Henge: Intent-Driven Multi-Tenant Stream Processing

Faria Kalim, Le Xu, Sharanya Bathey, Richa Meherwal, Indranil Gupta

Distributed Protocols Research Group

Department of Computer Science
University of Illinois at Urbana
Champaign

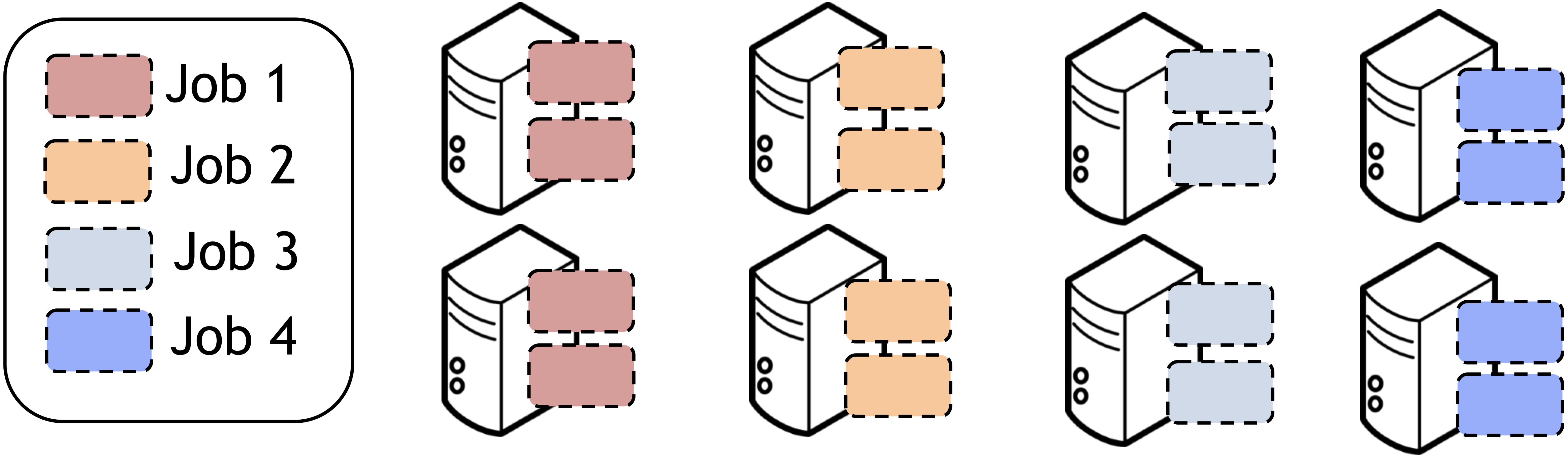
http://dprg.cs.uiuc.edu/

Henge allows stream processing jobs to satisfy **user-specified** performance requirements

while **reducing** costs

by performing online **resource reconfigurations** in a multi-tenant environment.

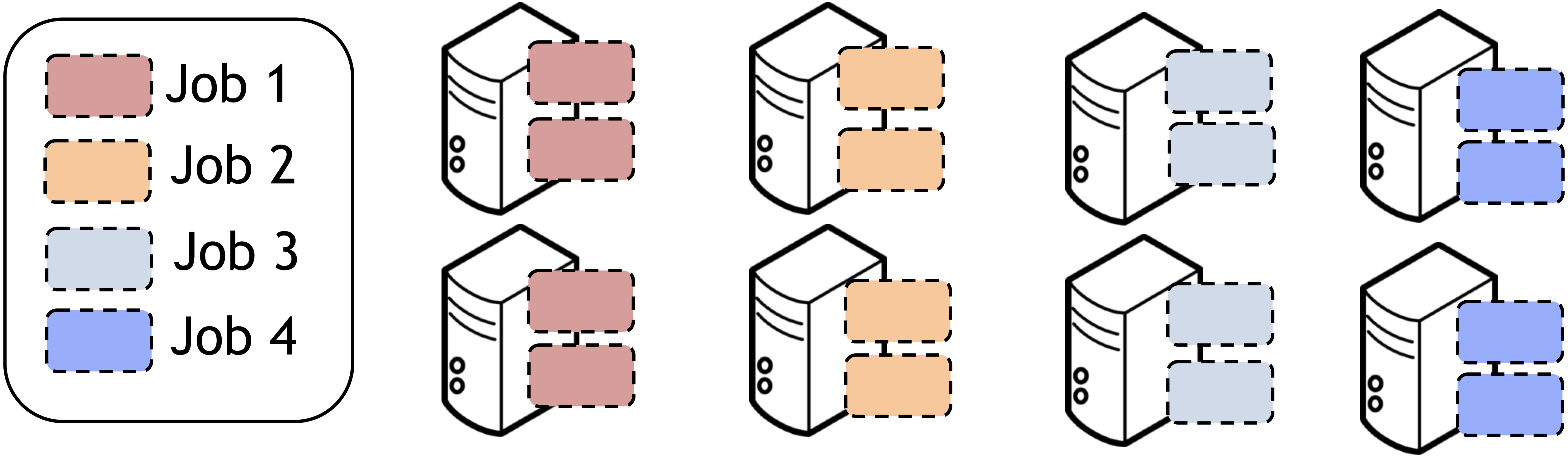
A Typical Deployment



Per-job clusters → overprovisioning

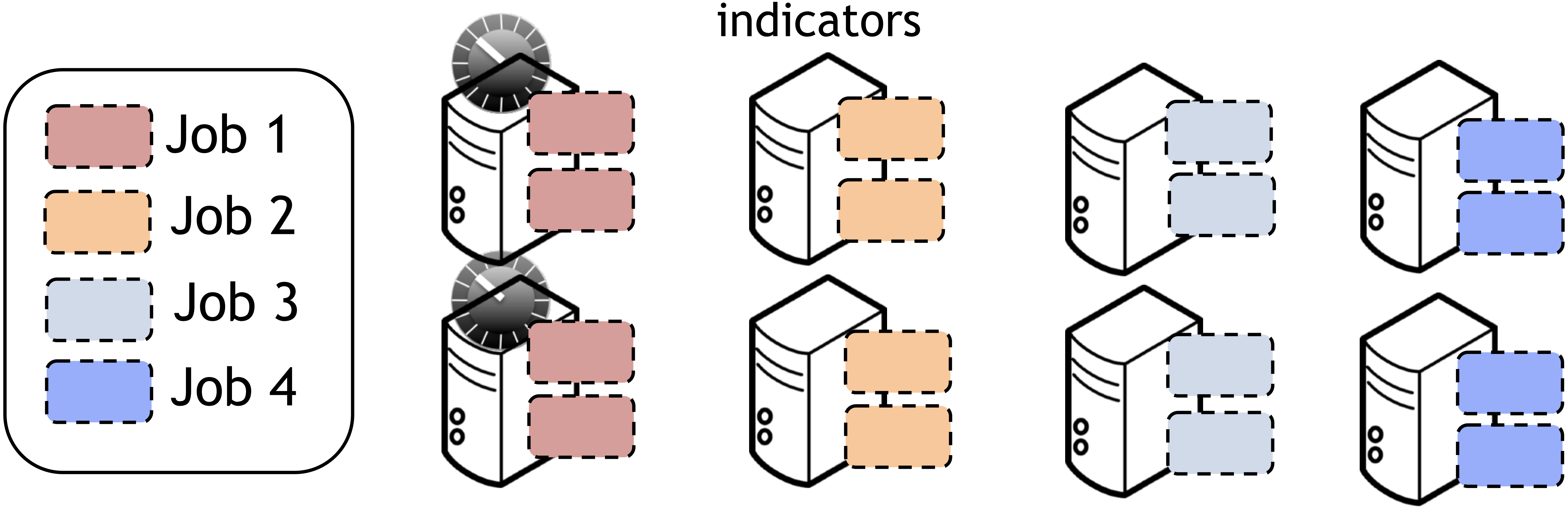
A Typical Deployment

Low level metrics e.g., queue sizes, CPU load as performance indicators



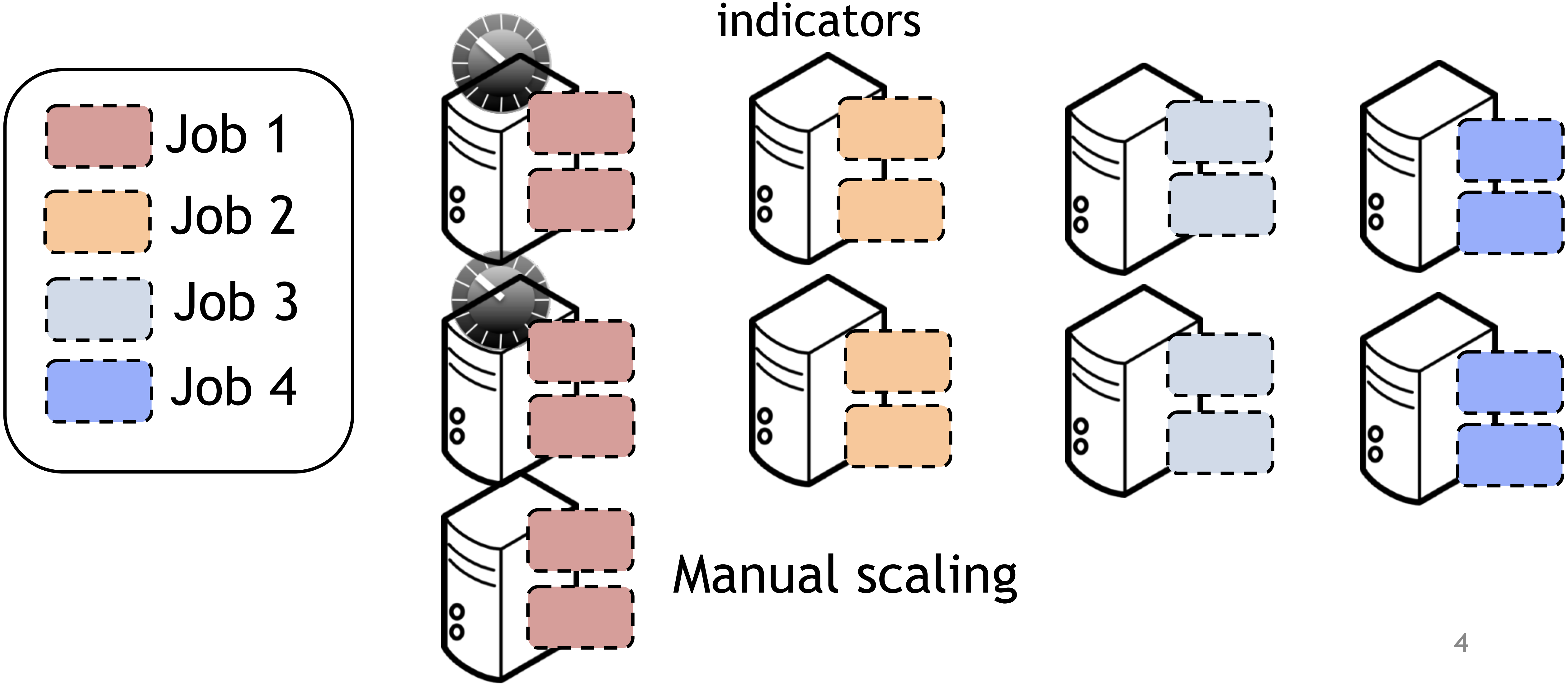
A Typical Deployment

Low level metrics e.g., queue sizes, CPU load as performance indicators



A Typical Deployment

Low level metrics e.g., queue sizes, CPU load as performance indicators



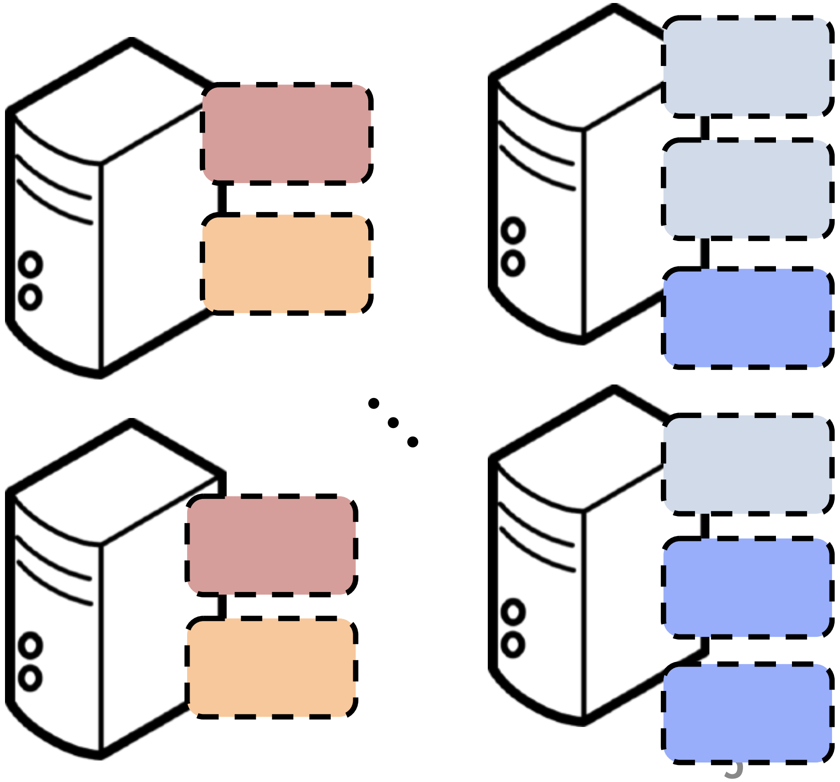
Manual scaling

Intent-Driven Multi-Tenancy

Intent-Driven Multi-Tenancy

Efficient resource usage across multiple users

➔ Multi-tenancy



Intent-Driven Multi-Tenancy

Efficient resource usage across multiple users

➔ Multi-tenancy

Application-aware adaptation to user requirements

➔ Intent-driven Multi-tenancy

Intent-Driven Multi-Tenancy

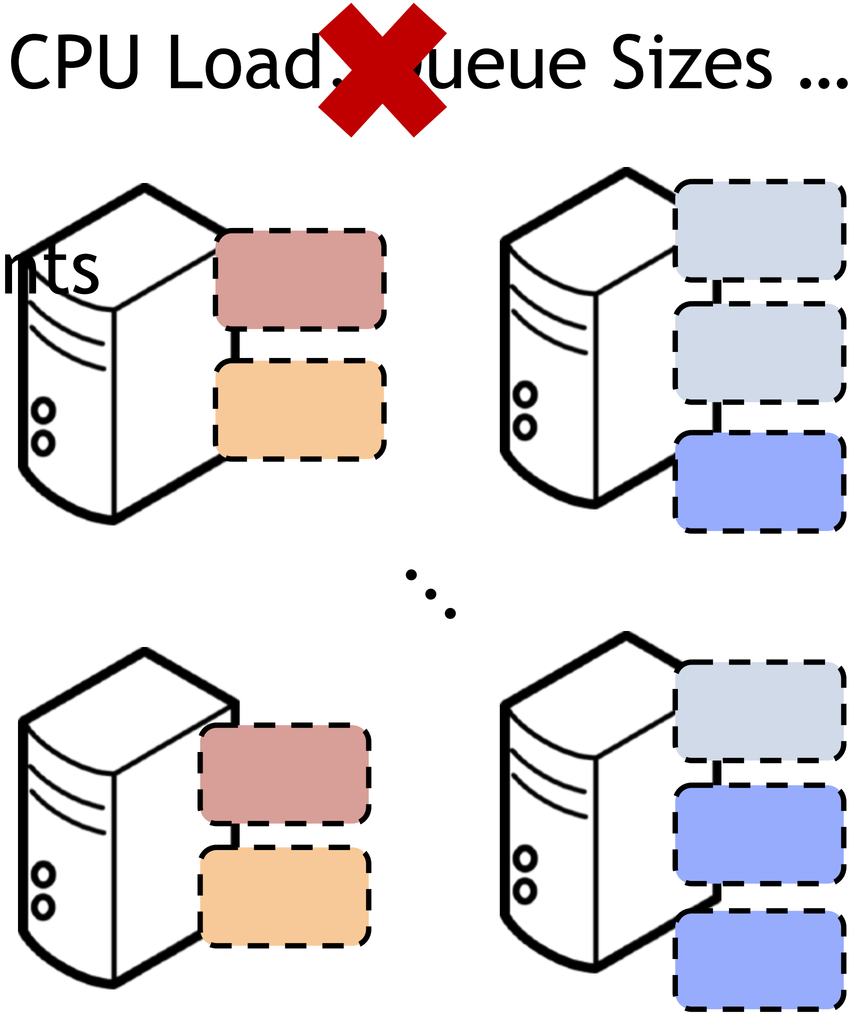
Efficient resource usage across multiple users

➔ Multi-tenancy

Application-aware adaptation to user requirements

➔ Intent-driven Multi-tenancy

Job	Description	Service Level Objective (SLO)
1	Finding ride price	Latency < 5 s
2	Analyzing earnings over	Throughput > 10K/hr.



