The Case for Tiny Tasks in Compute Clusters

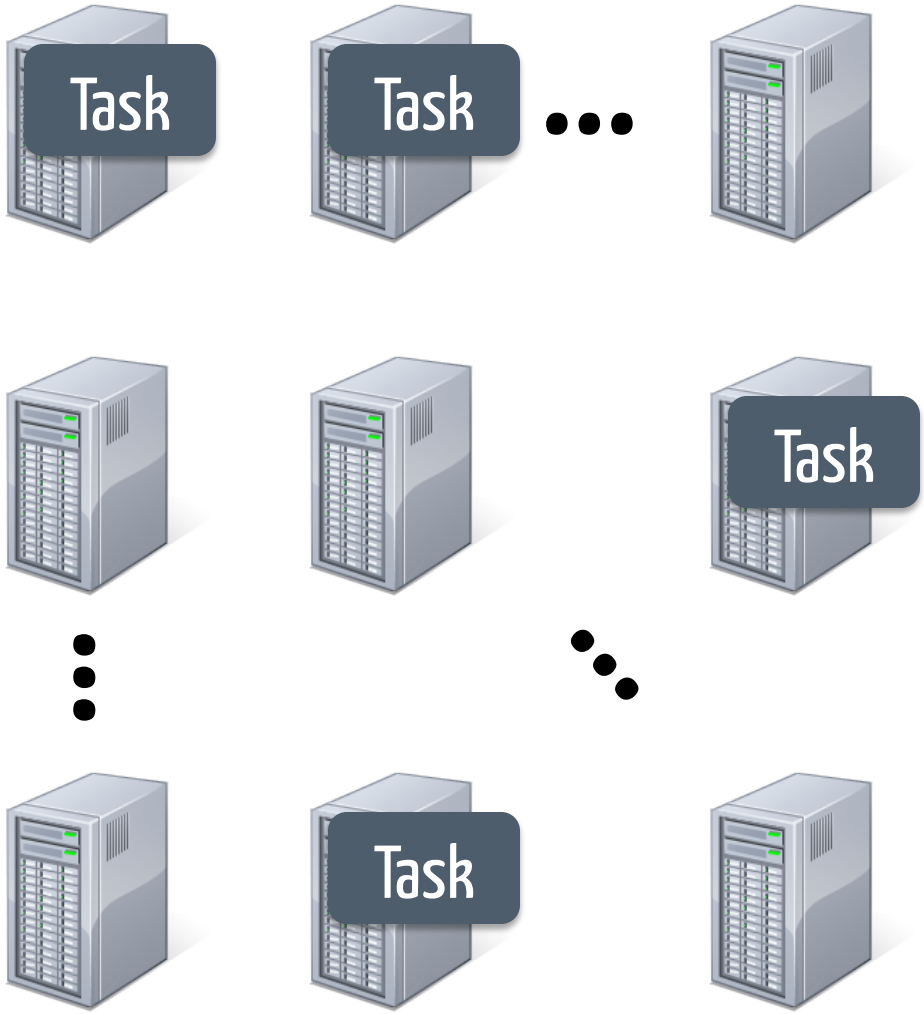
Kay Ousterhout*, Aurojit Panda*, Joshua Rosen*,
Shivaram Venkataraman*, Reynold Xin*,
Sylvia Ratnasamy*, Scott Shenker*+, Ion Stoica*

