Making Sense of Spark Performance

eecs.berkeley.edu/~keo/traces

Kay Ousterhout
UC Berkeley

In collaboration with Ryan Rasti, Sylvia Ratnasamy, Scott Shenker, and Byung-Gon Chun

About Me

PhD student in Computer Science at UC Berkeley

Thesis work centers around performance of large-scale distributed systems

Spark PMC member

About This Talk

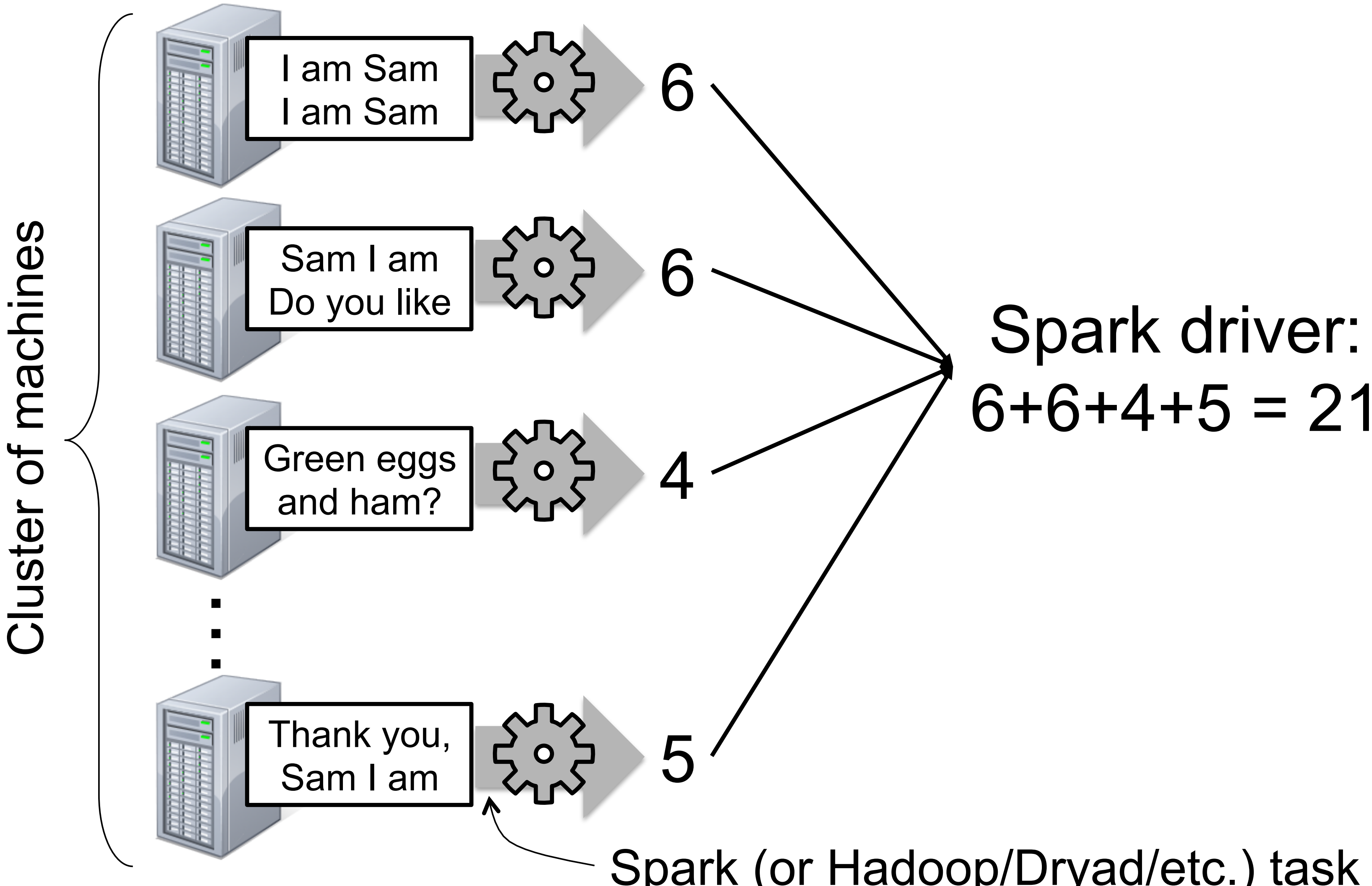
Overview of how Spark works

How we measured performance bottlenecks

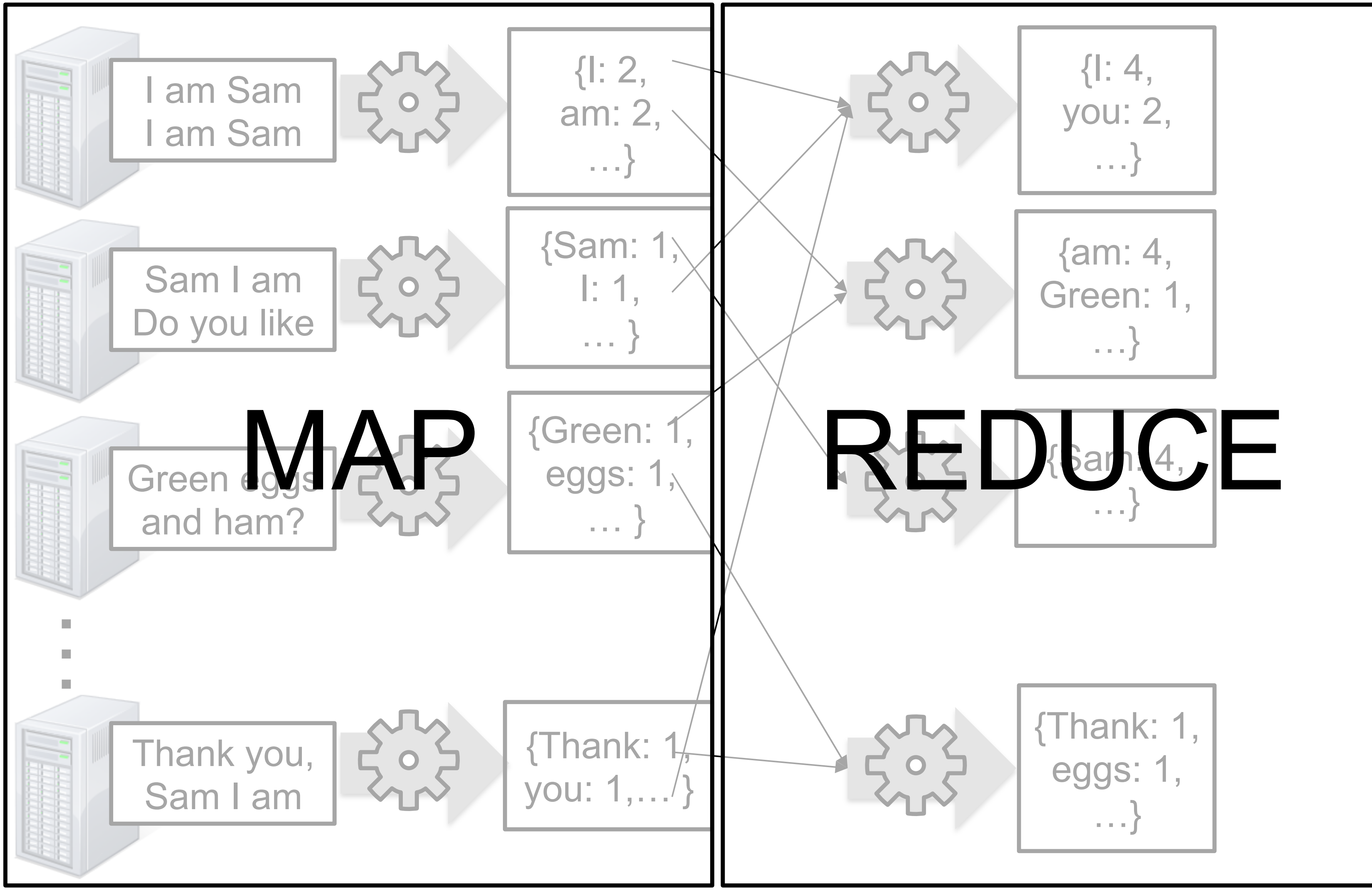
In-depth performance analysis for a few workloads

