Systems@ETH Zürich

SnailTrail

Generalizing Critical Paths for Online Analysis of Distributed Dataflows

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

Supported by

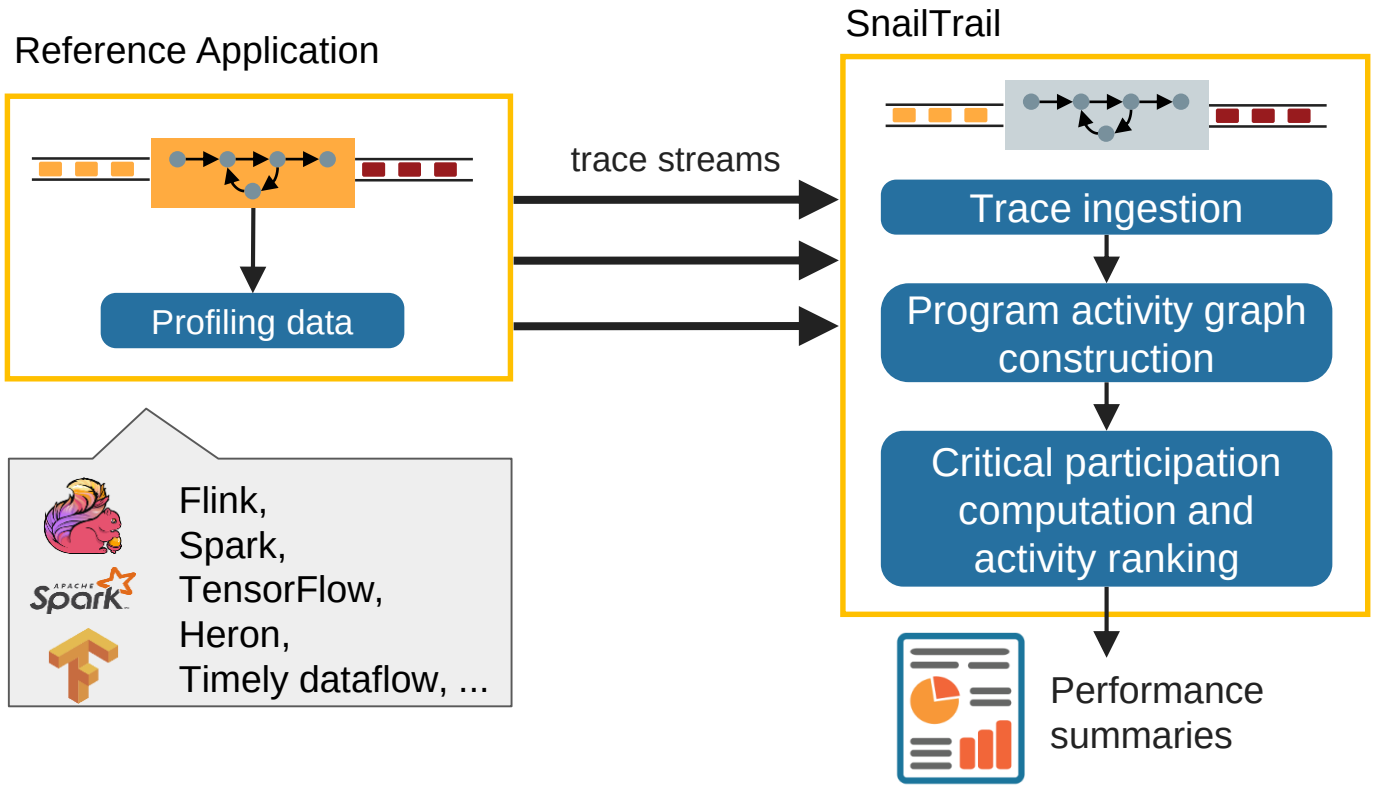
amadeus

FNSNF
Fonds national suisse
Schweizerischer Nationalfonds
Fondo nazionale svizzero
Swiss National Science Foundation

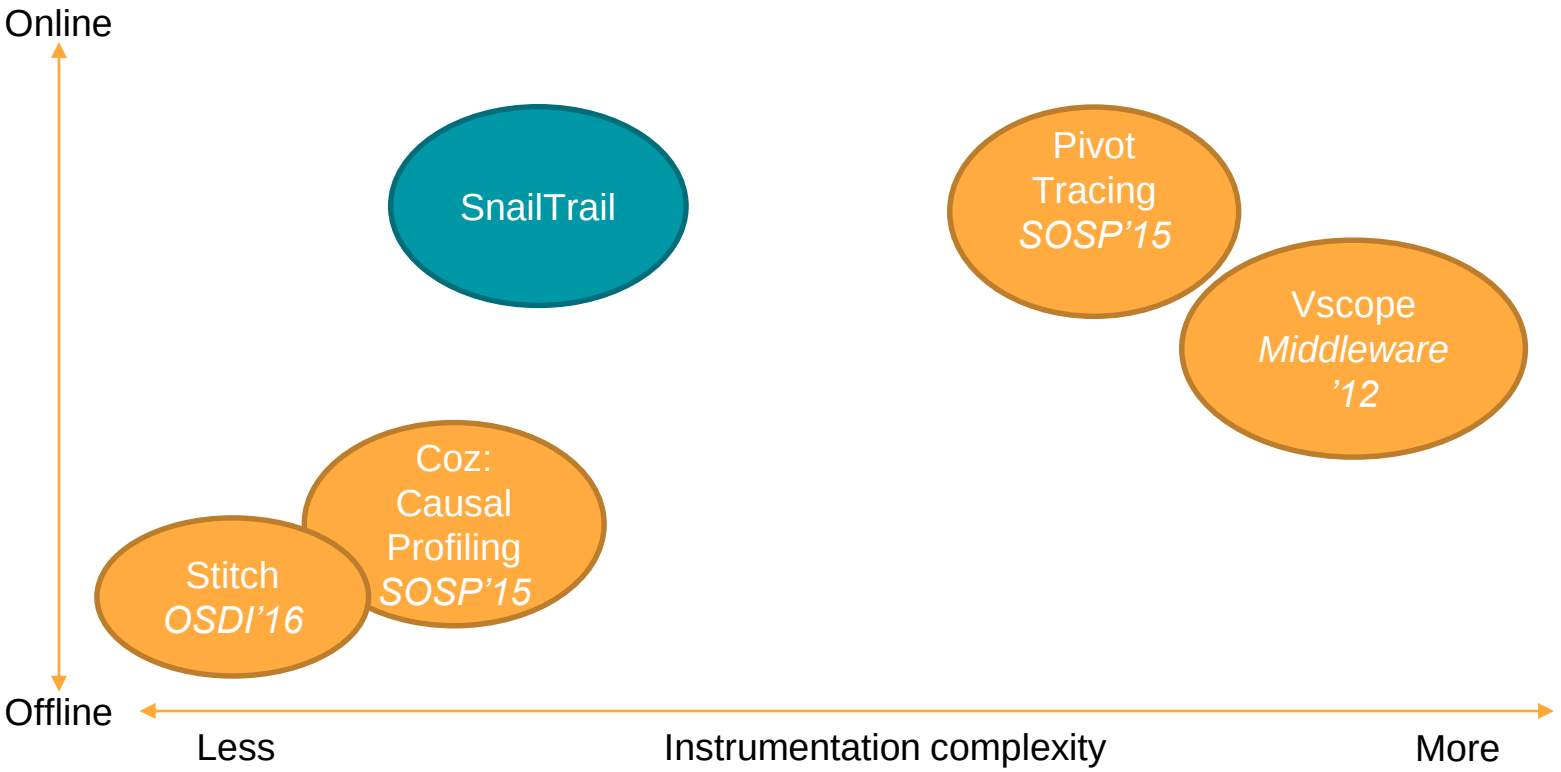
Google

SnailTrail: Diagnosing latency issues in dataflows

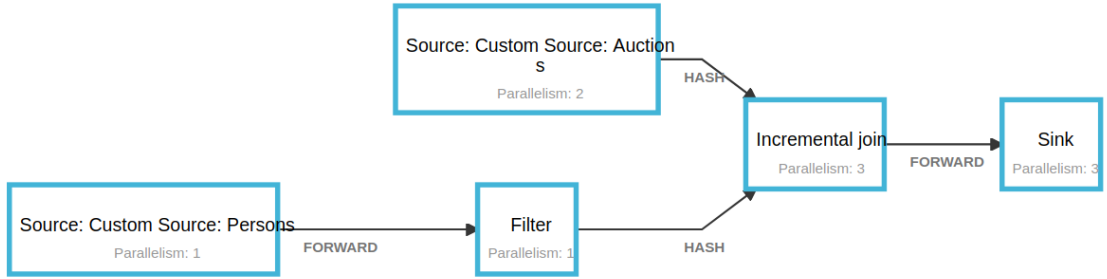
“Where is the latency bottleneck in my computation?”