CPU Load, Queue Sizes …

Problem

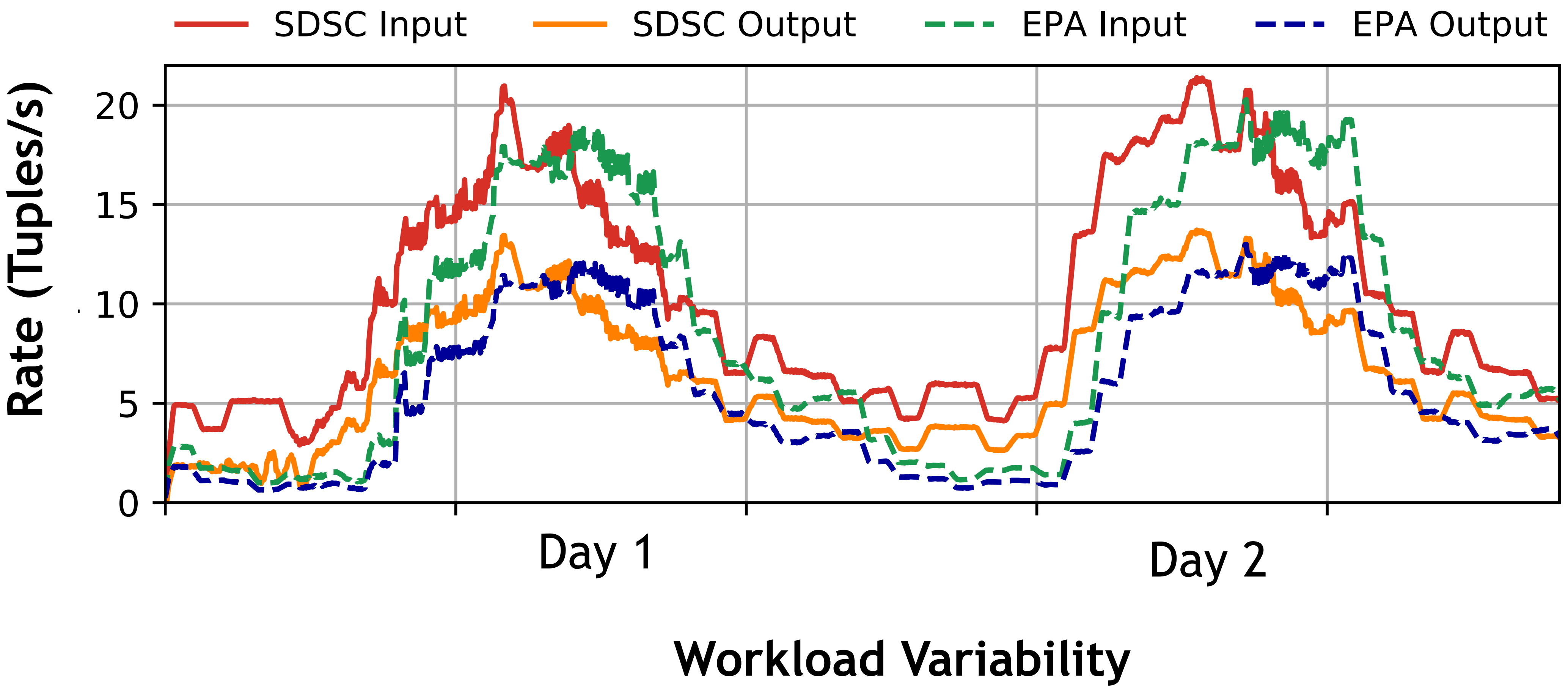
How can we
achieve user-facing service level objectives
for stream processing jobs
on multi-tenant clusters?

Problem

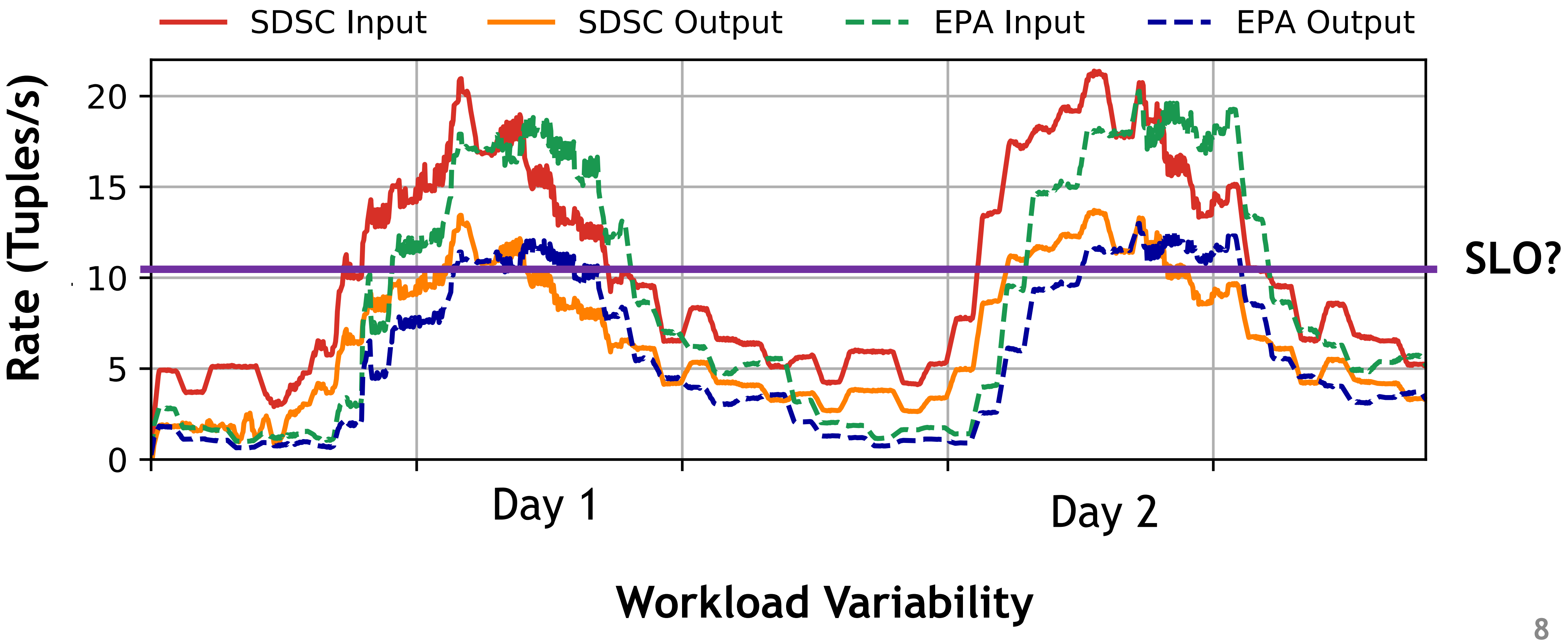
How can we
achieve user-facing service level objectives
for stream processing jobs
on multi-tenant clusters?

