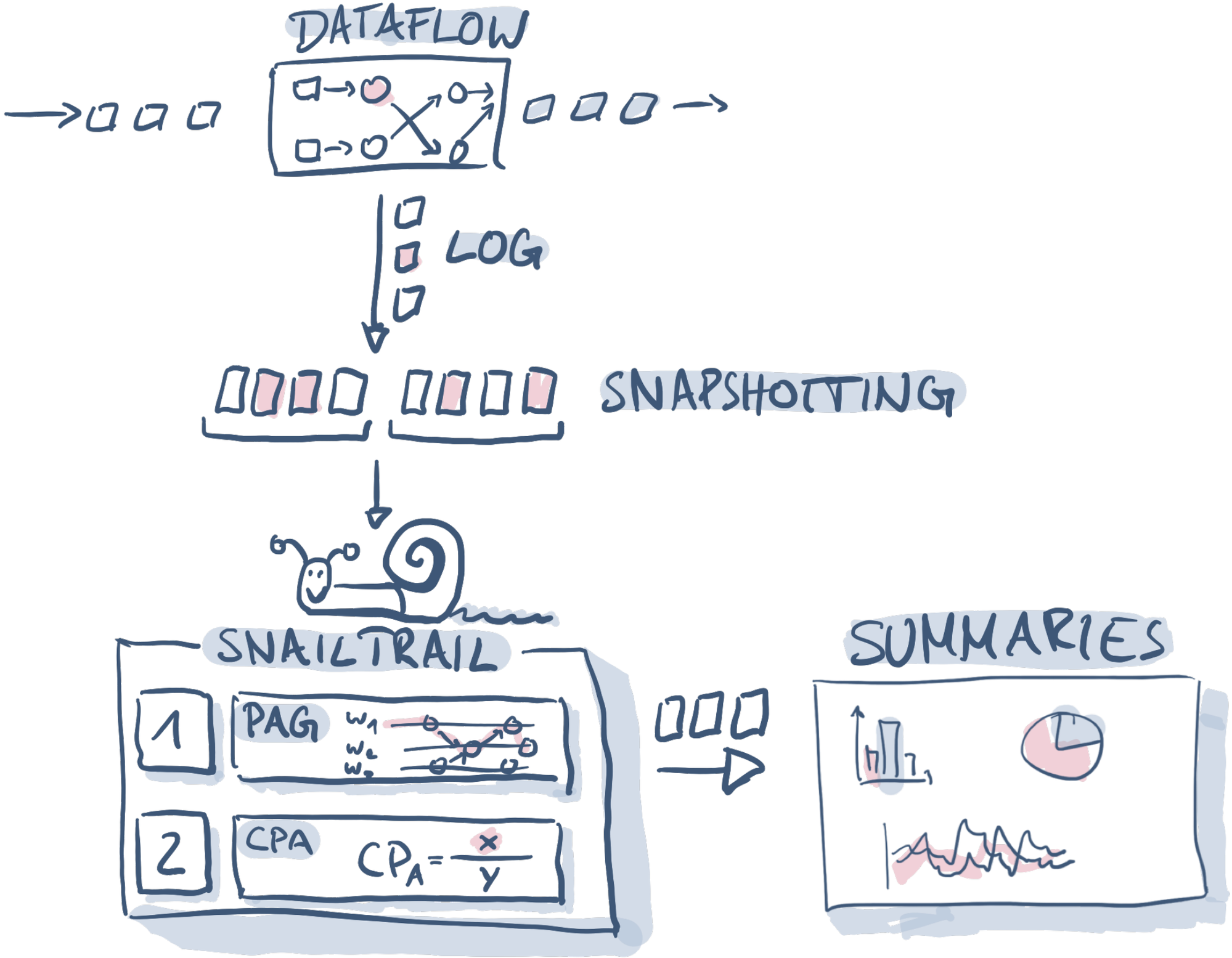
Moritz Hoffmann, Andrea Lattuada, John Liagouris, Vasiliki Kalavri,
Desislava Dimitrova, Sebastian Wicki, Zaheer Chothia, Timothy Roscoe

