DRIZZLE:
FAST AND ADAPTABLE STREAM PROCESSING AT SCALE

Shivaram Venkataraman, Aurojit Panda, **Kay Ousterhout**, Michael Armbrust, Ali Ghodsi, Michael Franklin, Benjamin Recht, Ion Stoica

Berkeley
UNIVERSITY OF CALIFORNIA

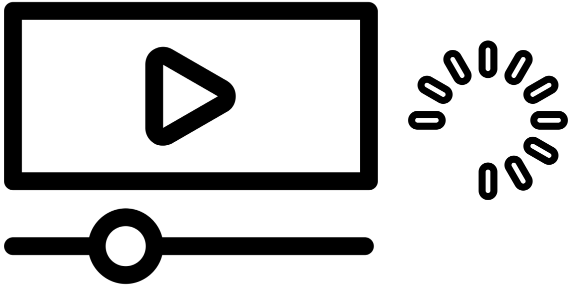
STREAMING WORKLOADS

STREAMING TRENDS: LOW LATENCY

Results power decisions by machines

Credit card fraud
↓
Disable account

Suspicious user logins
↓
Ask security questions



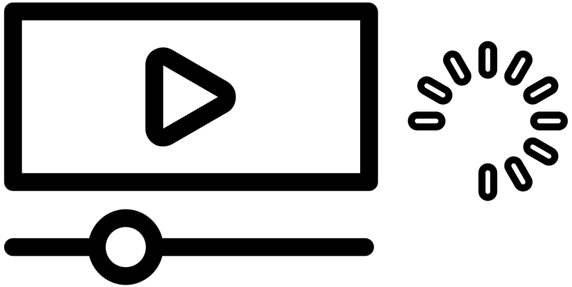
Slow video load
↓
Direct user to new CDN

STREAMING REQUIREMENTS: HIGH THROUGHPUT

Disable stolen accounts

Detect suspicious logins

Dynamically adjust application behavior



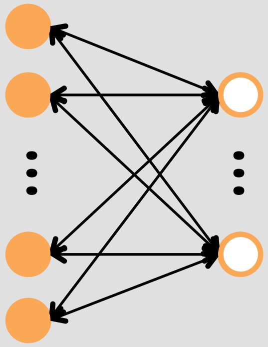
As many as 10s of millions of updates per second

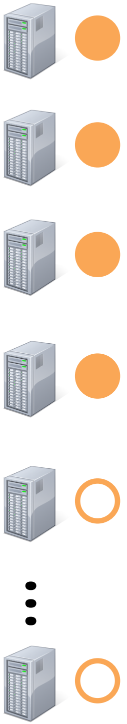
Need a distributed system

DISTRIBUTED EXECUTION MODELS

EXECUTION MODELS: CONTINUOUS OPERATORS

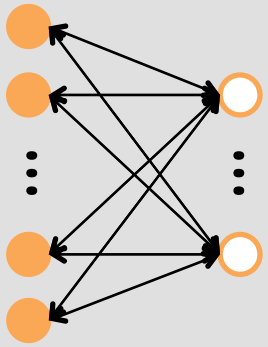
Group by user,
run anomaly detection

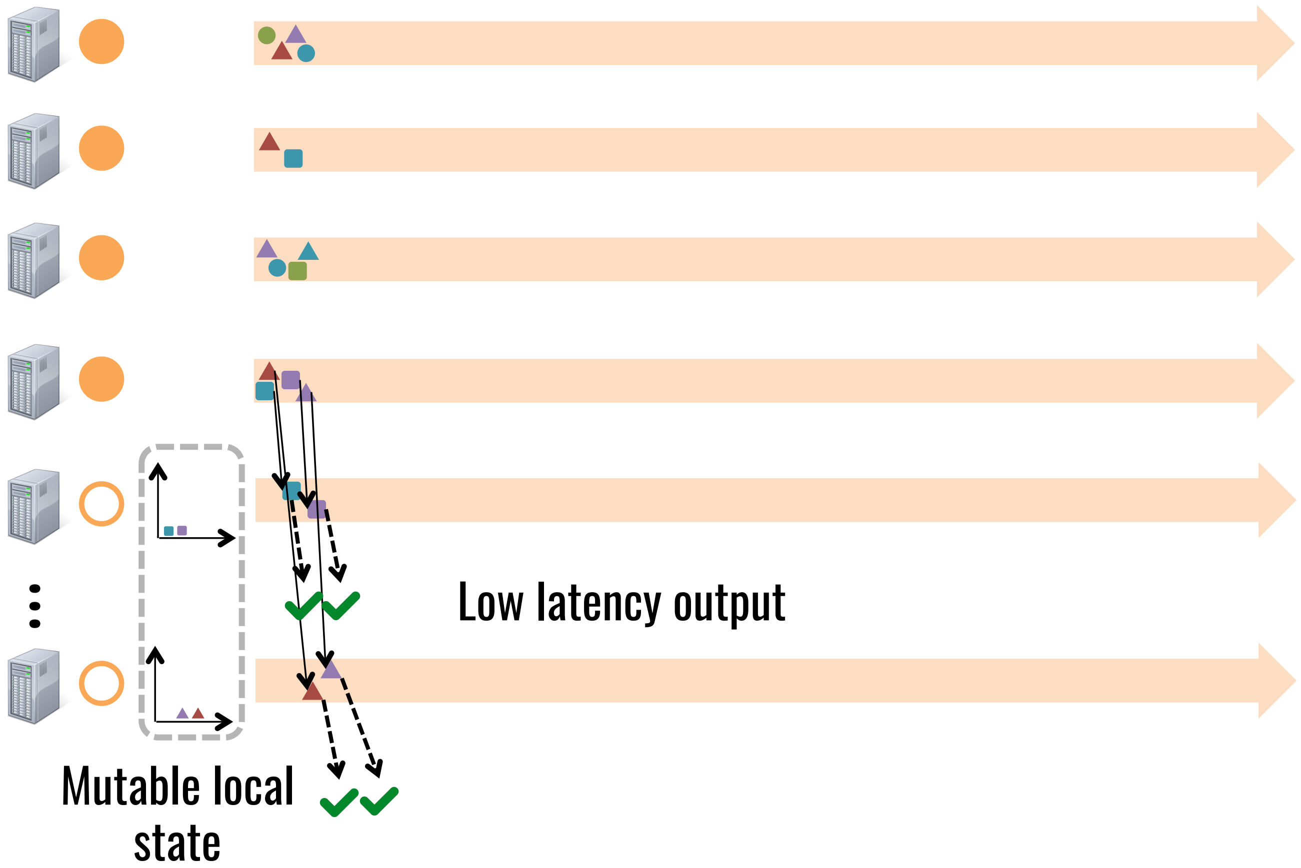




EXECUTION MODELS: CONTINUOUS OPERATORS

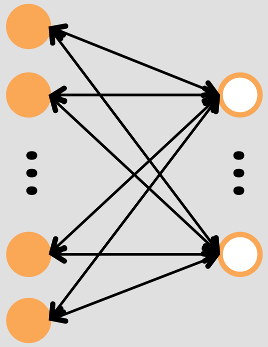
Group by user,
run anomaly detection

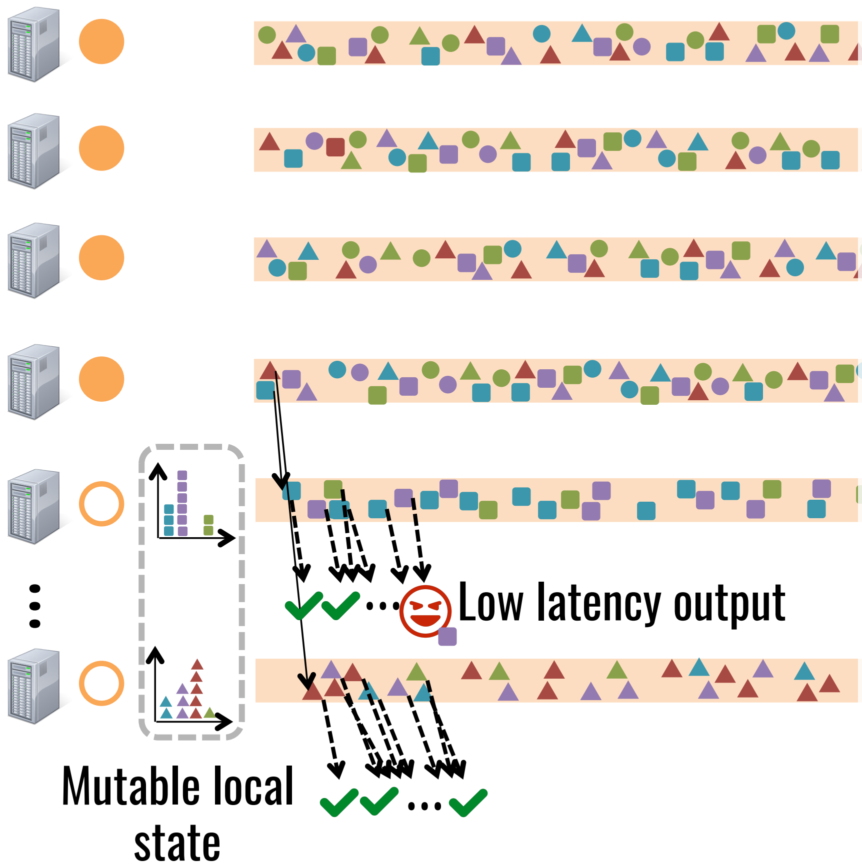




EXECUTION MODELS: CONTINUOUS OPERATORS

Group by user,
run anomaly detection





Systems:

Naiad

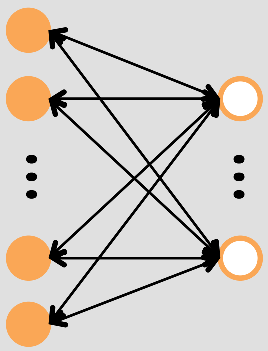
Flink

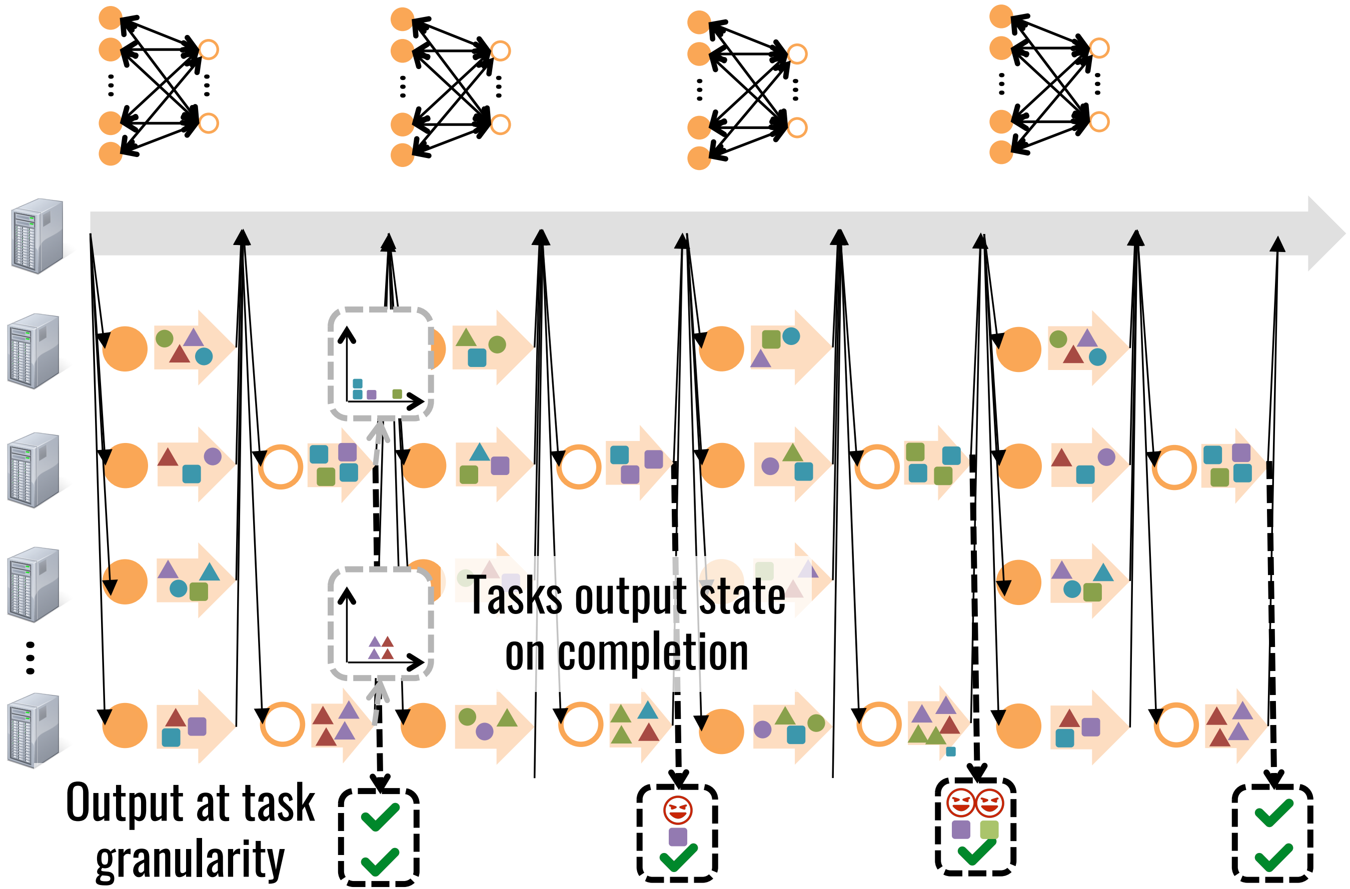
Google MillWheel

Streaming DBs:
Borealis, Flux etc

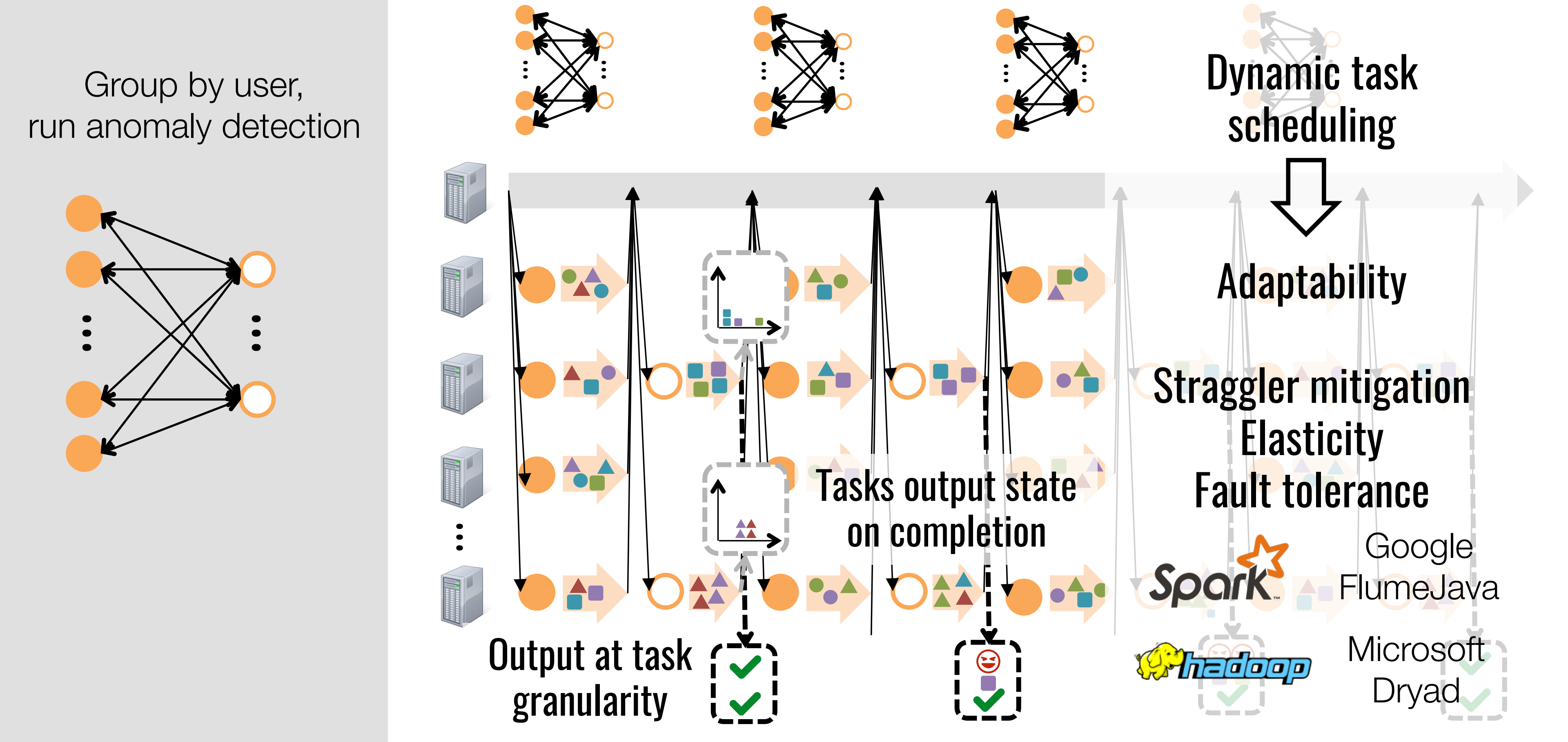
EXECUTION MODELS: MICRO-BATCH

Group by user,
run anomaly detection



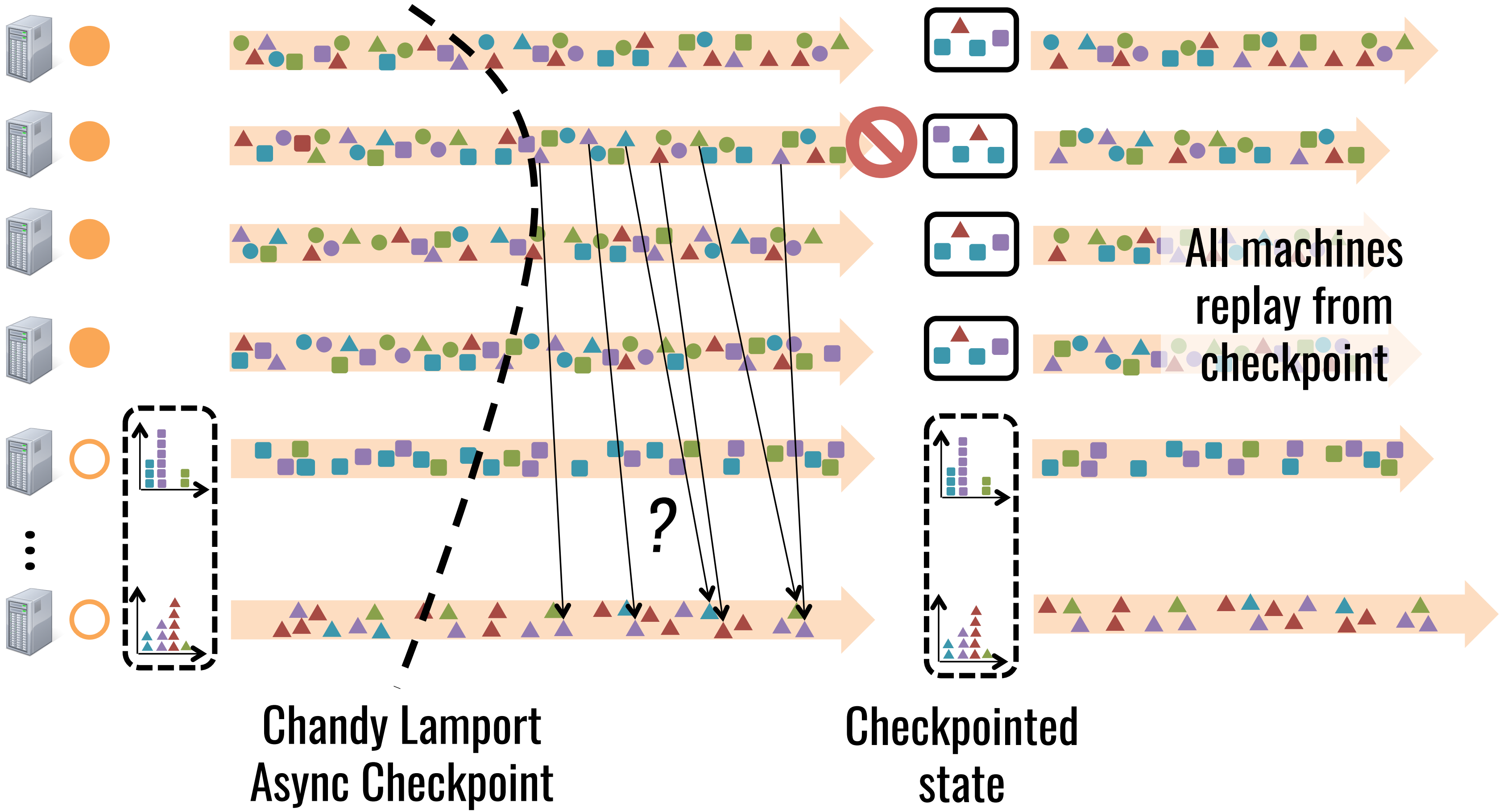


EXECUTION MODELS: MICRO-BATCH

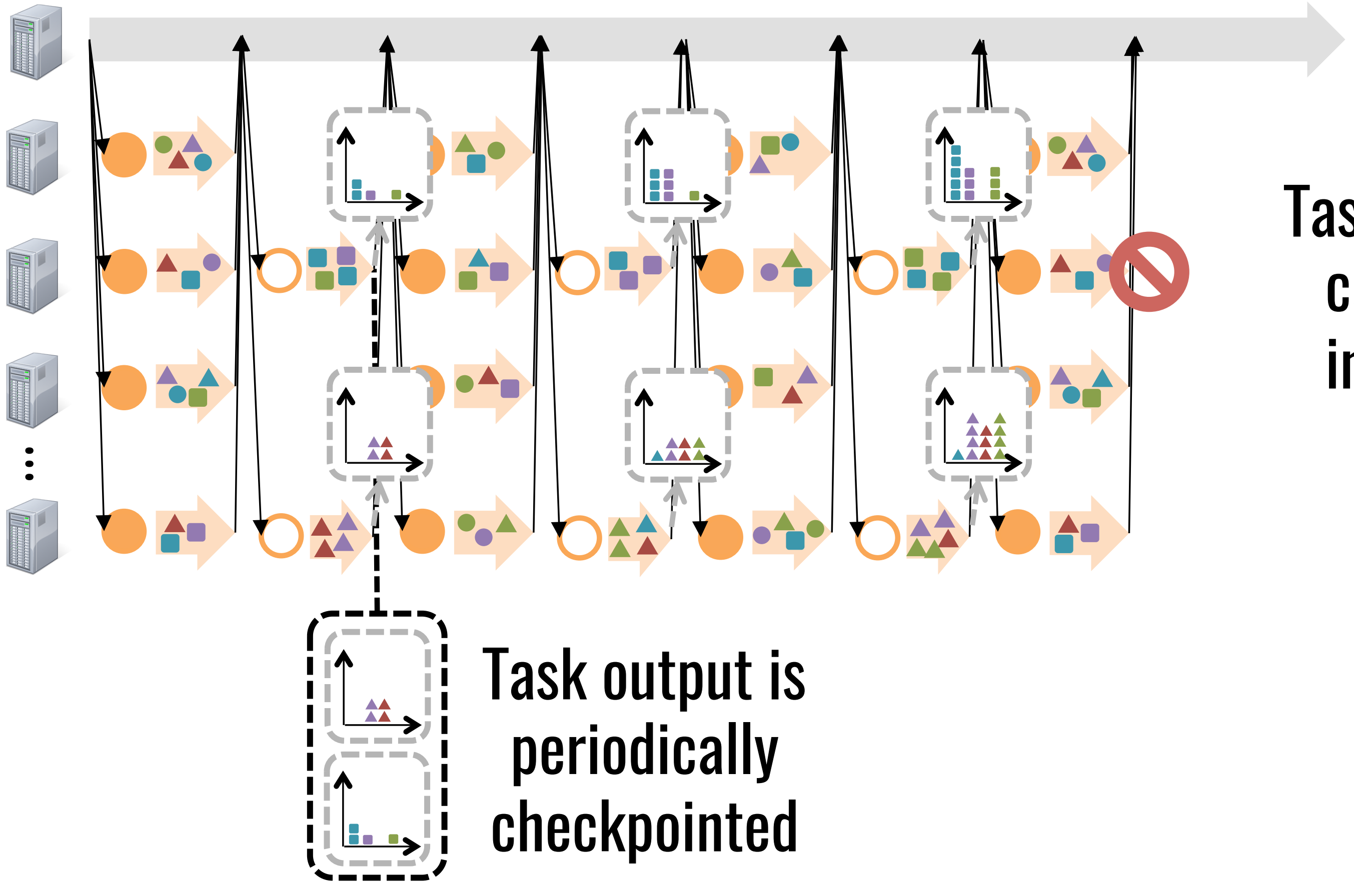


FAILURE RECOVERY

FAILURE RECOVERY: CONTINUOUS OPERATORS



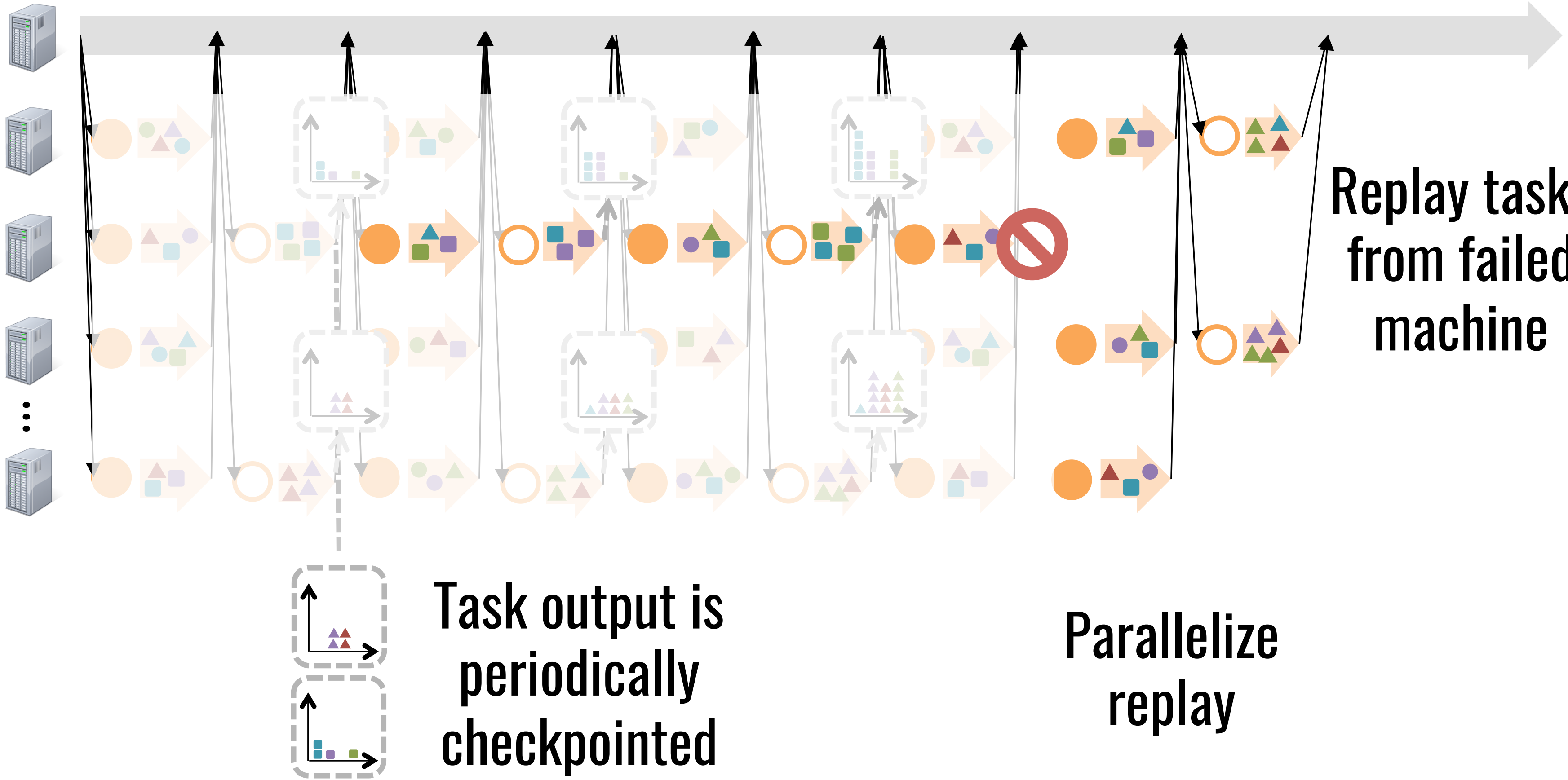
FAILURE RECOVERY: MICRO-BATCH



Task boundaries capture task interactions!

Task output is periodically checkpointed

FAILURE RECOVERY: MICRO-BATCH



Replay tasks from failed machine