* UC Berkeley, + ICSI

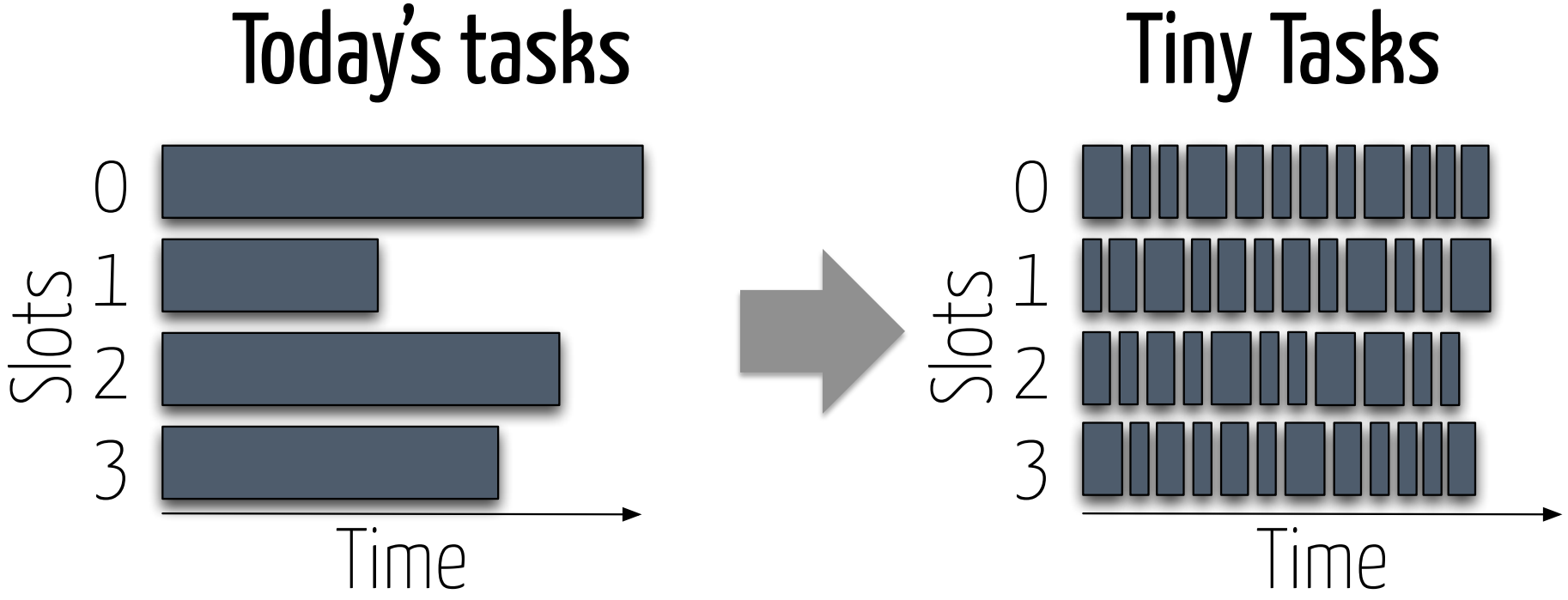
Setting

Map Reduce/Spark/Dryad

Job



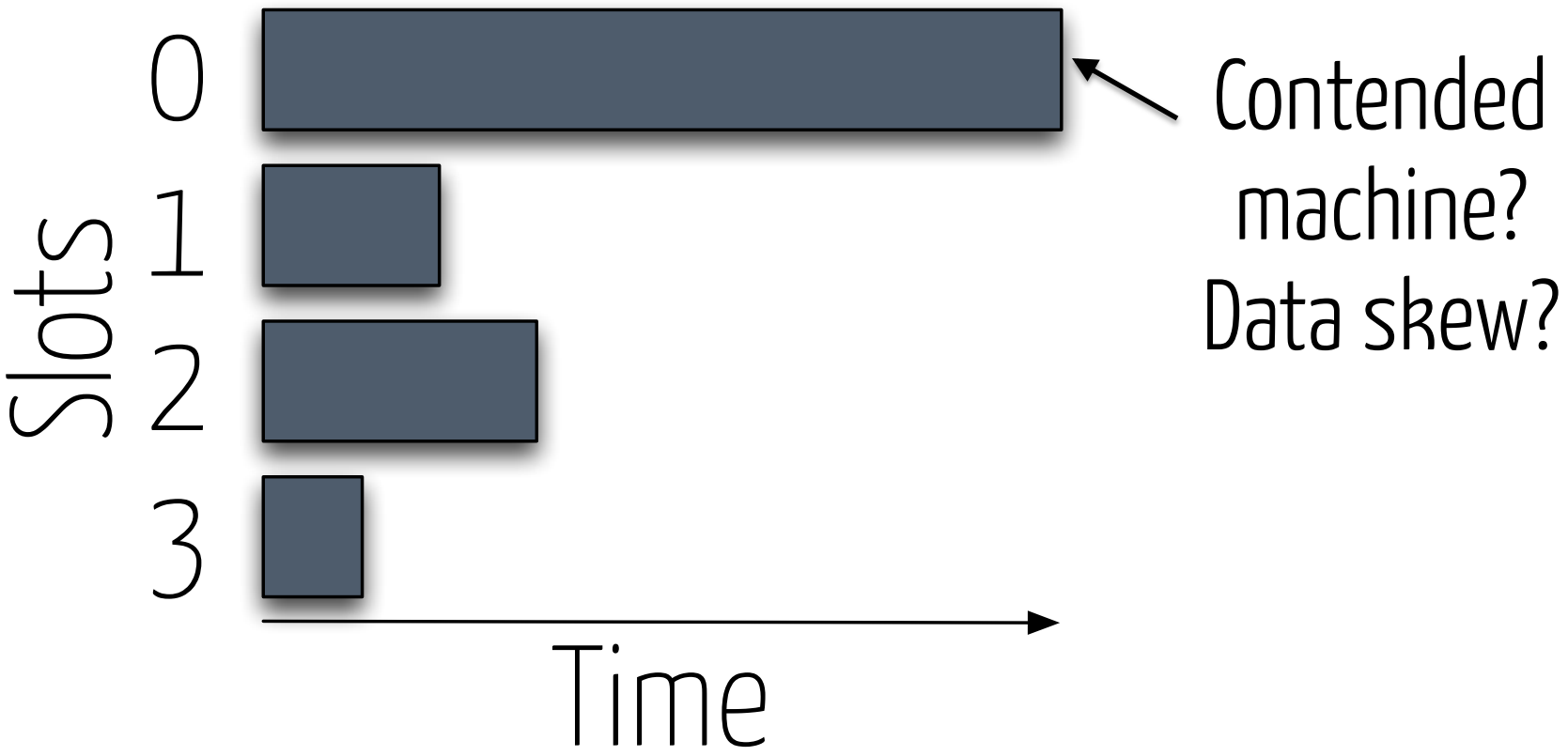
Use smaller tasks!