Systems Group, Department of Computer Science, ETH Zürich

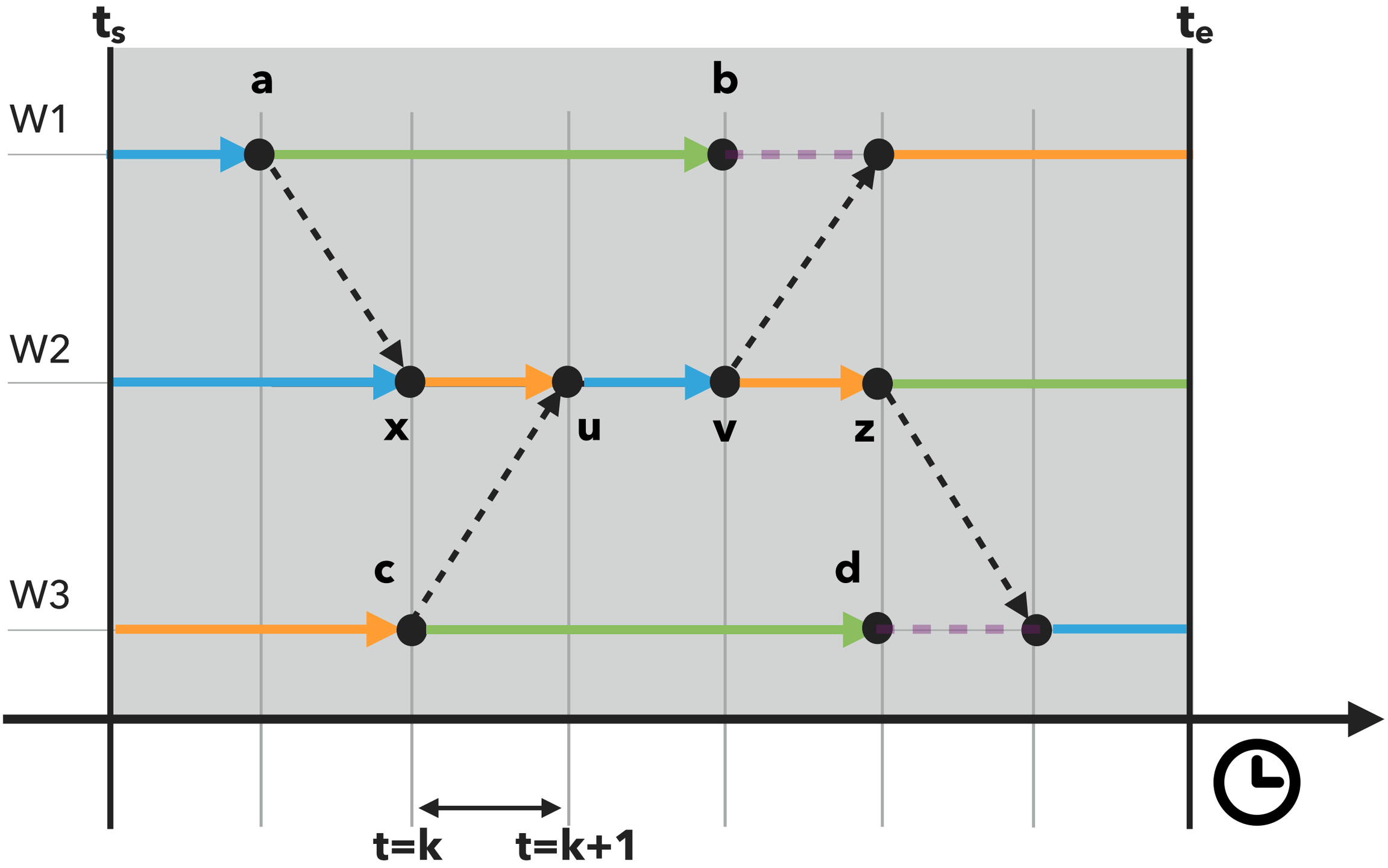
firstname.lastname@inf.ethz.ch

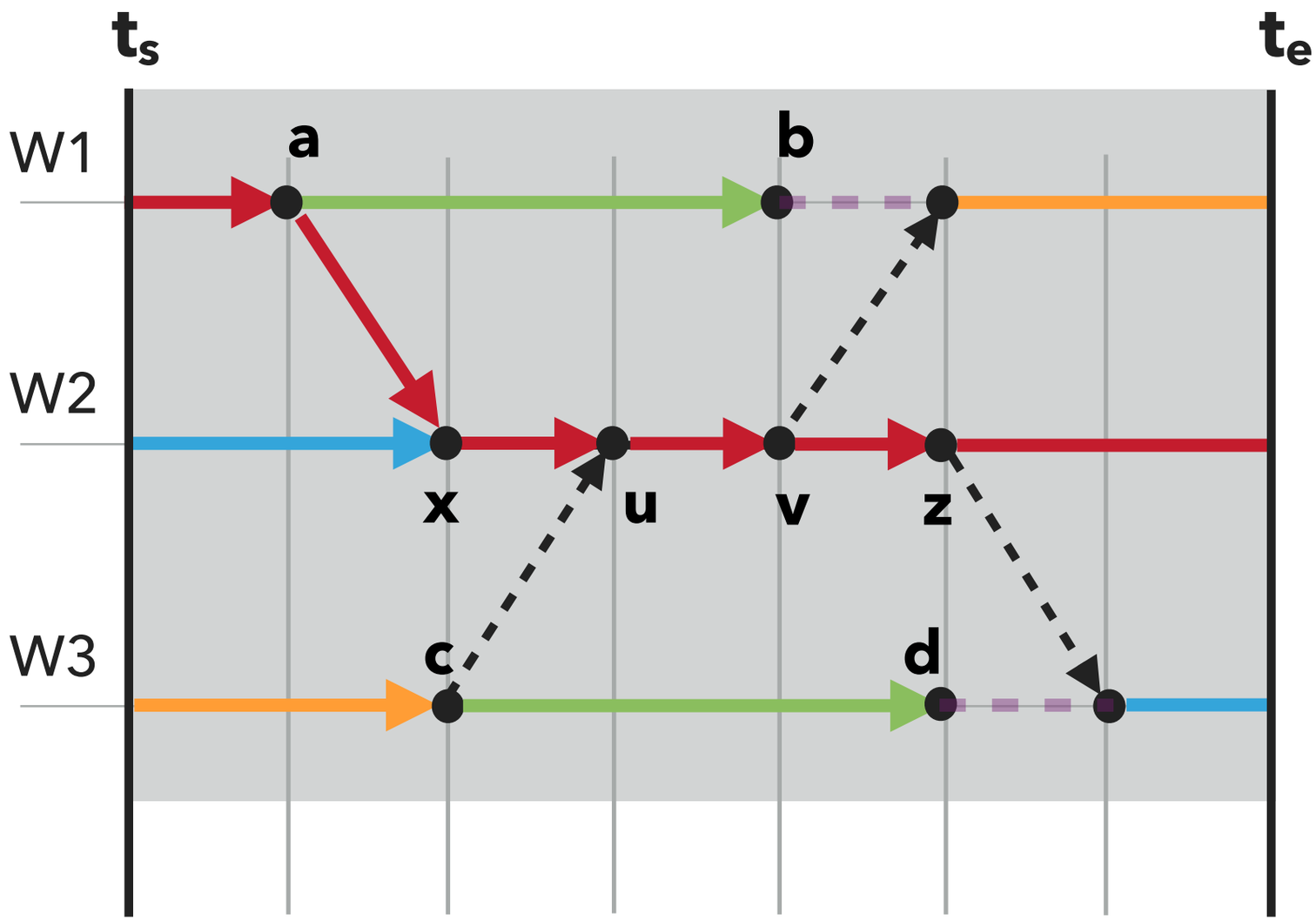
ONLINE CRITICAL PATH ANALYSIS

ONLINE ANALYSIS OF TRACE SNAPSHOTS

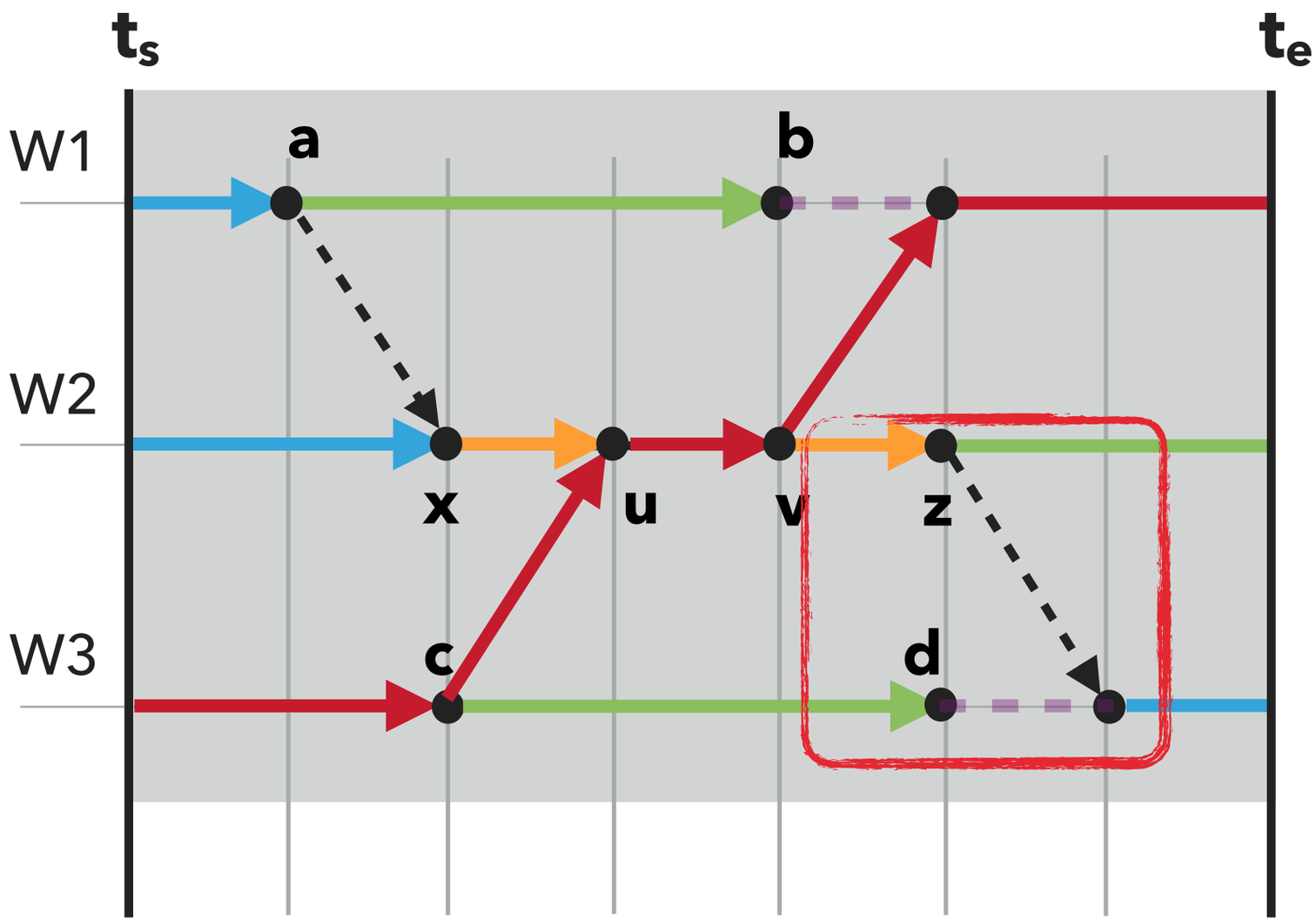