SnailTrail works online with minimal instrumentation



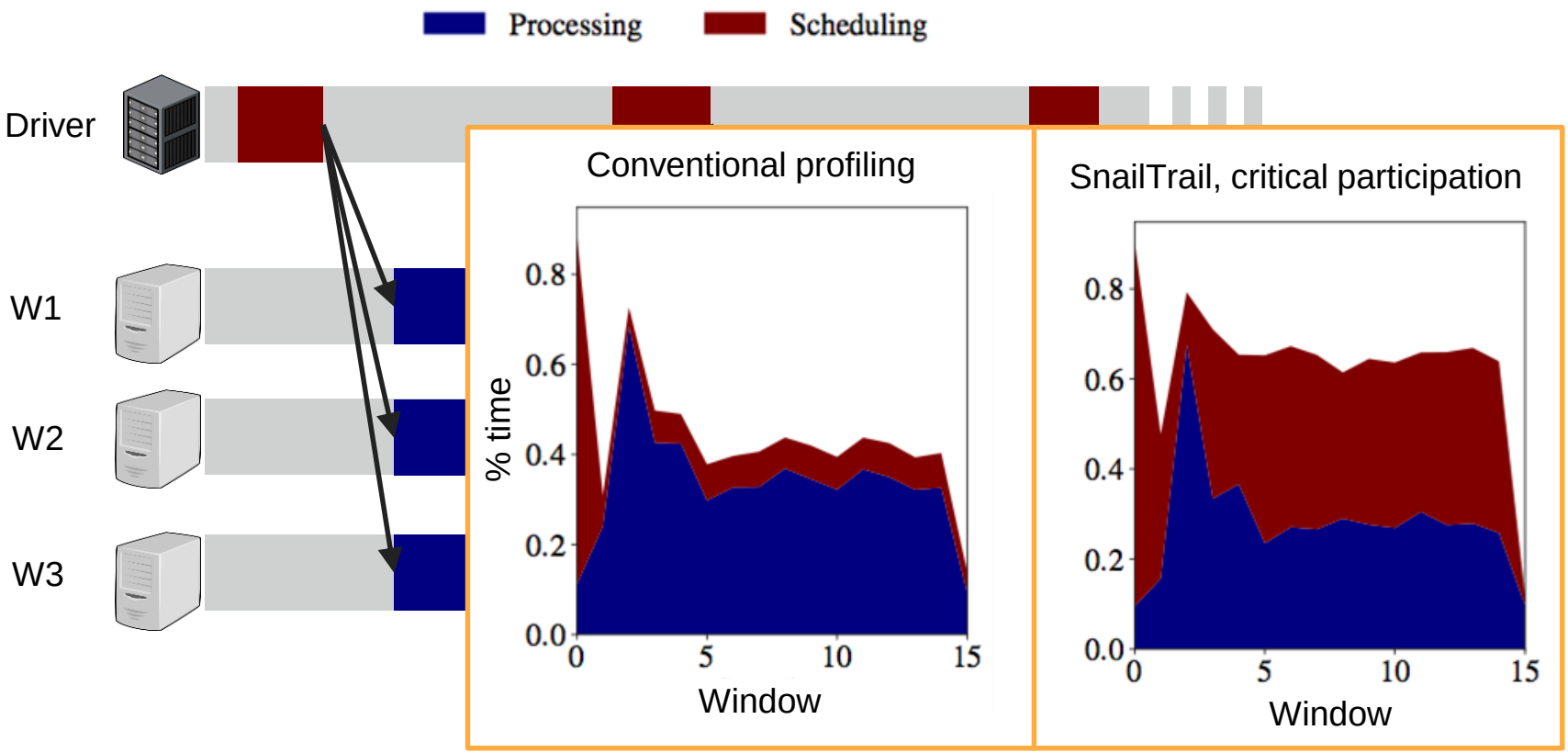
Example 1: Metrics in Flink's dashboard



Duration	Bytes received	Records received	Bytes sent	Records sent	Attempt
1m 28s	1.26 GB	2,871,683	3.20 MB	91,159	1
1m 28s	1.22 GB	2,764,835	3.52 MB	99,696	1
1m 28s	1.22 GB	2,777,058	3.15 MB	88,610	1

2018-04-04, 10:38:52		1m 28s	1.26 GB	2,871,683	3.20 MB	91,159	1	kalamari:40558	RUNNING
2018-04-04, 10:38:52		1m 28s	1.22 GB	2,764,835	3.52 MB	99,696	1	kalamari:40558	RUNNING
2018-04-04, 10:38:52		1m 28s	1.22 GB	2,777,058	3.15 MB	88,610	1	kalamari:40558	RUNNING
2018-04-04, 10:38:52	2018-04-04, 10:40:21	1m 28s	Sink		9.86 MB	279,465	0 B	0	3	0 0 3 5 0 0 0	RUNNING