Demo of performance analysis tool

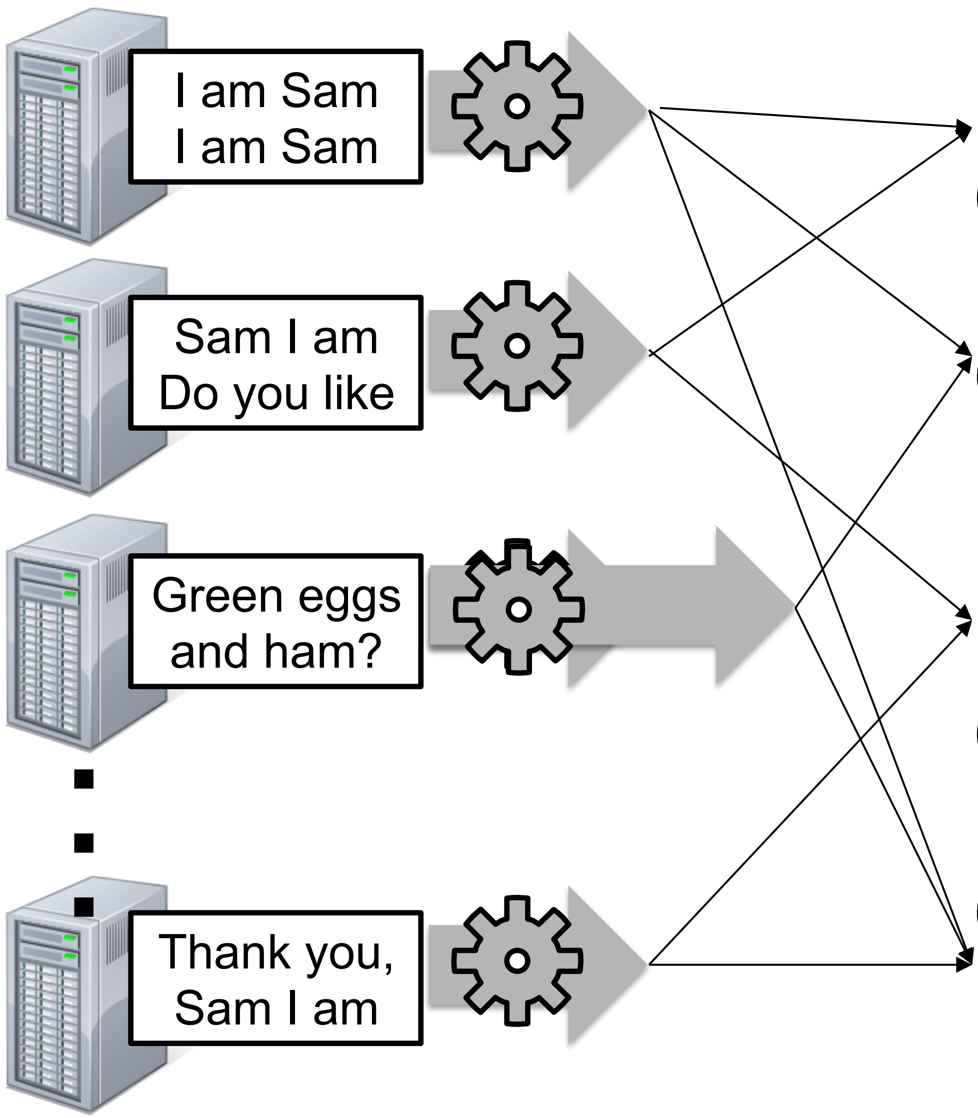
Count the # of words in the document



Count the # of occurrences of each word



Performance considerations



(1) Caching input data

(2) Scheduling: assigning tasks to machines

(3) Straggler tasks

(4) Network performance (e.g., during shuffle)

Caching PACMan [NSDI ’12], Spark [NSDI ’12], Tachyon [SoCC ’14]

Scheduling Sparrow [SOSP ‘13], Apollo [OSDI ’14], Mesos [NSDI ‘11], DRF [NSDI ‘11], Tetris [SIGCOMM ’14], Omega [Eurosys ’13], YARN [SoCC ’13], Quincy [SOSP ‘09], KMN [OSDI ’14]

Stragglers Scarlett [EuroSys ‘11], SkewTune [SIGMOD ‘12], LATE [OSDI ‘08], Mantri [OSDI ‘10], Dolly [NSDI ‘13], GRASS [NSDI ‘14], Wrangler [SoCC ’14]

Network VL2 [SIGCOMM ‘09], Hedera [NSDI ’10], Sinbad [SIGCOMM ’13], Orchestra [SIGCOMM ’11], Baraat [SIGCOMM ‘14], Varys [SIGCOMM ’14], PeriSCOPE [OSDI ‘12], SUDO [NSDI ’12], Camdoop [NSDI ’12], Oktopus [SIGCOMM ‘11]), EyeQ [NSDI ‘12], FairCloud [SIGCOMM ’12]

Generalized programming model

Dryad [Eurosys ‘07], Spark [NSDI ’12]

Caching PACMan [NSDI '12], Spark [NSDI '12], Tachyon [SoCC '14]

Scheduling Sparrow [SOSP '13], Apollo [OSDI '14], Mesos [NSDI '11], DRF [NSDI '11], Tetris [SIGCOMM '14], Omega [Eurosys '13], YARN [SoCC '13], Quincy [SOSP '09], KMN [OSDI '14]

Stragglers [OSDI '08], Mantri [OSDI '10], Dolly [NSDI '13], GRASS [NSDI '14], Wrangler [SoCC '14]

Network and disk I/O are bottlenecks

Network VL2 [SIGCOMM '09], Hedera [NSDI '10], Sinbad [SIGCOMM '13], Orchestra [SIGCOMM '11], Baraat [SIGCOMM '14], Varys [SIGCOMM '14], PeriSCOPE [OSDI '12], SUDO [NSDI '12], Camdoop [NSDI '12], Oktopus [SIGCOMM '11]), EyeQ [NSDI '12], FairCloud [SIGCOMM '12]

Stragglers are a major issue with unknown causes

Generalized programming model Dryad [Eurosys '07], Spark [NSDI '12]

This Work

(1) Methodology for quantifying performance bottlenecks

(2) Bottleneck measurement for 3 SQL workloads (TPC-DS and 2 others)