PROGRAM ACTIVITY GRAPH SNAPSHOT

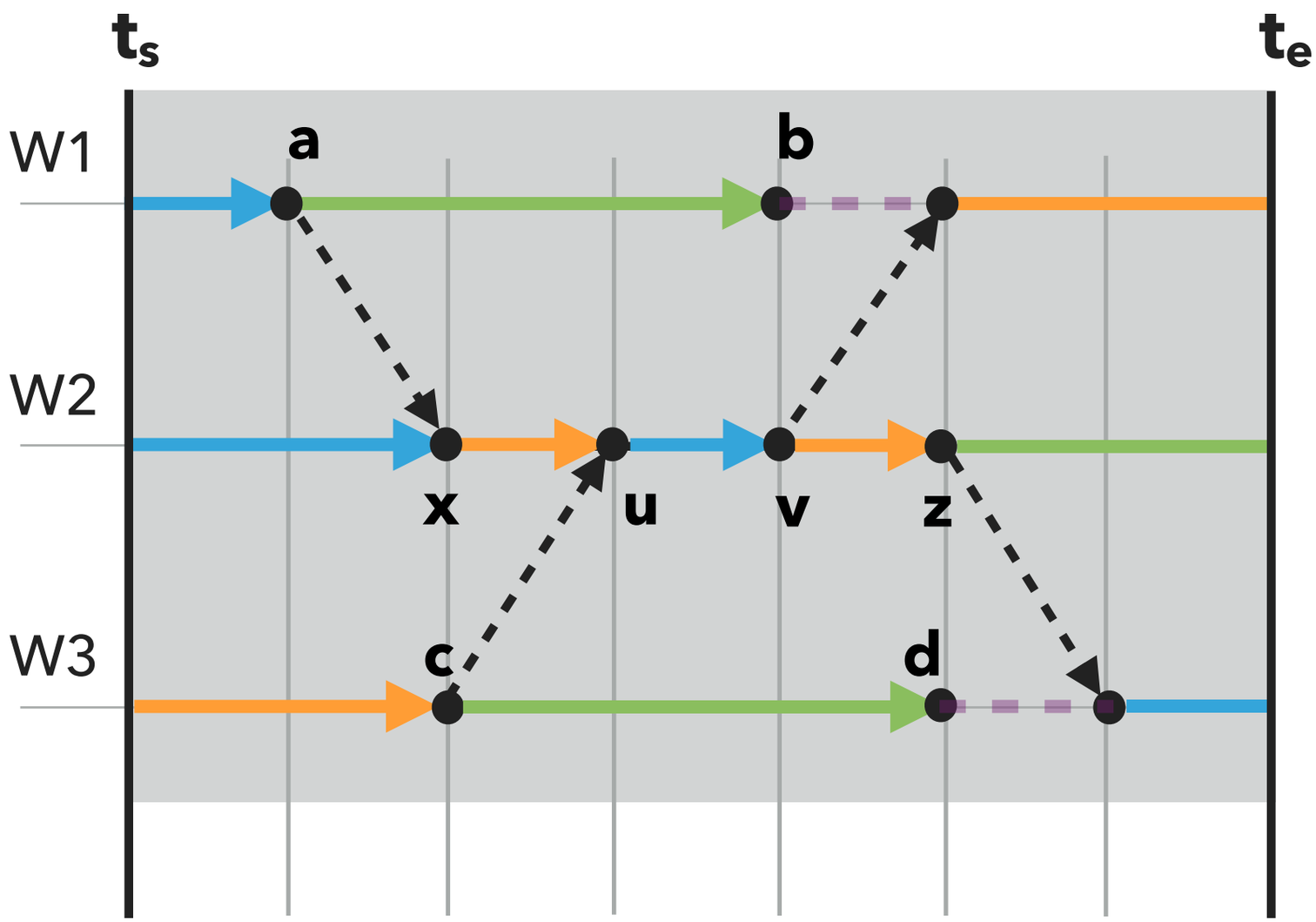




- All paths have the same length: $t_e - t_s$

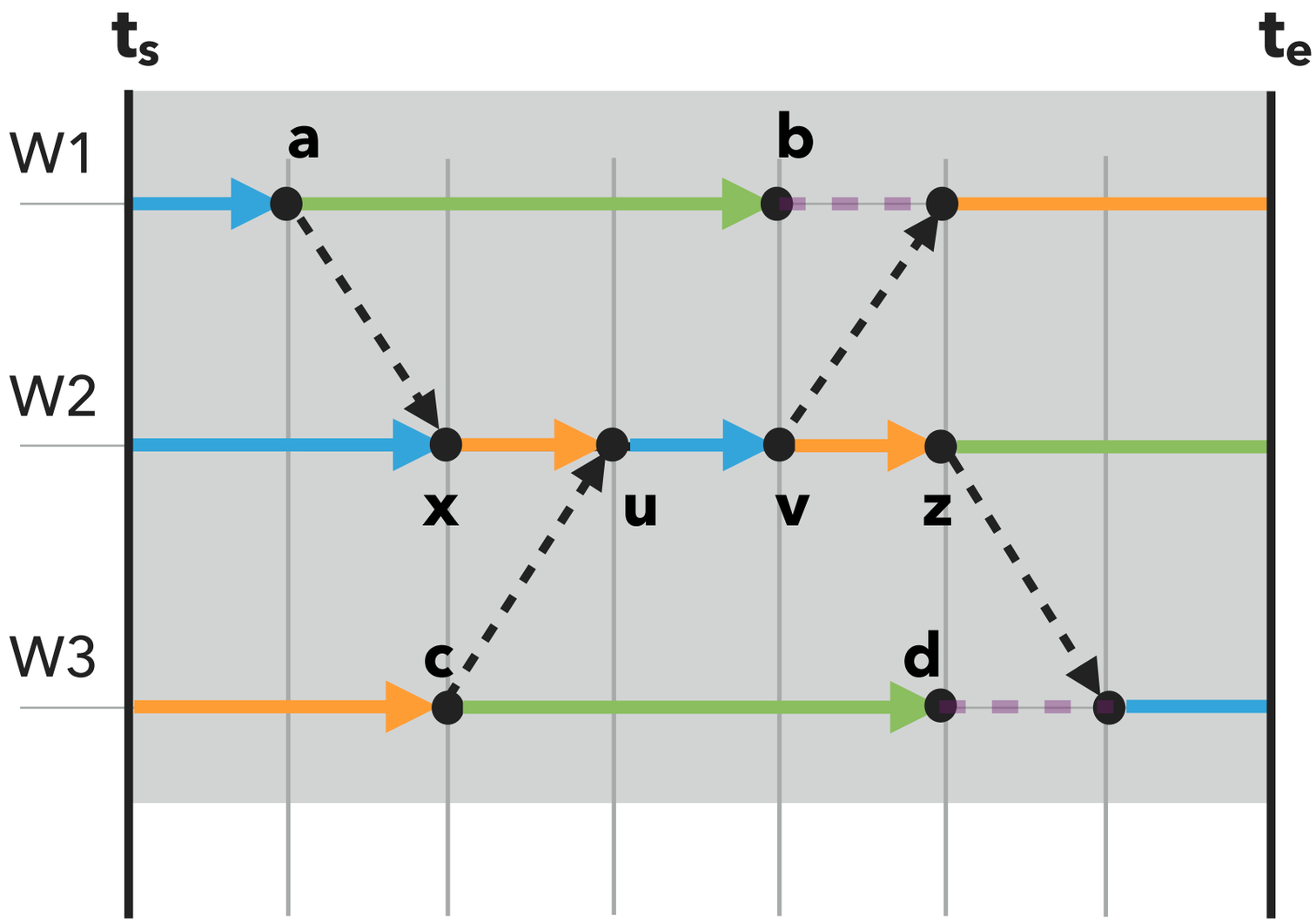


- All paths have the same length: $t_e - t_s$
- Choosing a random path might miss critical activities