Example 2: Task Scheduling in Spark

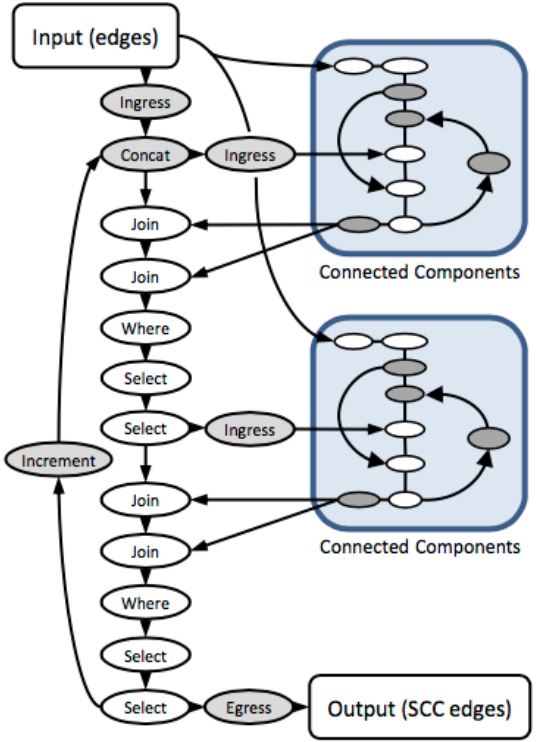


The real-world is more complex

Many tasks, activities, operators, dependencies

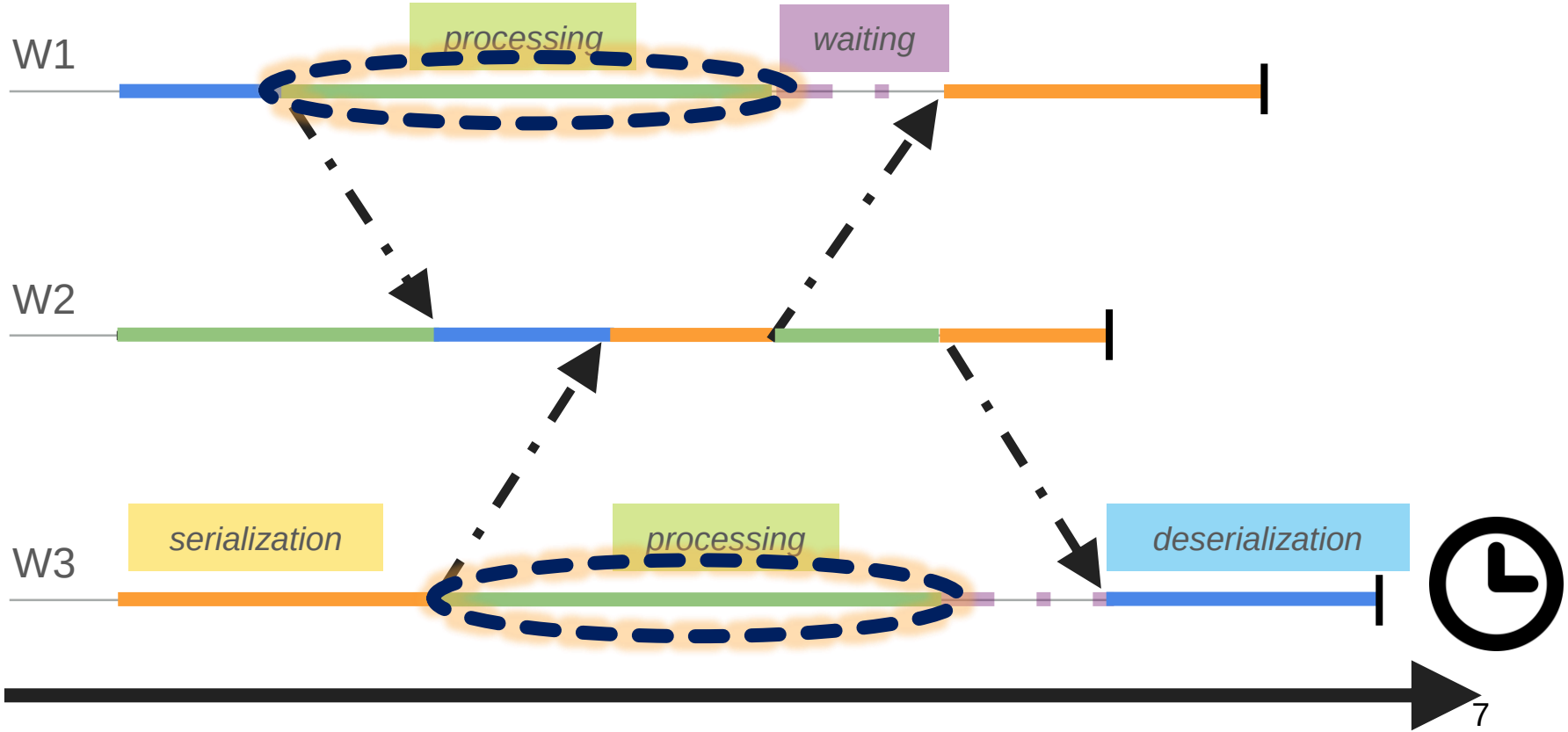
Long-running, dynamic workloads

Bottlenecks not isolated

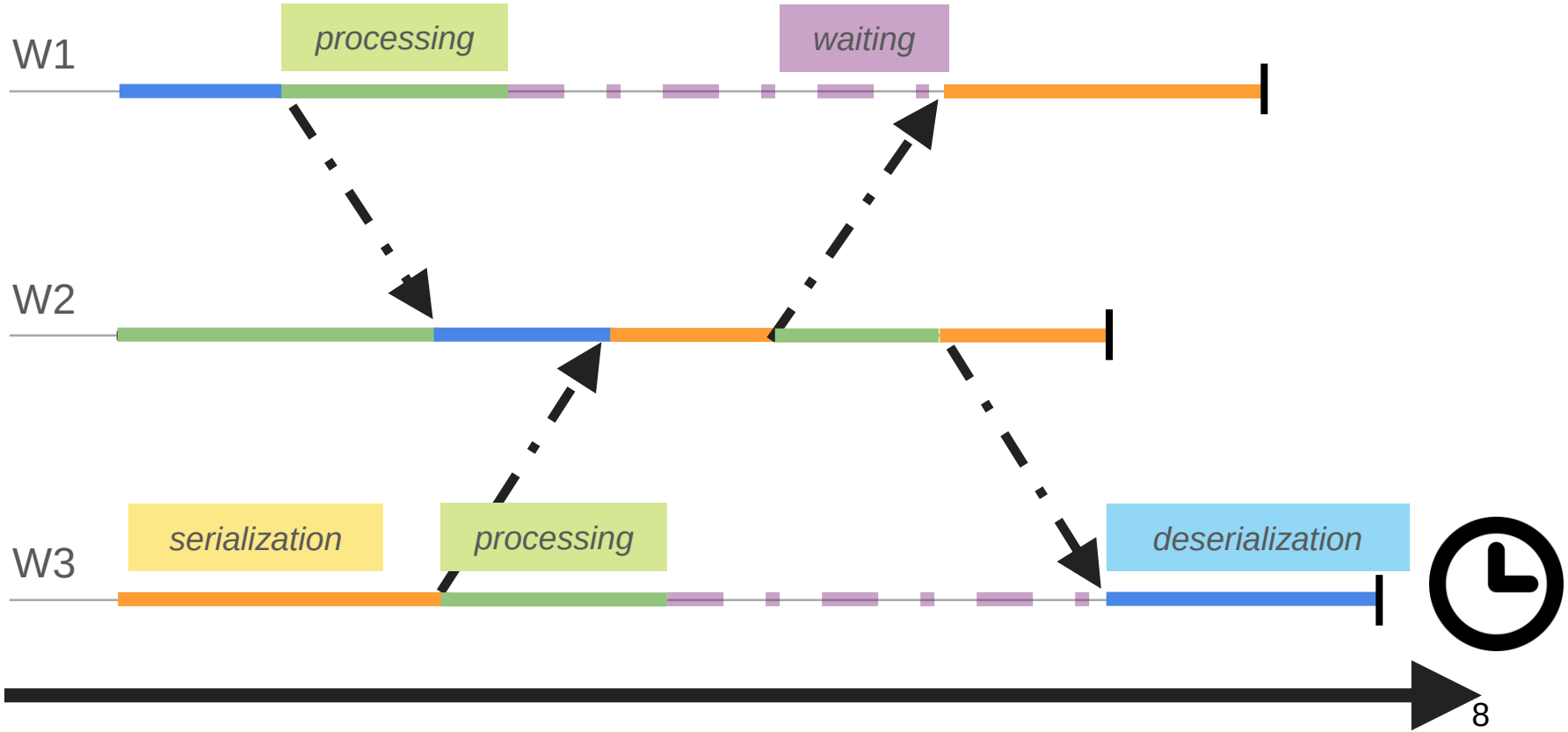


Credits: Frank McSherry, “Tracking progress in timely dataflow”