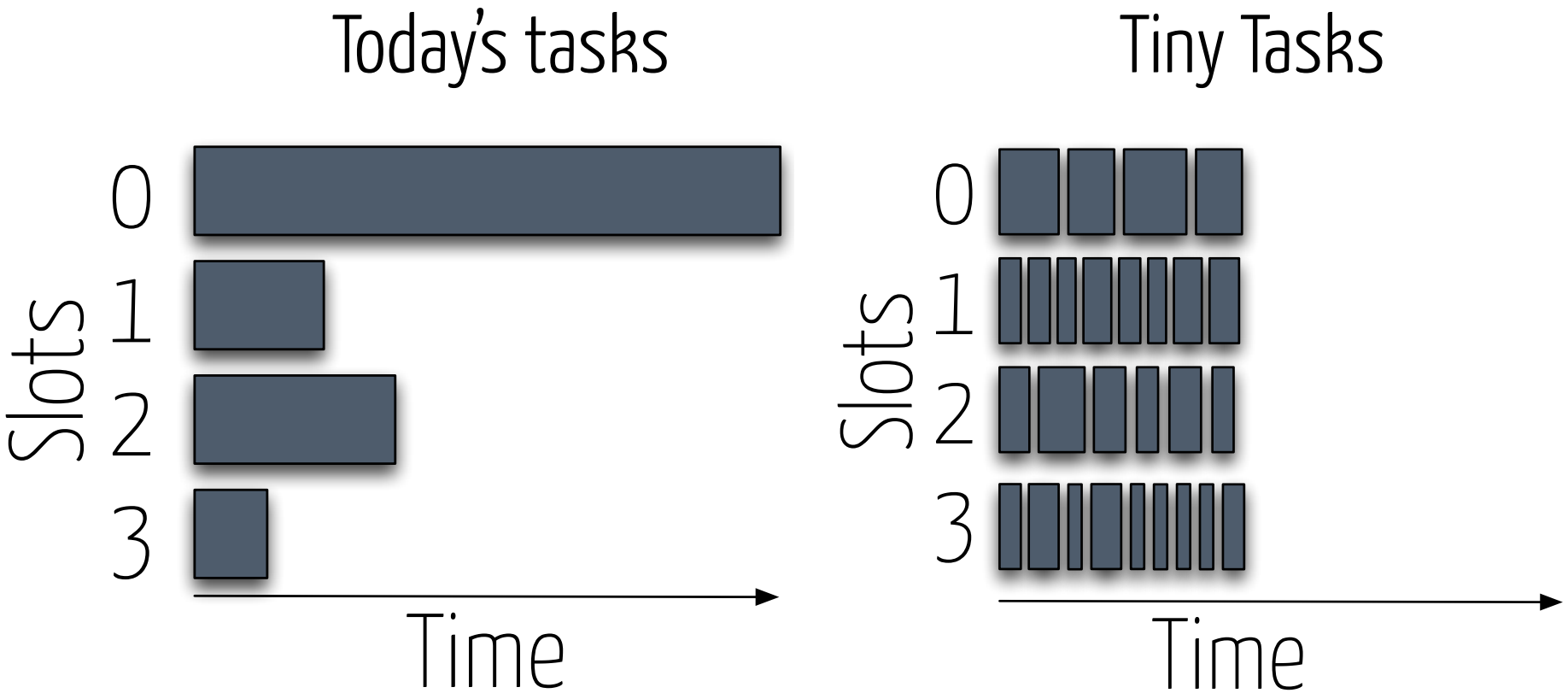
Why? How? Where?