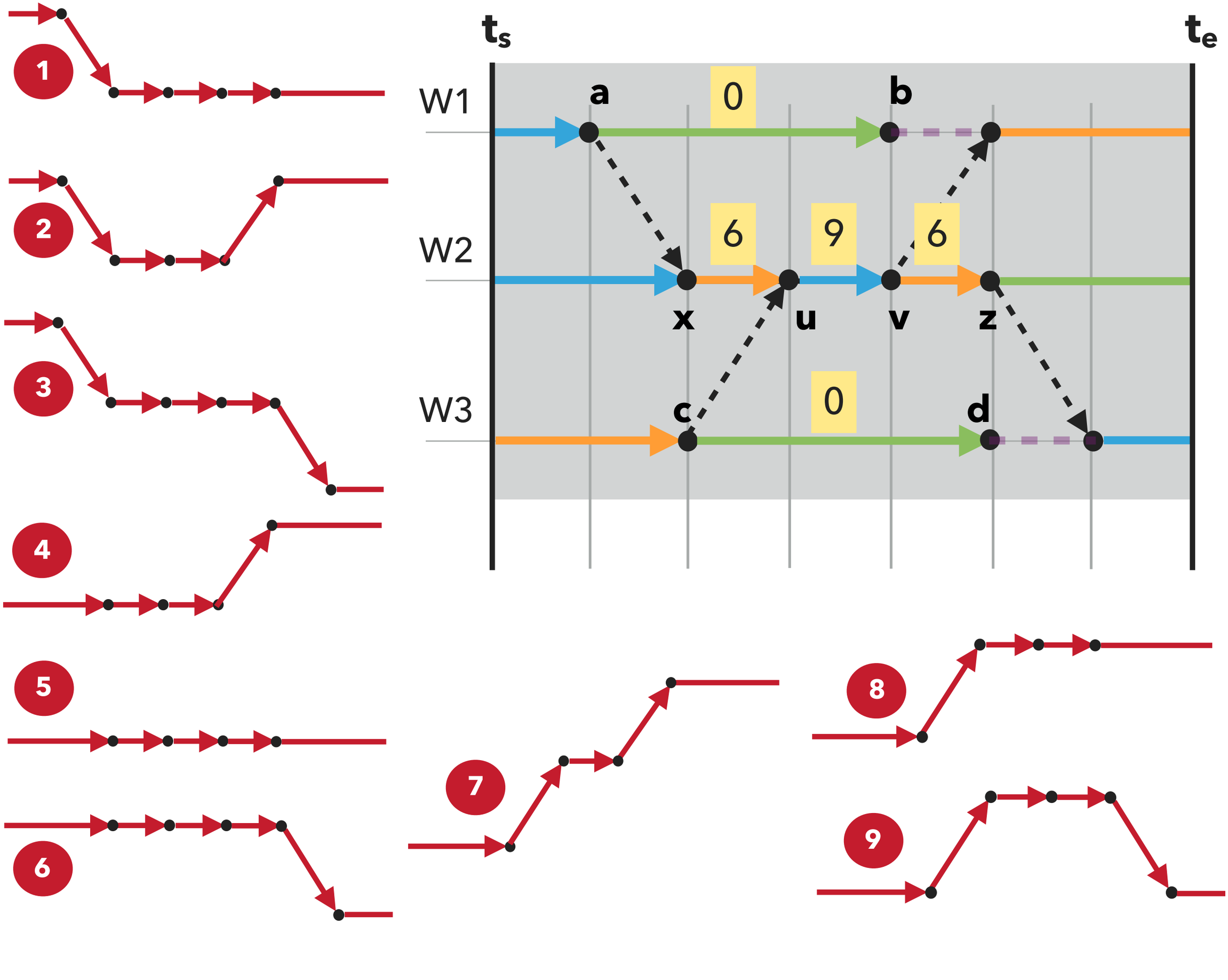
- All paths have the same length: $t_e - t_s$
- Choosing a random path might miss critical activities
- Enumerating all paths is impractical

All paths are potentially part of the evolving critical path



How to rank activities with regard to criticality?

Intuition: the more paths an activity appears on the more probable it is that this activity is critical


1
2
3
4
5
6
7
8
9
t_s
t_e
W1
W2
W3
a
b
x
u
v
z
c
d
0
6
9
6
0

CRITICAL PARTICIPATION (CP METRIC)

An estimation of the activity's participation in the critical path

centrality: the number of paths this activity appears on

activity duration: edge weight

$$CP_a = \frac{C(a) \cdot a_w}{N(t_e - t_s)}$$

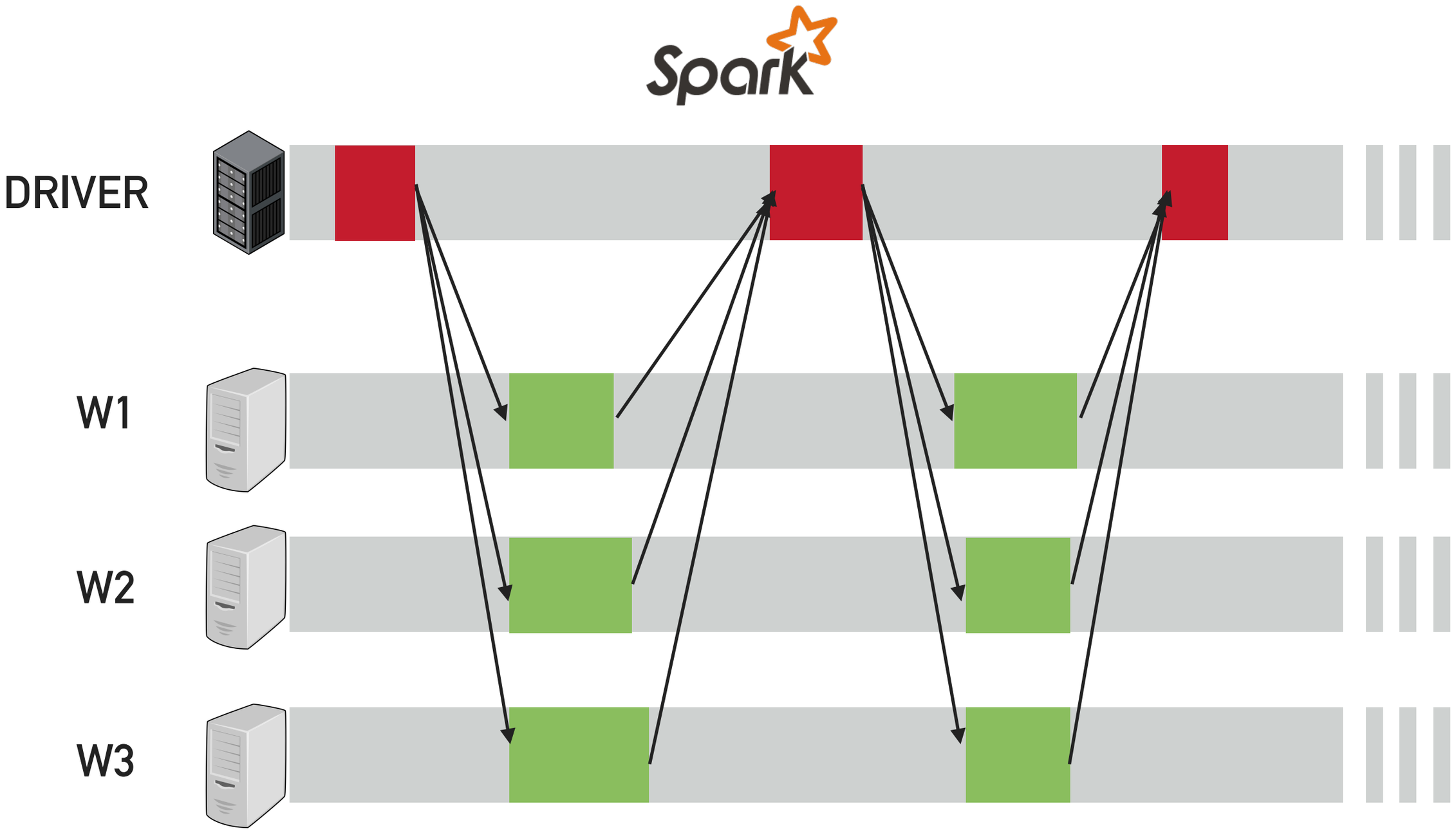
Can be computed without path enumeration!