Conventional profiling can indicate wrong bottleneck

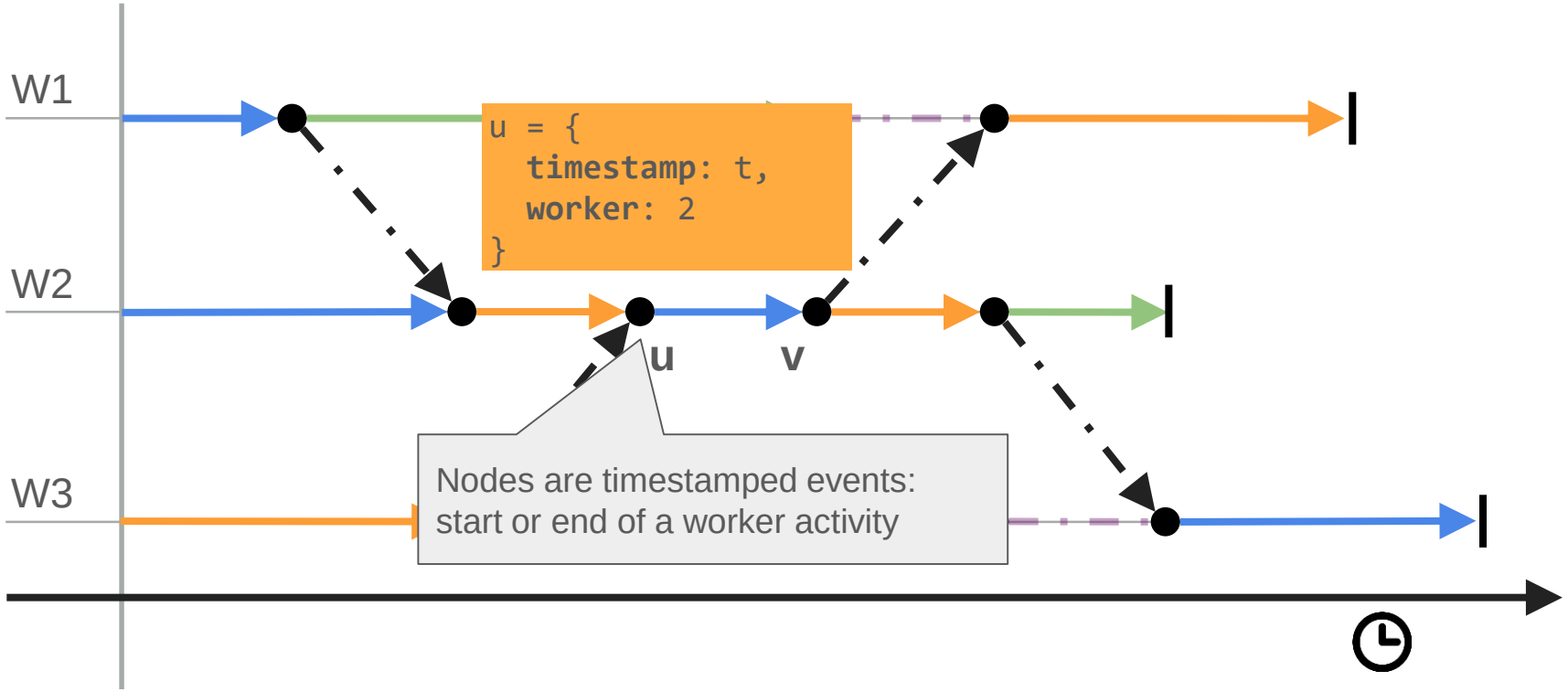


Conventional profiling can indicate wrong bottleneck

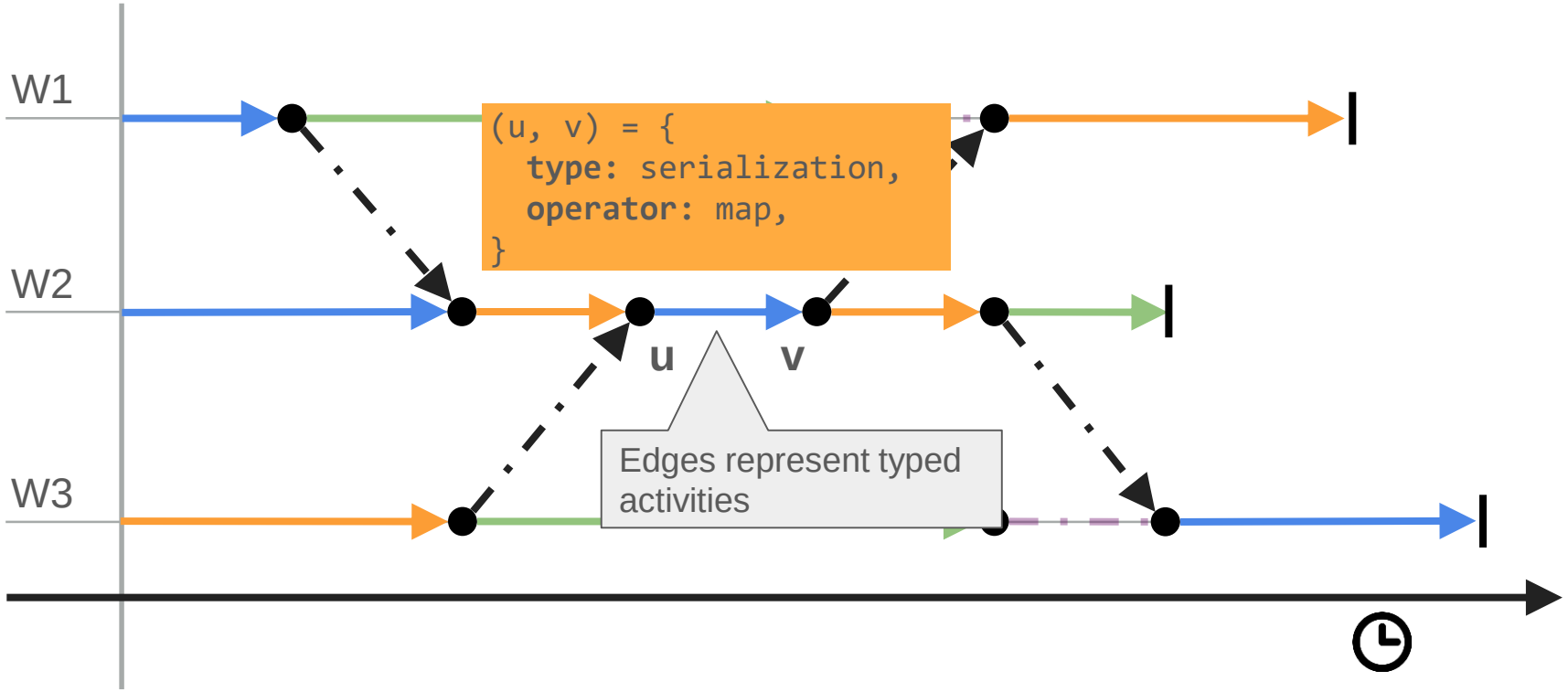


A quick review of critical path analysis

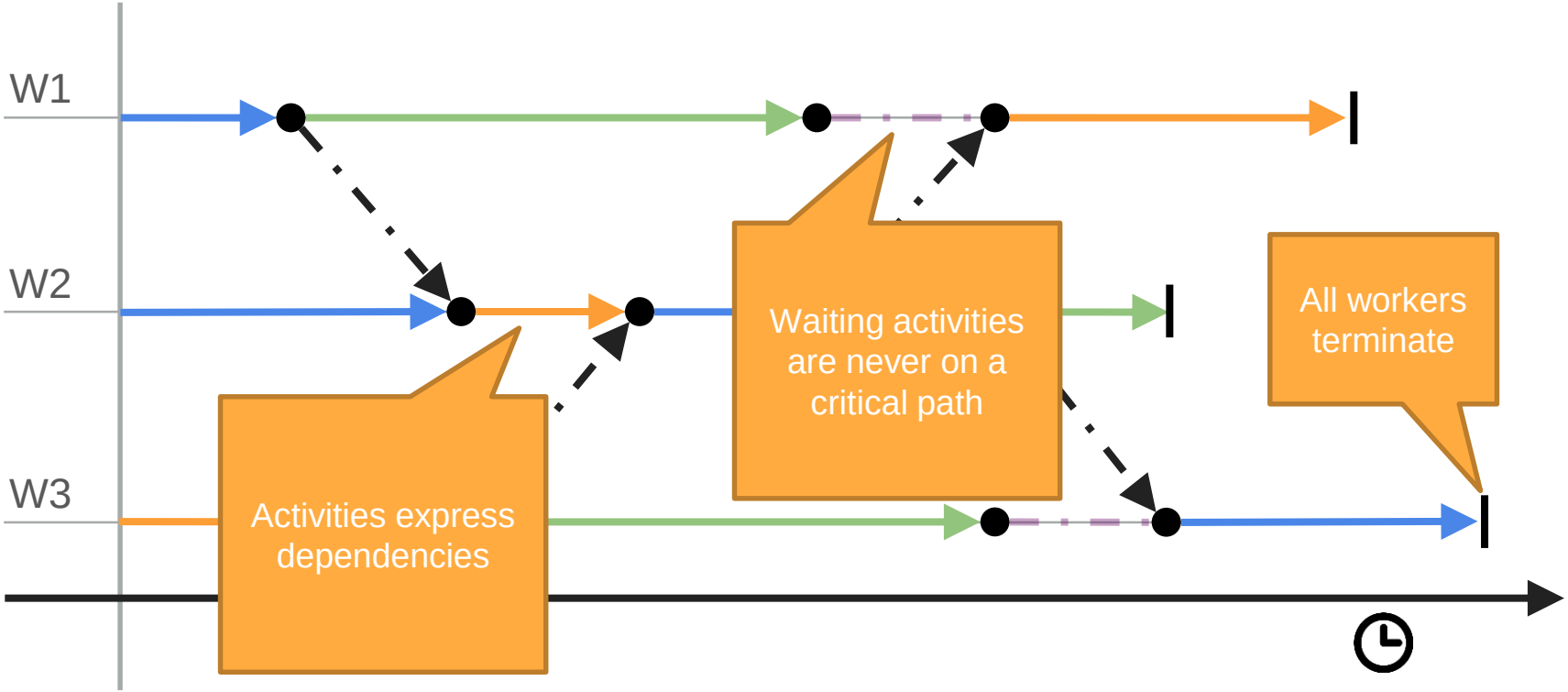
The program activity graph



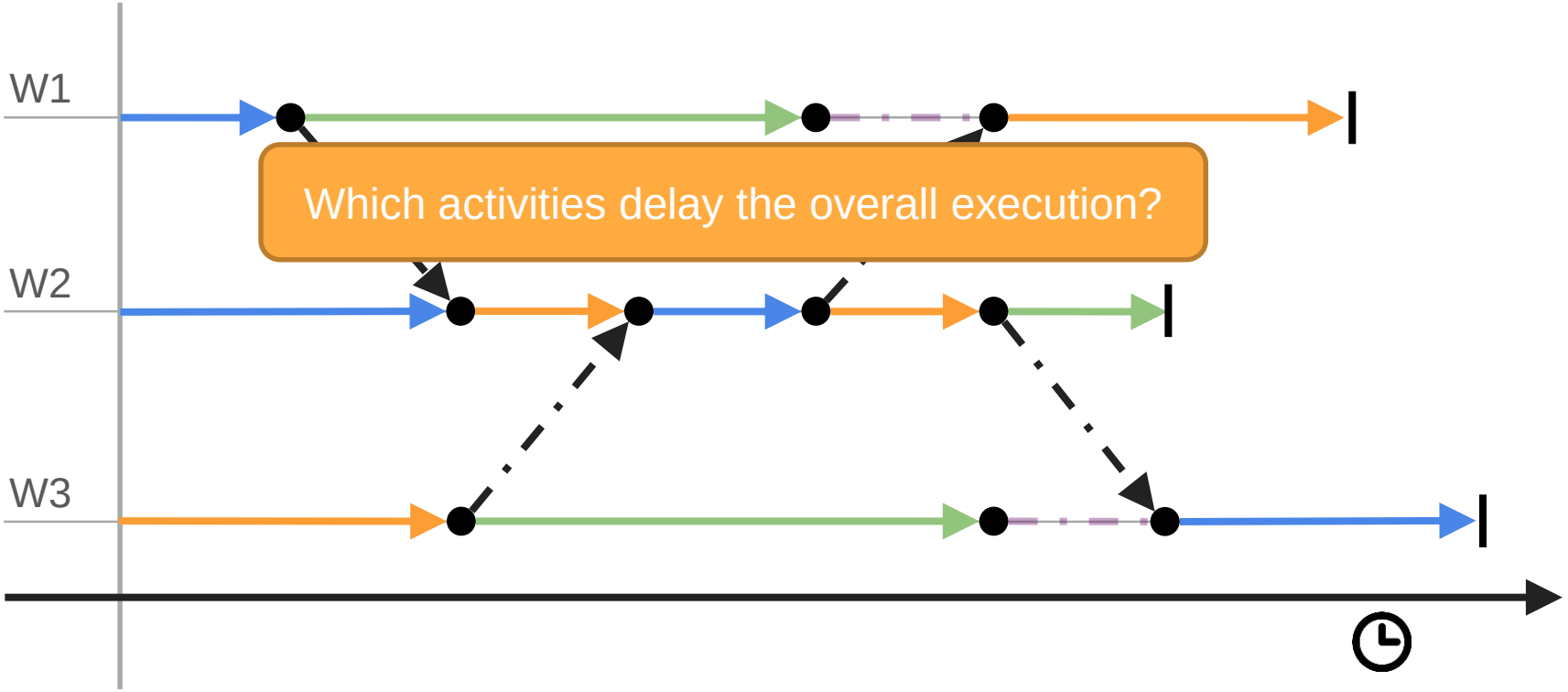
The program activity graph



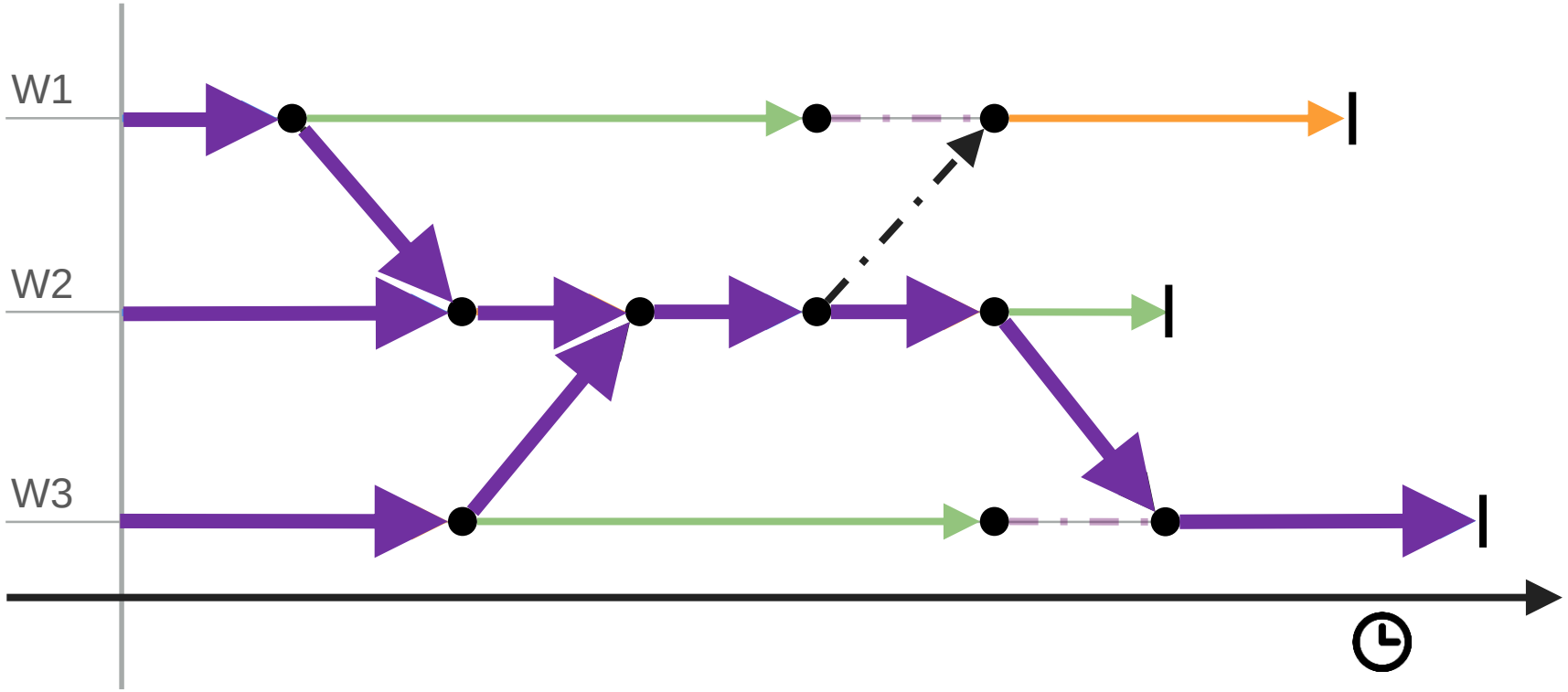
The program activity graph