Latency,
Throughput

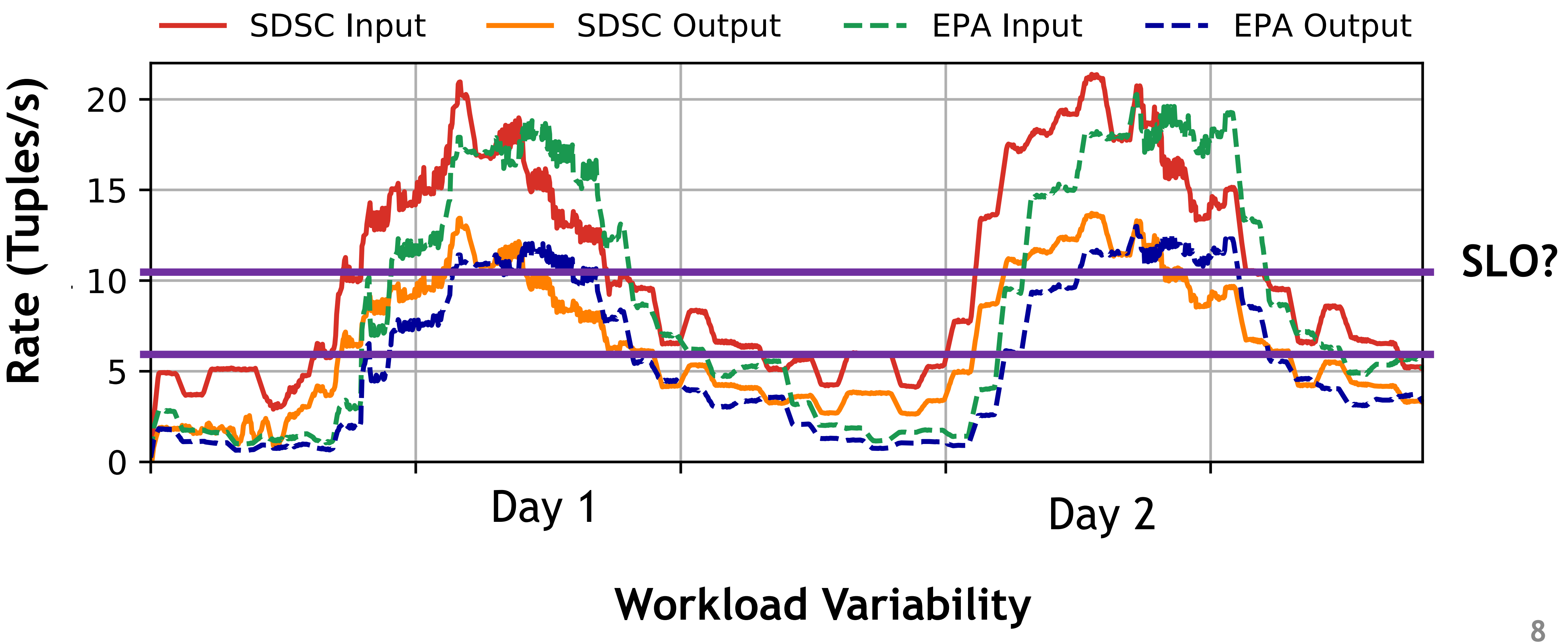
Absolute Throughput SLOs are not Useful



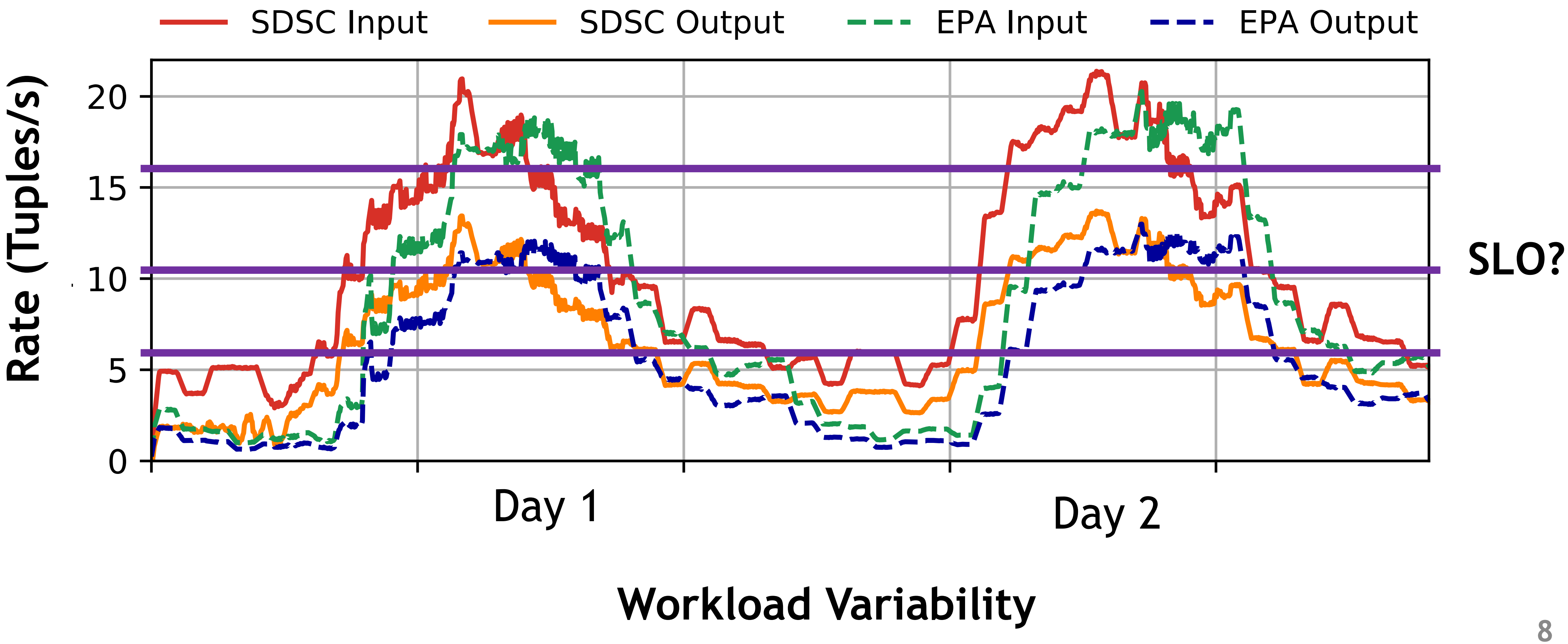
Absolute Throughput SLOs are not Useful



Absolute Throughput SLOs are not Useful

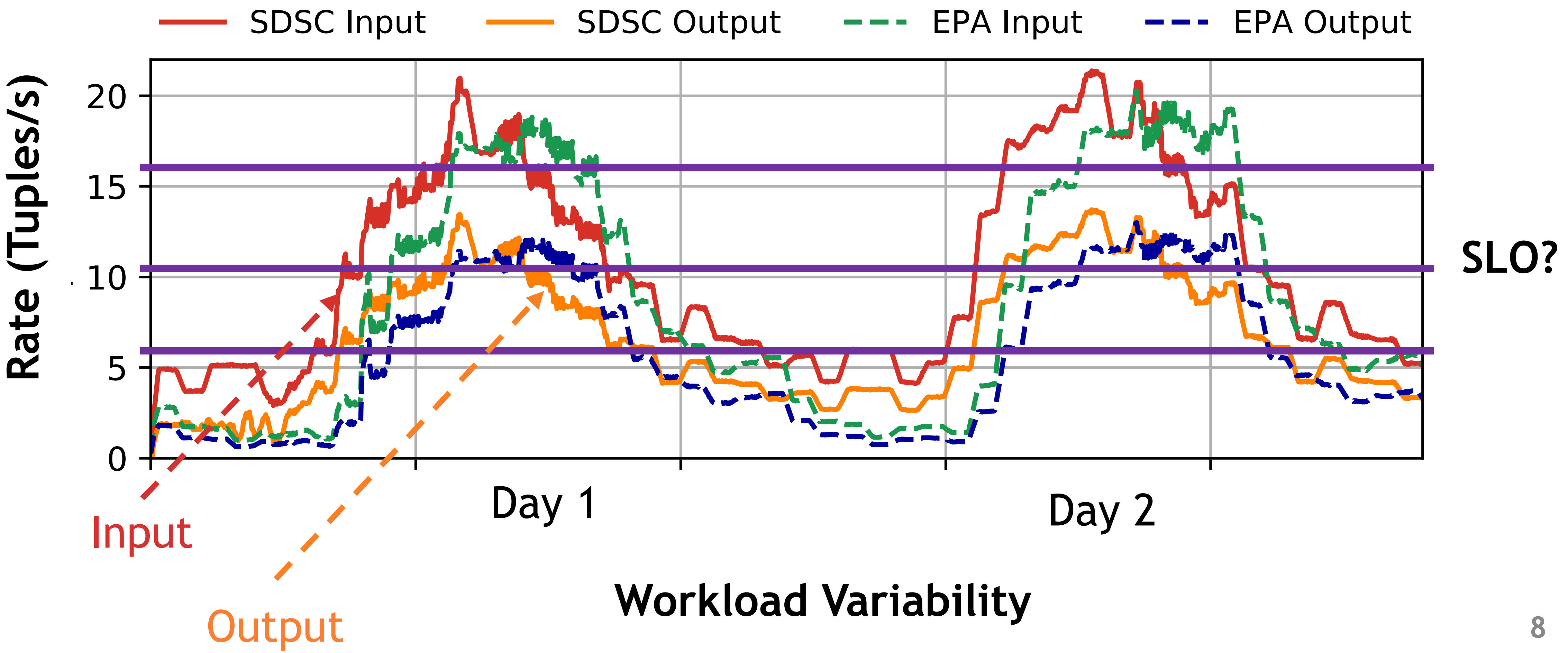


Absolute Throughput SLOs are not Useful



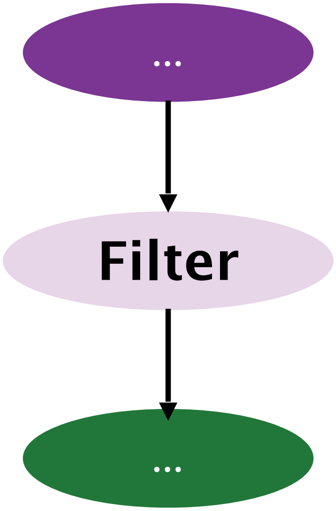
SLO?

Absolute Throughput SLOs are not Useful

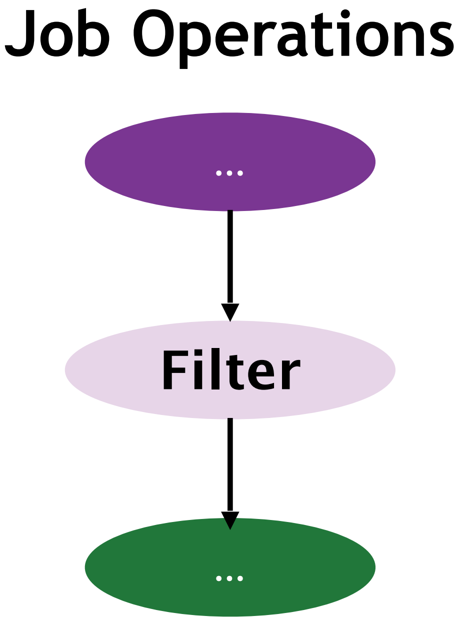


Absolute Throughput SLOs are not Useful

Job Operations



Absolute Throughput SLOs are not Useful



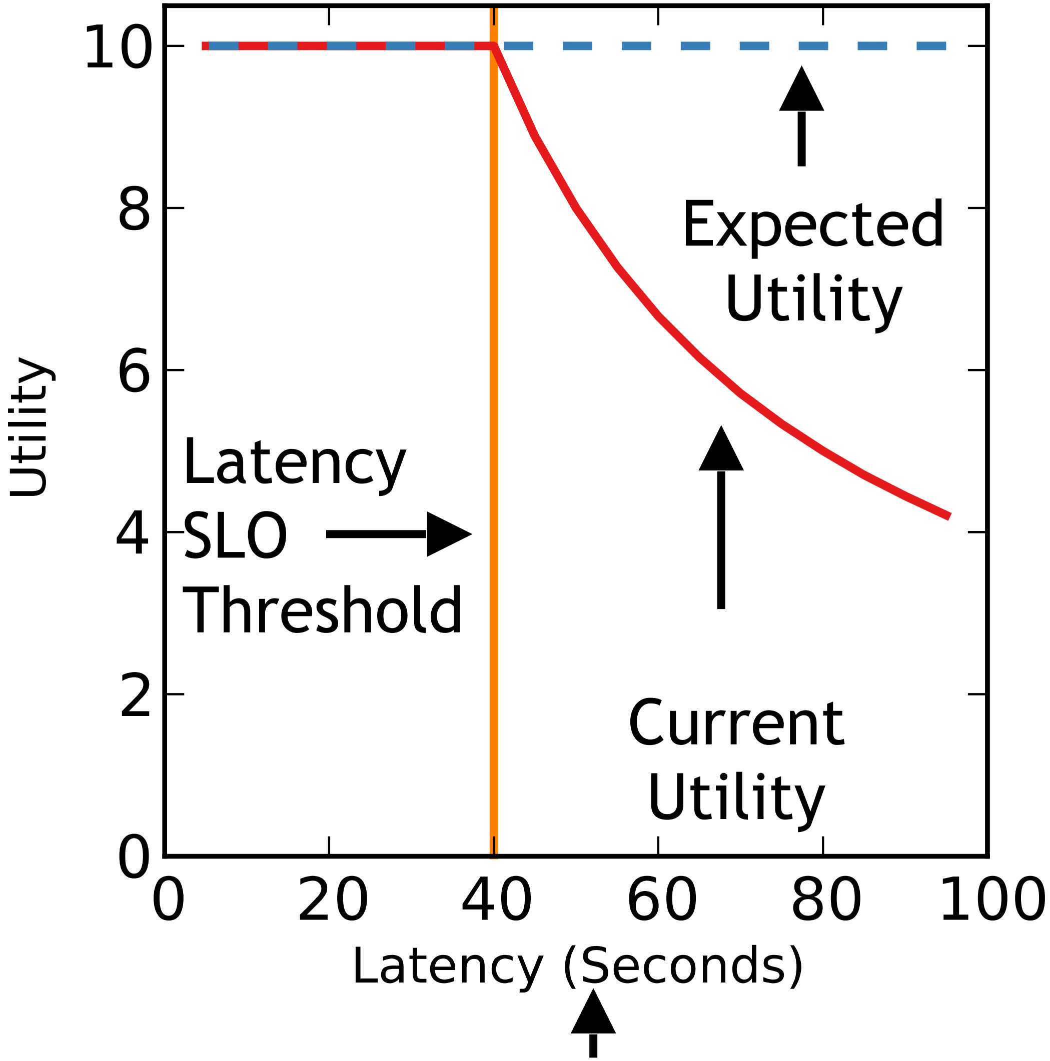
Juice: fraction* of the input data processed by the job per unit time.

Jobs benefit even below SLO threshold

Job Utility Functions

Jobs benefit even below SLO threshold

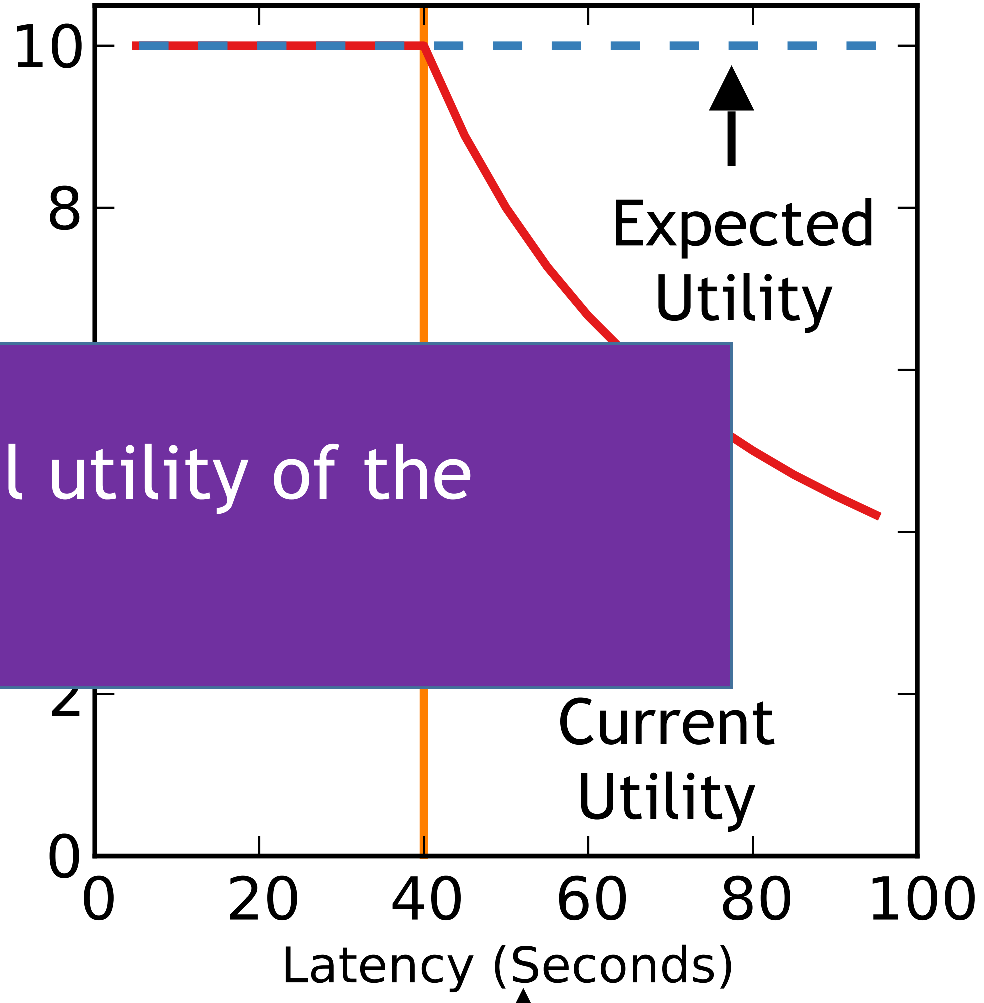
Job Utility Functions



Utility function for a single job

Jobs benefit even below SLO threshold

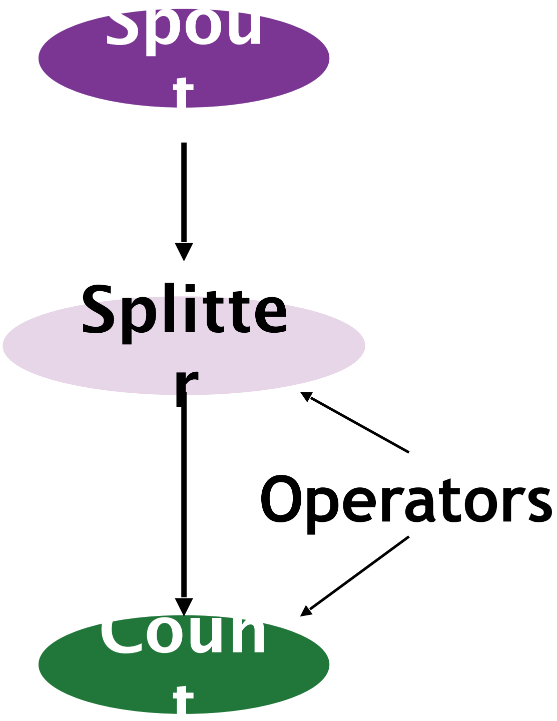
Job Utility Functions



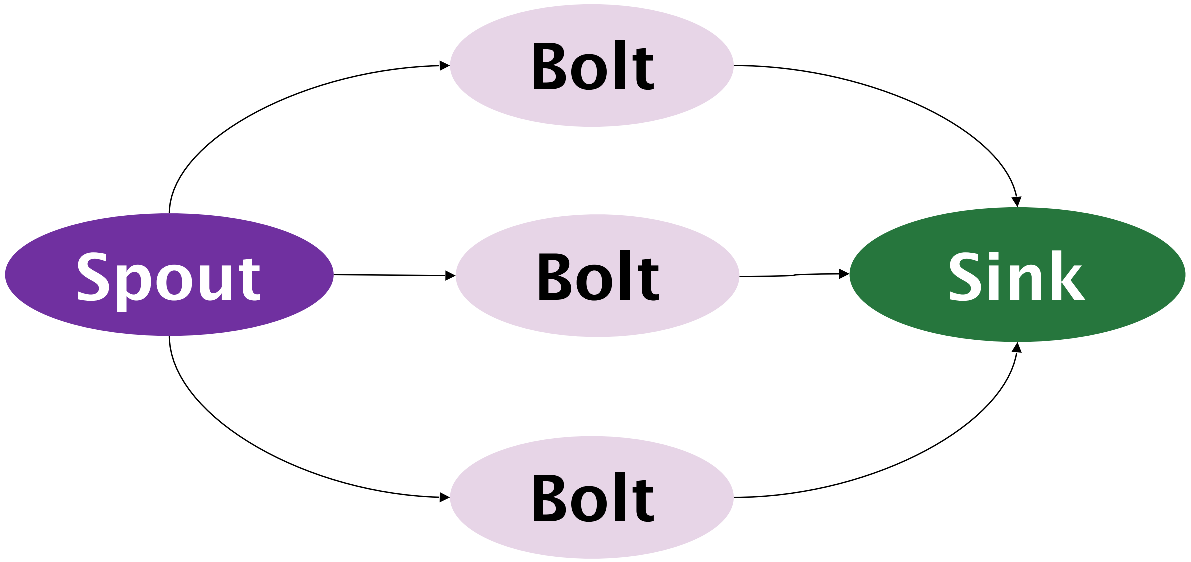
Henge’s goal → Maximize the total utility of the cluster

Utility function for a single job

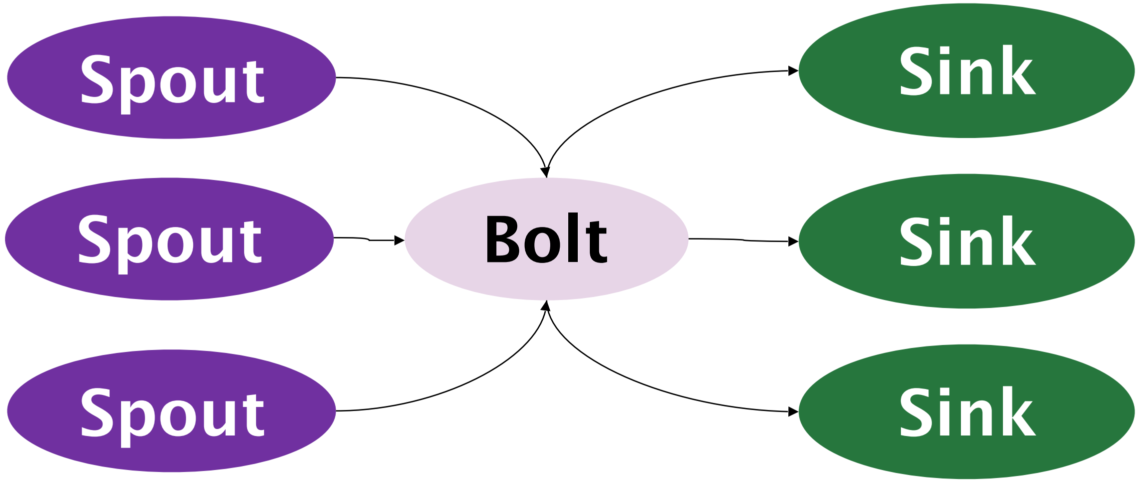
Background: Stream Processing Topologies (Jobs)