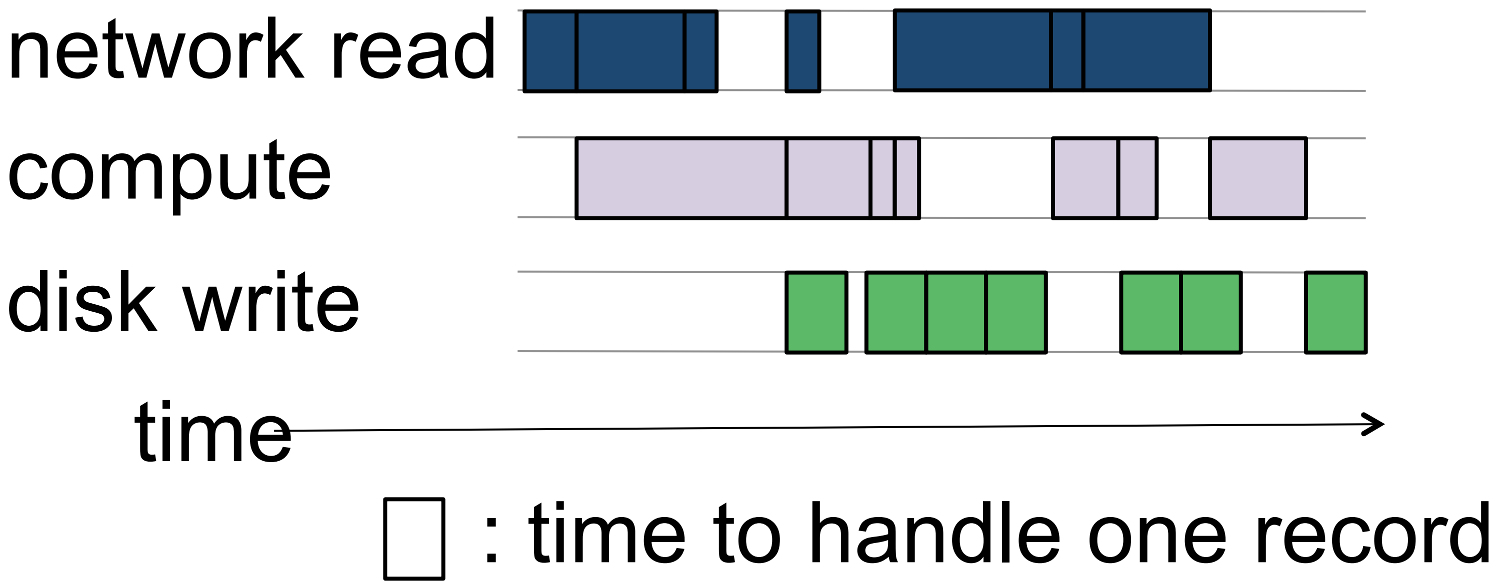
Network optimizations

can reduce job completion time by **at most 2%**

CPU (not I/O) often the bottleneck

Most straggler causes can be identified and fixed

Example Spark task:



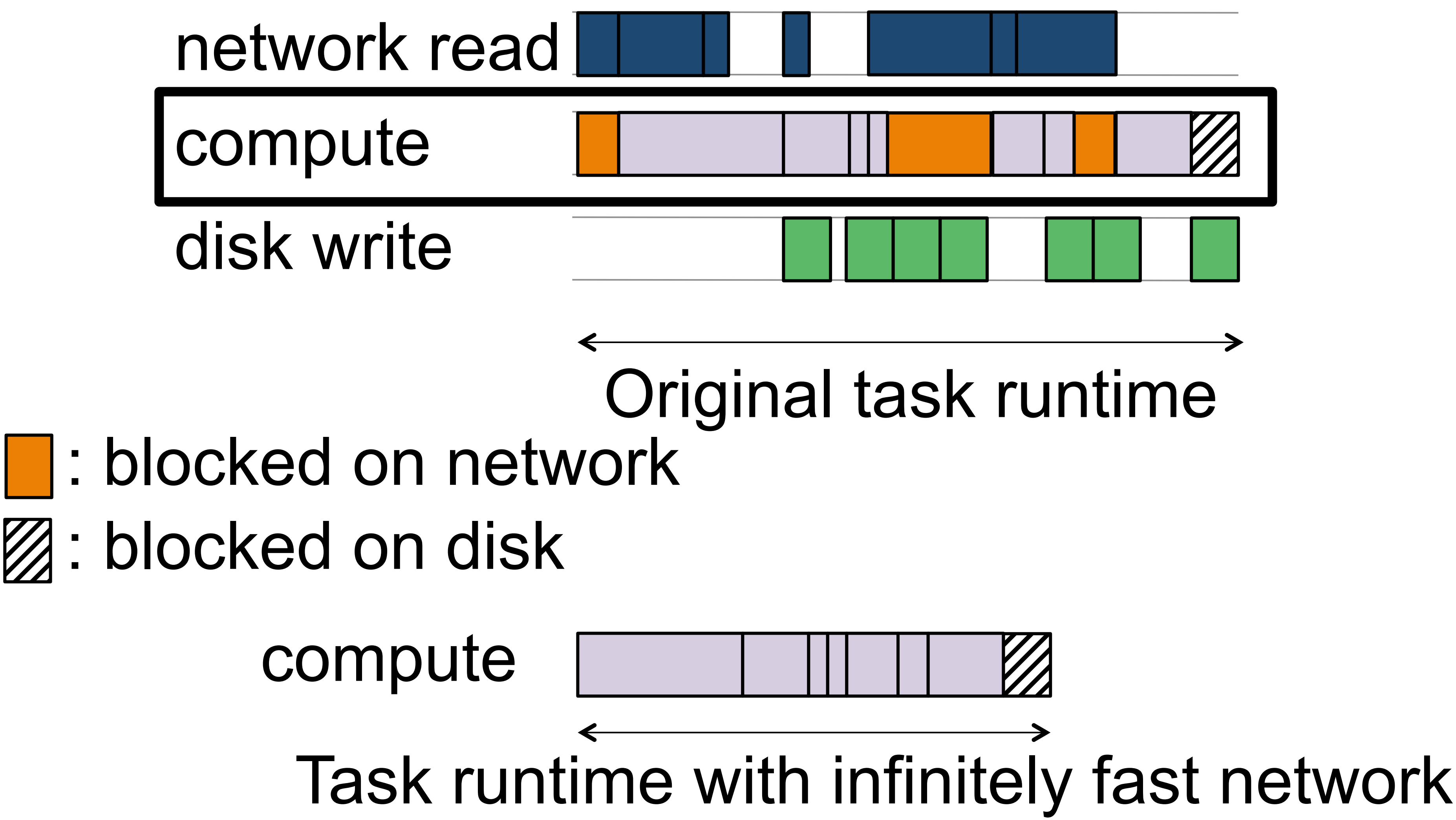
Fine-grained instrumentation needed to understand performance

How much faster would a job run if the network were infinitely fast?