total number of paths in the snapshot

path length (identical for all paths)

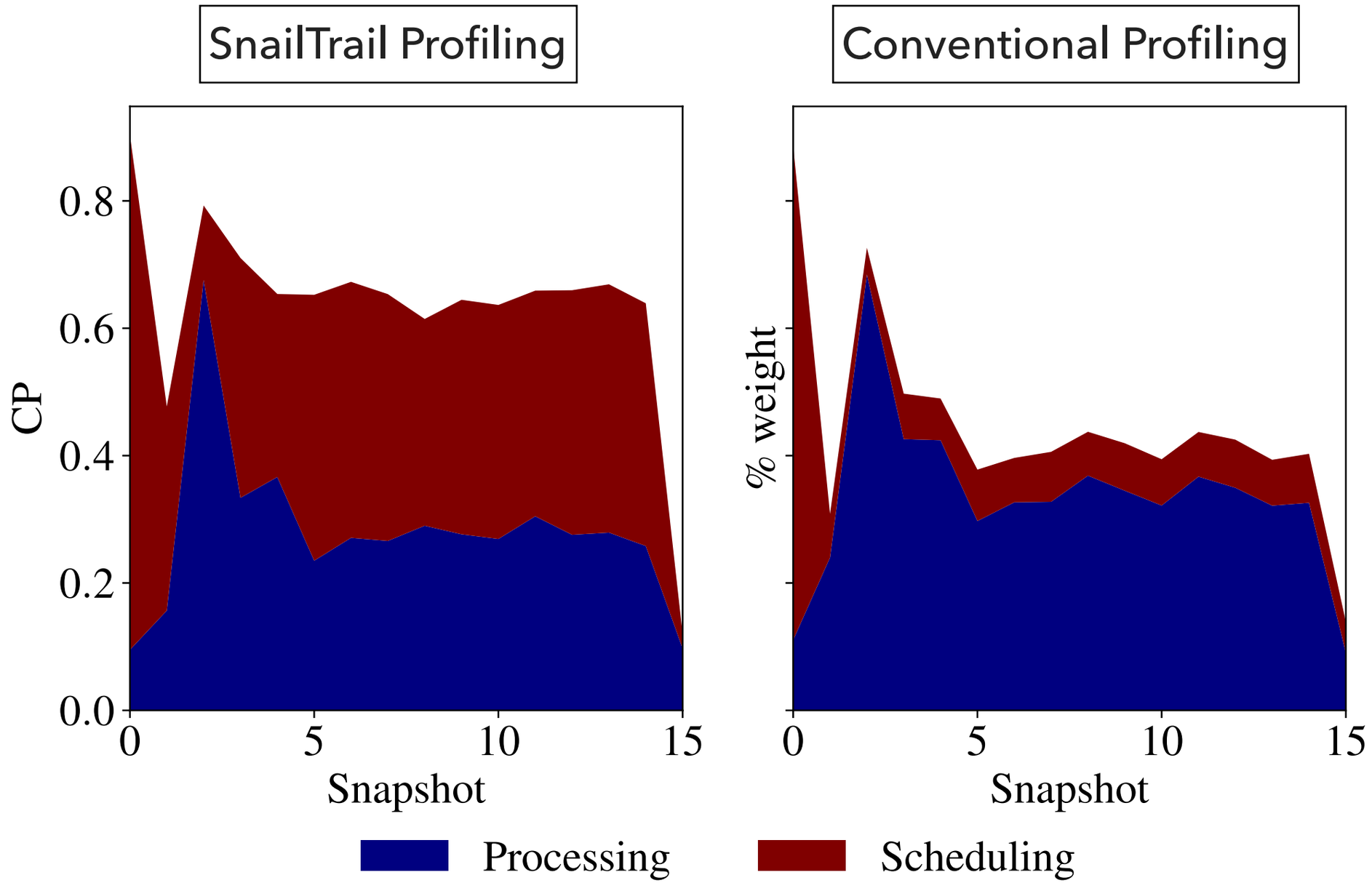
ONLINE PERFORMANCE ANALYSIS WITH SNAILTRAIL

EXAMPLE: TASK SCHEDULING IN APACHE SPARK



Venkataraman, Shivaram, et al. "Drizzle: Fast and adaptable stream processing at scale." Spark Summit (2016).