Logical DAG for a Word Count Job

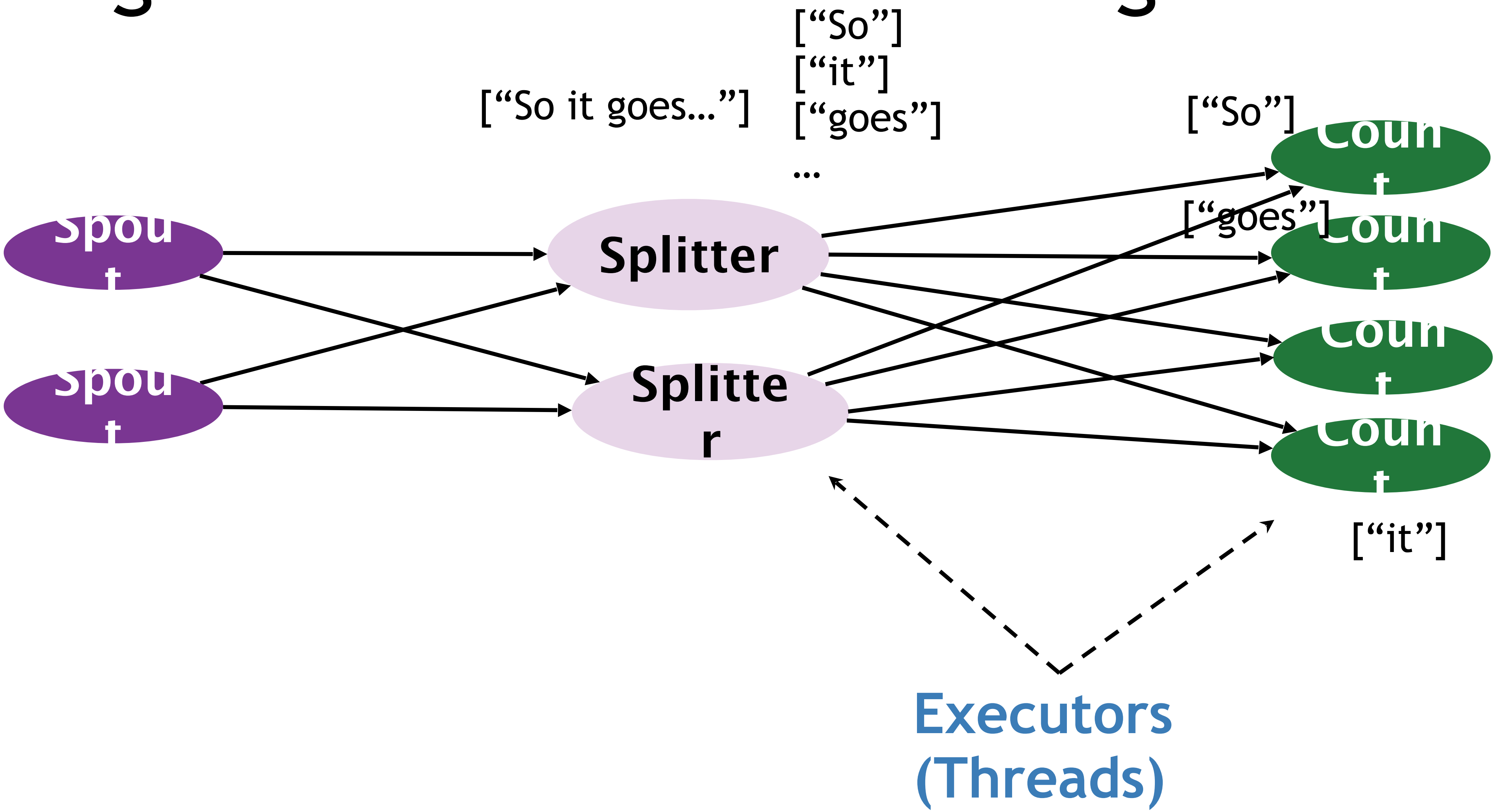


Diamond Topology

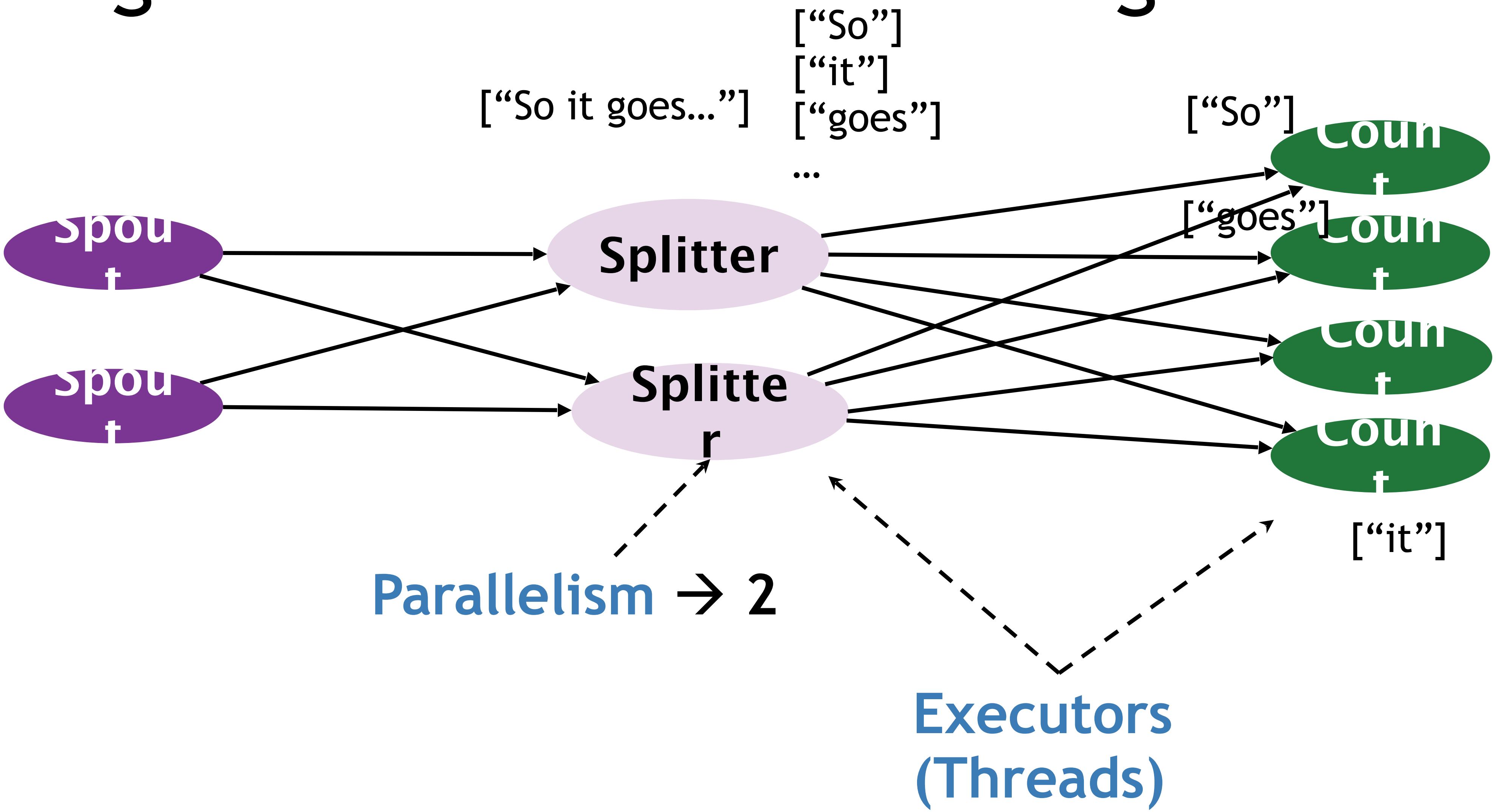


Star Topology

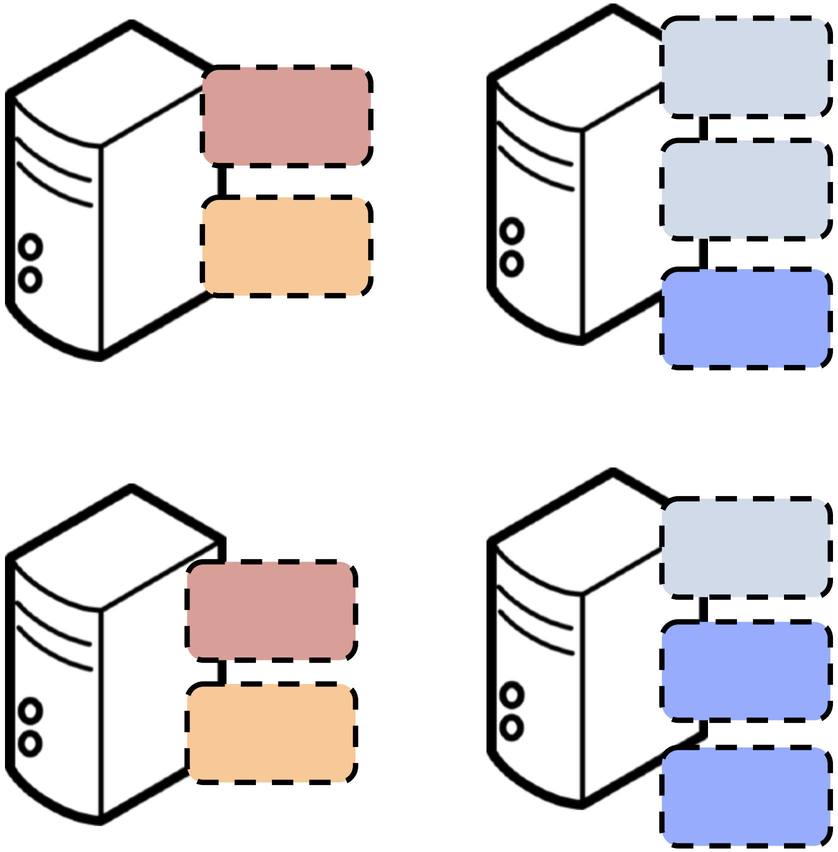
Background: Stream Processing Jobs



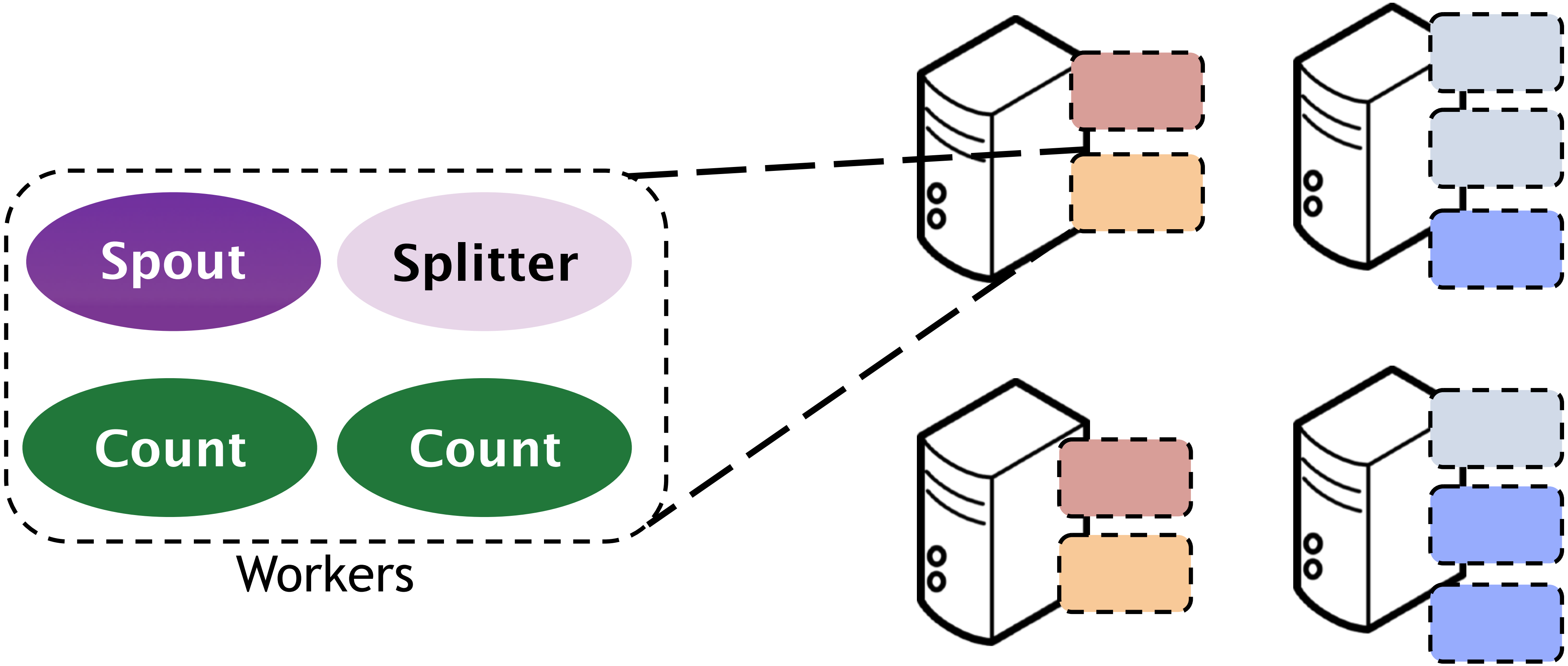
Background: Stream Processing Jobs



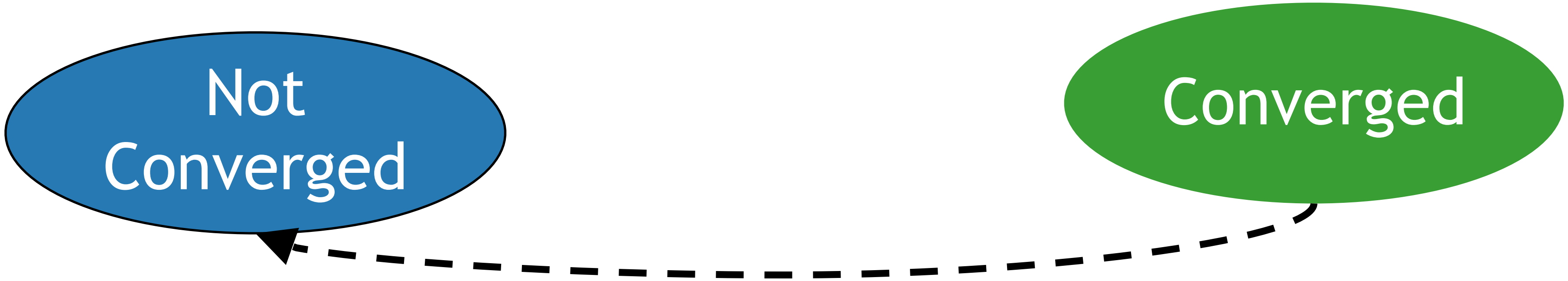
Background: A Physical Deployment



Background: A Physical Deployment



Henge’s Cluster-Wide State Machine

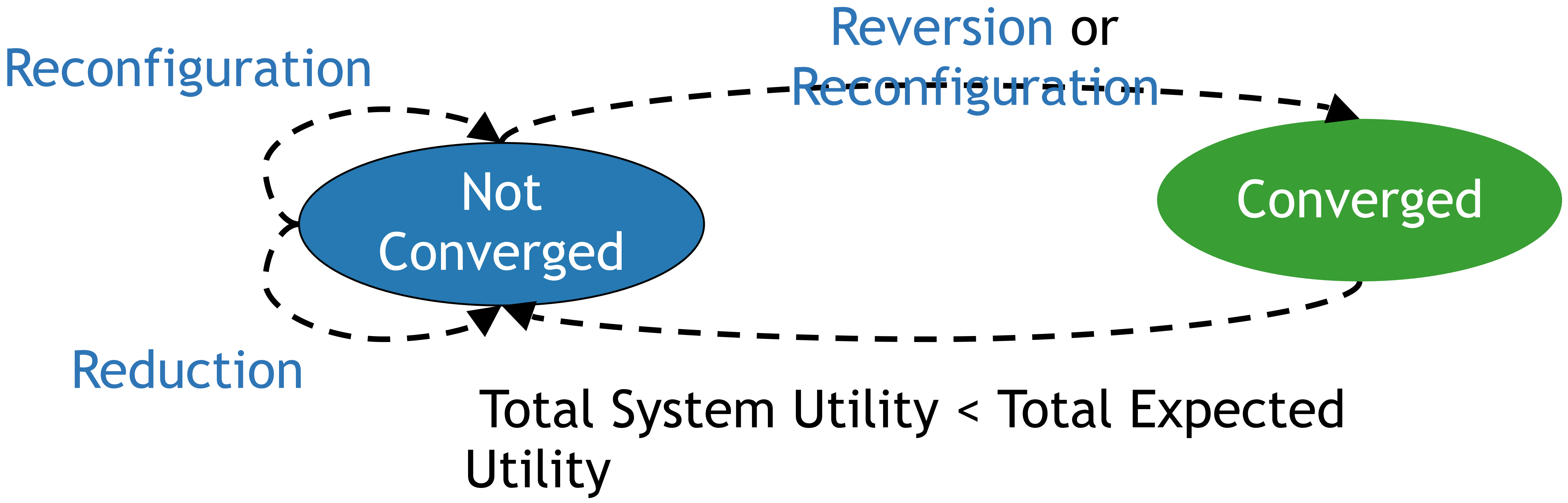


Not Converged

Converged

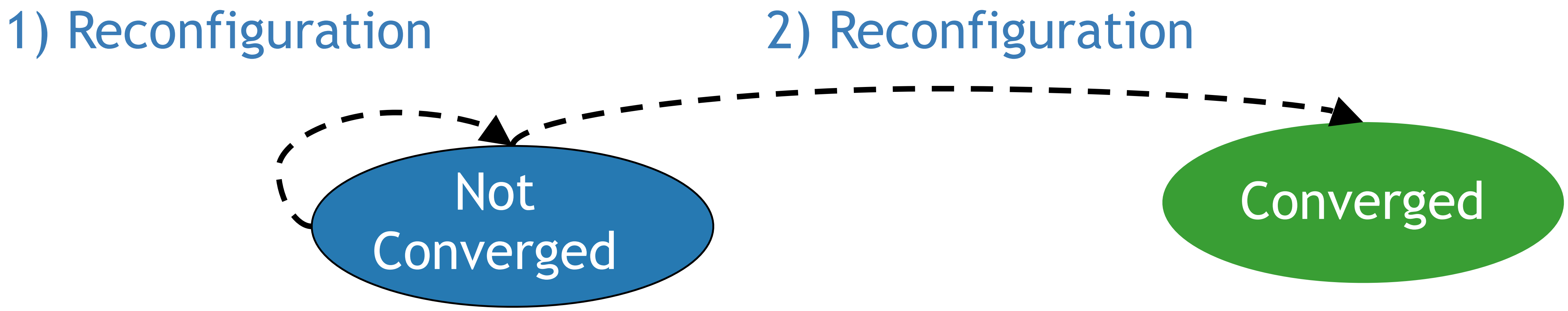
Total System Utility < Total Expected Utility