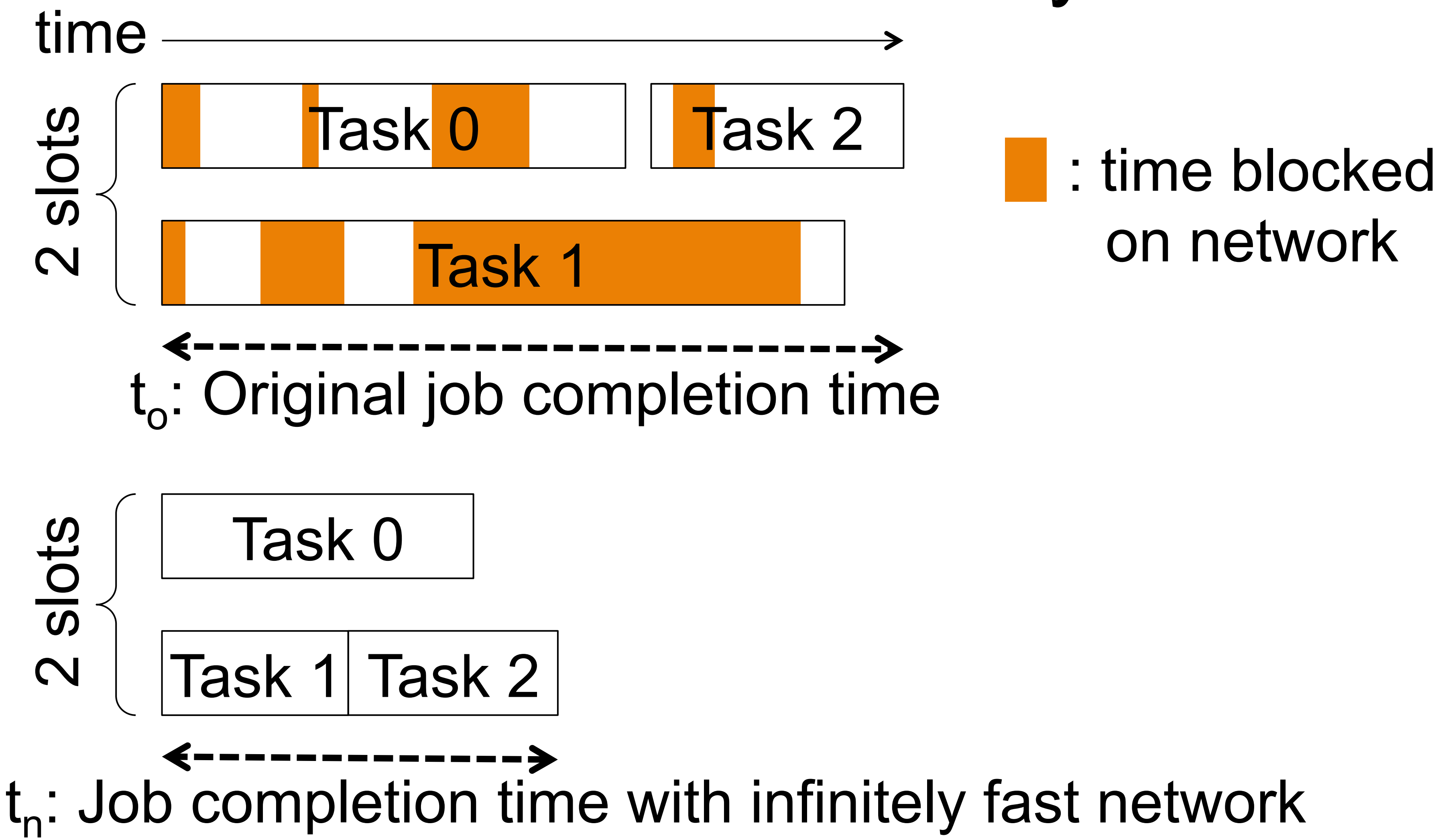
↓

What's an upper bound on the improvement from network optimizations?

How much faster could a **task** run if the network were infinitely fast?



How much faster would a job run if the network were infinitely fast?



SQL Workloads

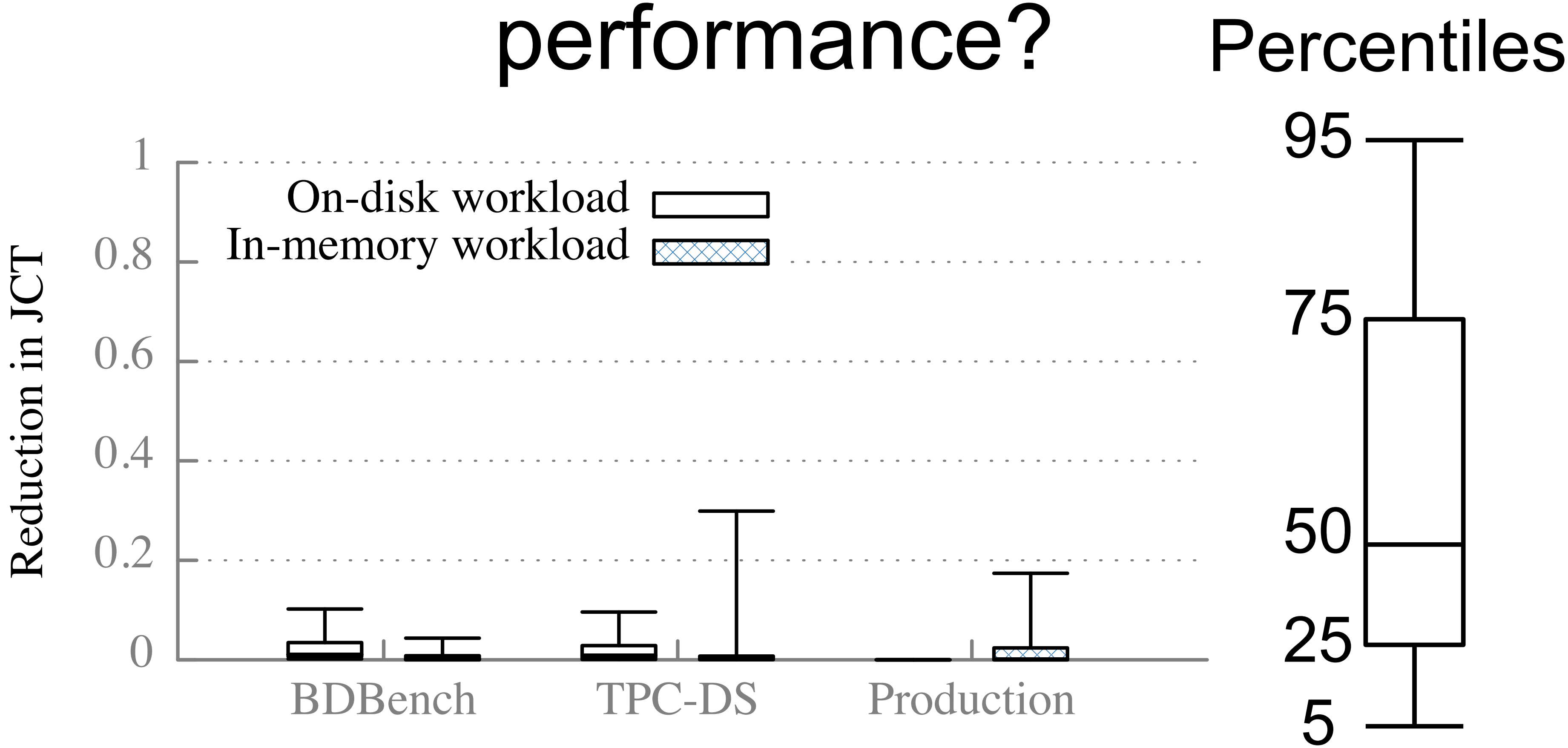
TPC-DS (20 machines, 850GB;
60 machines, 2.5TB)
www.tpc.org/tpcds