The program activity graph



Classical critical path analysis



What is the equivalent of a critical path for **continuously** running, distributed **streaming** applications, with potentially **unbounded** input?

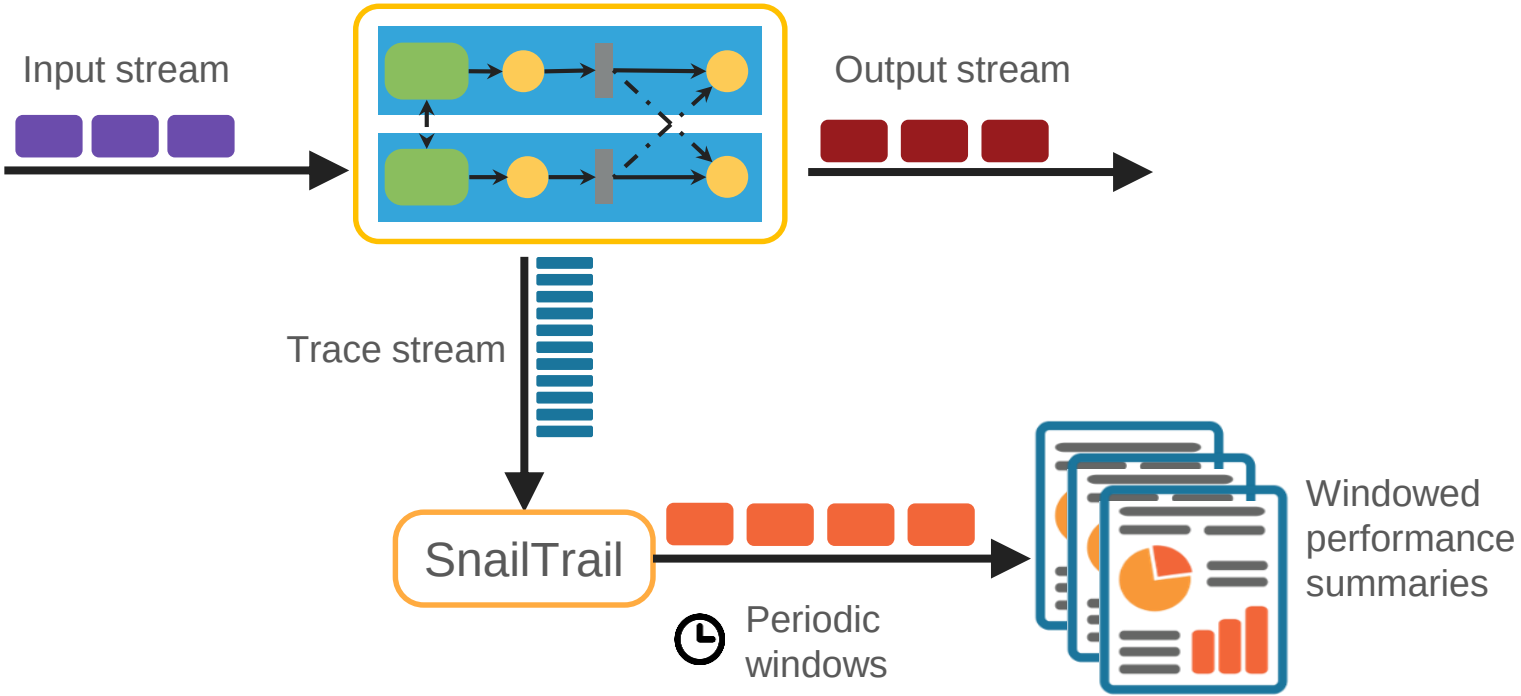
There might be no “job end”

The program activity graph and critical paths change continuously

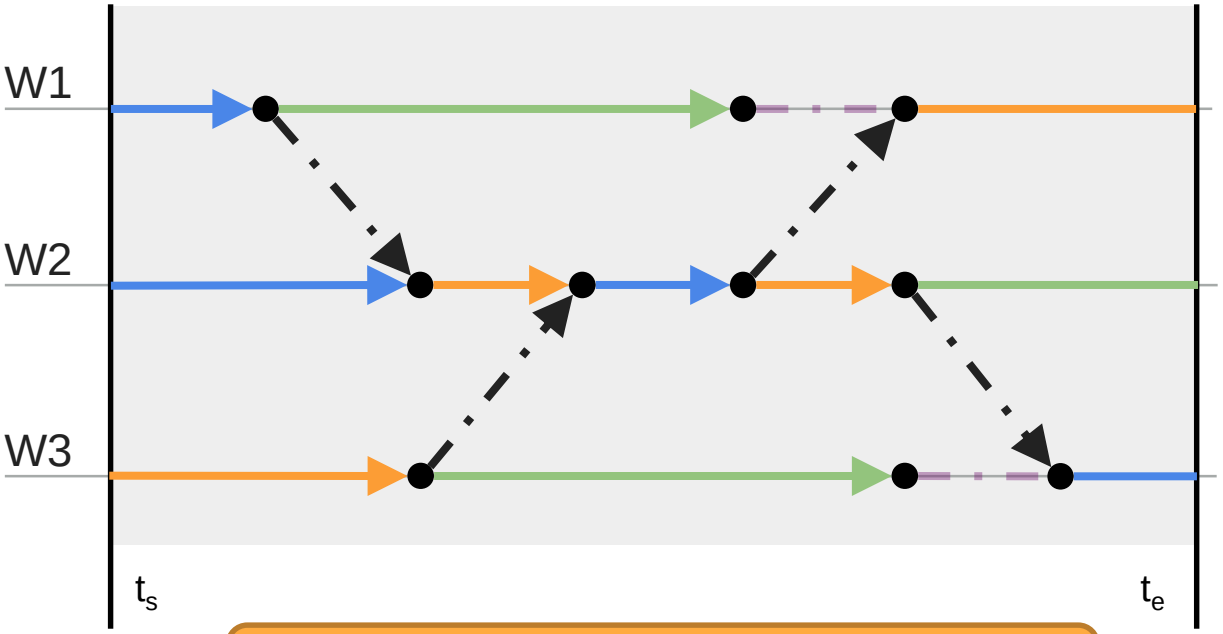
Profiling information can quickly become stale