Task output is periodically checkpointed

Parallelize replay

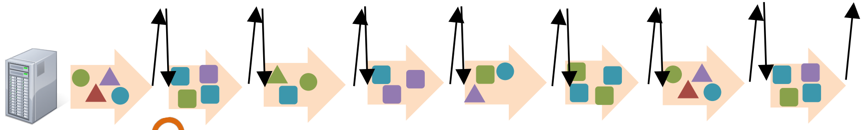
EXECUTION MODELS

Continuous operators

Static scheduling
Inflexible
Slow failover

✔ Low latency

Micro-batch



Scheduling granularity

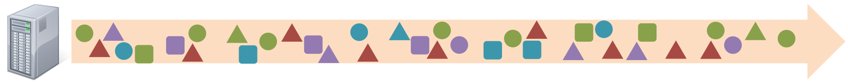
Processing granularity

✔ Dynamic scheduling
Adaptable
Parallel recovery
Straggler mitigation

Higher latency

EXECUTION MODELS

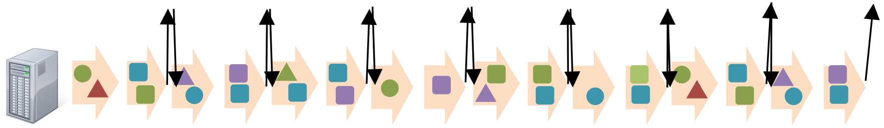
Continuous operators



Static scheduling

✔ Low latency

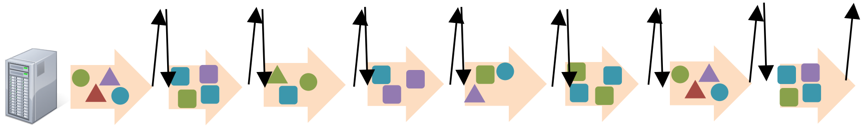
Drizzle



✔ Dynamic scheduling (coarse granularity)

✔ Low latency (fine-grained processing)