Big Data Benchmark (5 machines, 60GB)
amplab.cs.berkeley.edu/benchmark

Databricks (9 machines, tens of GB)
databricks.com

2 versions of each: in-memory, on-disk

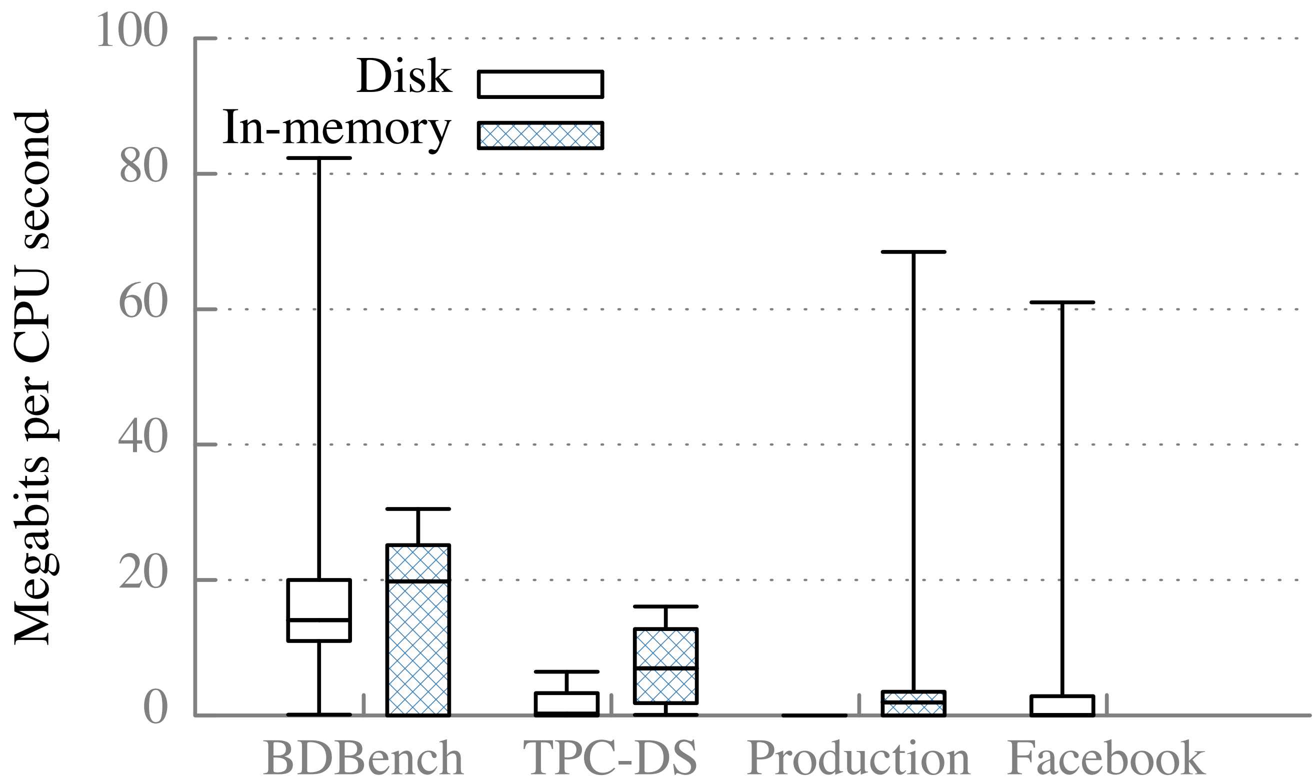
How much faster could jobs get from optimizing network performance?



Median improvement at most 2%

How can we sanity check these numbers?

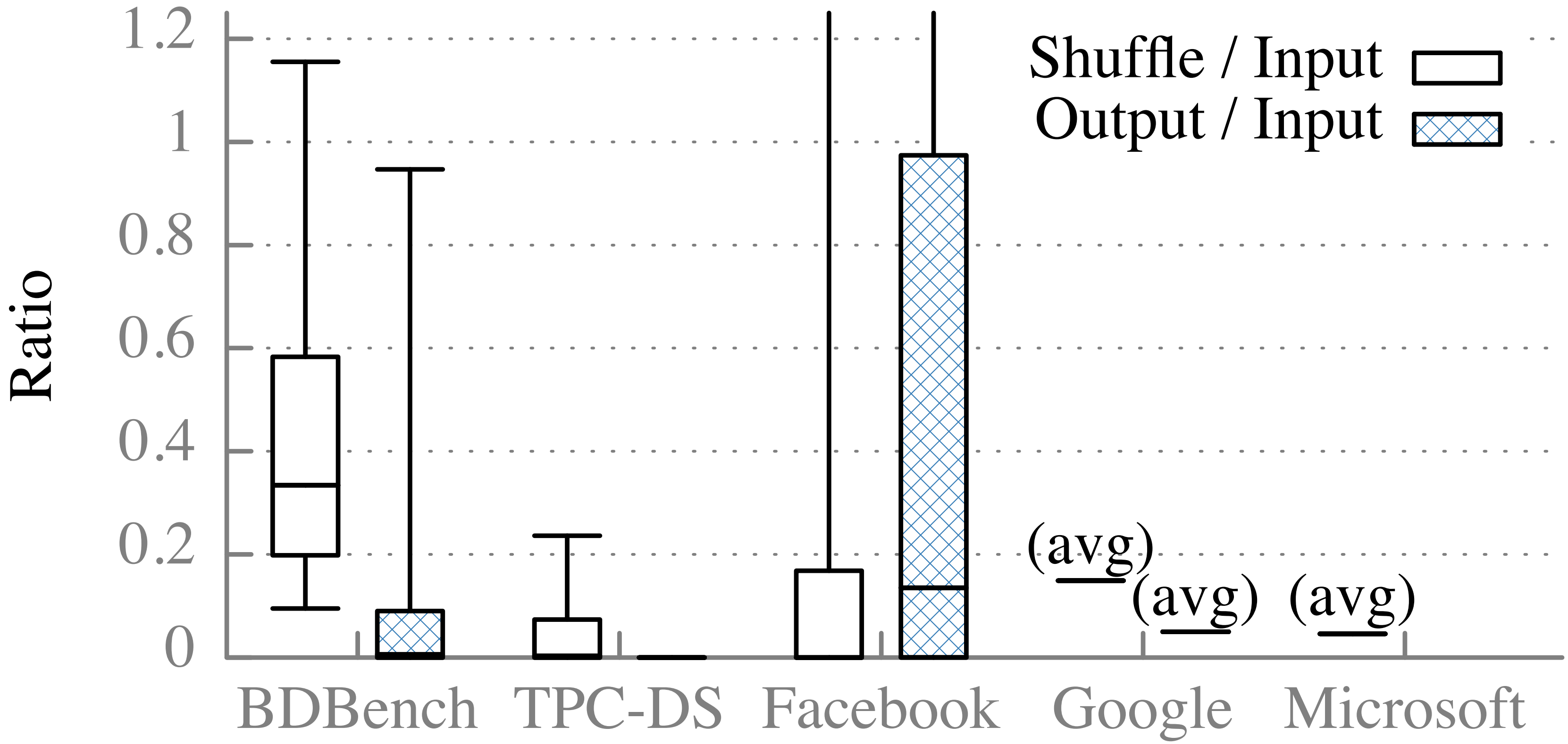
How much data is transferred per CPU second?