Henge's Cluster-Wide State Machine


Reconfiguration
Reversion or Reconfiguration
Not Converged
Converged
Reduction
Total System Utility < Total Expected Utility

Reconfiguration

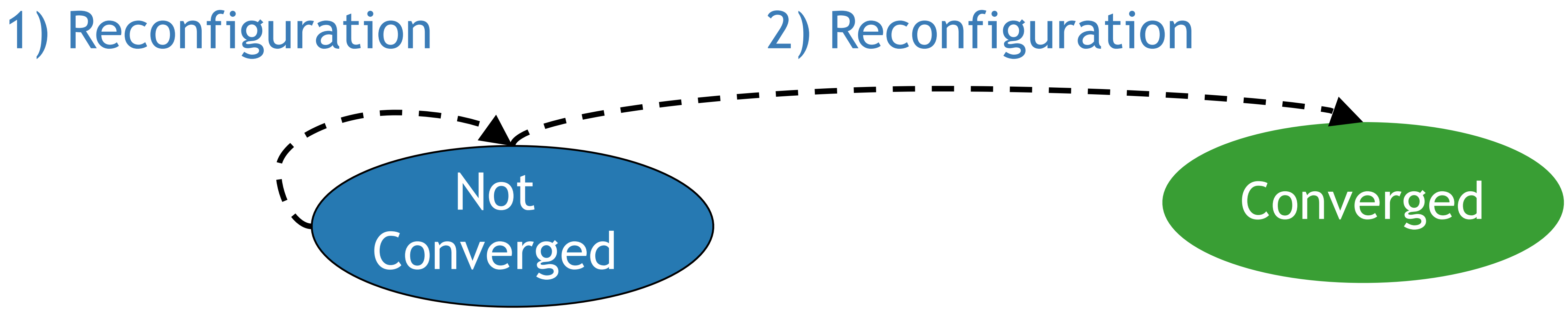
De-congest operator by increasing parallelism level of executors



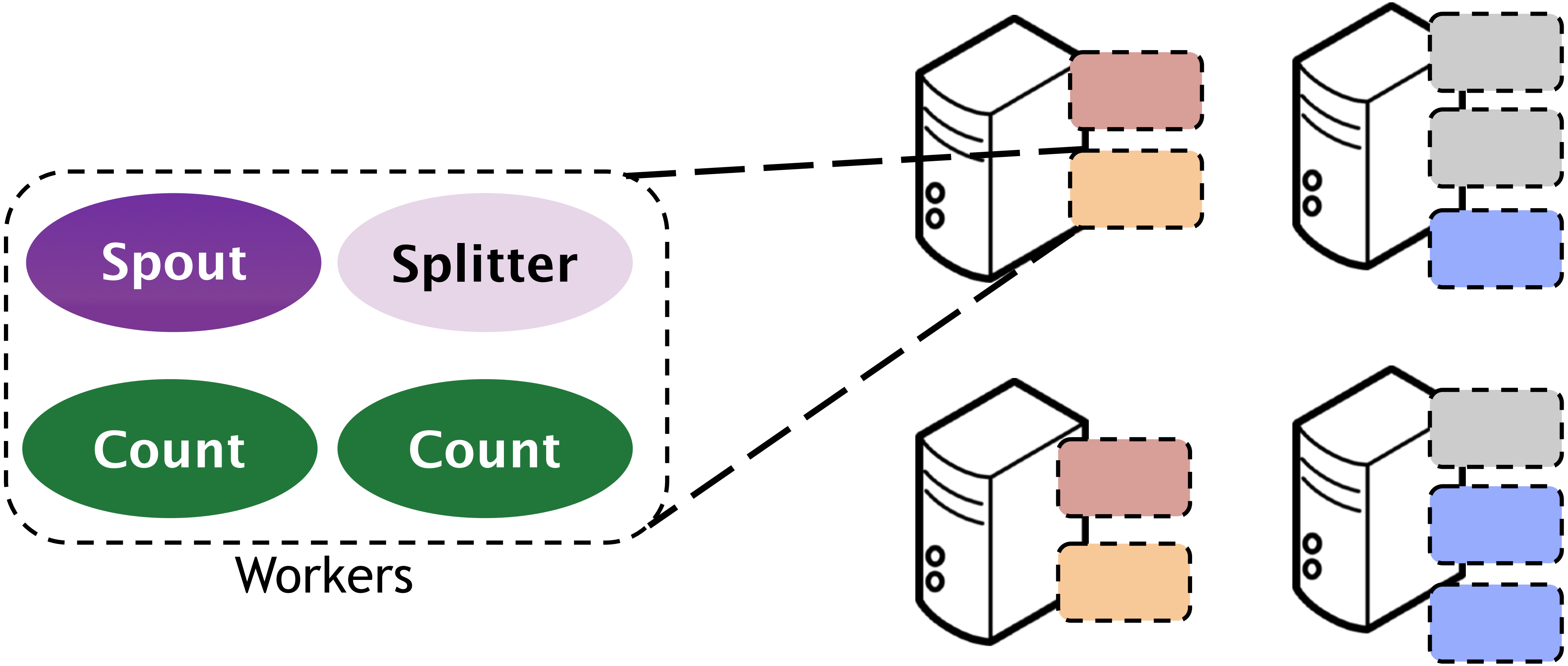
Reconfiguration

De-congest operator by increasing parallelism level of executors

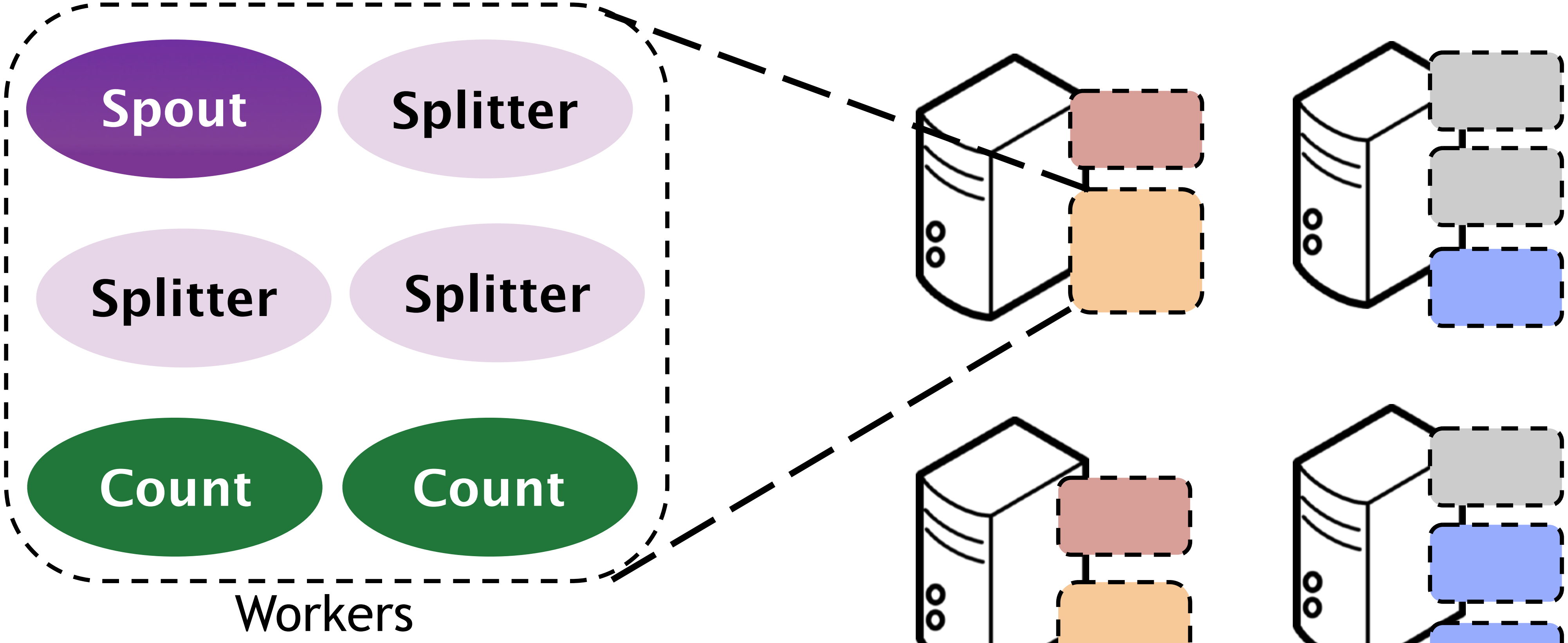
3) **Black-list** topologies that show less than Δ% improvement



Bottlenecks



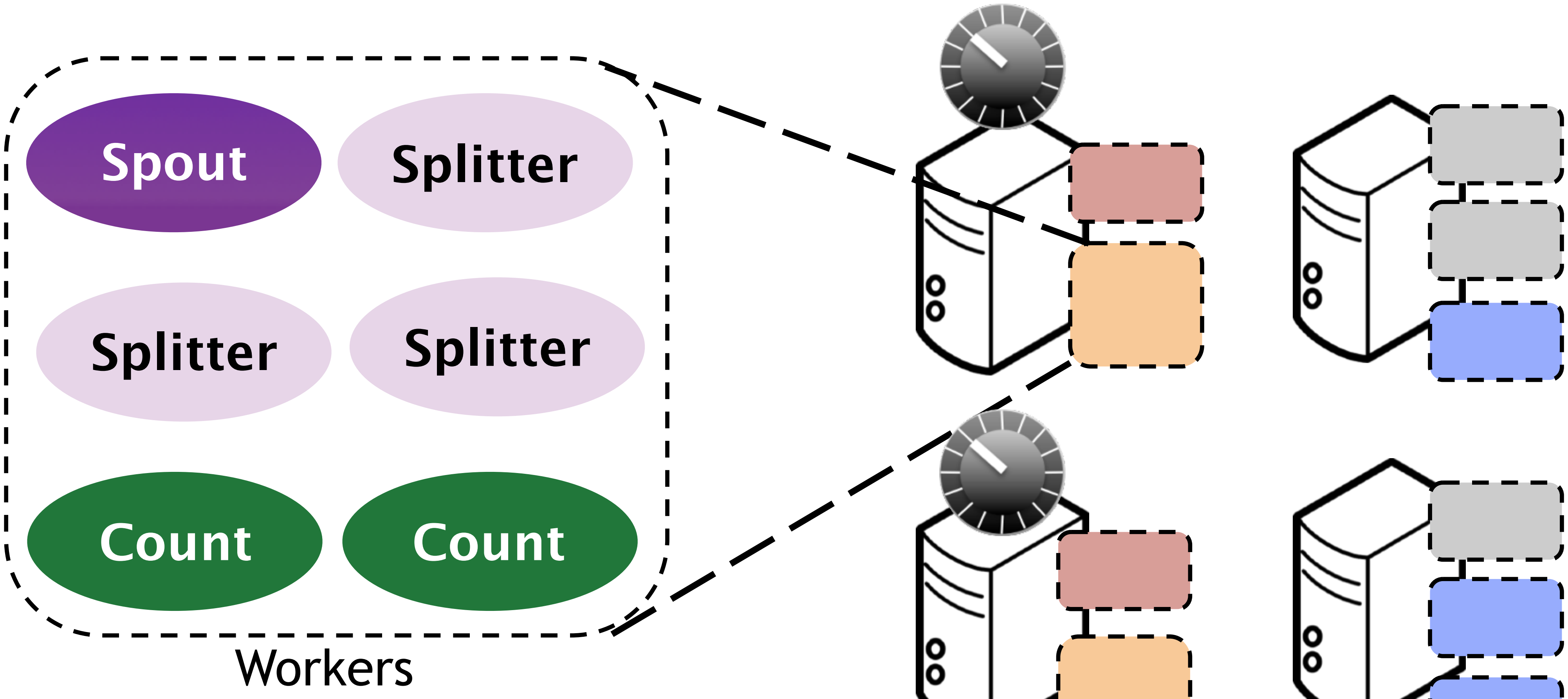
Reconfiguration



Bottlenecks

Reconfigs.