SCHEDULING BOTTLENECK IN APACHE SPARK

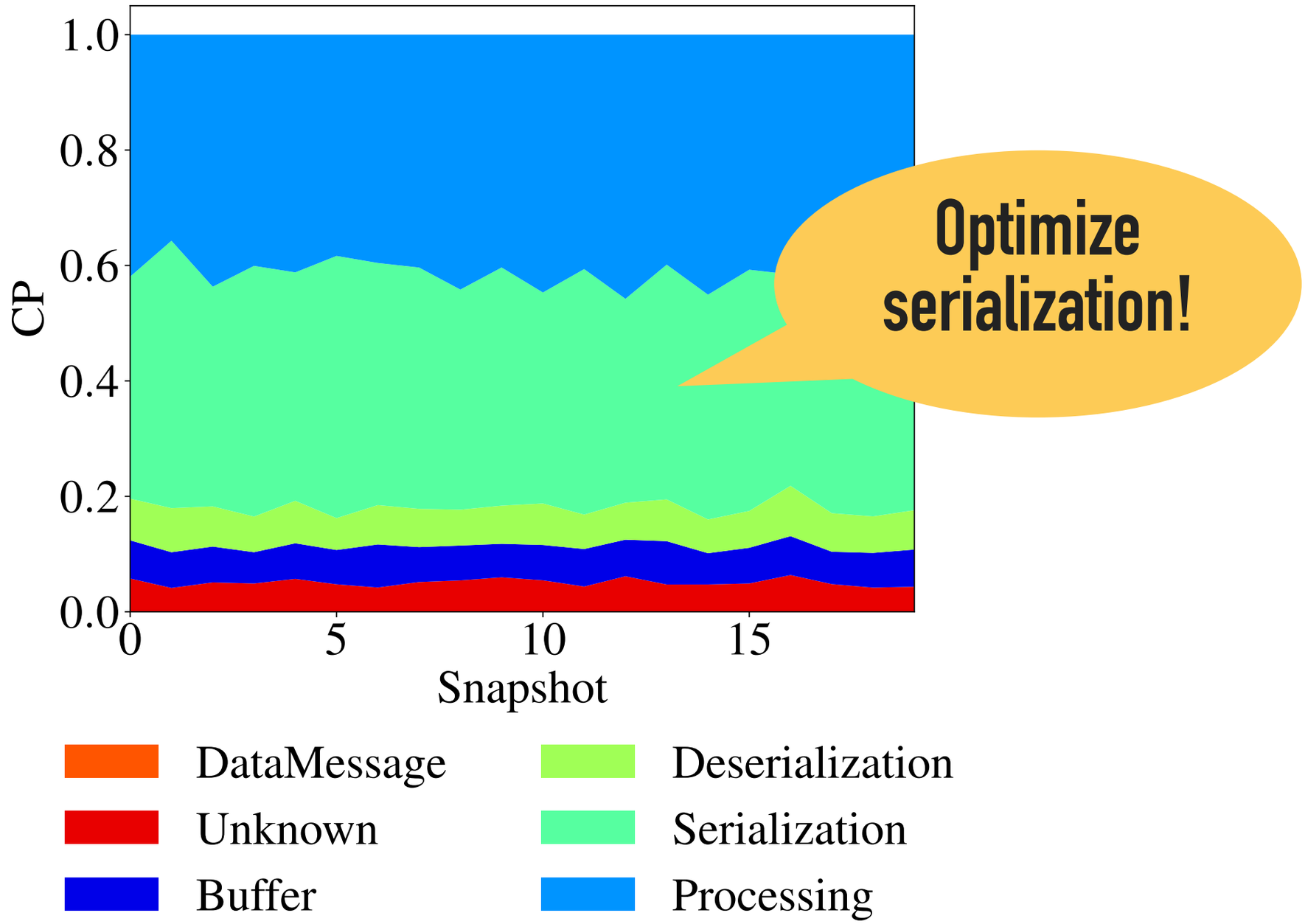


Apache Spark: Yahoo! Streaming Benchmark, 16 workers, 8s snapshots

SNAILTRAIL CP-BASED SUMMARIES

- Activity Summary
  - *which **activity type** is a bottleneck?*

Activity Summary

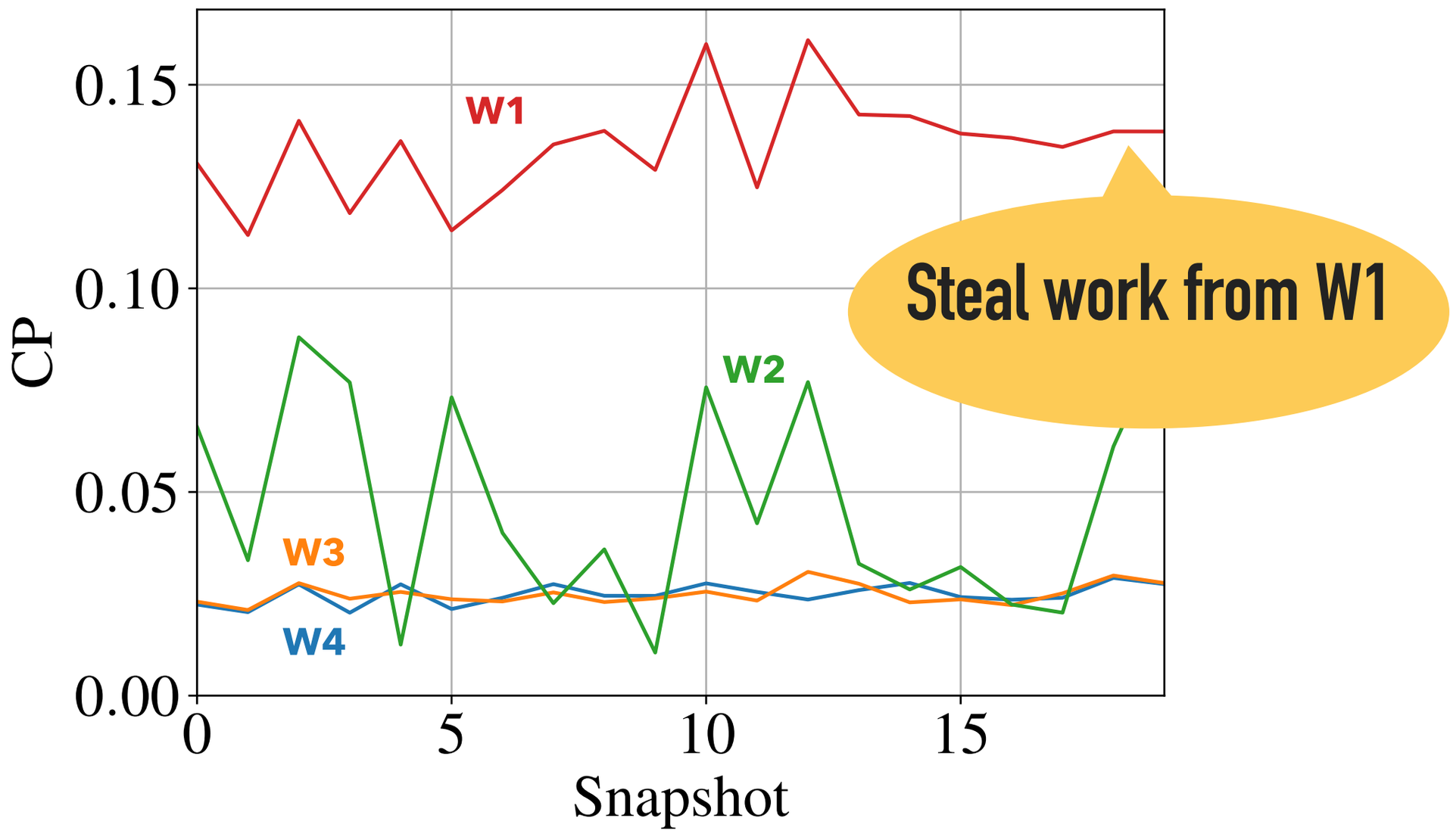


Apache Flink: Dhalion WordCount Benchmark, 4 workers, 1s snapshots

SNAILTRAIL CP-BASED SUMMARIES

- Activity Summary
  - *which **activity type** is a bottleneck?*
- Straggler Summary
  - *which **worker** is a bottleneck?*