Microsoft ’09-’10: **1.9–6.35 Mb / task second**

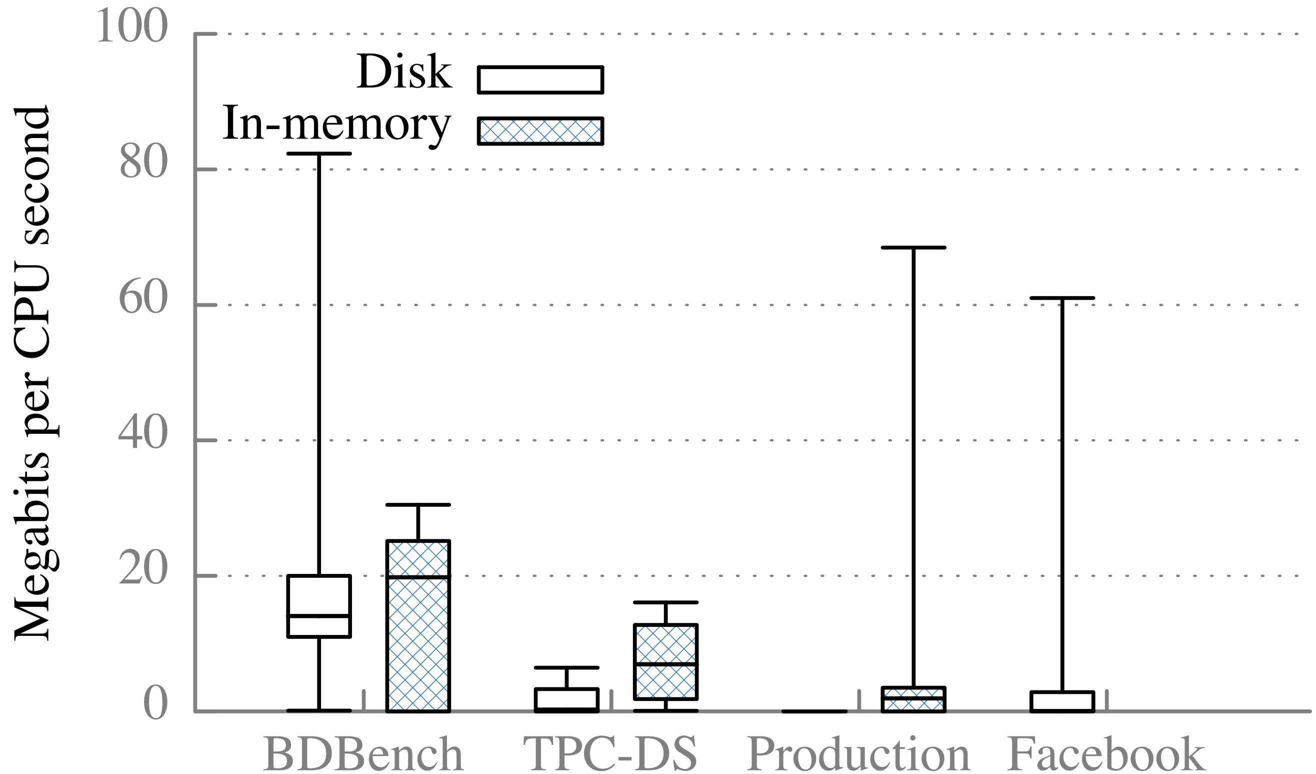
Google ’04-‘07: **1.34–1.61 Mb / machine second**

How can this be true?



Shuffle Data < Input Data

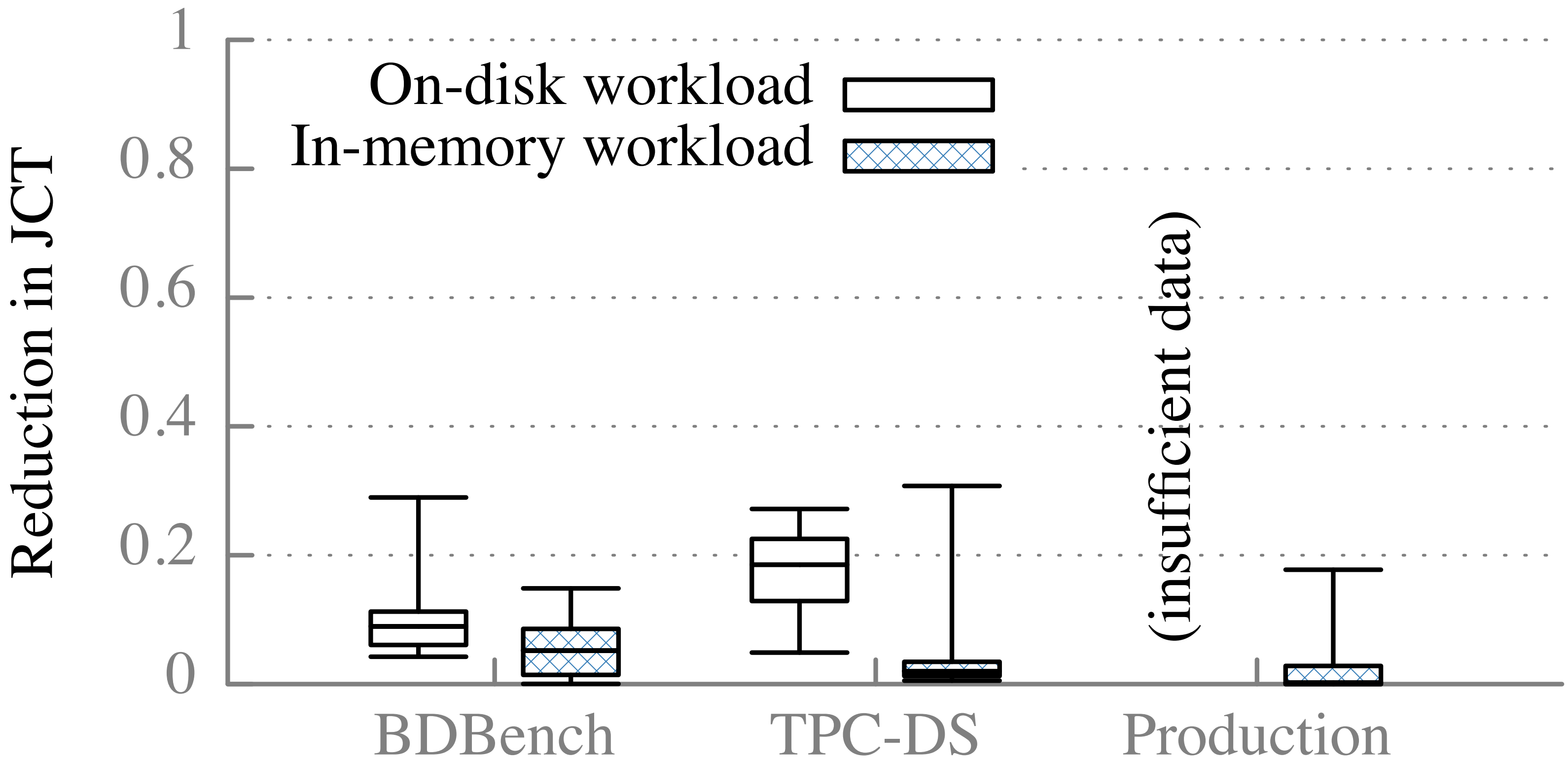
What kind of hardware should I buy?