Online critical path analysis

SnailTrail: Online analysis of trace **windows**

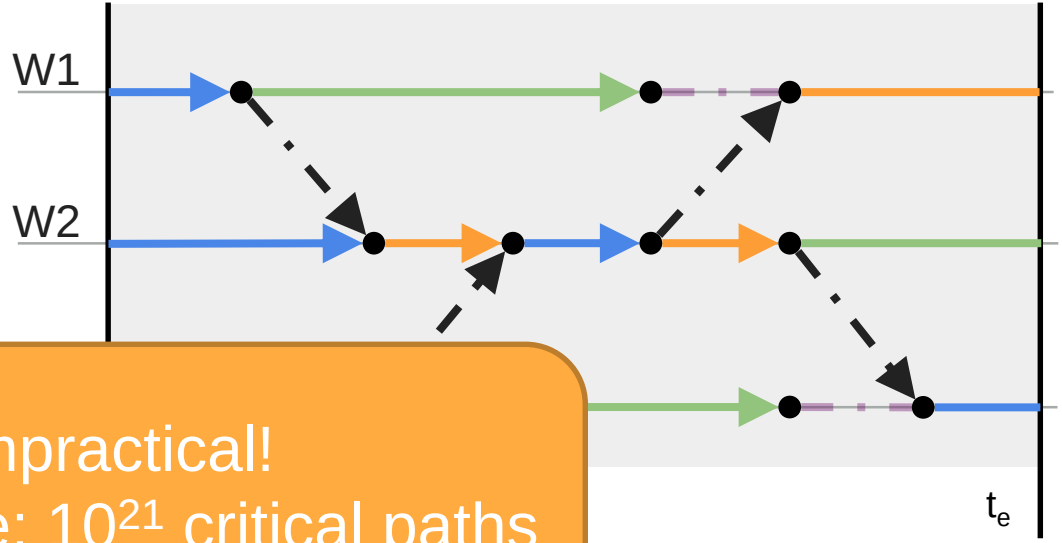


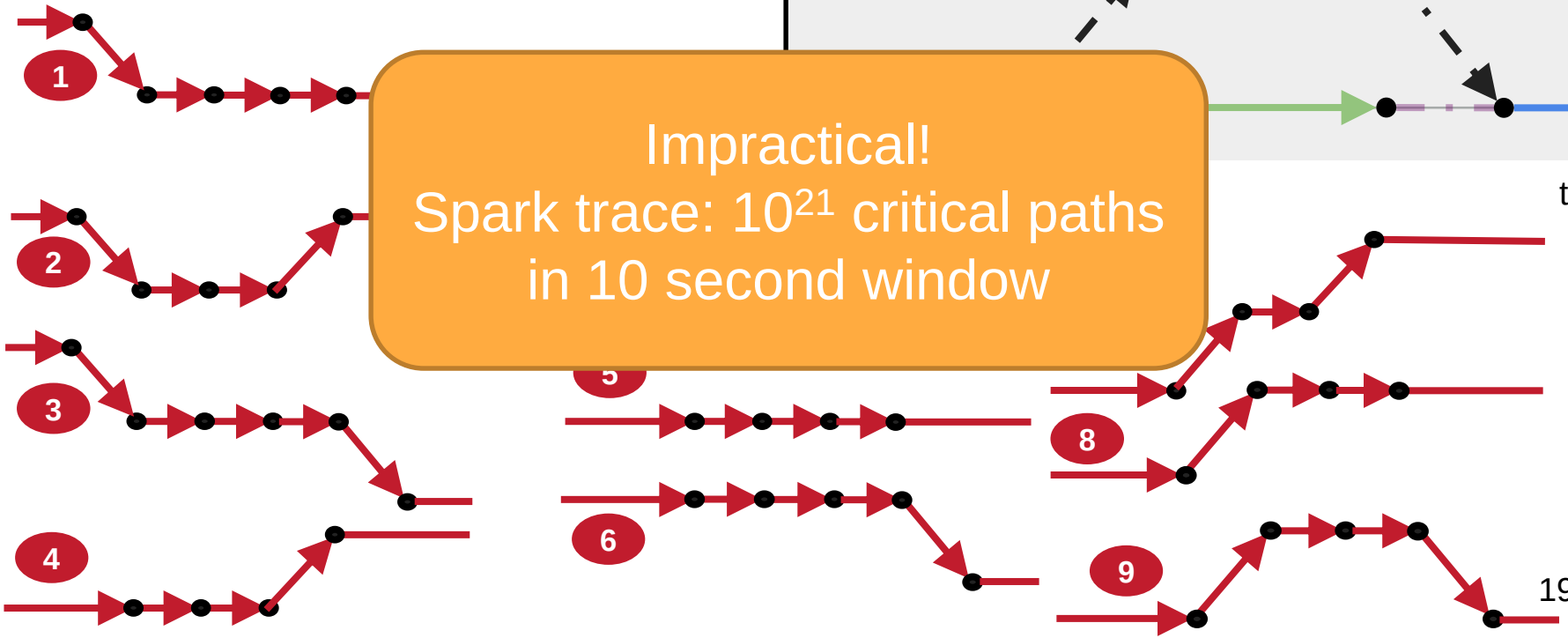
Program activity graph window



All critical paths have equal length

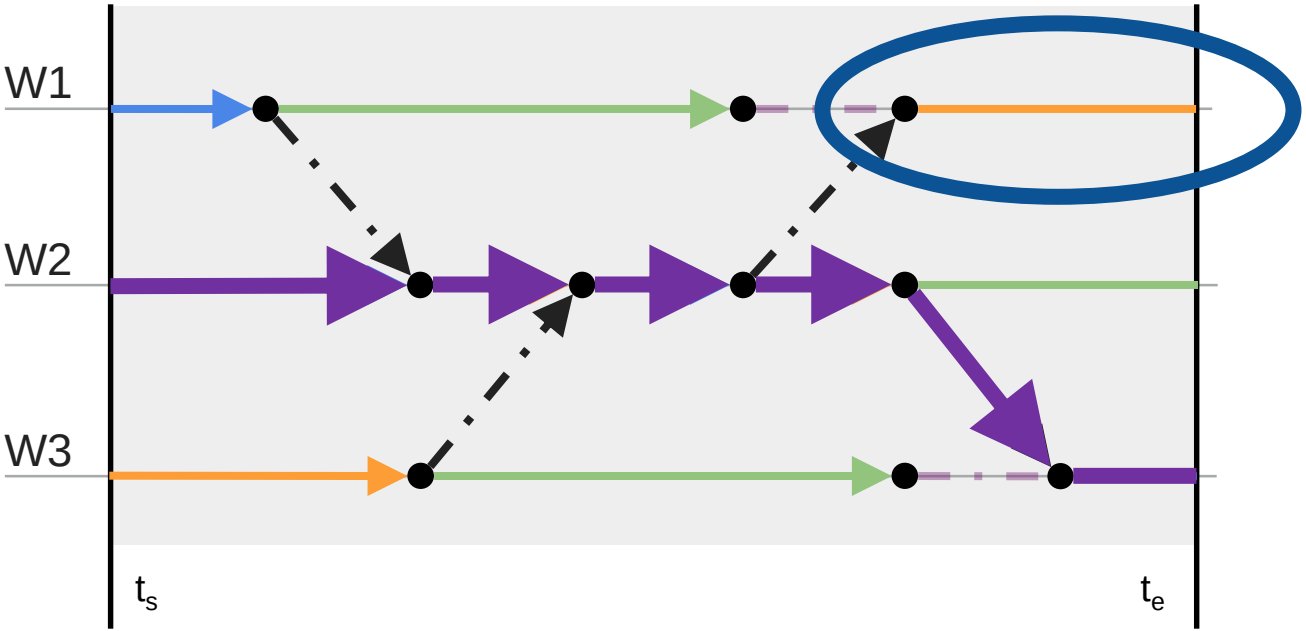
Cannot enumerate all critical paths





Impractical!
Spark trace: 10^{21} critical paths
in 10 second window

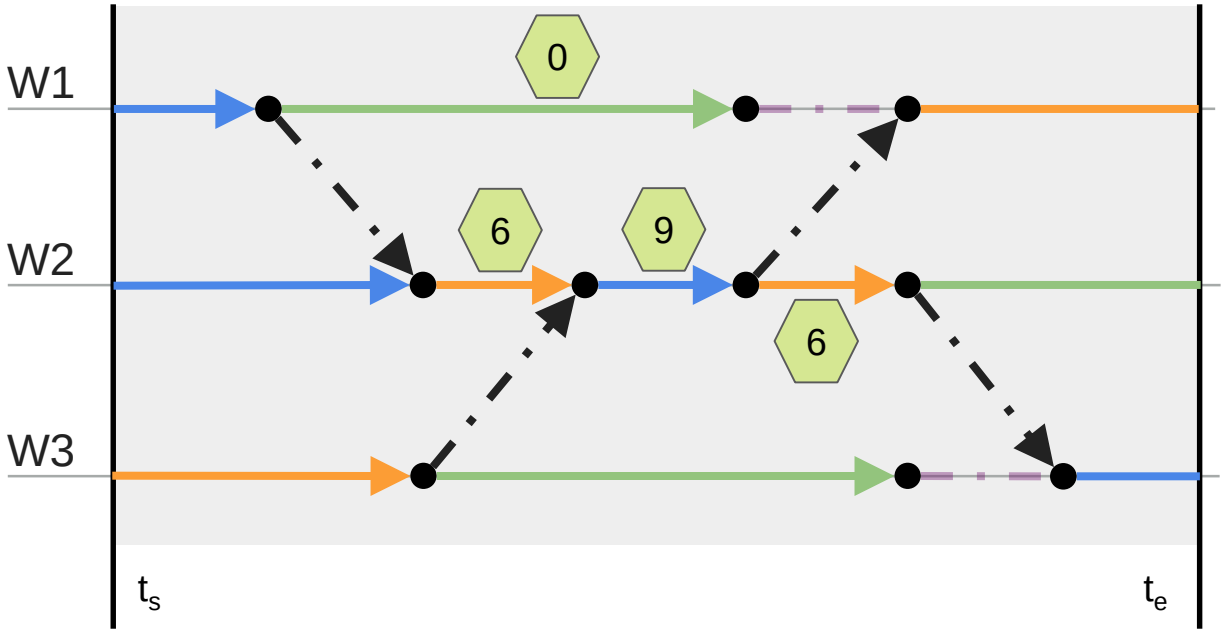
Sampling critical paths misses critical activities



We **rank activities** across all critical paths to capture their relative importance.