Why? How? Where?

Problem: Skew and Stragglers



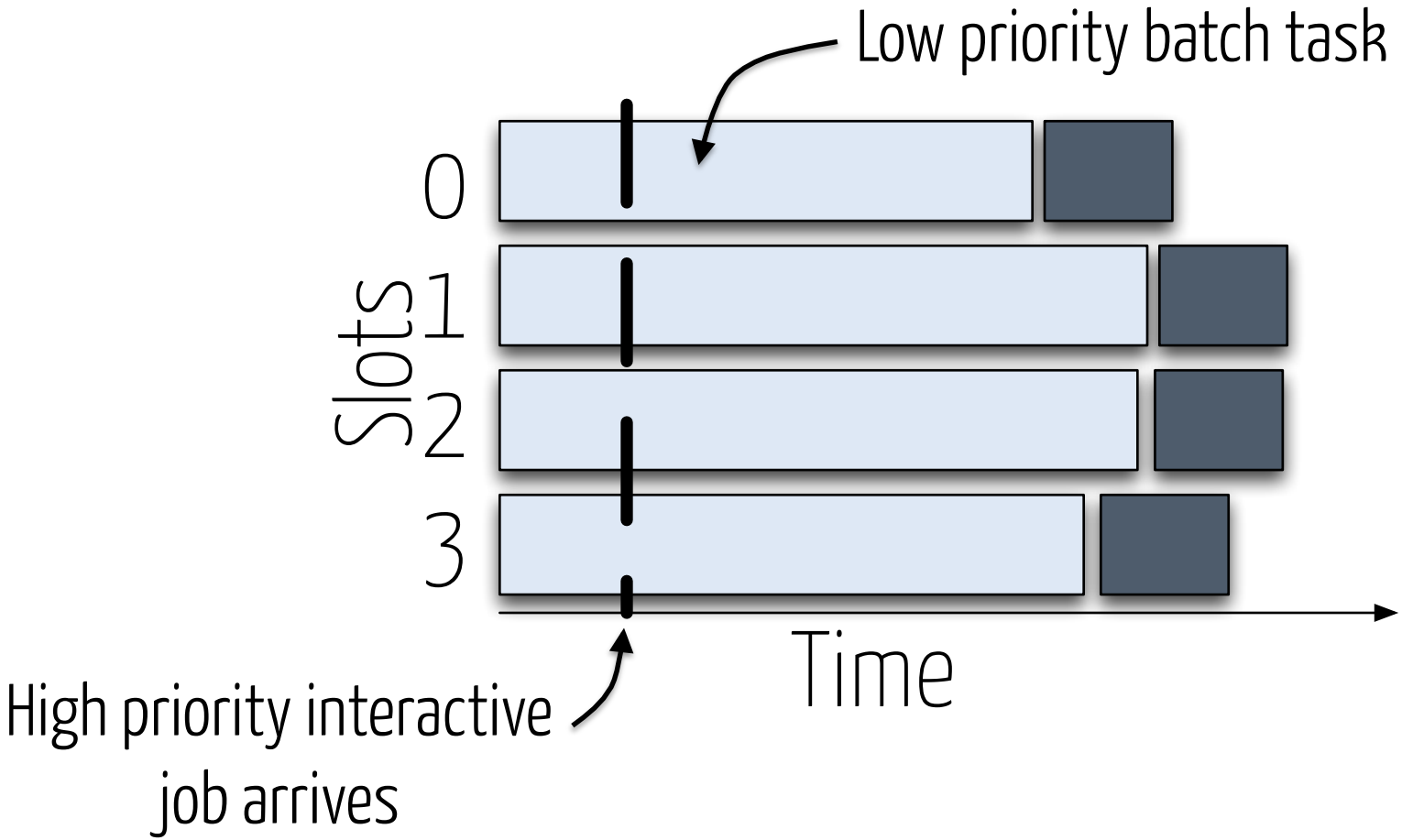
Benefit: Handling of Skew and Stragglers