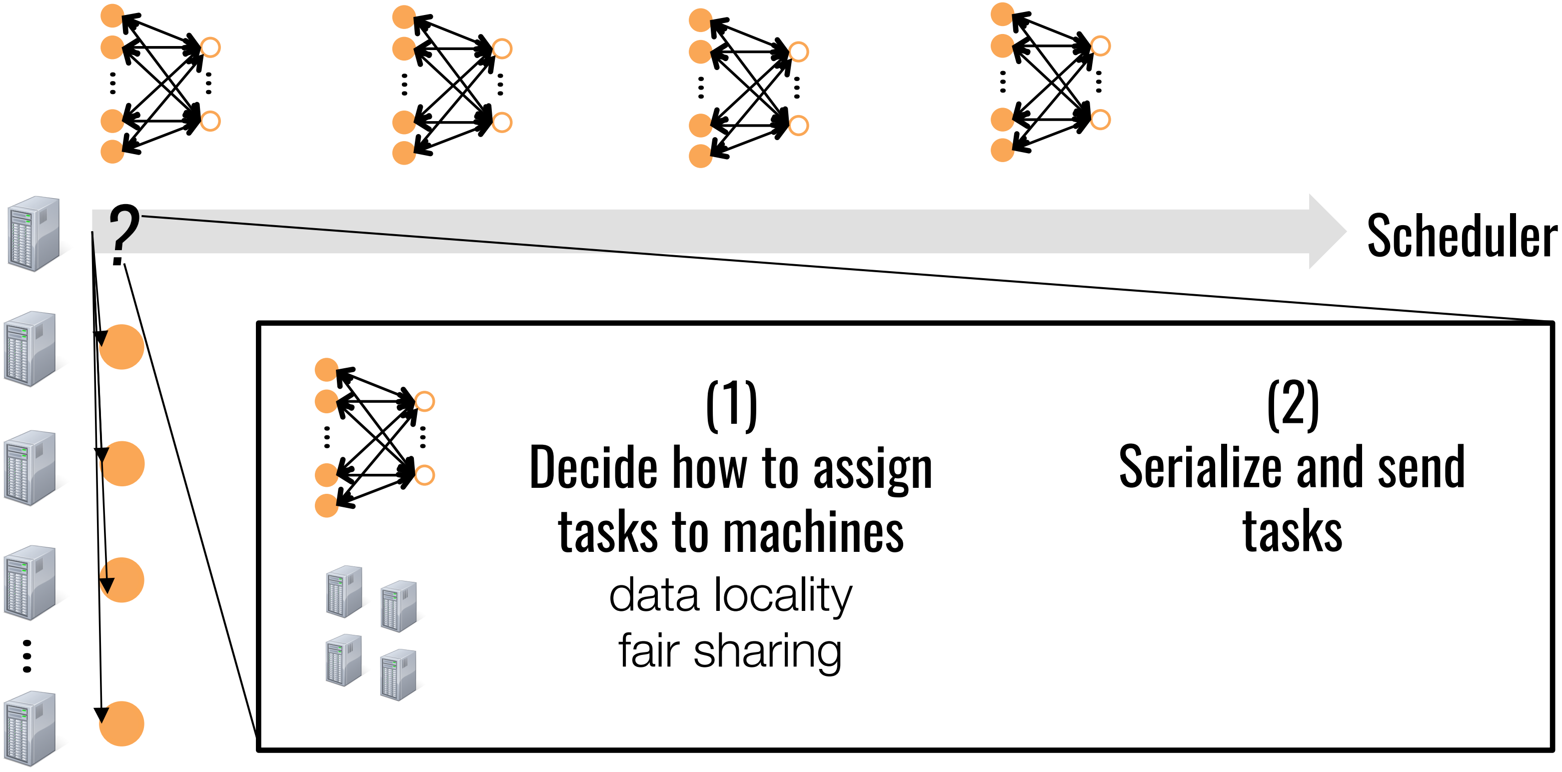
Micro-batch



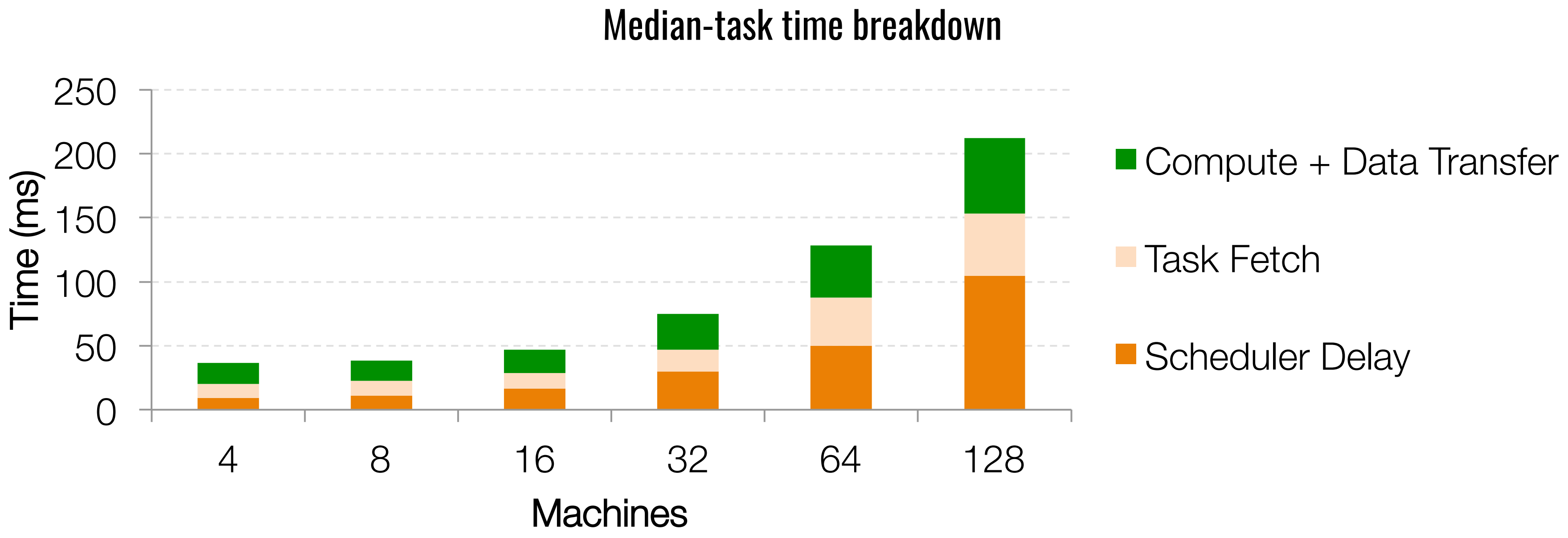
✔ Dynamic scheduling (coarse granularity)

Higher latency (coarse-grained processing)