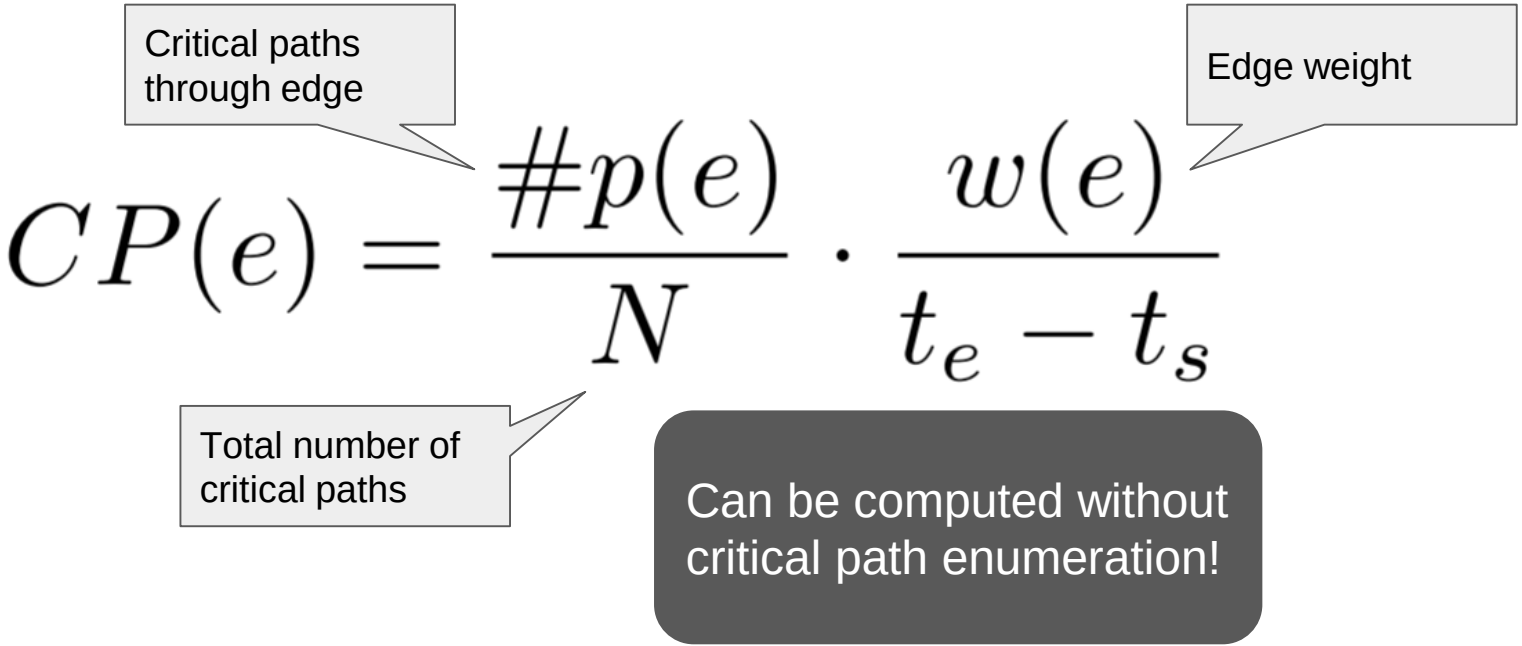
Intuition: The more critical paths go through an activity, the more critical it might be

Counting over enumerating

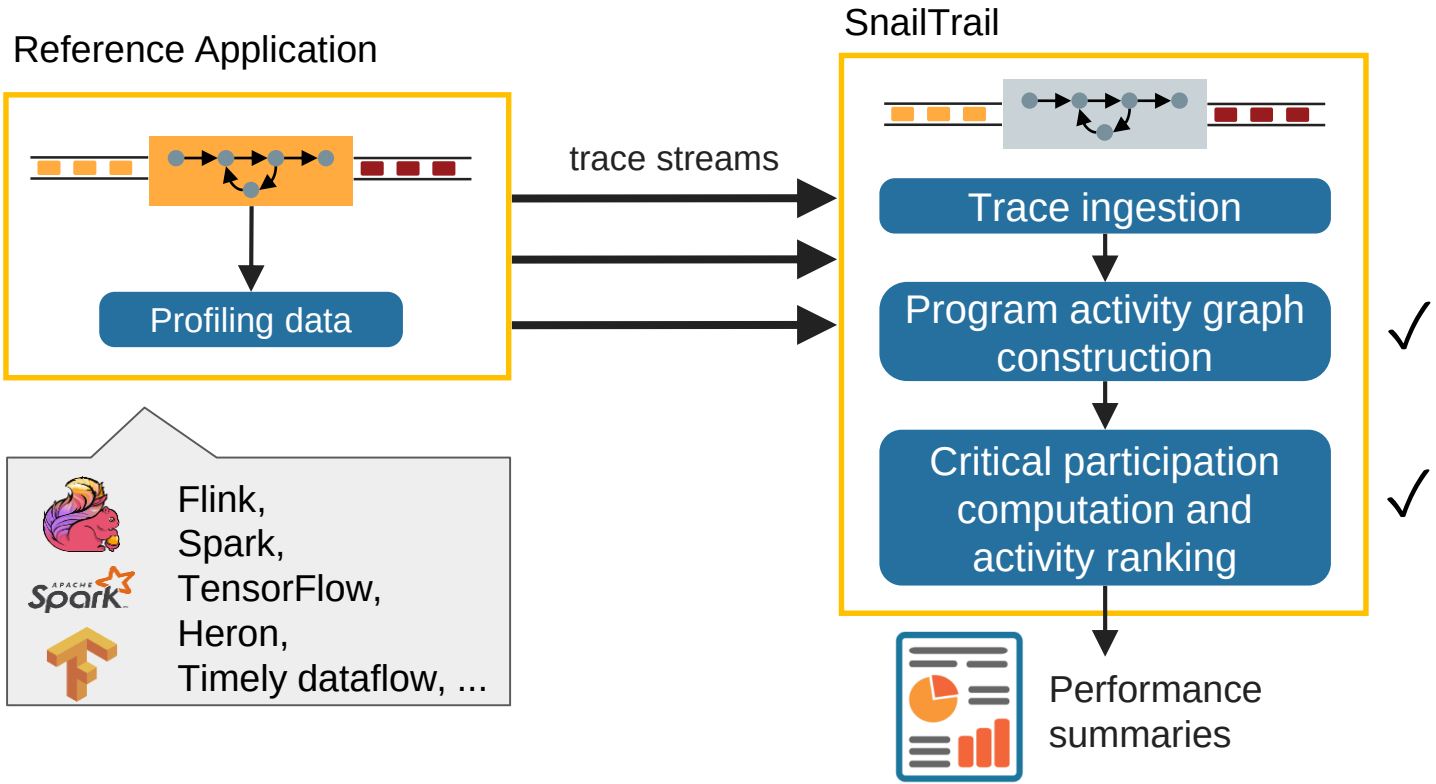


The Critical Participation metric

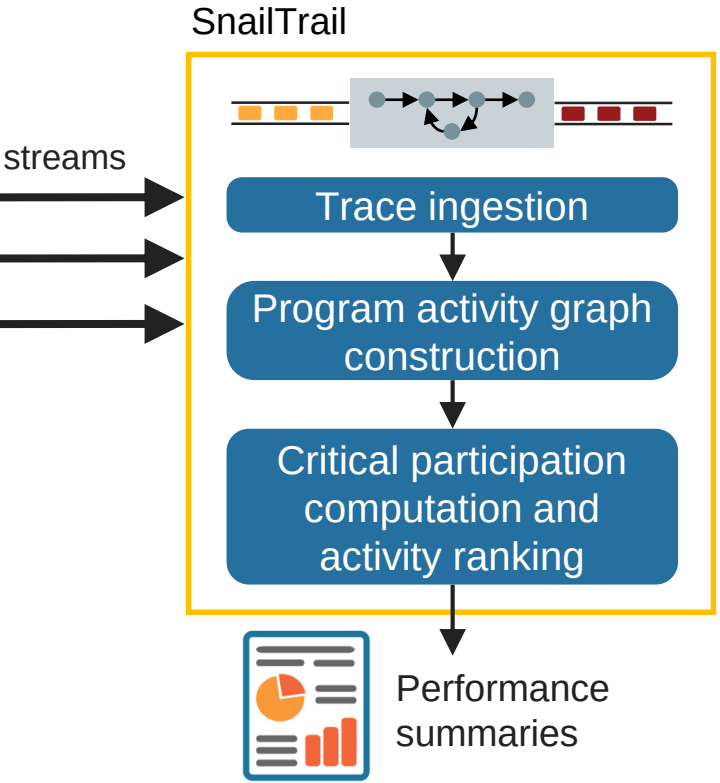
Fraction of an edge's time contribution across all critical paths