10Gbps networking hardware likely not necessary!

How much faster would jobs complete if the disk were infinitely fast?

How much faster could jobs get from optimizing disk performance?



Median improvement at most 19%

Disk Configuration

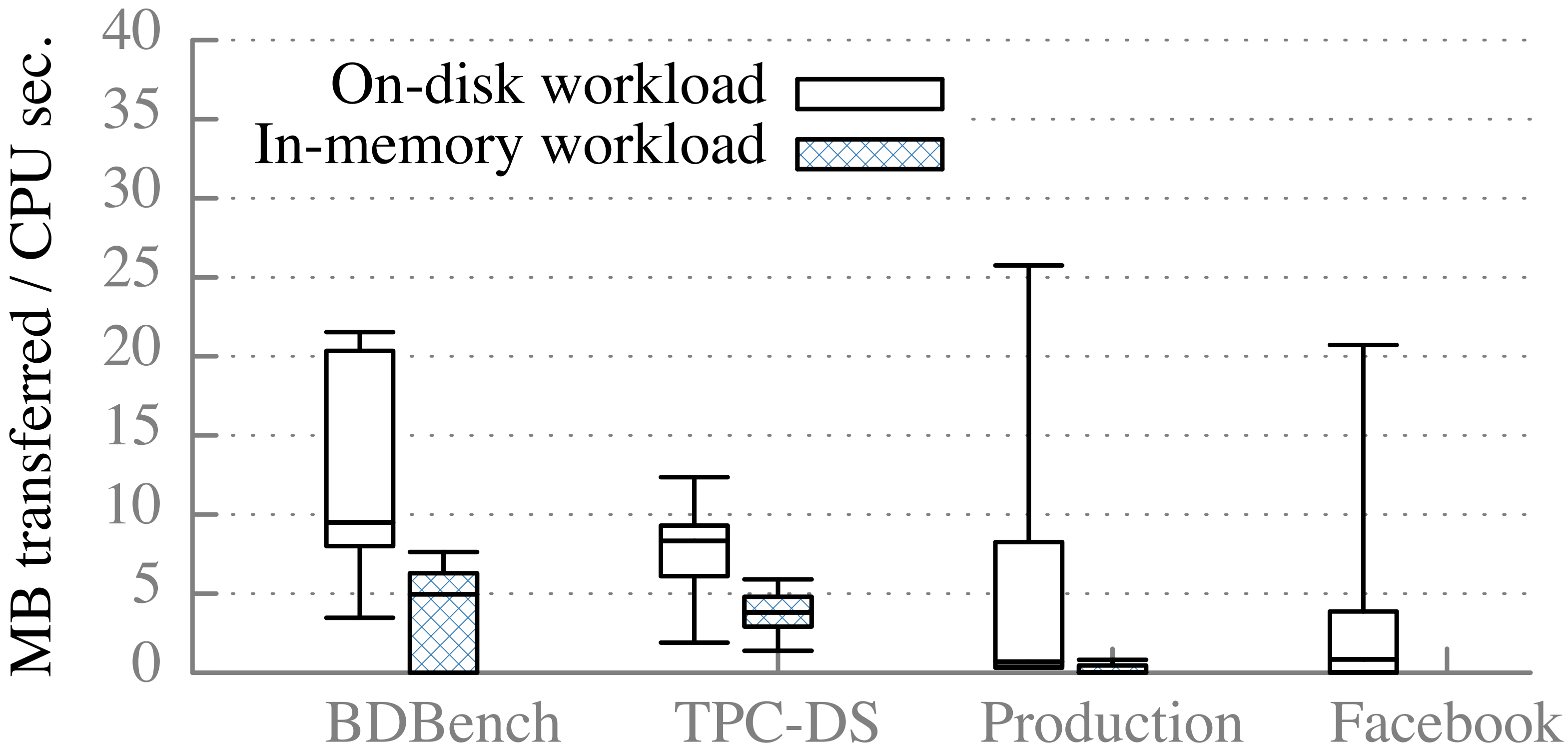
Our instances: 2 disks, 8 cores

Cloudera:

- At least 1 disk for every 3 cores
- As many as 2 disks for each core

Our instances are under provisioned → results are upper bound

How much data is transferred per CPU second?



Google: **0.8-1.5 MB / machine second**

Microsoft: **7-11 MB / task second**

What does this mean about Spark versus Hadoop?

This work:
19%

serialized + compressed **on-disk** data

serialized + compressed **in-memory** data

Faster

This work says nothing about Spark vs. Hadoop!

up to 10x
spark.apache.org

This work:
19%

6x or more
amplab.cs.berkeley.edu/benchmark/

hadoop
(on-disk data)

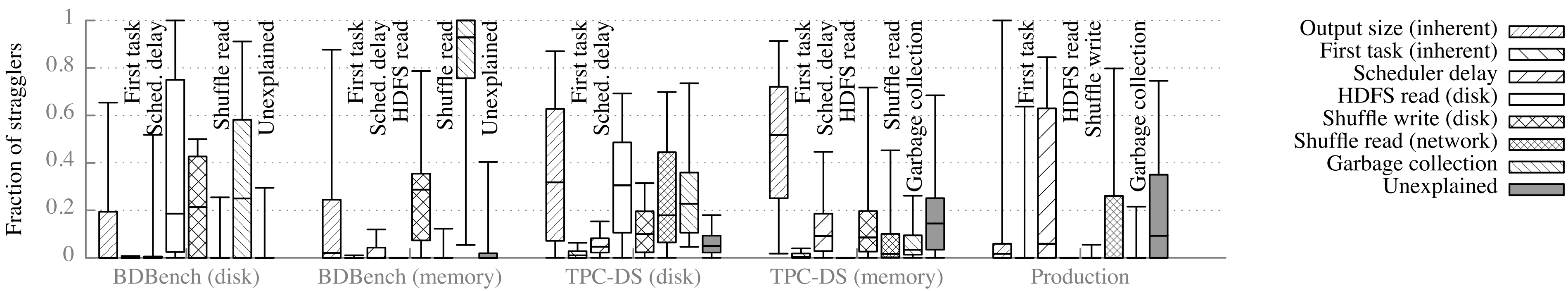
Spark
serialized + compressed **on-disk** data