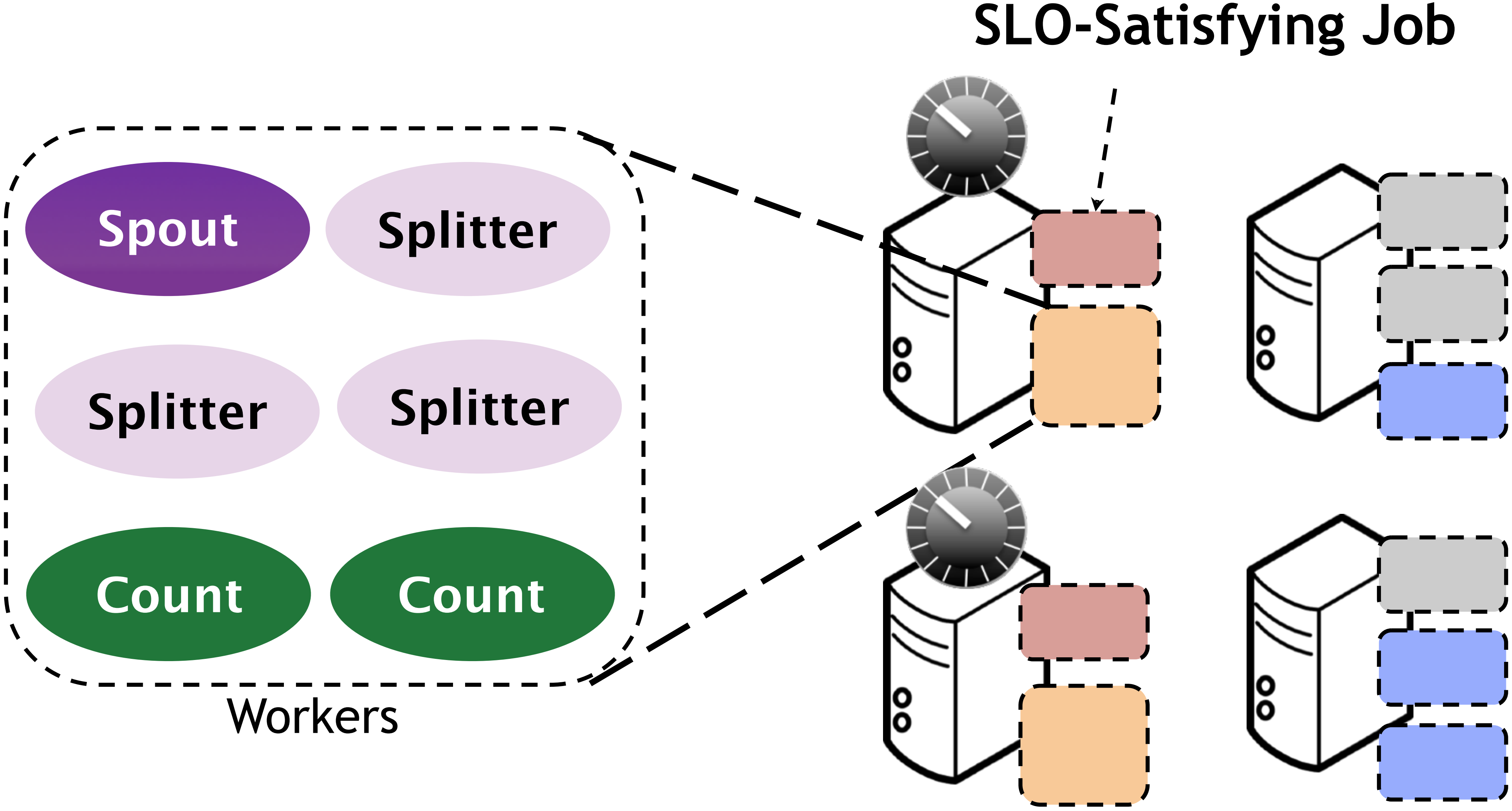
High Load



Bottlenecks

Reconfigs.

High Load

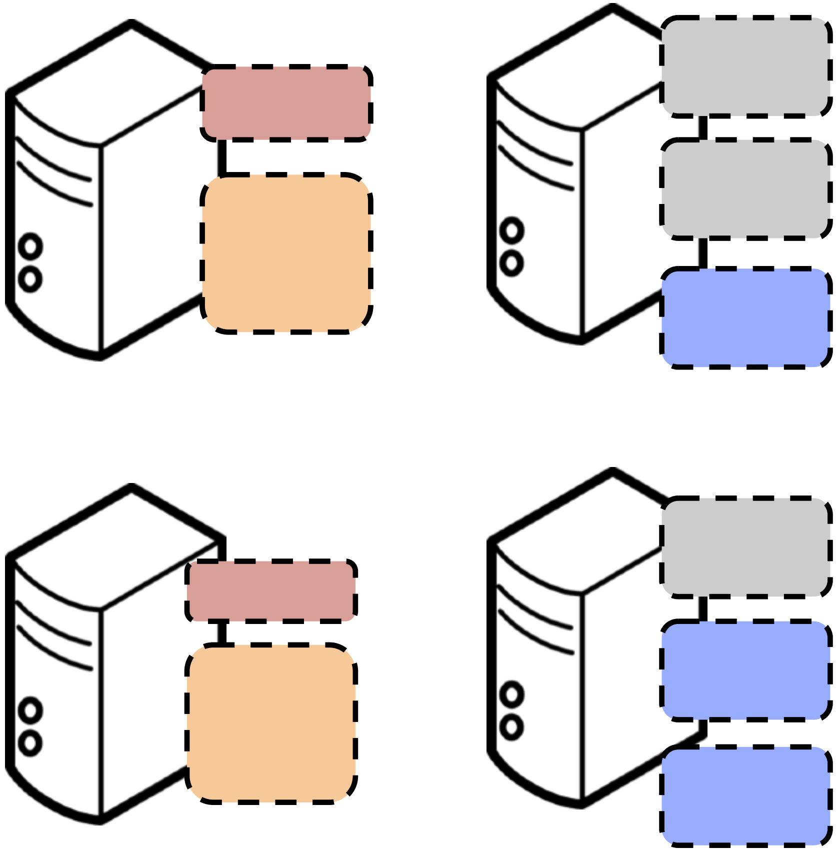


Bottlenecks

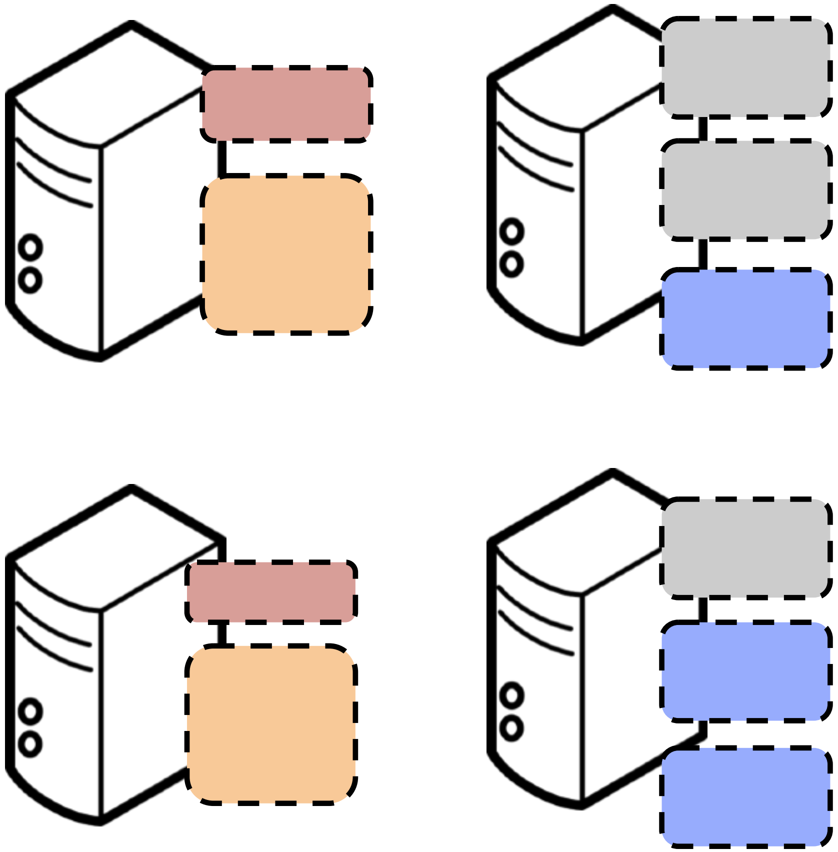
Reconfigs.

High
Load

Reduction

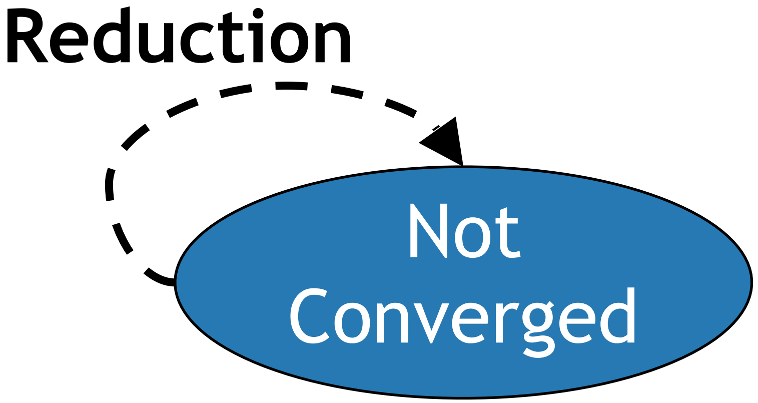


Reduction



Reduction

Reconfigurations → drop in utility



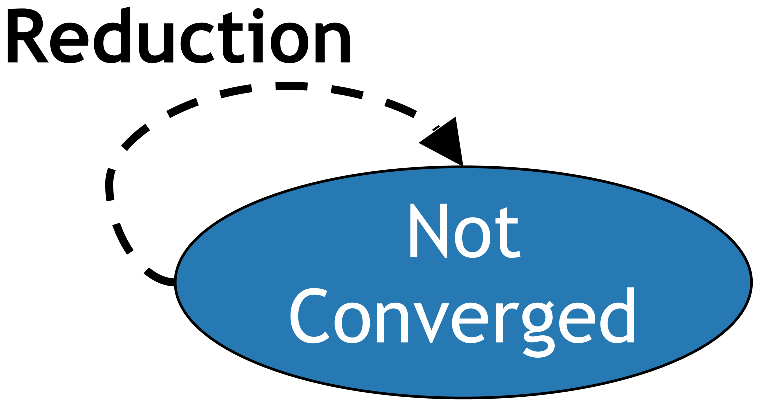
Reduction

Reconfigurations → drop in utility

If high CPU load on majority of machines, **reduce** parallelism for operators that

a) are in topologies that satisfy their SLO

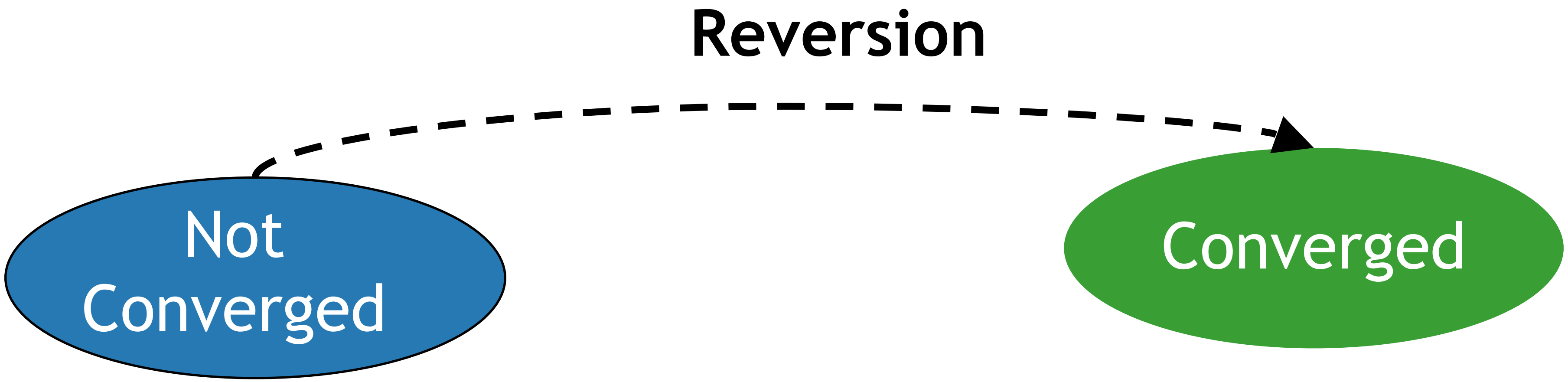
b) are not congested



Reversion

Reconfigurations → drop in utility *and reduction is not possible*

Revert to a past configuration that provided best utility



Evaluation

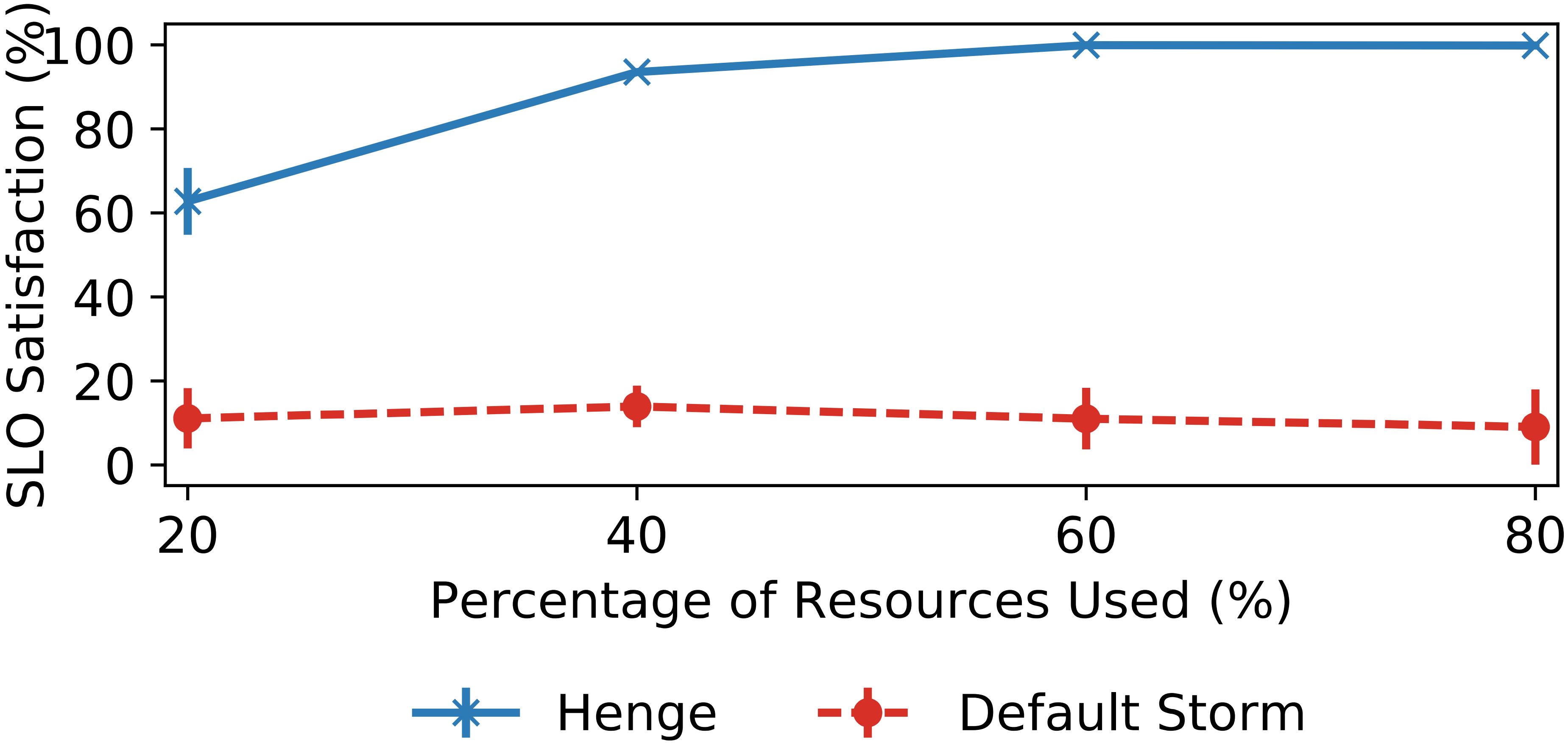
Real-world workloads:

- Yahoo!
- Twitter
- Web log traces

Experimental Setup:

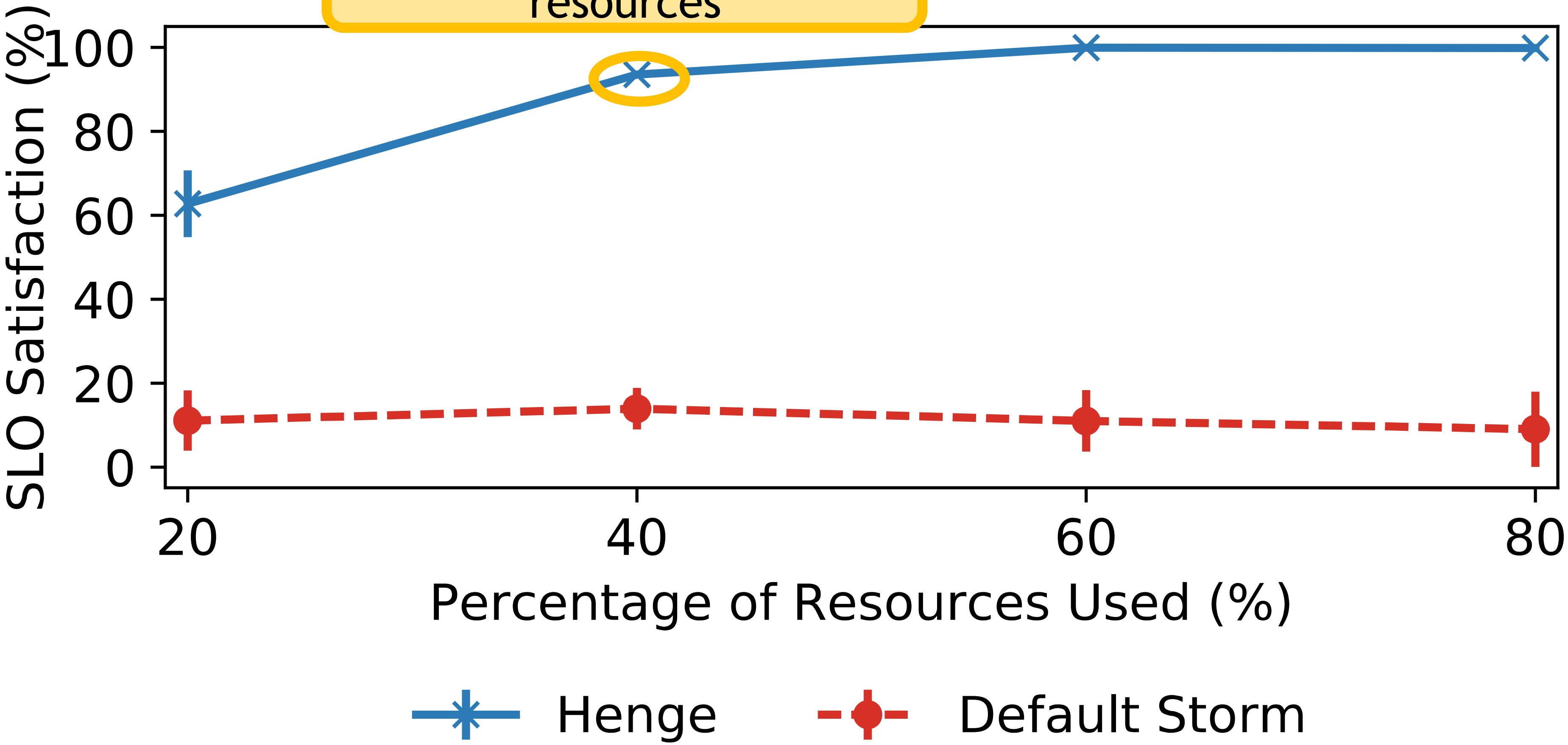
- 10-40 node Emulab cluster

Reducing cost and achieving high utilities

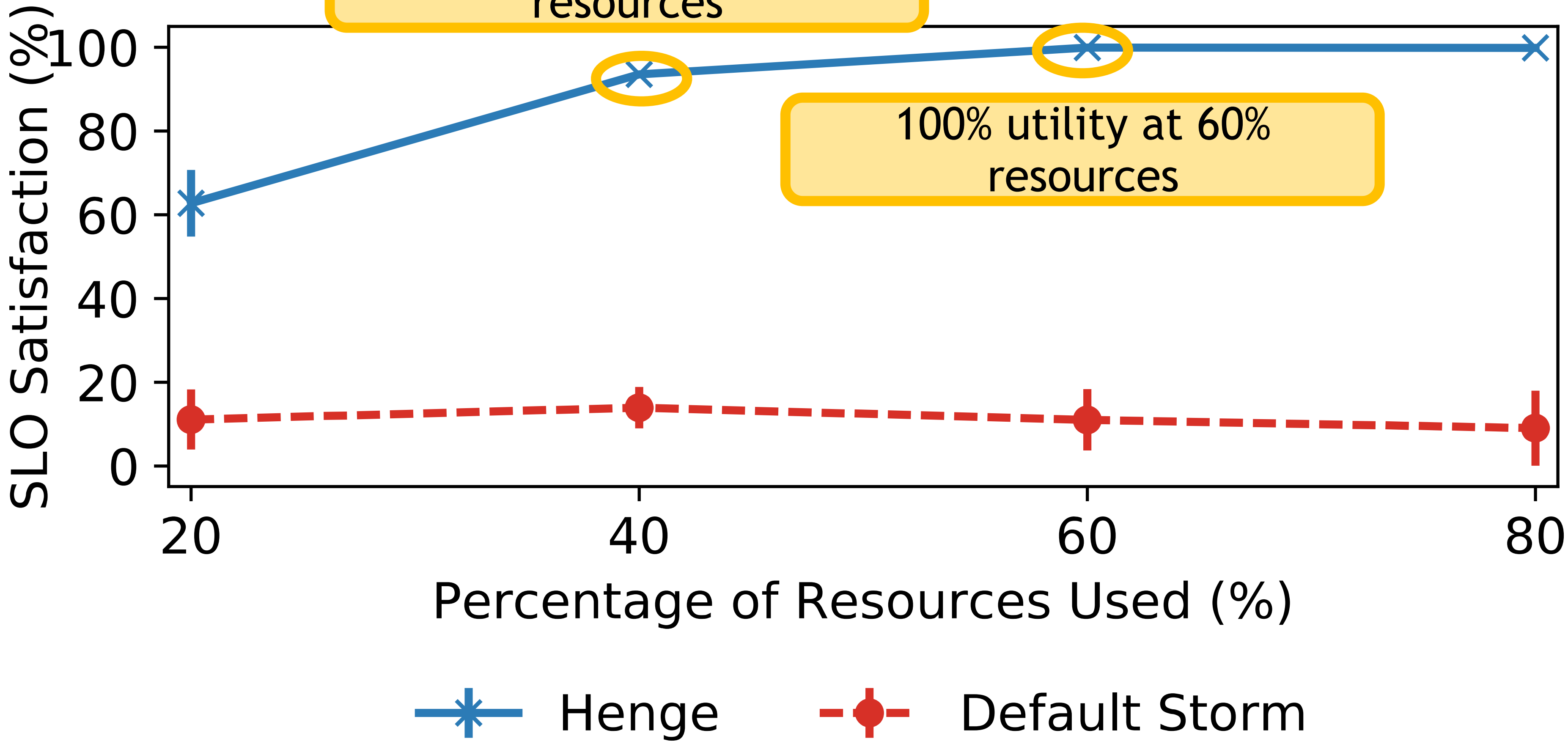


Reducing cost and achieving high utilities

93.5% utility at 40% resources



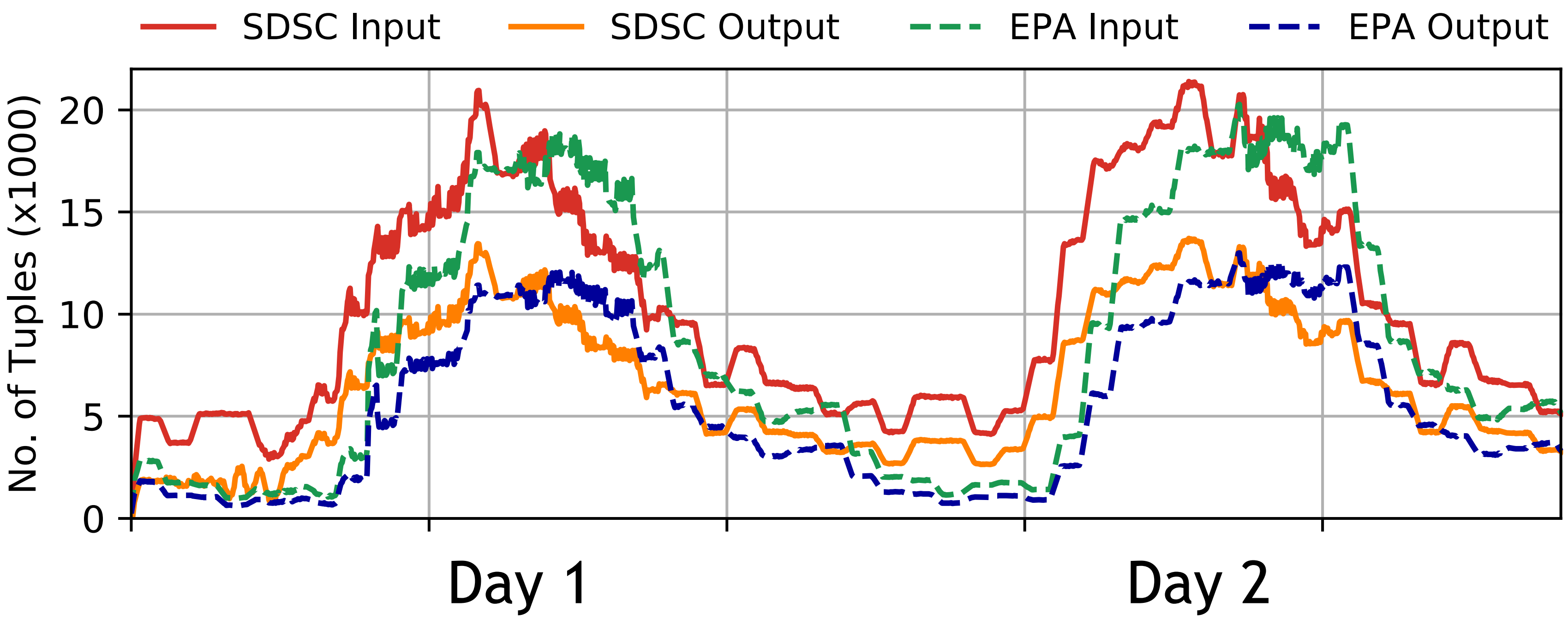
Reducing cost and achieving high utilities