INSIDE THE SCHEDULER



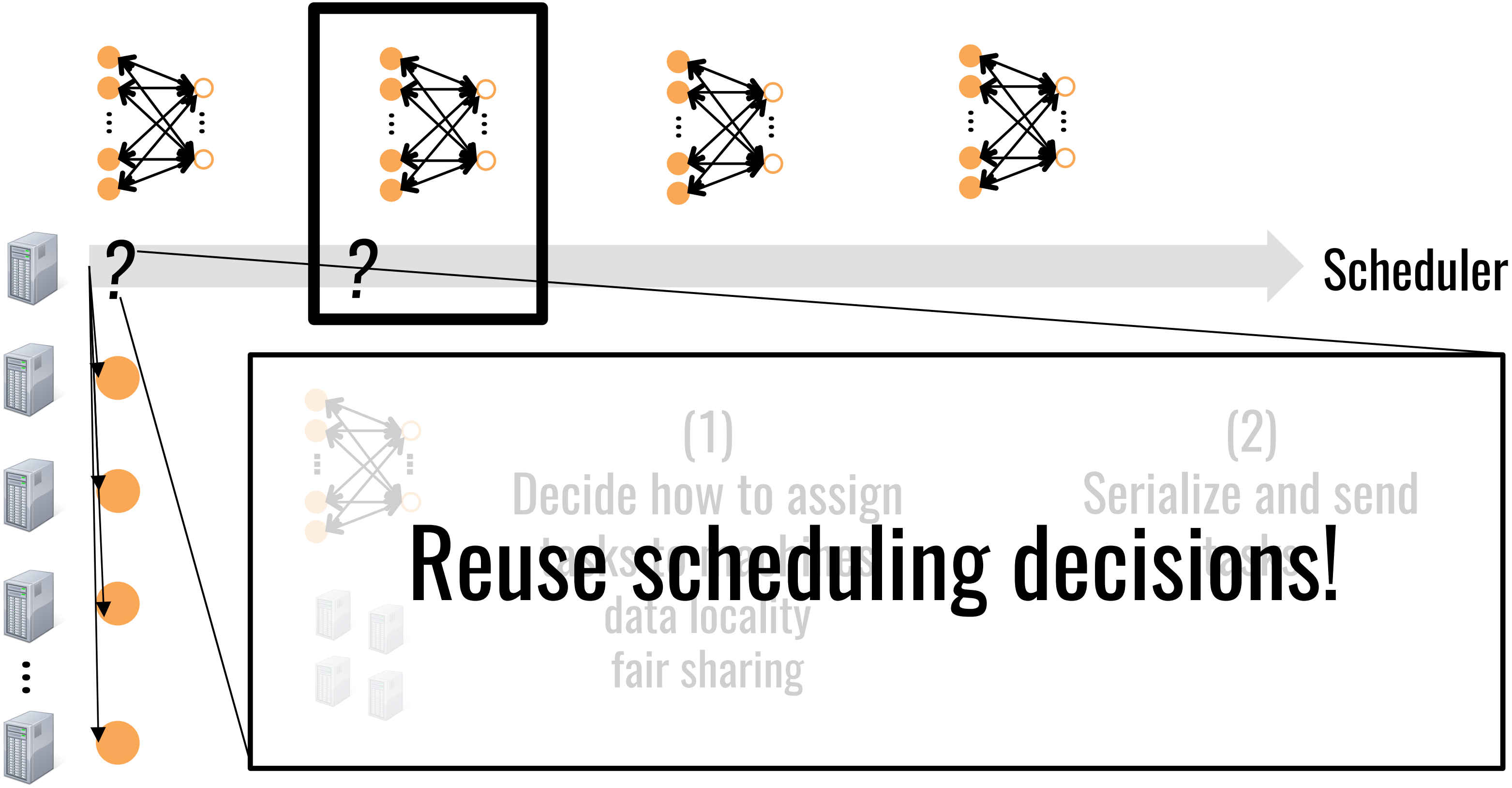
SCHEDULING OVERHEADS



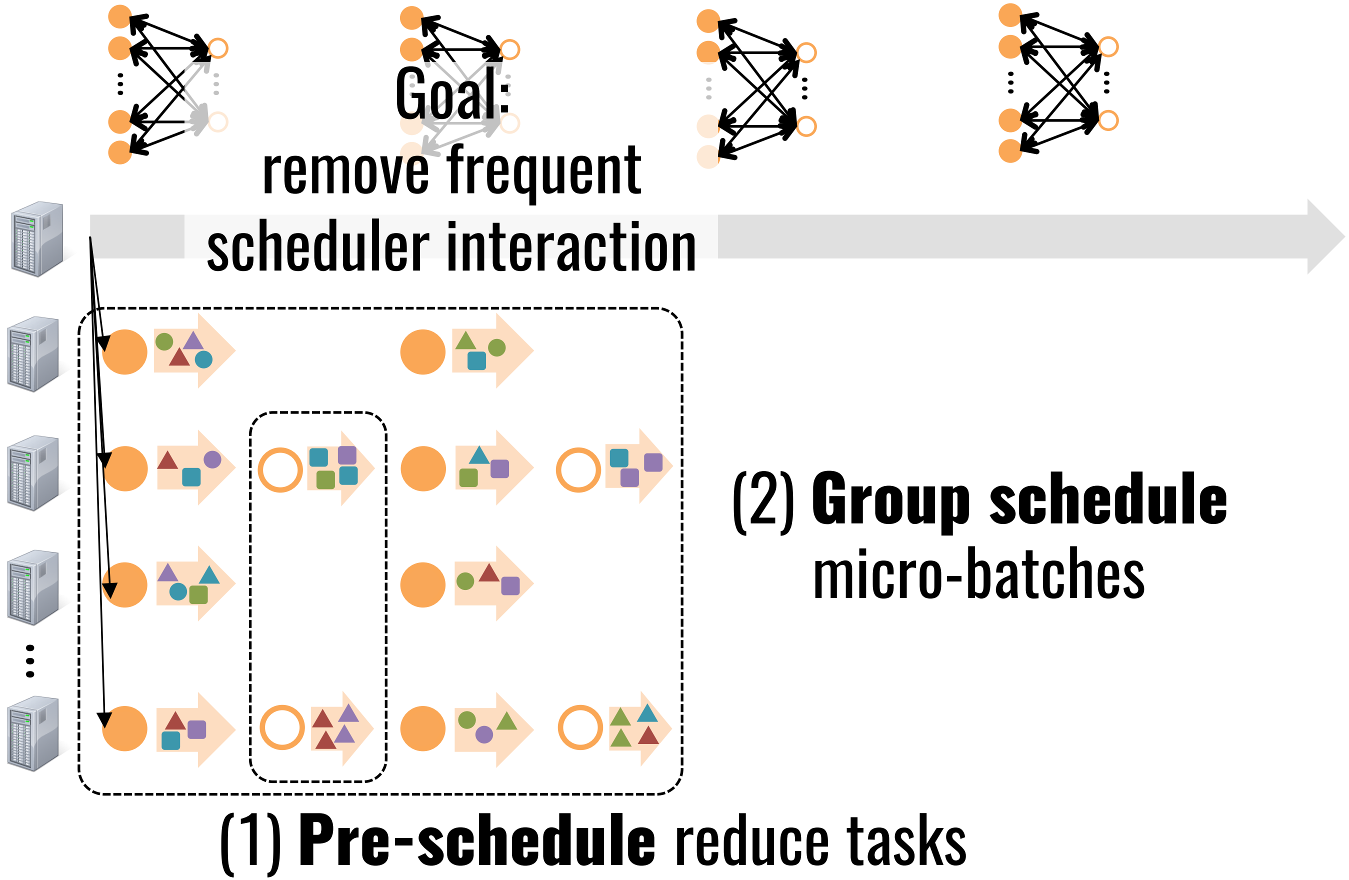
Cluster: 4 core, r3.xlarge machines

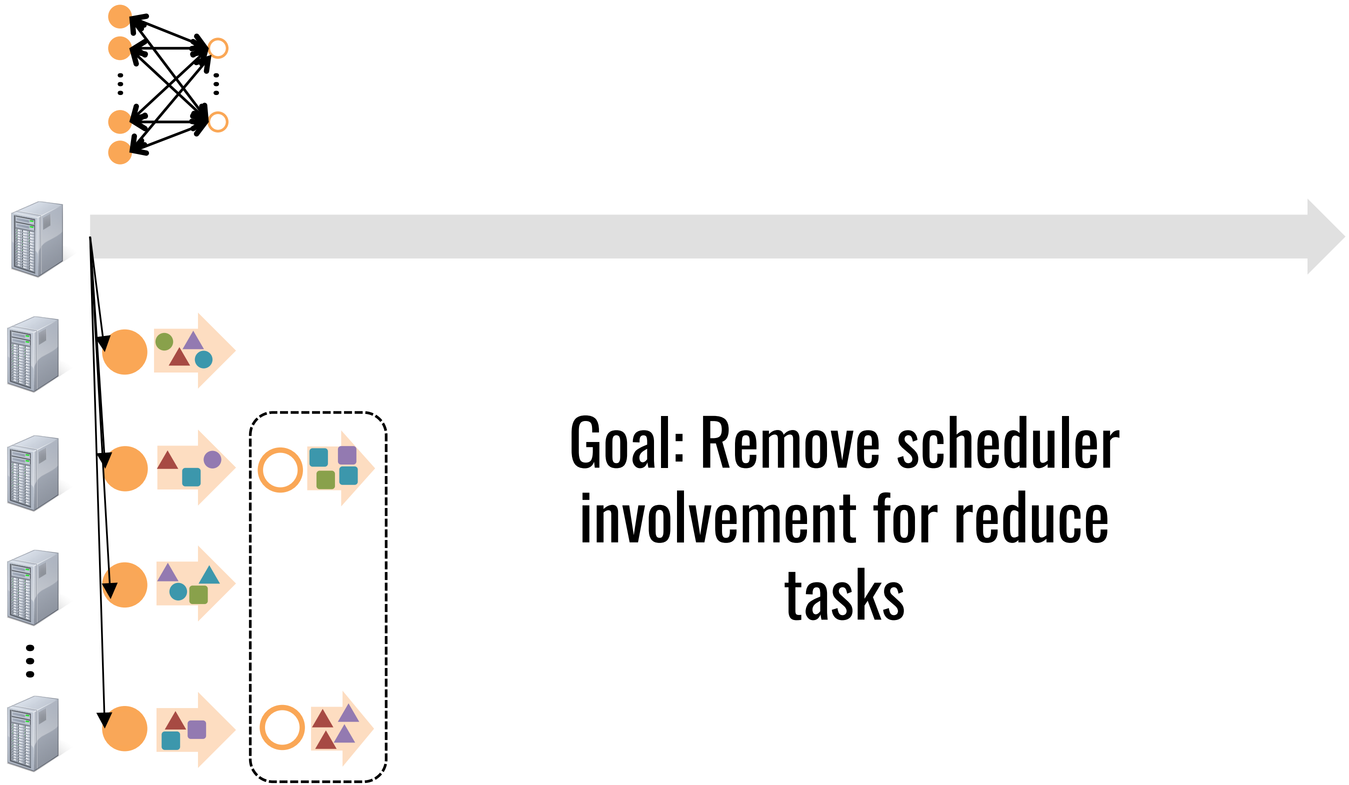
Workload: Sum of 10k numbers per-core

INSIDE THE SCHEDULER



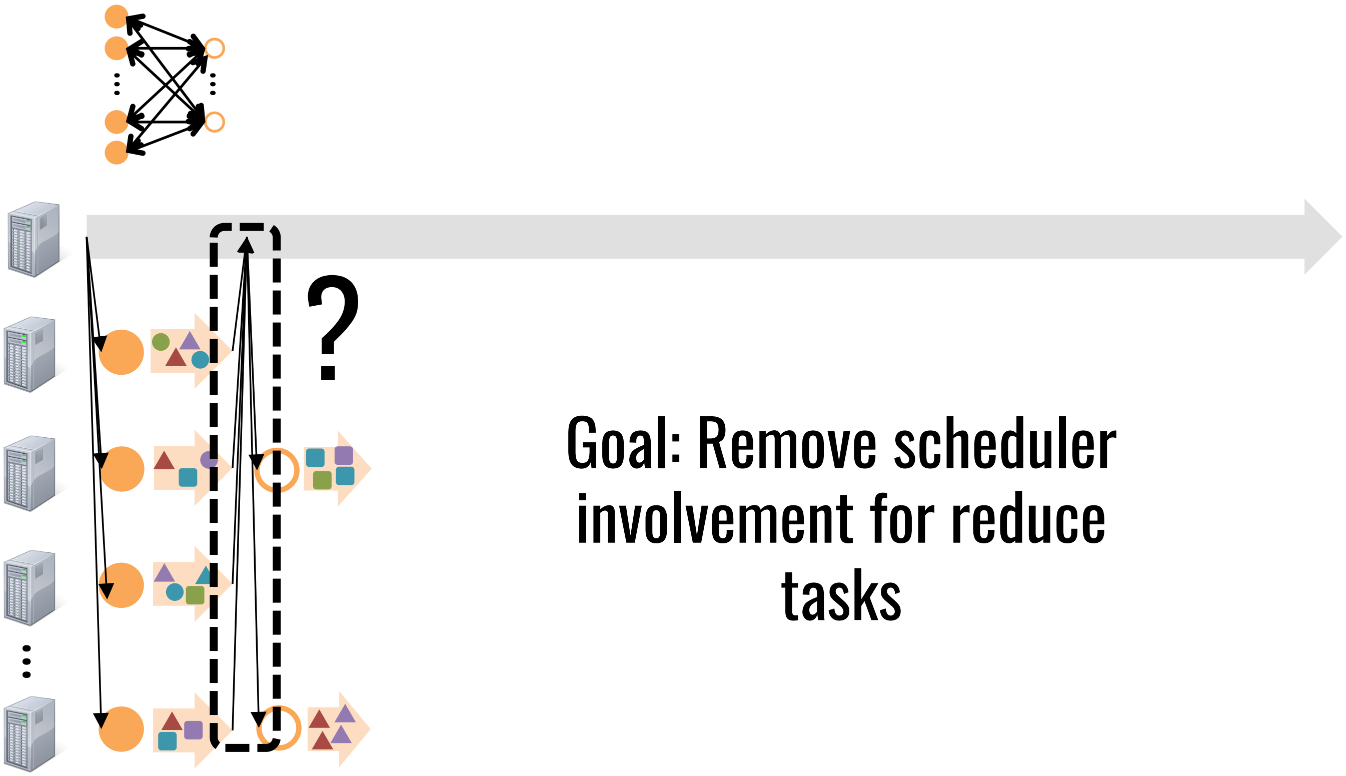
DRIZZLE


Goal:
remove frequent
scheduler interaction
(2) **Group schedule**
micro-batches
(1) **Pre-schedule** reduce tasks