Straggler Summary

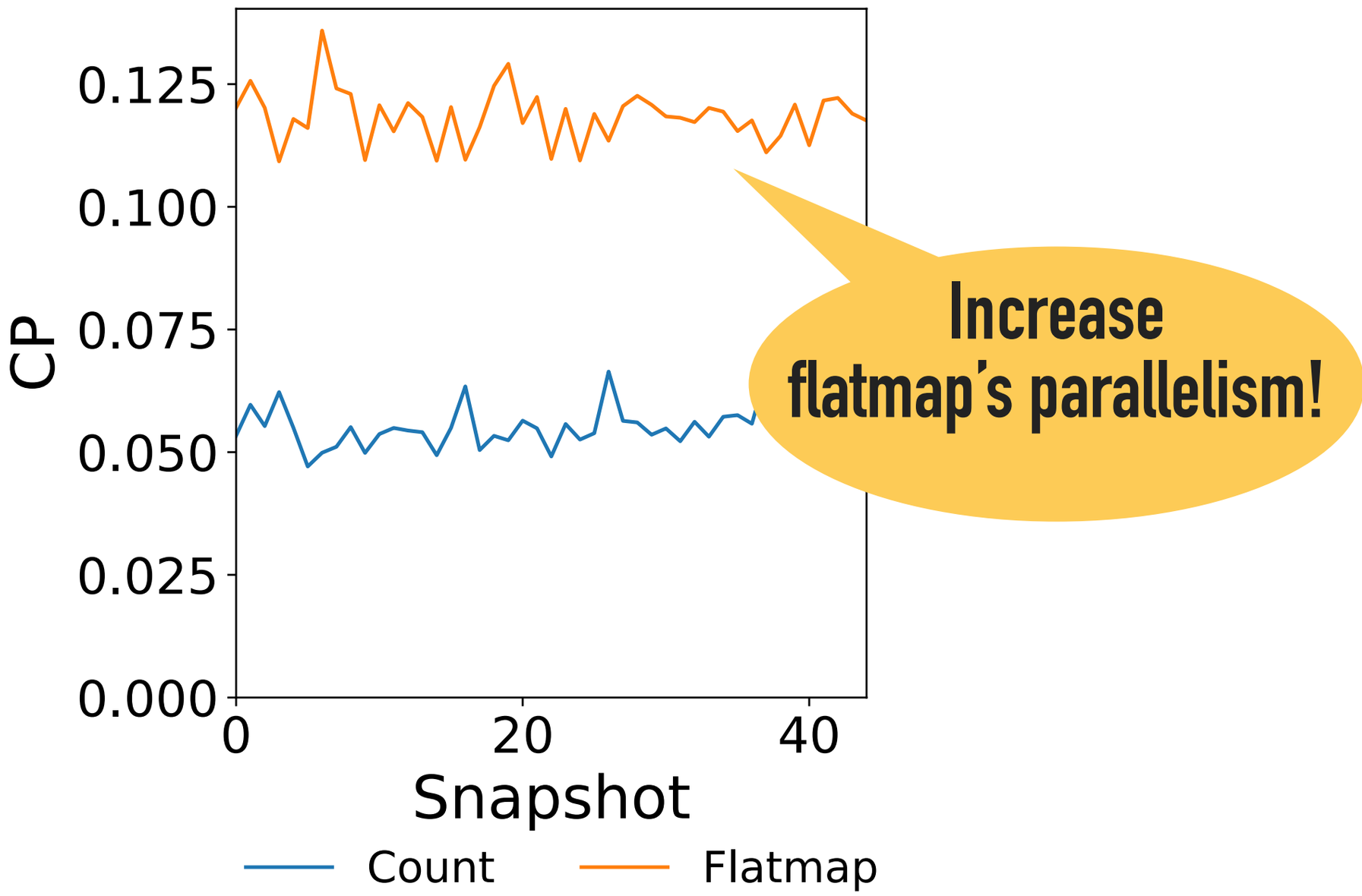


Apache Flink: Dhalion WordCount Benchmark, 4 workers, 1s snapshots

SNAILTRAIL CP-BASED SUMMARIES

- Activity Summary
  - *which **activity type** is a bottleneck?*
- Straggler Summary
  - *which **worker** is a bottleneck?*
- Operator Summary
  - *which **operator** is a bottleneck?*

Operator Summary

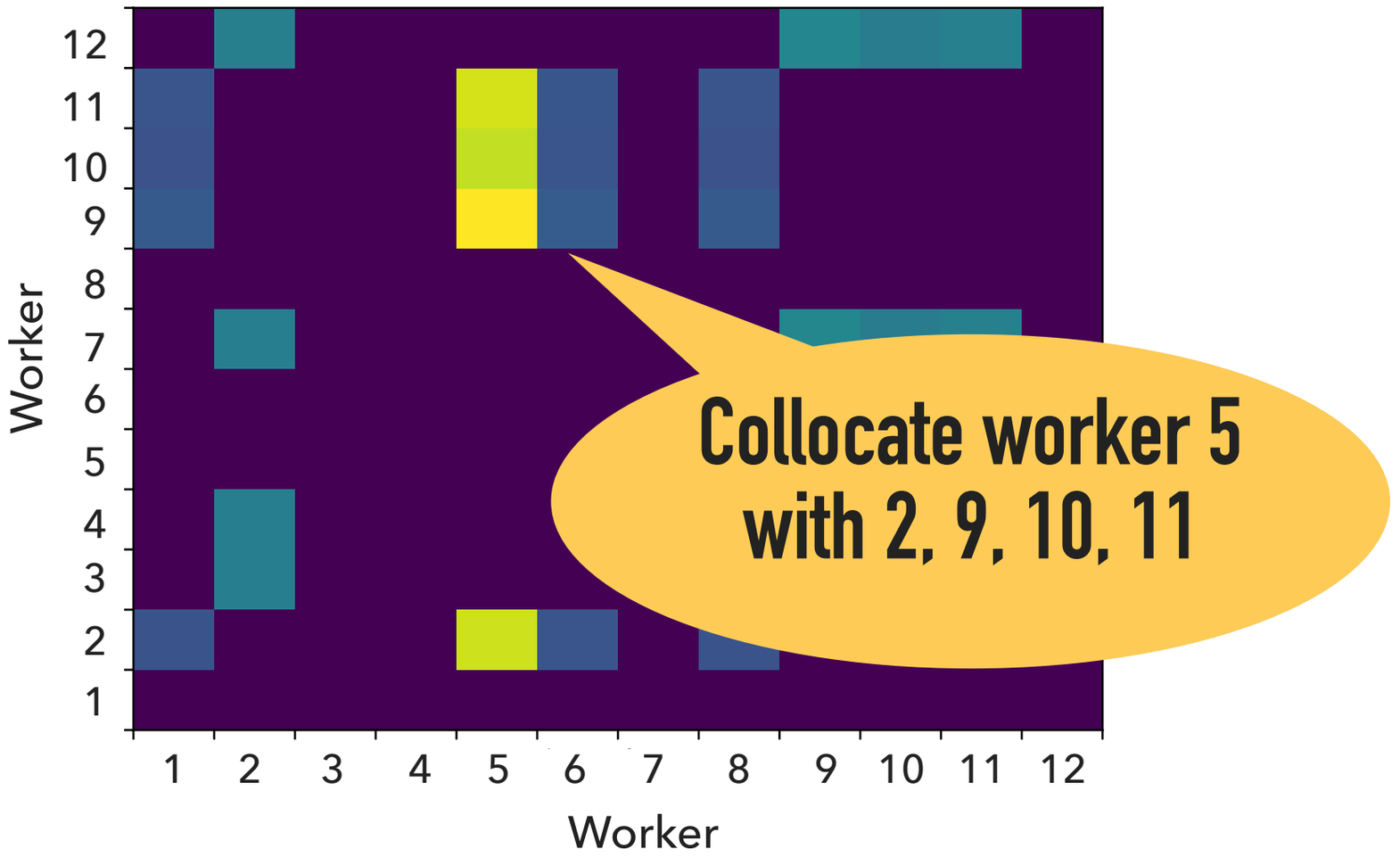


Apache Flink: Dhalion WordCount Benchmark, 10 workers, 1s snapshots

SNAILTRAIL CP-BASED SUMMARIES

- Activity Summary
  - *which **activity type** is a bottleneck?*
- Straggler Summary
  - *which **worker** is a bottleneck?*
- Operator Summary
  - *which **operator** is a bottleneck?*
- Communication Summary
  - *which communication **channels** are bottlenecks?*