SnailTrail in action



Interpreting critical participation-based summaries



Stream of tuples:
(Activity type, Operator, Worker, …, Critical participation)

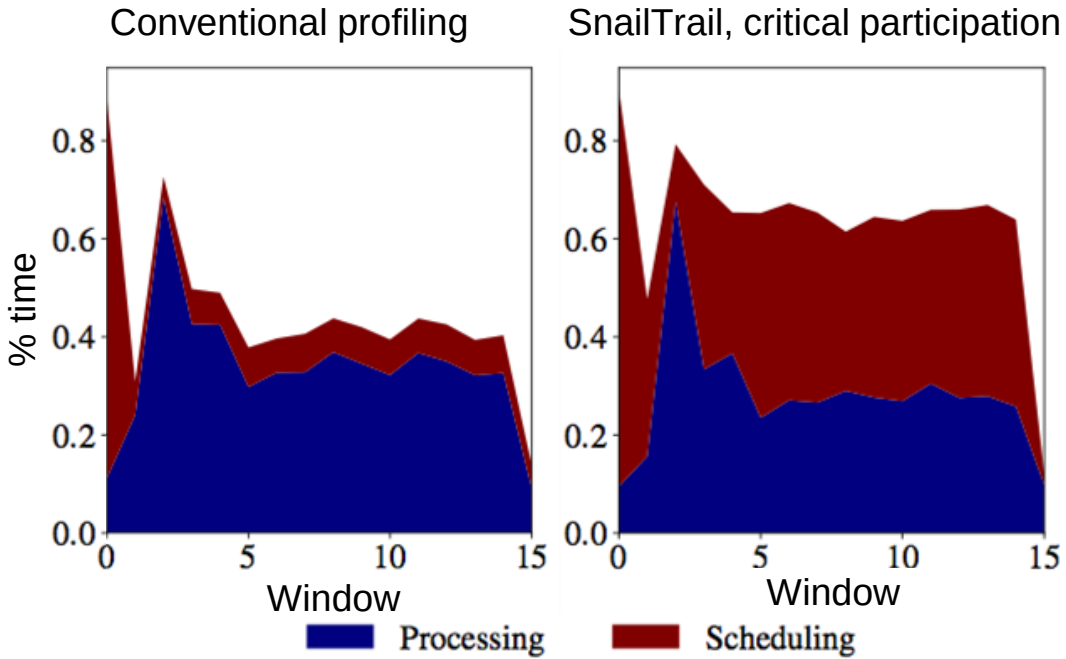
Examples:

Activity type bottleneck analysis
Operator bottleneck analysis

(More in the paper!)

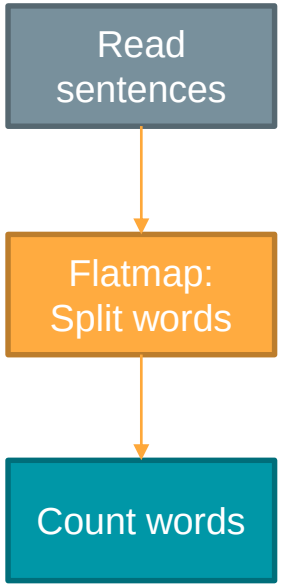
Activity type bottleneck analysis (Spark)

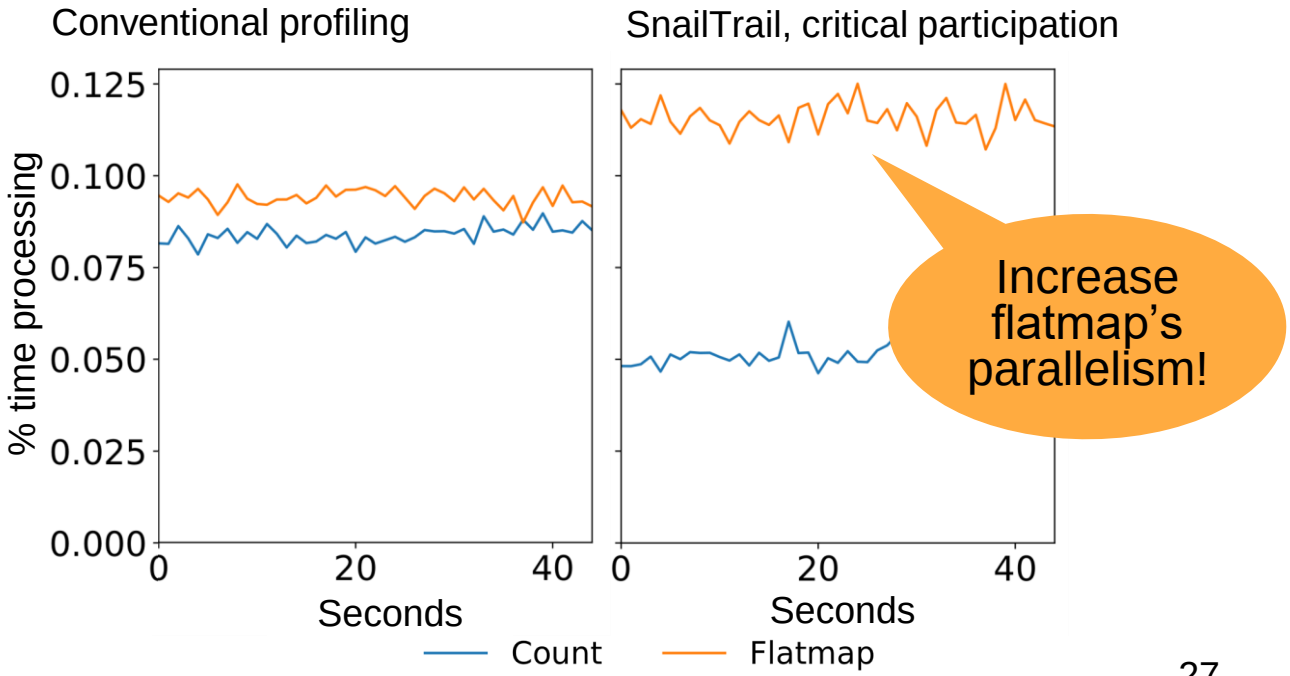
Apache Spark: Yahoo! Streaming Benchmark, 16 workers, 8s windows



Operator bottleneck analysis (Flink)

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





SnailTrail performance

Low instrumentation overhead

Spark, TensorFlow

No observed overhead

Flink, Timely

~10% overhead compared to logging disabled

High throughput

1.2 million events/s

8 workers

Always online

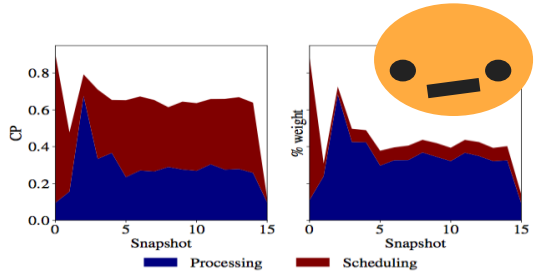
1s of traces in 6ms (100x)

256s of traces in < 25s (10x)

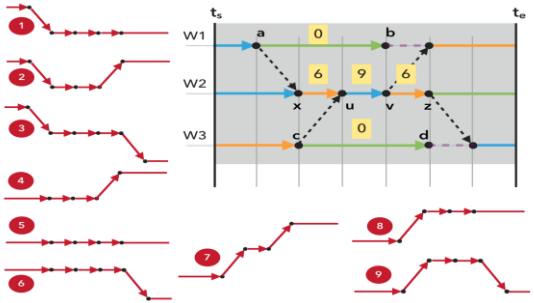
SnailTrail on Intel Xeon E5-4640, 2.40GHz, 32 cores, 512GiB RAM

Trace: Apache Flink Sessionization, 48 workers, 1s-256s windows

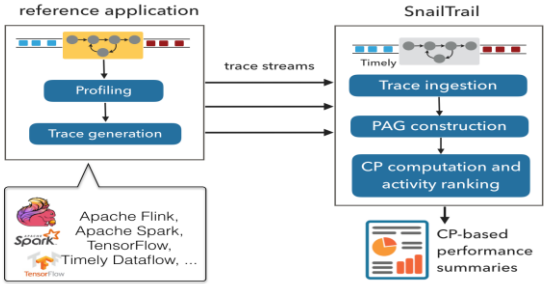
Summary



Conventional profiling is misleading



CP-metric: online critical path analysis



SnailTrail: online CP-based summaries

strymon.systems.ethz.ch/
github.com/strymon-system/snailtrail