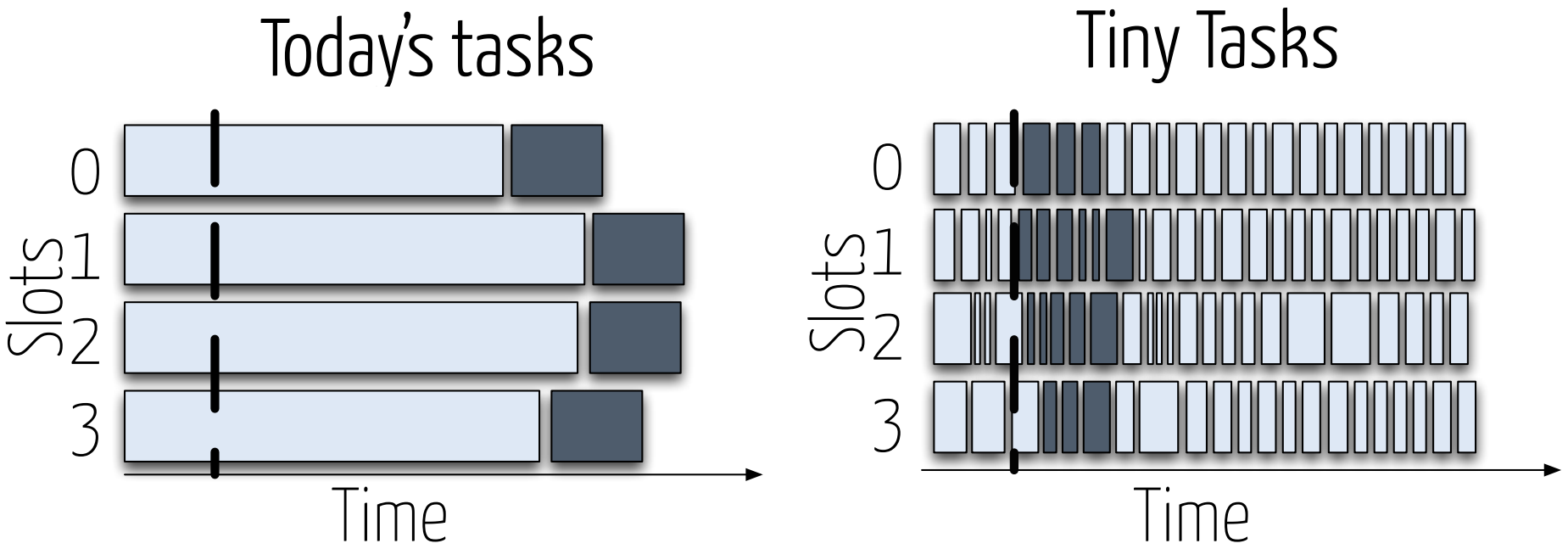
As much as 5.2x reduction in job completion time!

Problem: Batch and Interactive Sharing

Clusters forced to trade off utilization and responsiveness!



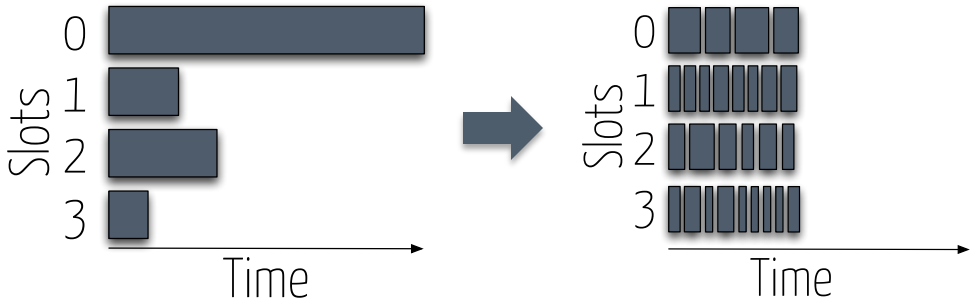
Benefit: Improved Sharing



High-priority tasks not subject to long wait times!

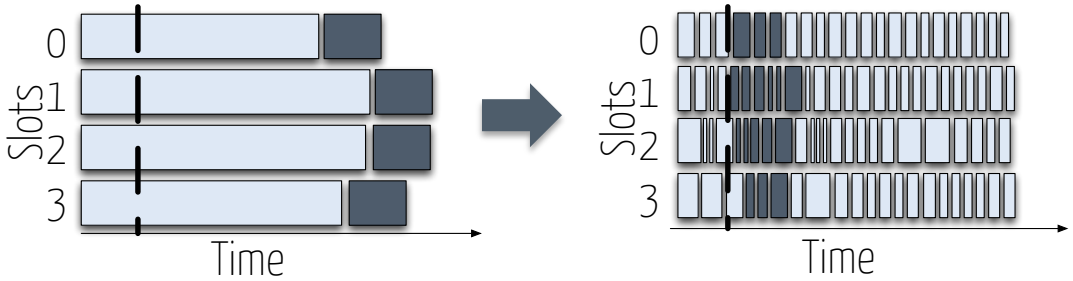
Benefits: Recap

(1) Straggler mitigation



Mantri (OSDI '10)
Scarlett (EuroSys '11)
SkewTune (SIGMOD '12)
Dolly (NSDI '13)
...