Goal: Remove scheduler involvement for reduce tasks

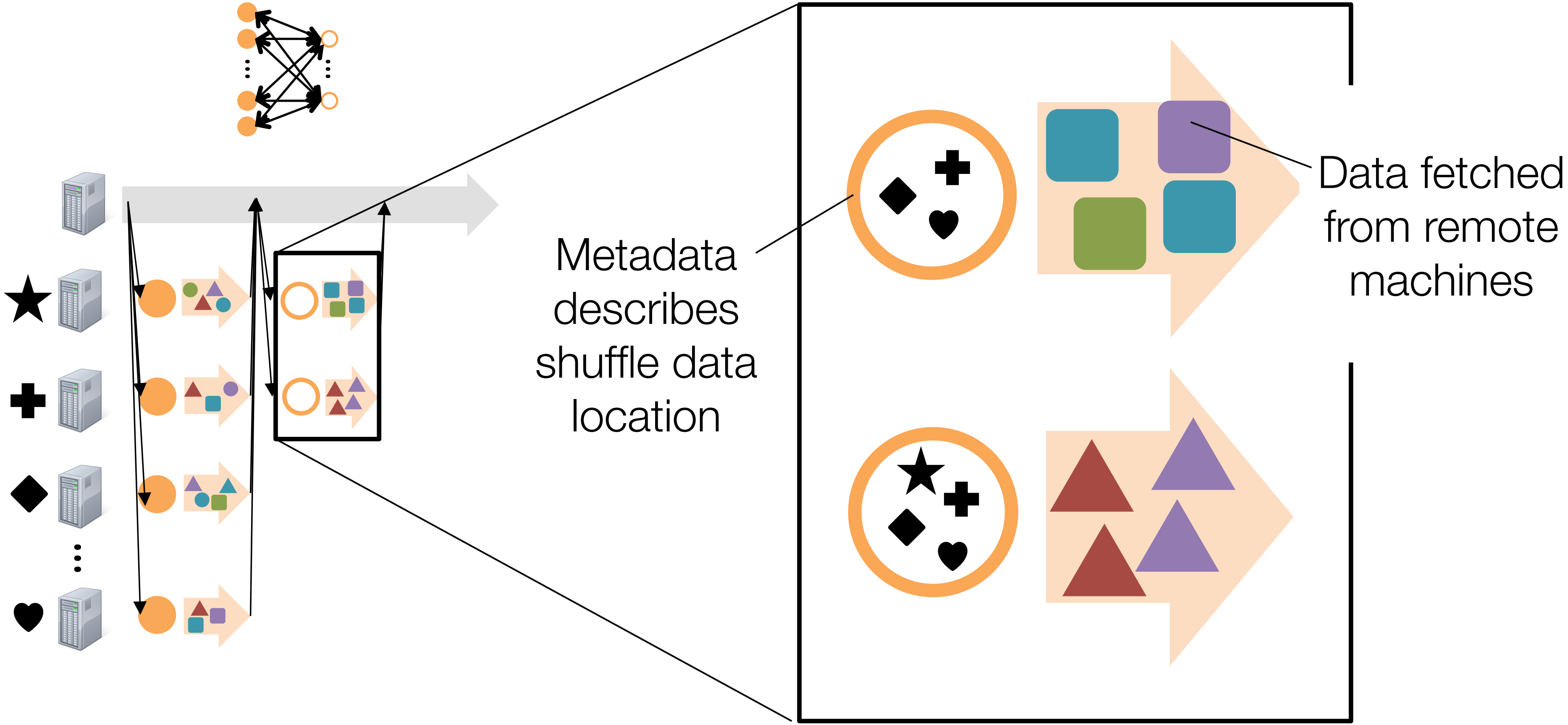
(1) **Pre-schedule** reduce tasks



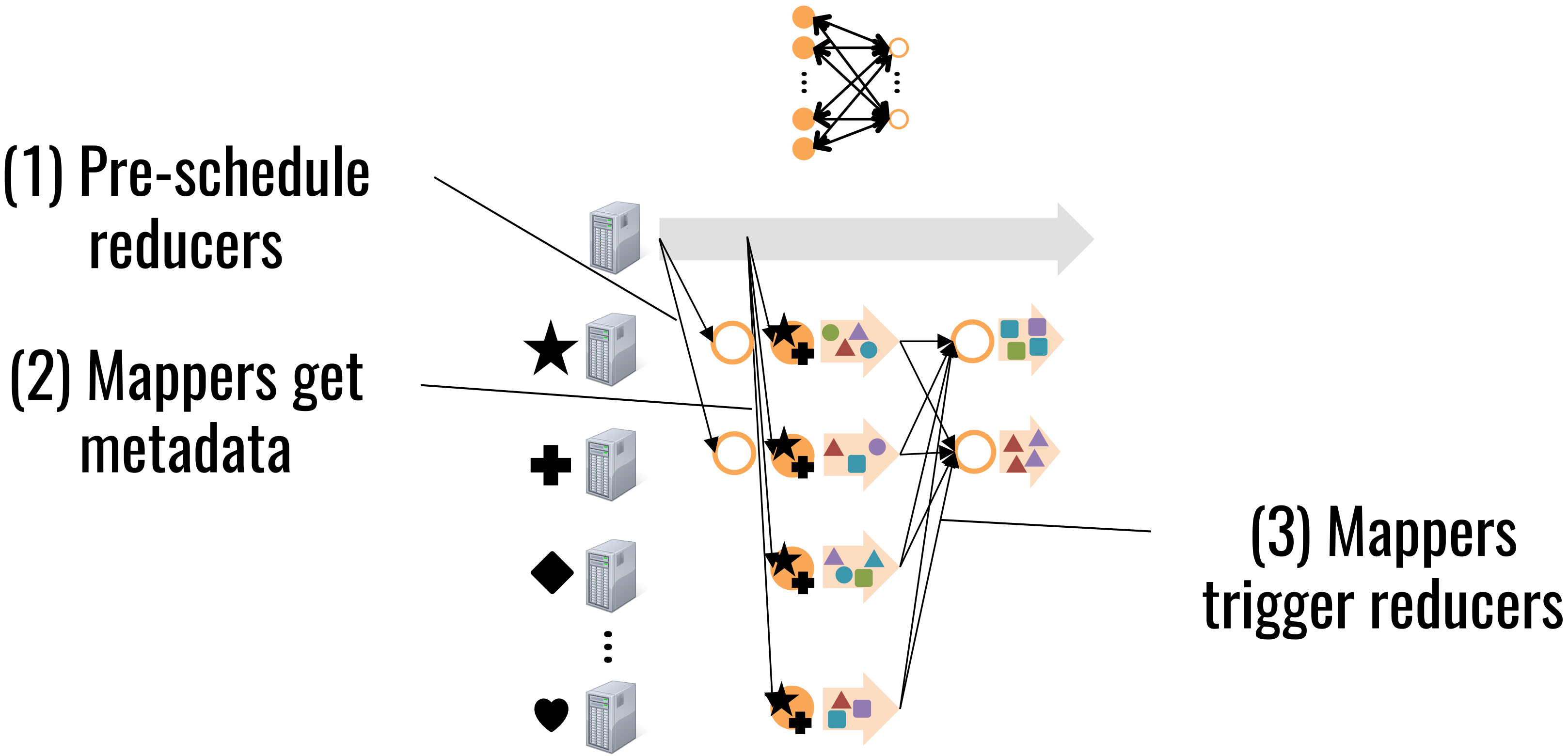
Goal: Remove scheduler involvement for reduce tasks