Communication Summary



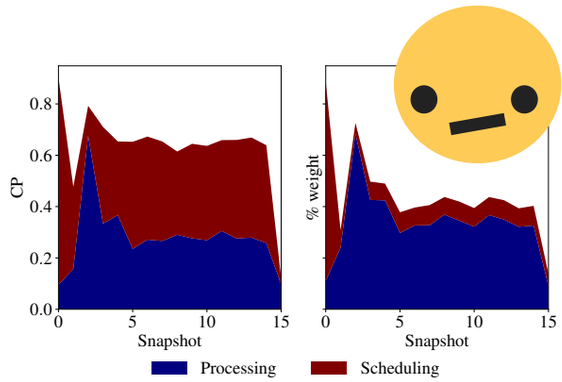
Communication Criticality

SNAILTRAIL PERFORMANCE

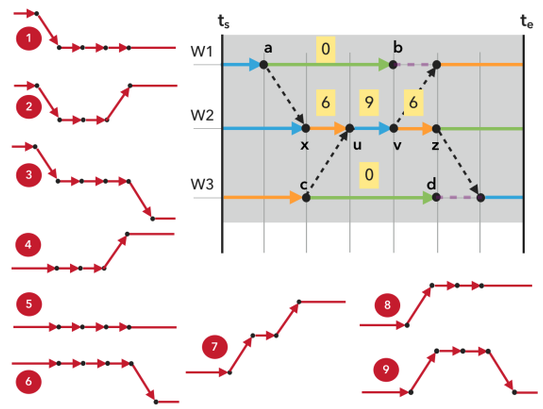
- Low instrumentation overhead
  - < 10% for all reference systems
- High throughput
  - 1.2 million events per second
- Always online
  - 1s of traces in 6ms, 256s of traces in < 25s

SnailTrail on Intel Xeon E5-4640, 2.40GHz, 32 cores, 512GB RAM
Traces from Apache Flink Sessionization, 48 workers, 1s-256s snapshots

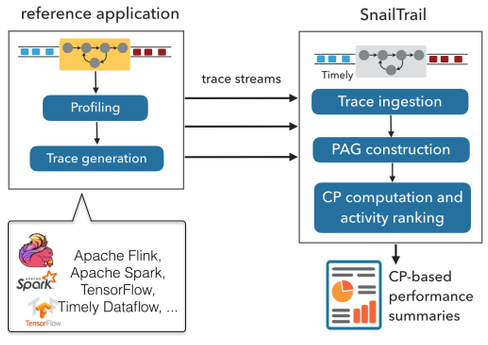
RECAP



Conventional profiling is misleading

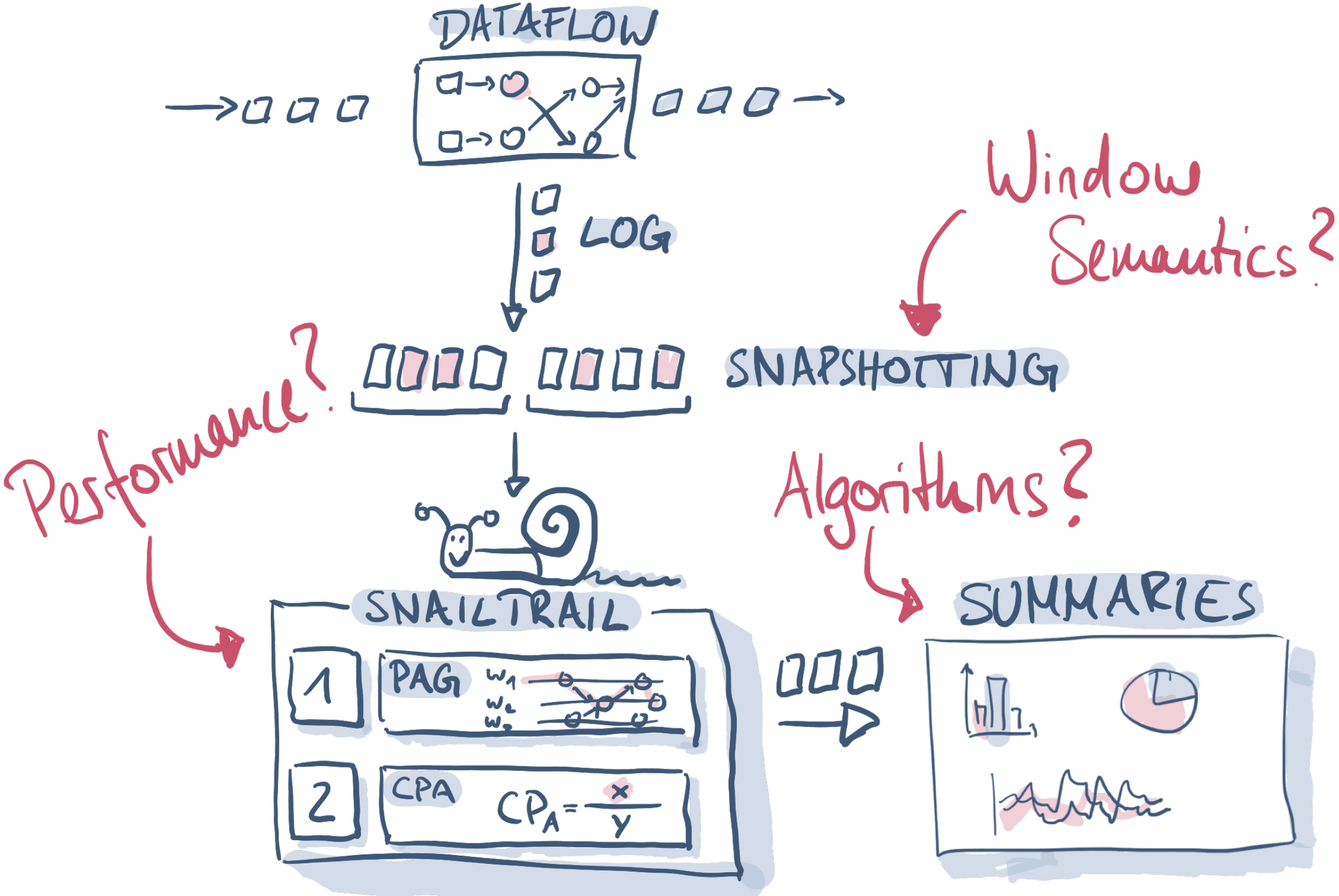


CP-metric: online critical path analysis



SnailTrail: online CP-based summaries

FUTURE WORK (ALSO, MY THESIS)



SnailTrail: Generalizing Critical Paths for Online Analysis of Distributed Dataflows*

strymon.systems.ethz.ch

Moritz Hoffmann, Andrea Lattuada, John Liagouris, Vasiliki Kalavri,
Desislava Dimitrova, Sebastian Wicki, Zaheer Chothia, Timothy Roscoe

Systems Group, Department of Computer Science, ETH Zürich

firstname.lastname@inf.ethz.ch

SnailTrail (stable): github.com/strymon-system/snailtrail
SnailTrail (very much WIP): github.com/li1/snailtrail
Strymon Ecosystem: github.com/strymon-system
Timely, Differential: github.com/timelydataflow

Malte Sandstede (samalt@ethz.ch)