(2) Improved sharing



Quincy (SOSP '09)
Amoeba (SOCC '12)
...

Why? **How?** Where?

Schedule task

Scheduling requirements:

High Throughput (millions per second)

Low Latency (milliseconds)

Distributed Scheduling
(e.g., Sparrow Scheduler)

Use existing thread pool to launch tasks

Schedule task

Launch task

Use existing thread pool to launch tasks

+

Cache task binaries

Task launch = RPC time (<1ms)

Schedule task

Launch task

Read input data

Smallest efficient file block size:

8MB

Distribute Metadata

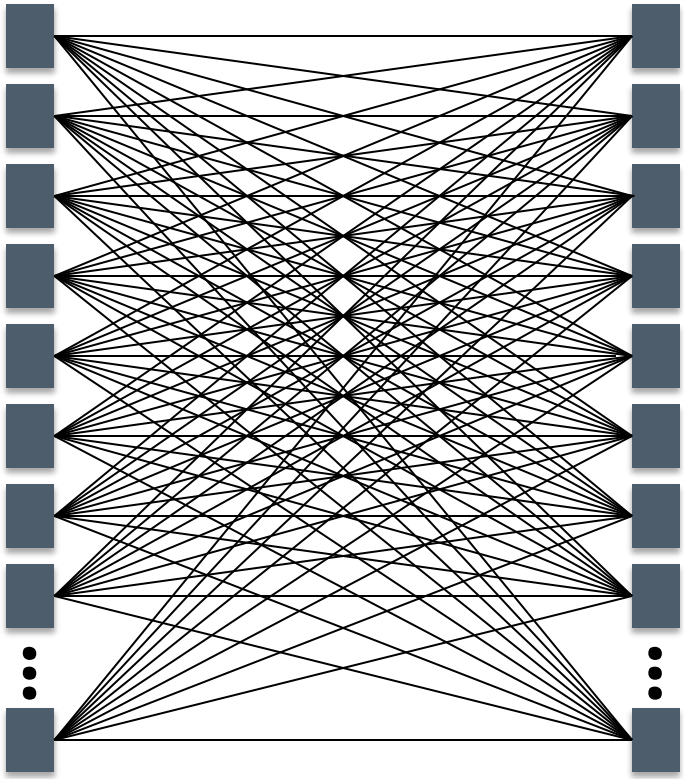
(à la Flat Datacenter Storage, OSDI '12)

Schedule task

Launch task

Read input data

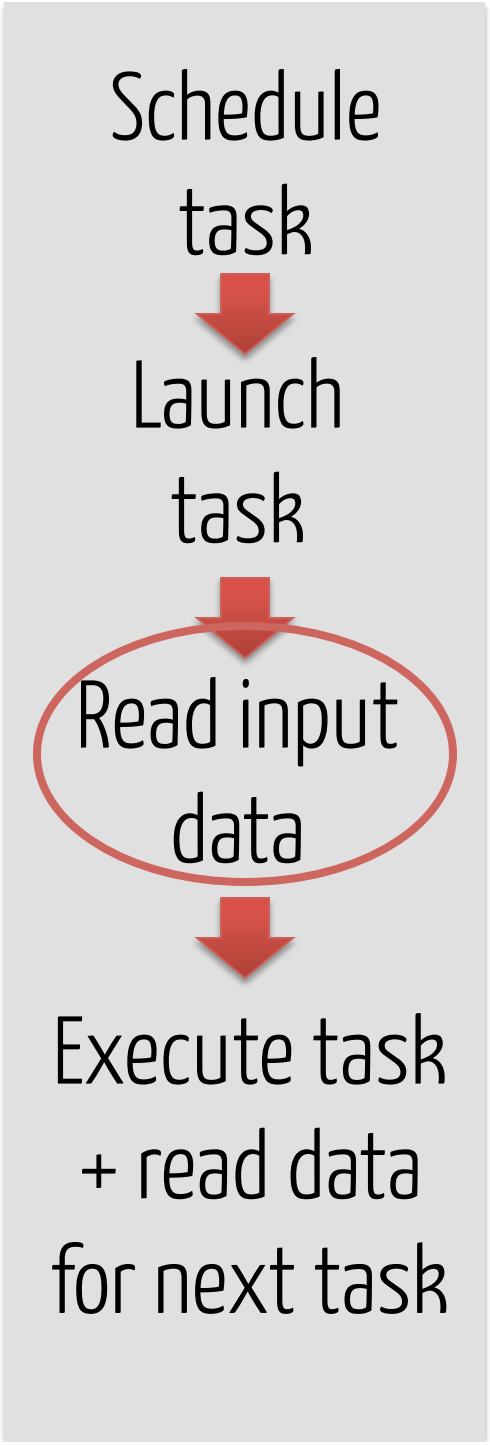
Execute task + read data for next task



Tons of tiny transfers!