(1) **Pre-schedule** reduce tasks

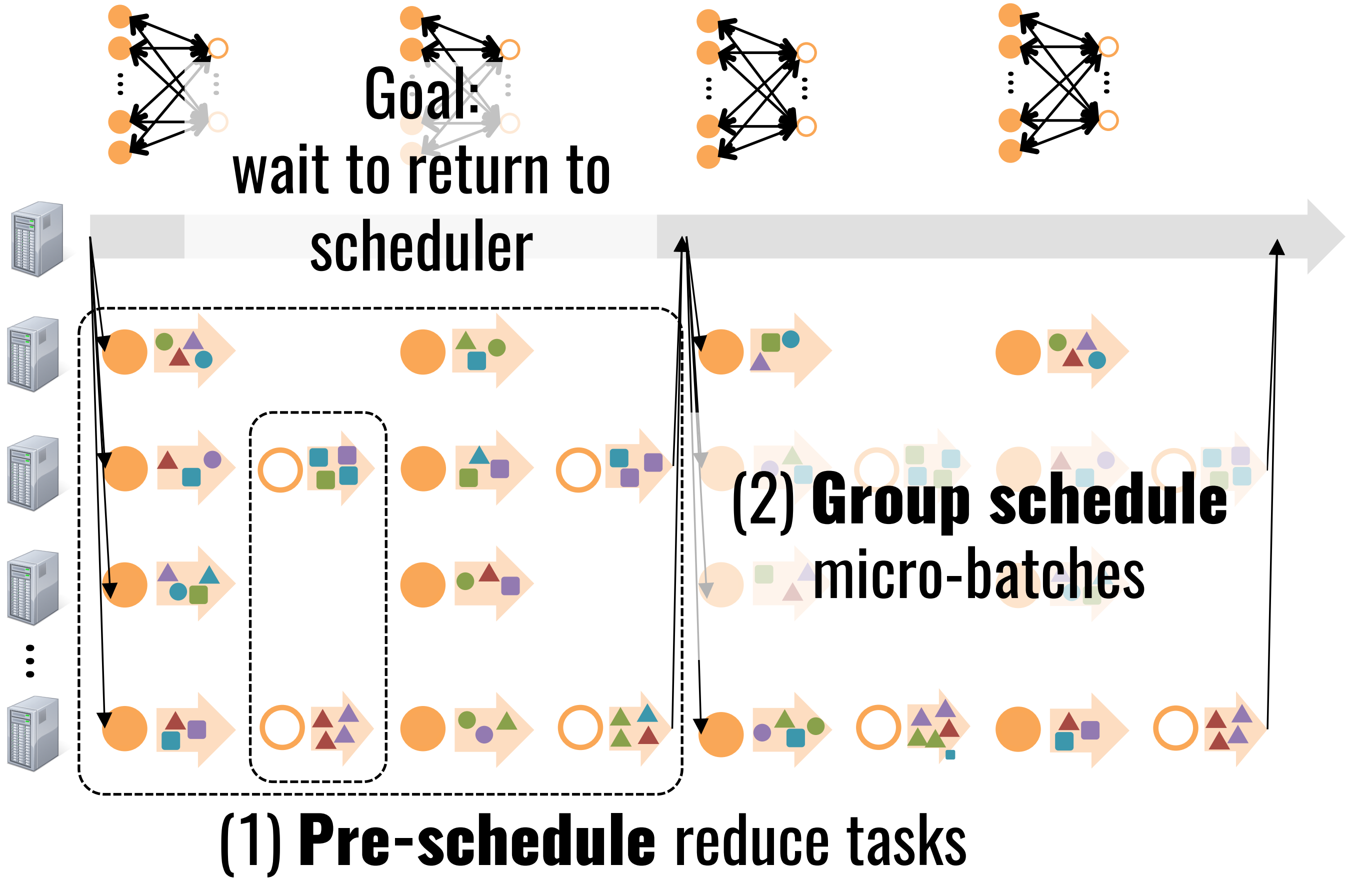
COORDINATING SHUFFLES: EXISTING SYSTEMS



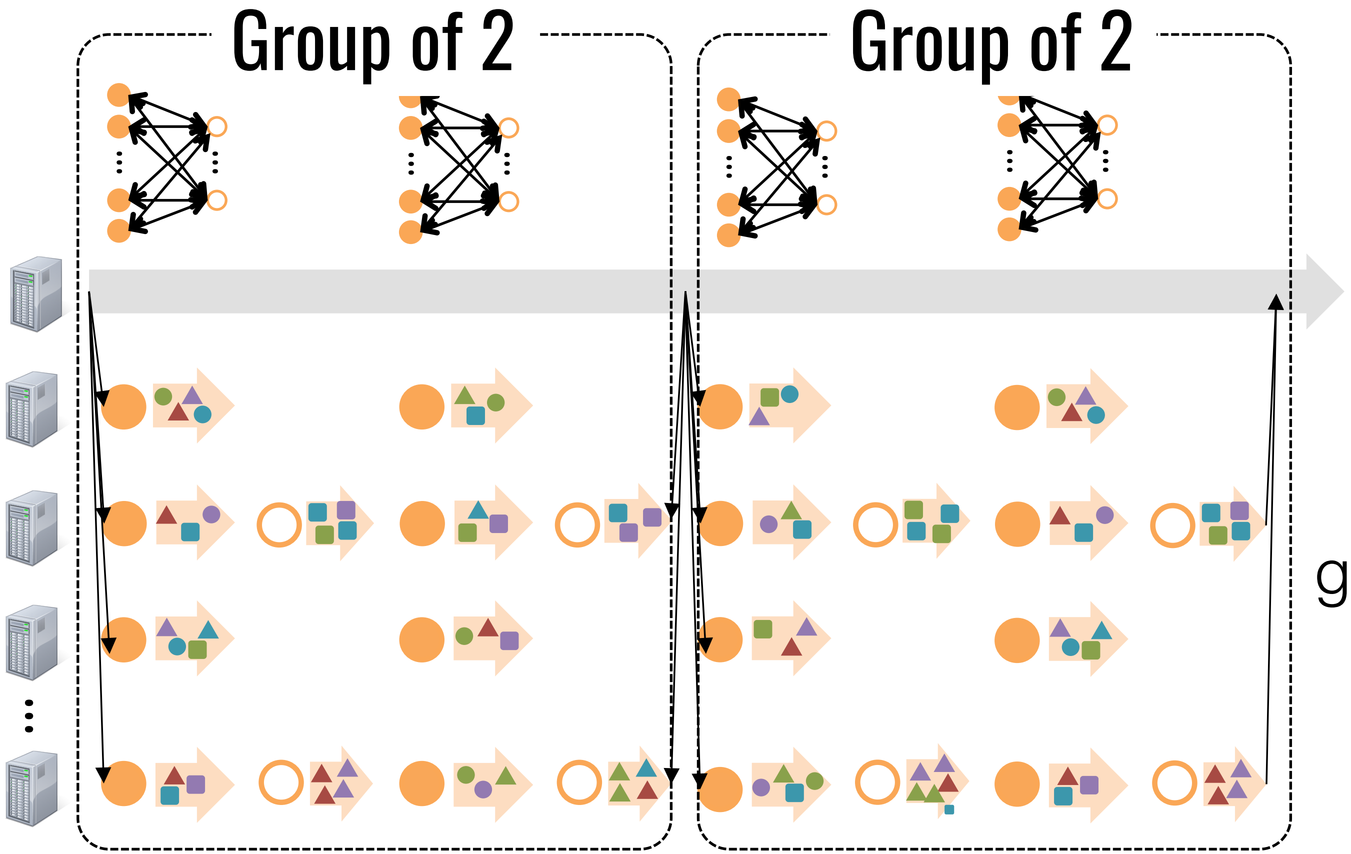
COORDINATING SHUFFLES: PRE-SCHEDULING



DRIZZLE


Goal:
wait to return to
scheduler
(2) **Group schedule**
micro-batches
(1) **Pre-schedule** reduce tasks

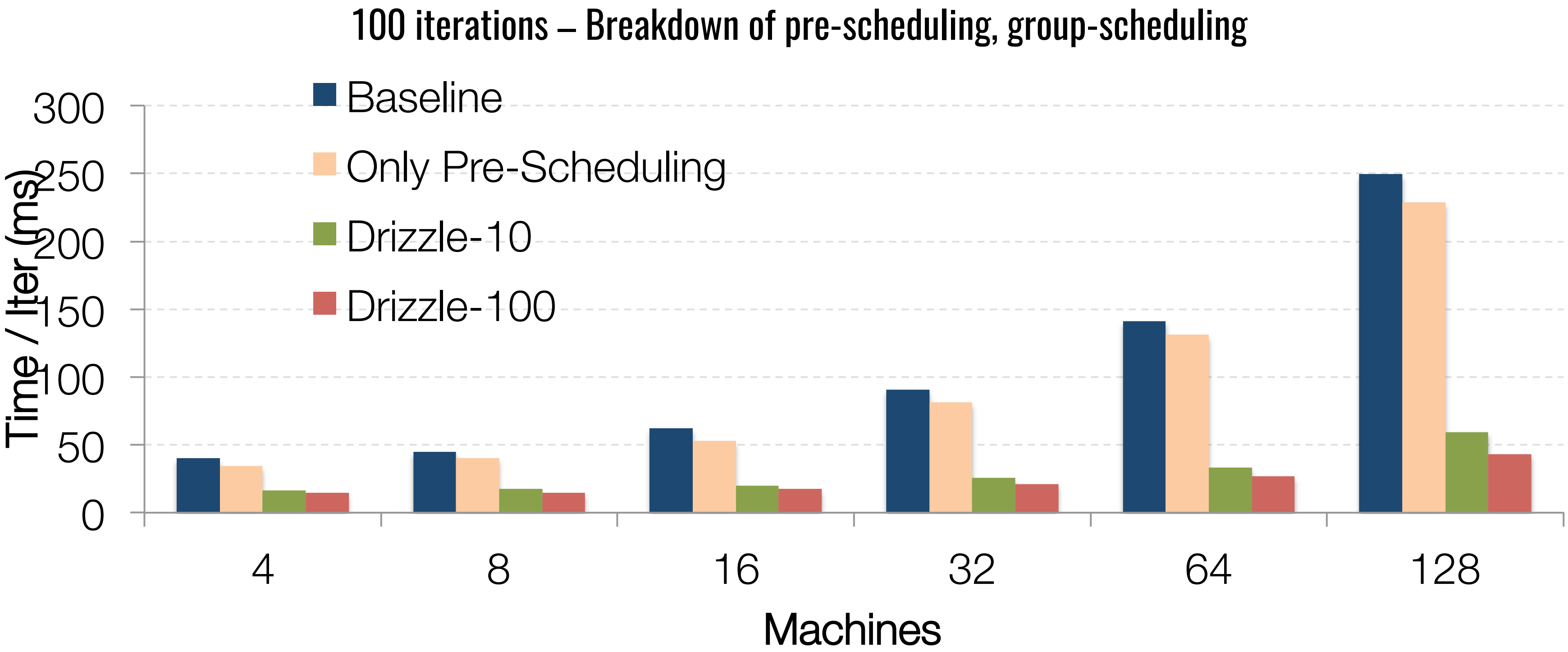
GROUP SCHEDULING



Schedule **group** of micro-batches at once

Fault tolerance, scheduling at group boundaries

MICRO-BENCHMARK: 2-STAGES



In the paper: group size auto-tuning

EVALUATION

1. Latency?

Continuous operators

Static scheduling

✓ Low latency

Drizzle

✓ Dynamic scheduling (coarse granularity)

✓ Low latency (fine-grained processing)

Micro-batch

✓ Dynamic scheduling (coarse granularity)

2. Adaptability?

Higher latency (coarse-grained processing)

EVALUATION: LATENCY

Yahoo! Streaming Benchmark

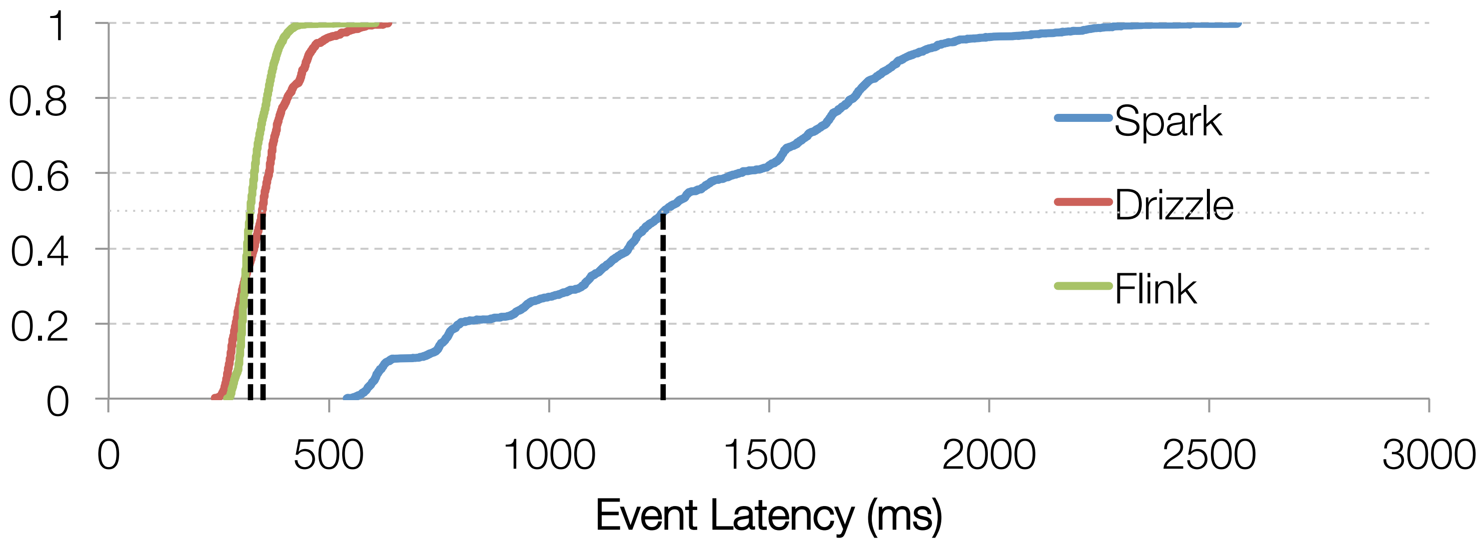
Input: JSON events of ad-clicks
Compute: Number of clicks per campaign
Window: Update every 10s

Comparing Spark 2.0, Flink 1.1.1, Drizzle
128 Amazon EC2 r3.xlarge instances

STREAMING BENCHMARK - PERFORMANCE

Yahoo Streaming Benchmark: 20M JSON Ad-events / second, 128 machines

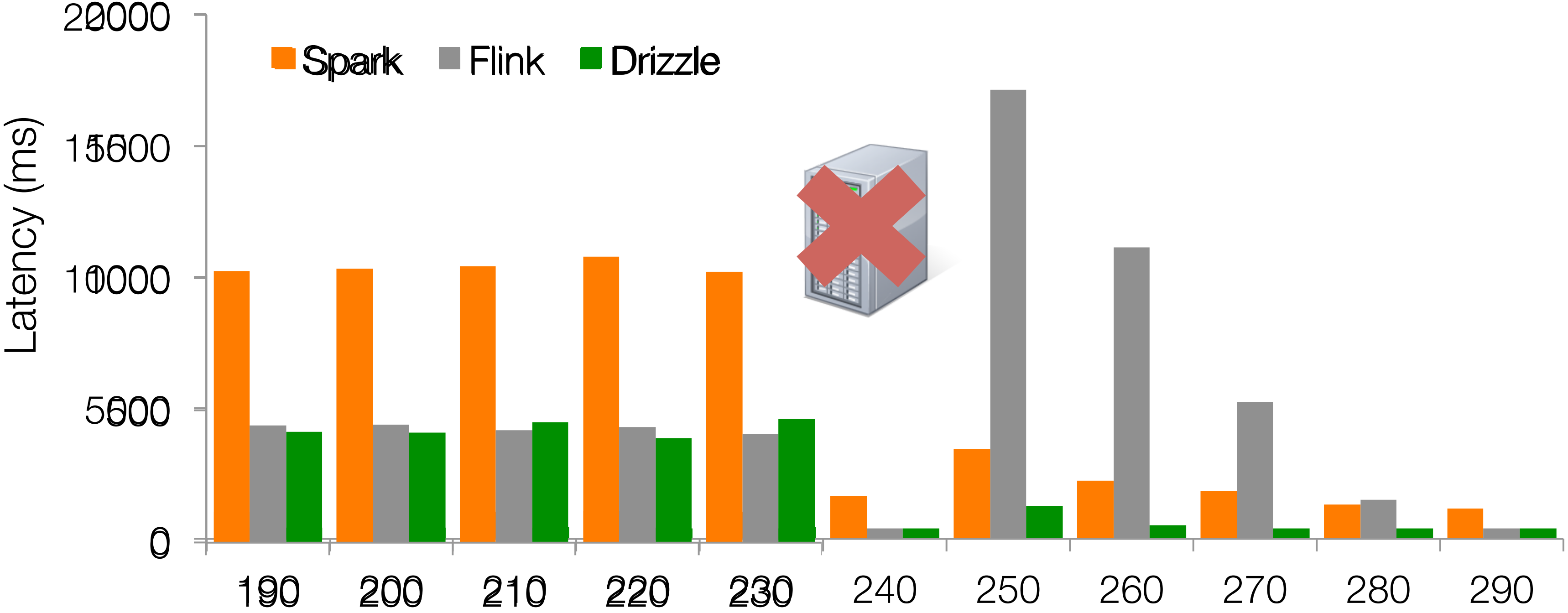
Event Latency: Difference between window end, processing end



ADAPTABILITY: FAULT TOLERANCE

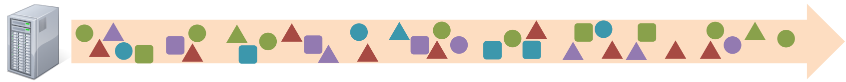
Yahoo Streaming Benchmark: 20M JSON Ad-events / second, 128 machines

Inject machine failure at 240 seconds



EXECUTION MODELS

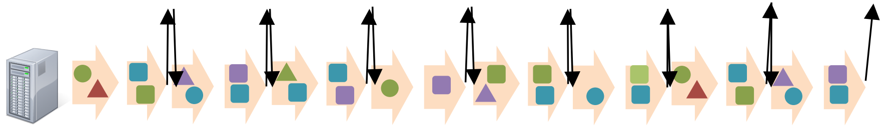
Continuous operators



Static scheduling

✔ Low latency

Drizzle

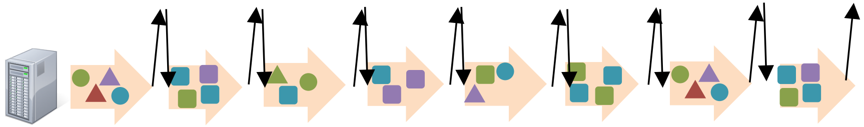


✔ Dynamic scheduling (coarse-granularity)

✔ Low latency (fine-grained processing)

✔ Optimization of batches

Micro-batch



✔ Dynamic scheduling

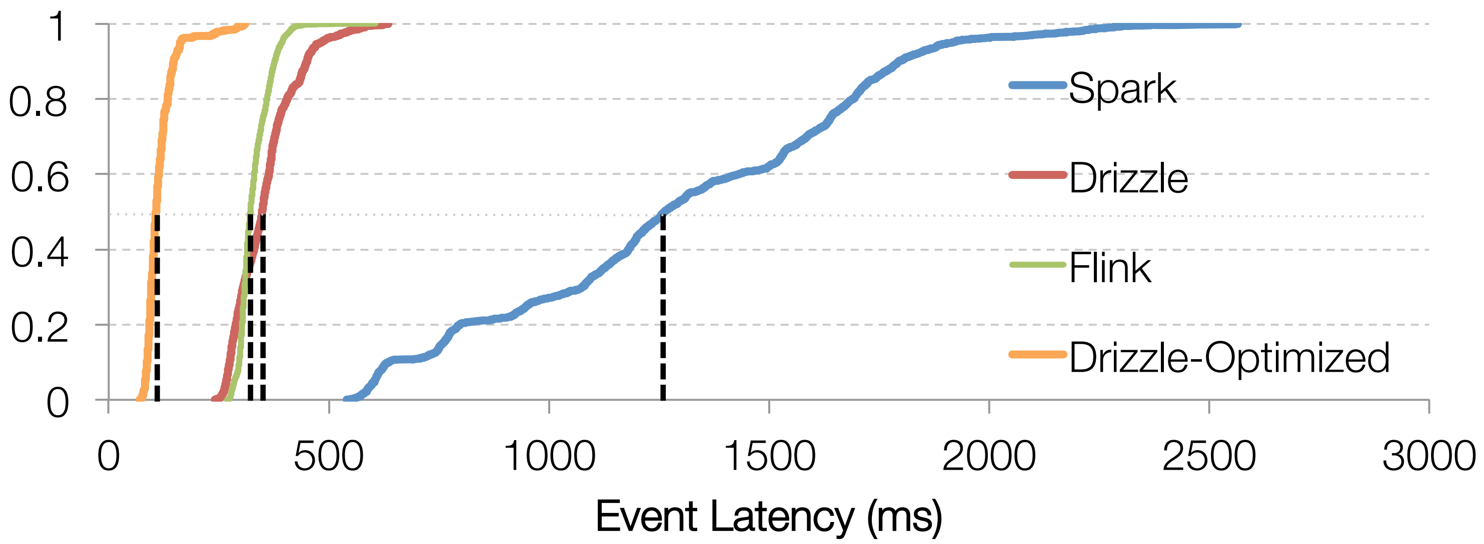
Higher latency

✔ Optimization of batches

INTRA-BATCH QUERY OPTIMIZATION

Yahoo Streaming Benchmark: 20M JSON Ad-events / second, 128 machines

Optimize execution of each micro-batch by pushing down aggregation



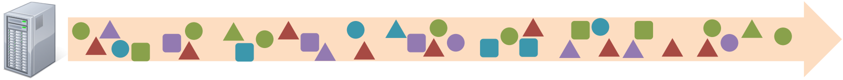
EVALUATION

End-to-end Latency Fault tolerance Query optimization	Yahoo Streaming Benchmark
Throughput Elasticity Group-size tuning	Synthetic micro-benchmarks Video Analytics Shivaram’s Thesis: Iterative ML Algorithms

CONCLUSION

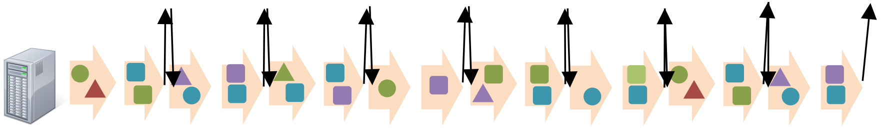
Shivaram is answering questions on sli.do

Continuous operators



Static scheduling ✔ Low latency

Drizzle



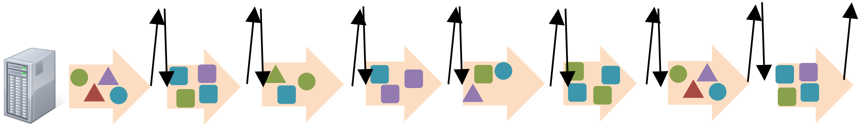
✔ Dynamic scheduling (coarse granularity)

✔ Low latency (fine-grained processing)

✔ Optimization of batches

Source code: https://github.com/amplab/drizzle-spark

Micro-batch



✔ Dynamic scheduling (coarse granularity)

Higher latency (coarse-grained processing)

✔ Optimization of batches