Spark
serialized + compressed **in-memory** data

Spark
deserialized in-memory data

Faster →

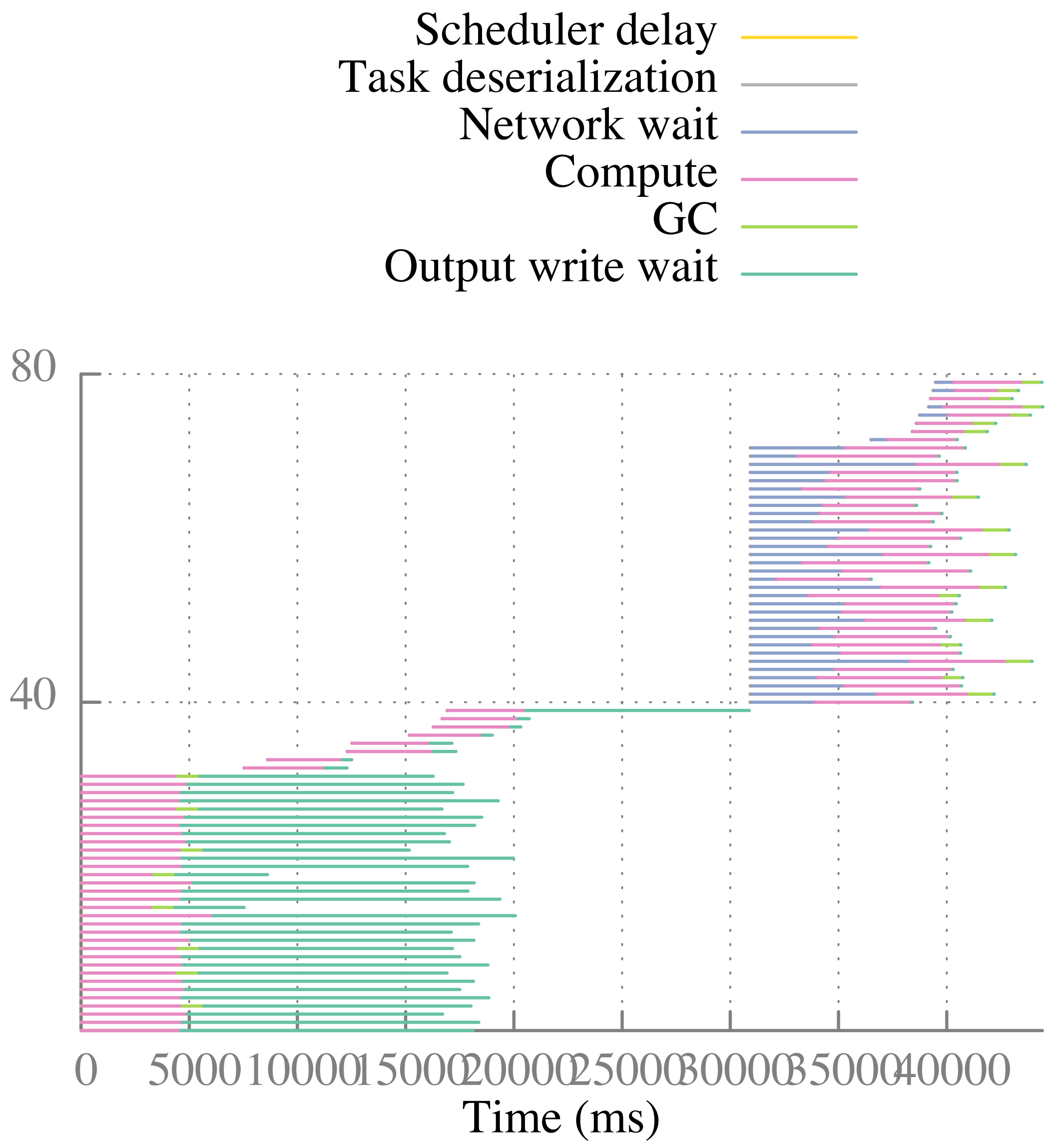
What causes stragglers?



Takeaway: causes depend on the workload, but disk and garbage collection common

Fixing straggler causes can speed up other tasks too

Live demo



eecs.berkeley.edu/~keo/traces

I want your workloads!

`spark.eventLog.enabled true`

keo@cs.berkeley.edu

Network optimizations
can reduce job completion time by **at most 2%**

CPU (not I/O) often the bottleneck
19% reduction in completion time from optimizing disk

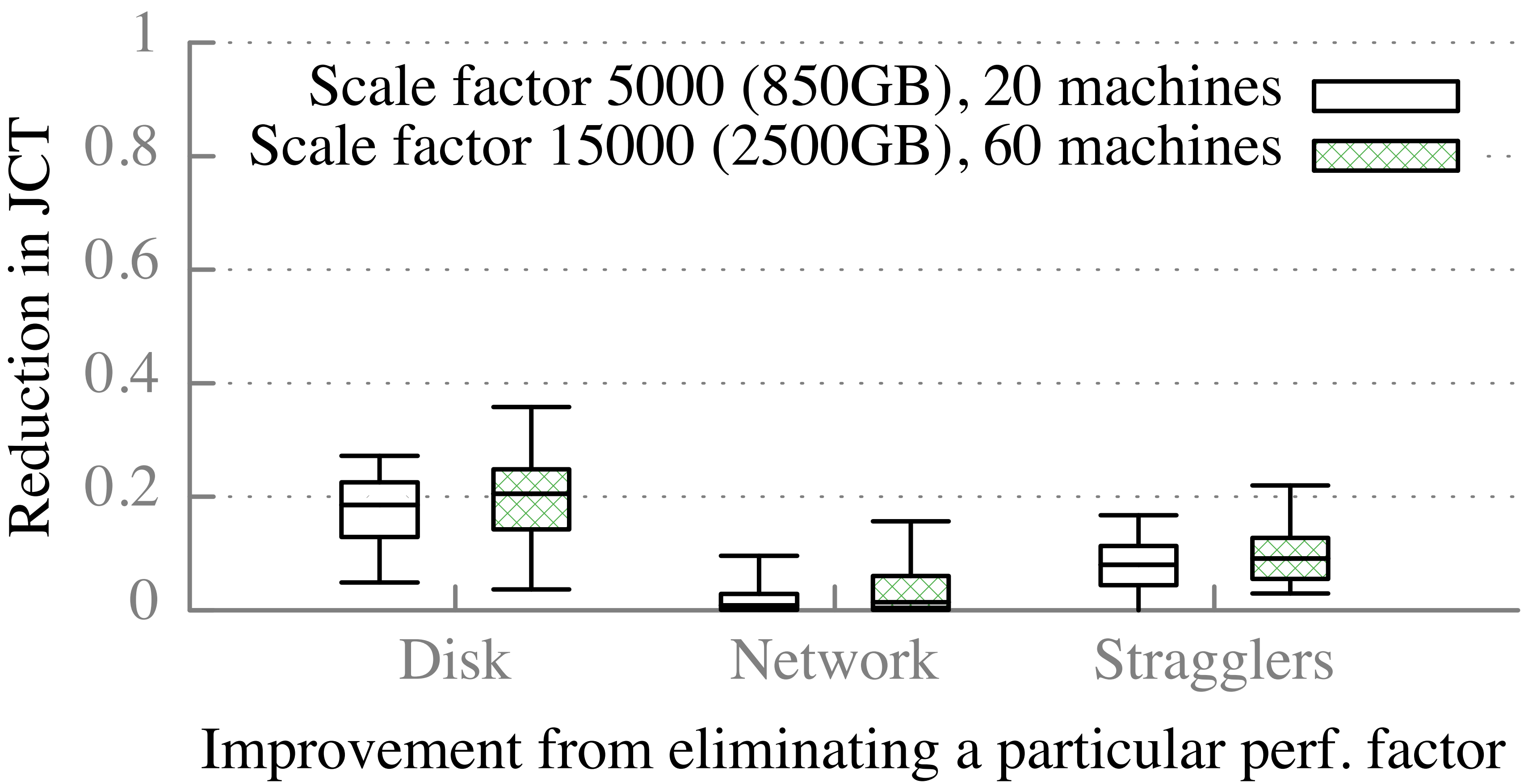
Many straggler causes can be identified and fixed

Project webpage (with links to paper and tool):
eecs.berkeley.edu/~keo/traces

Contact: keo@cs.berkeley.edu, @kayousterhout

Backup Slides

How do results change with scale?



How does the utilization compare?