Framework-Controlled I/O

(enables optimizations, e.g., pipelining)

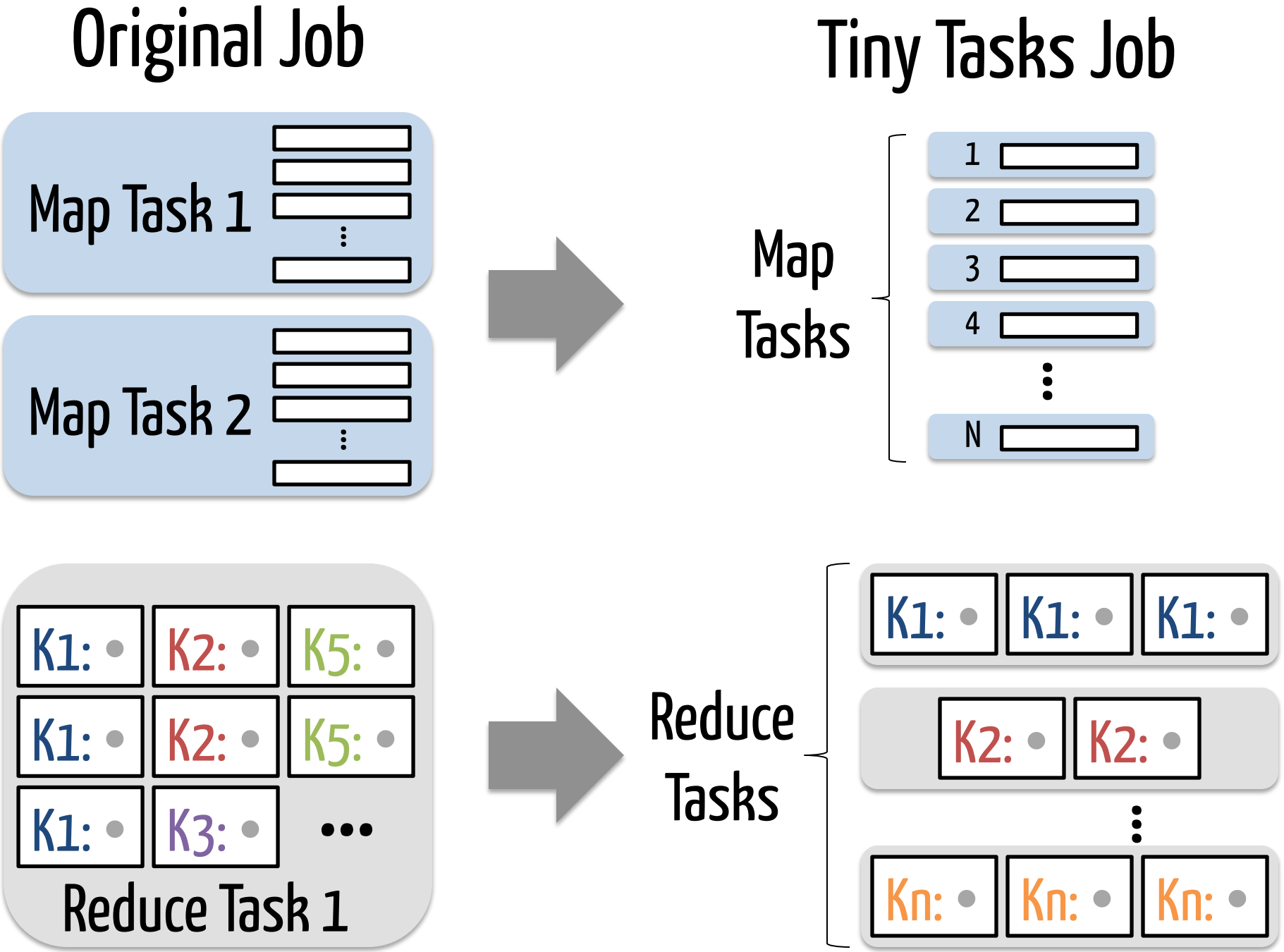


How low can you go?