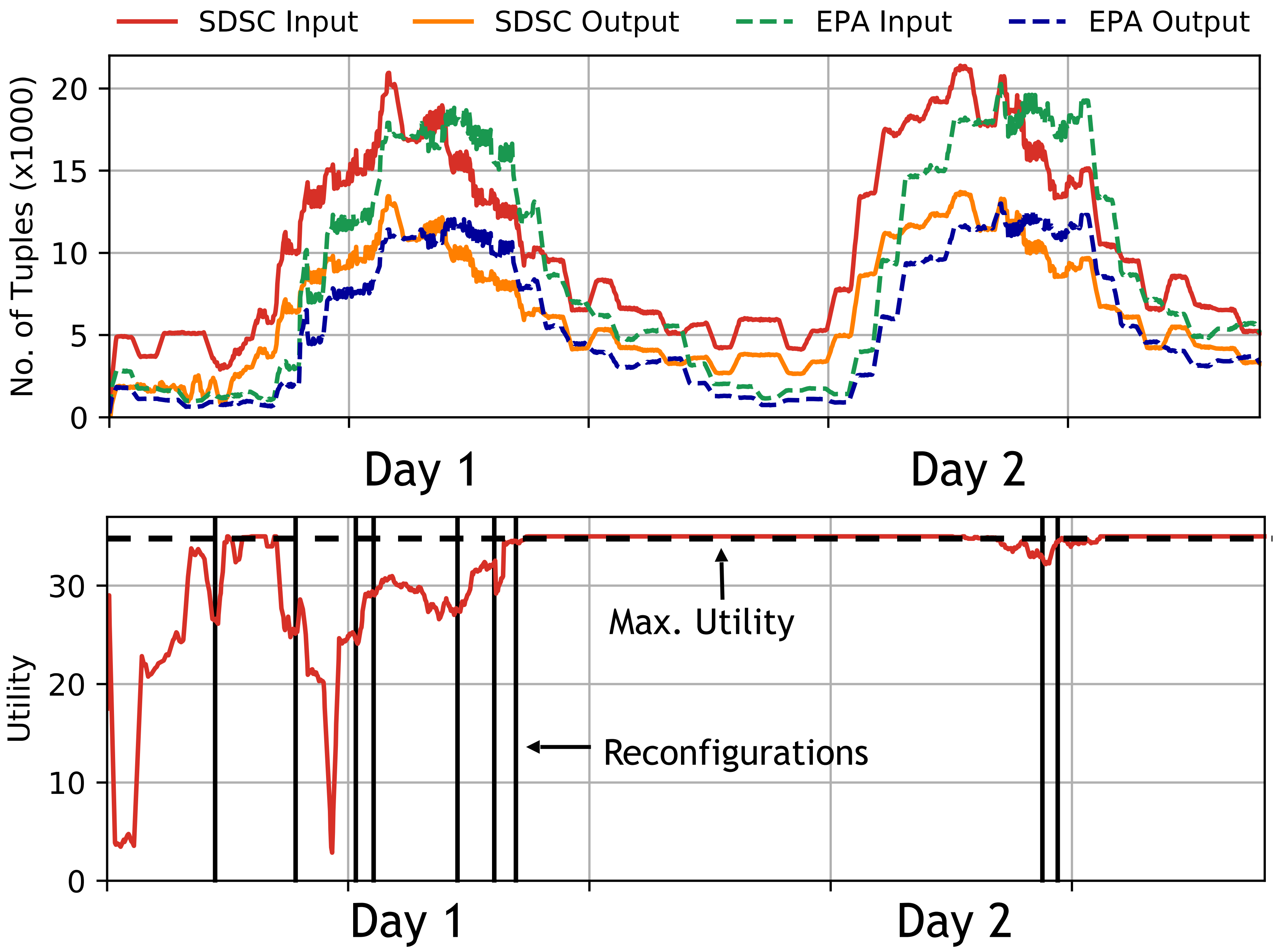


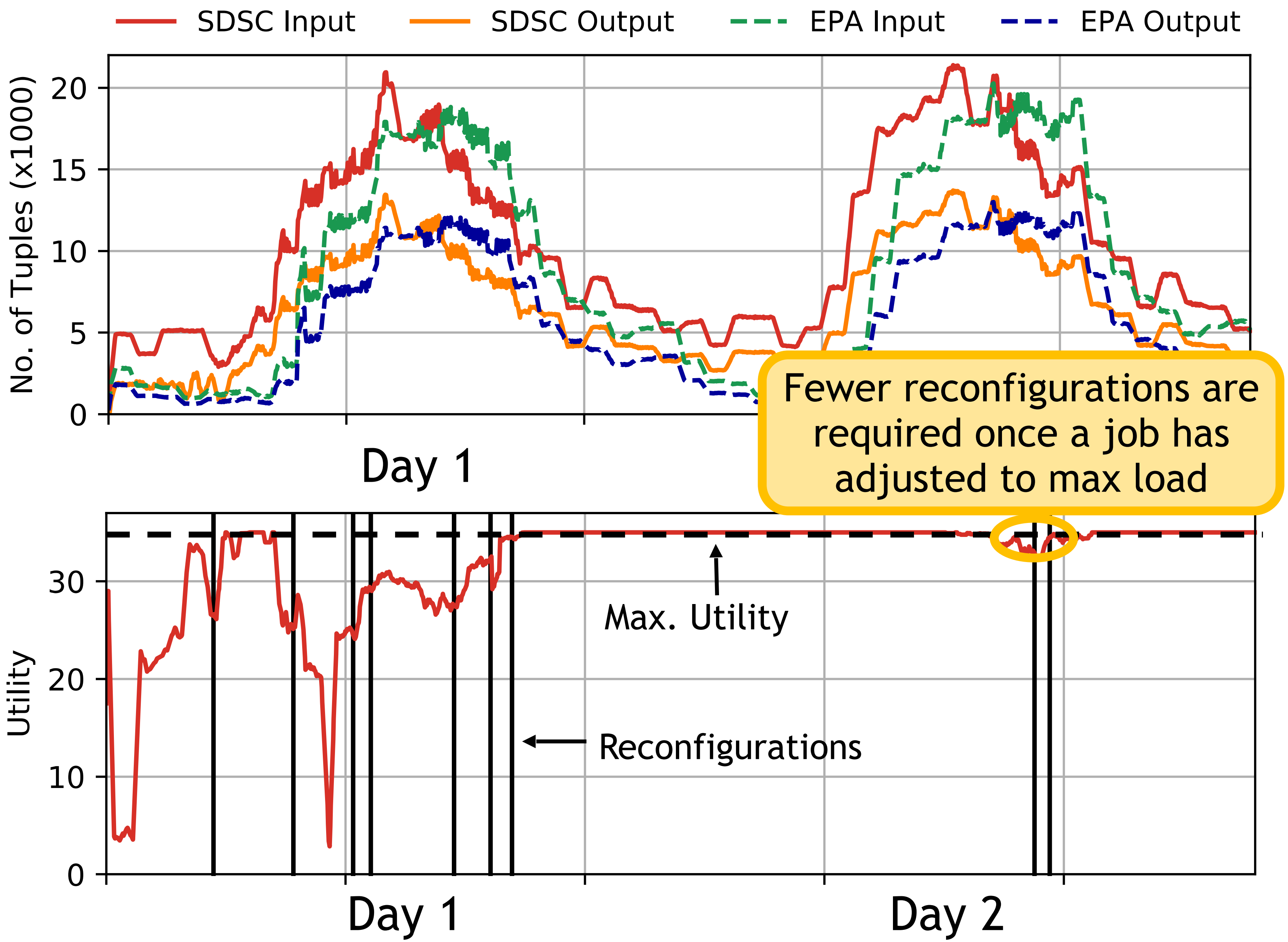
93.5% utility at 40% resources

100% utility at 60% resources

Adapting to a Diurnal Pattern

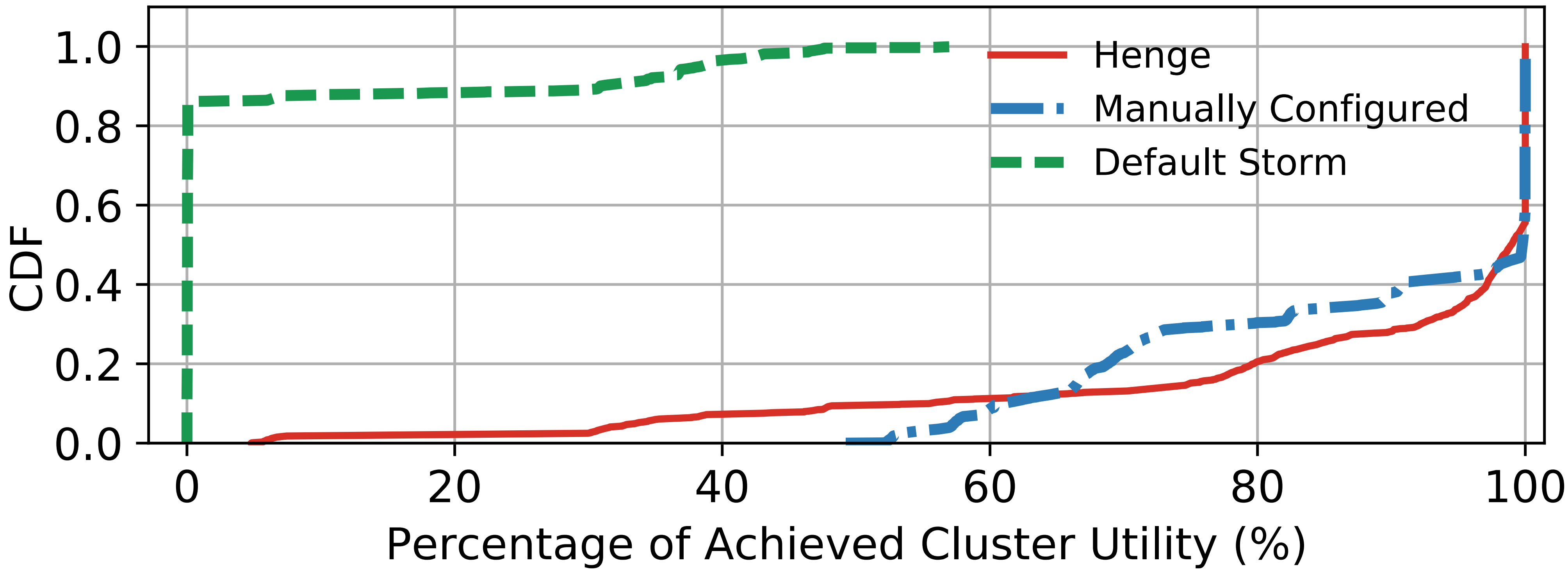

SDSC Input
SDSC Output
EPA Input
EPA Output
No. of Tuples (×1000)
0
5
10
15
20
Day 1
Day 2


SDSC Input
SDSC Output
EPA Input
EPA Output
No. of Tuples (x1000)
0
5
10
15
20
Day 1
Day 2
Utility
0
10
20
30
Max. Utility
Reconfigurations
Day 1
Day 2

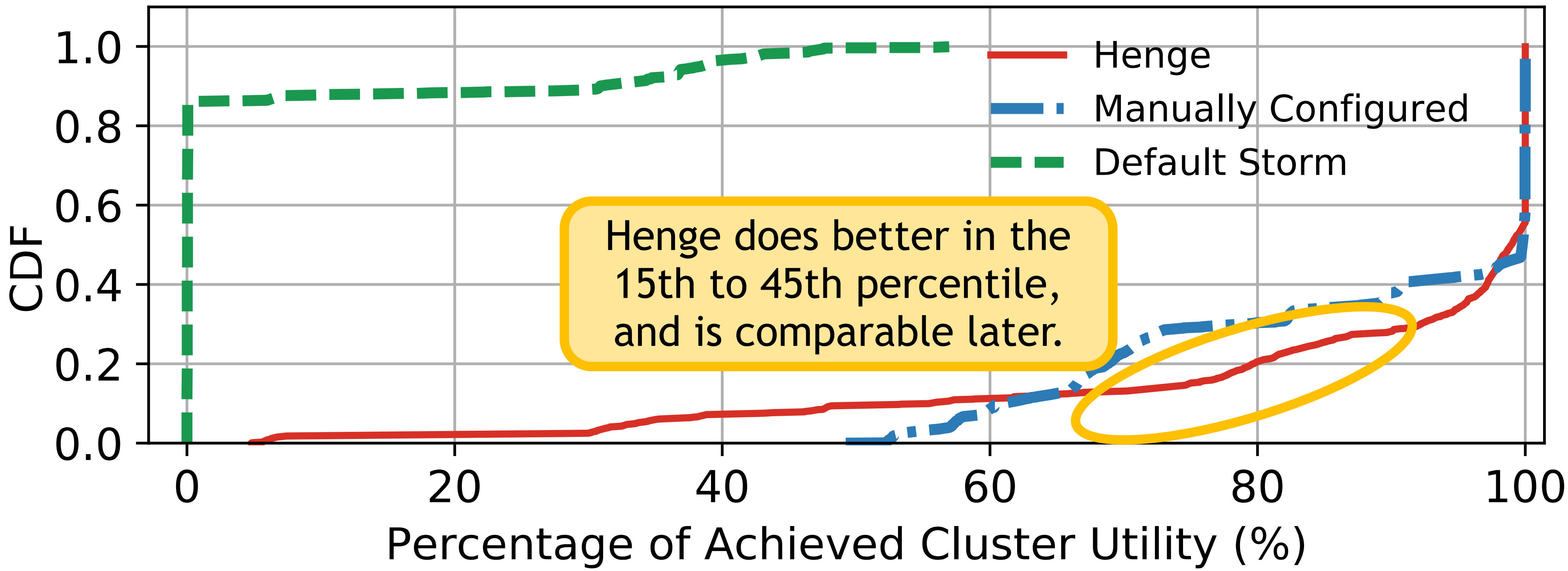


Fewer reconfigurations are required once a job has adjusted to max load

Can Henge do better than manual configuration?

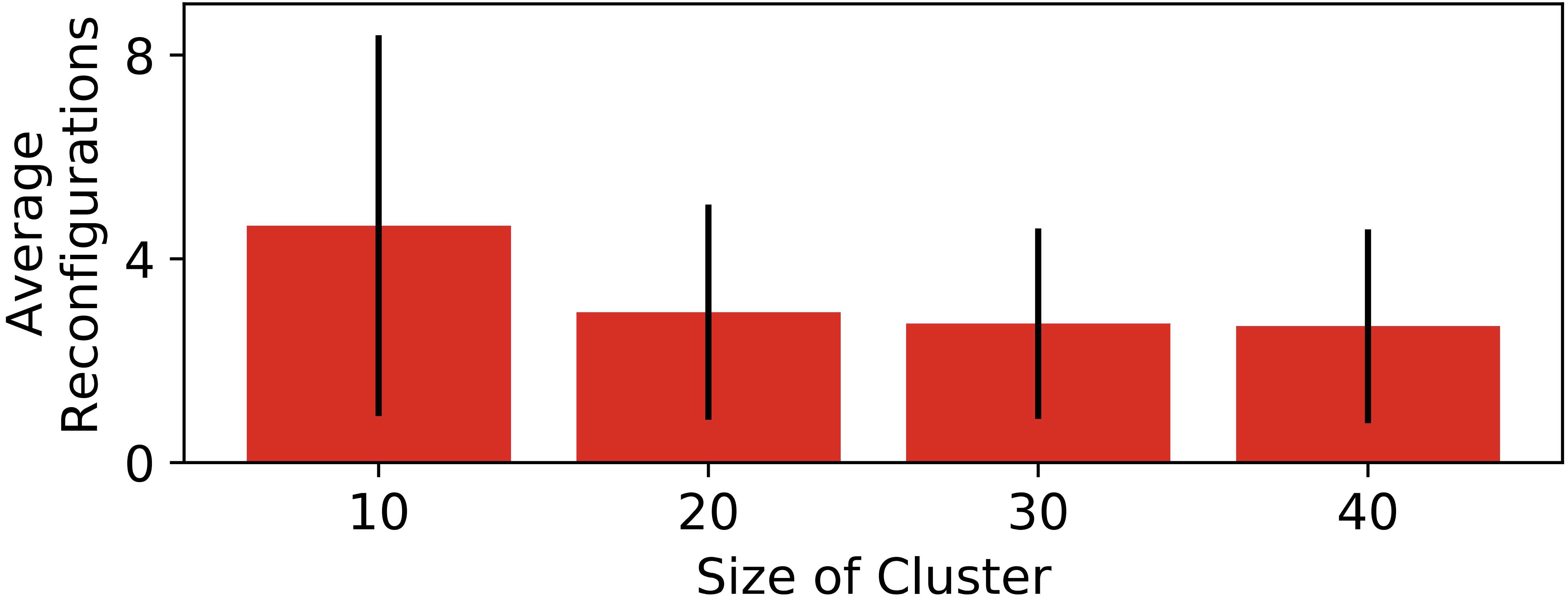


Can Henge do better than manual configuration?

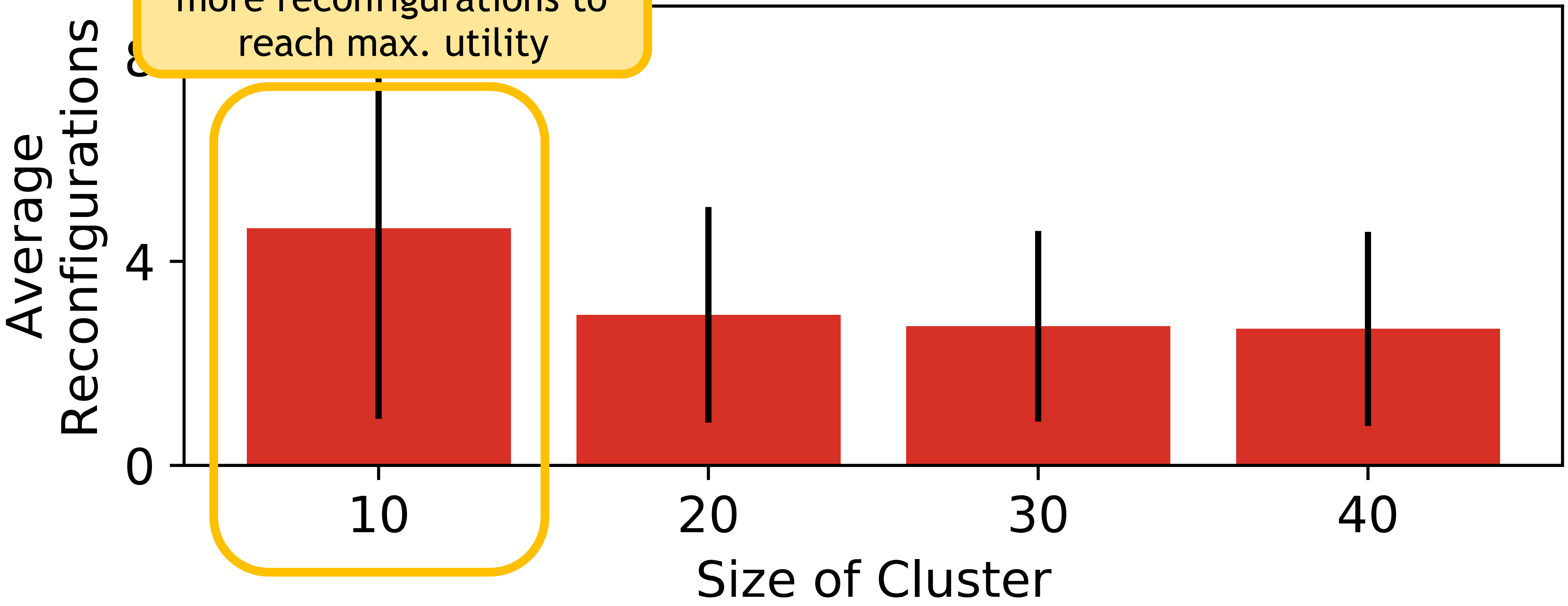


Henge does better in the 15th to 45th percentile, and is comparable later.

Scaling Cluster Size



Scaling Cluster Size



Limited resources entail more reconfigurations to reach max. utility

More Results

Henge can:

- handle dynamic workloads
    - abrupt e.g., spikes & natural fluctuations
    - gradual e.g., diurnal patterns
- satisfy hybrid SLOs
- scale with number of jobs & cluster size
- gracefully handle failures

Summary

- Henge allows users to specify **performance intents** for their jobs
- Henge's goal is to **maximize cluster-wide utility**
- The scheduler performs fine-grained **reconfigurations** to allow stream processing jobs to meet user-specified intents