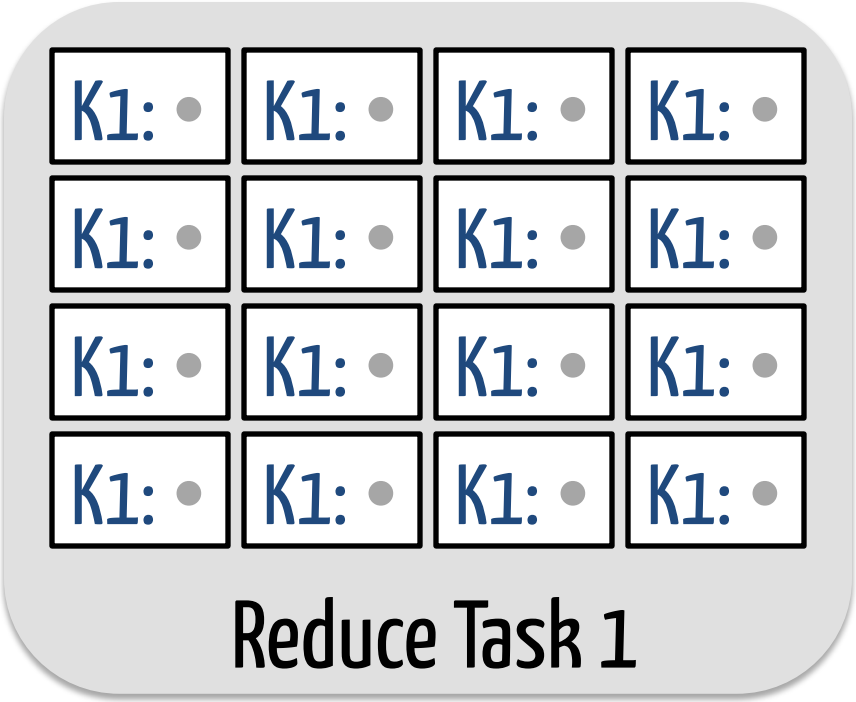
8MB disk block

100's of milliseconds

Why? How? **Where?**


Original Job
Map Task 1
Map Task 2
K1:
K2:
K5:
K1:
K2:
K5:
K1:
K3:
Reduce Task 1
Tiny Tasks Job
Map Tasks
1
2
3
4
N
Reduce Tasks
K1:
K1:
K1:
K2:
K2:
Kn:
Kn:
Kn:

Original Reduce Phase



Tiny Tasks = ?

Splitting Large Tasks

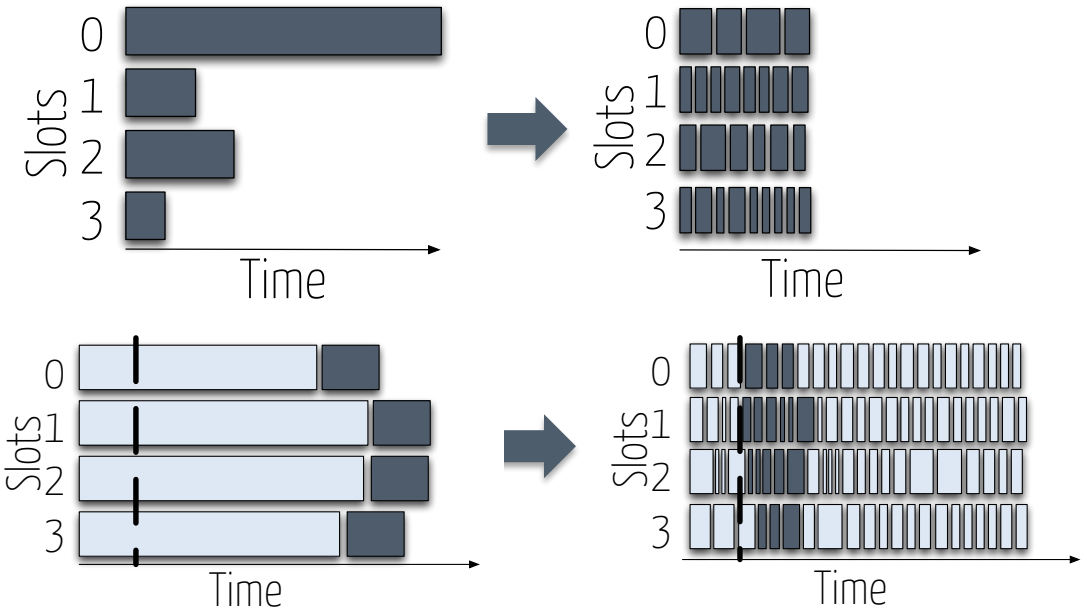
- Aggregation trees
  - Works for functions that are associative and commutative
- Framework-managed temporary state store
- Ultimately, need to allow a small number of large tasks

Tiny tasks

mitigate stragglers

+

Improve sharing



Distributed scheduling → Launch task in existing thread pool → Distributed file metadata → Pipelined task execution

Questions? Find me or Shivaram: