Reducing Job Slowdown Variability for Data-Intensive Workloads

Bogdan Ghit and Dick Epema

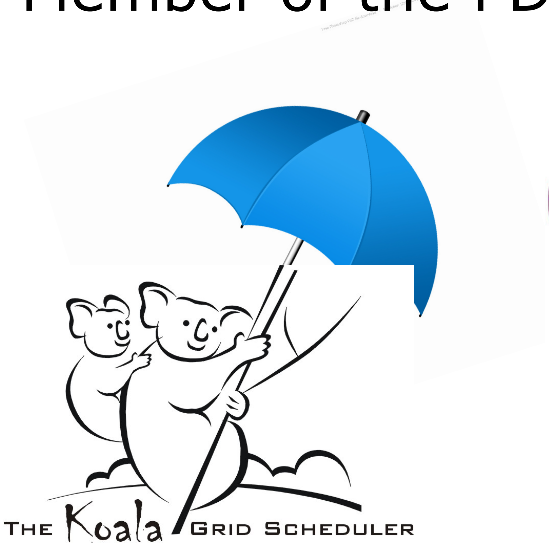
Delft University of Technology

About me

PhD candidate at TU Delft, advised by Dick Epema.

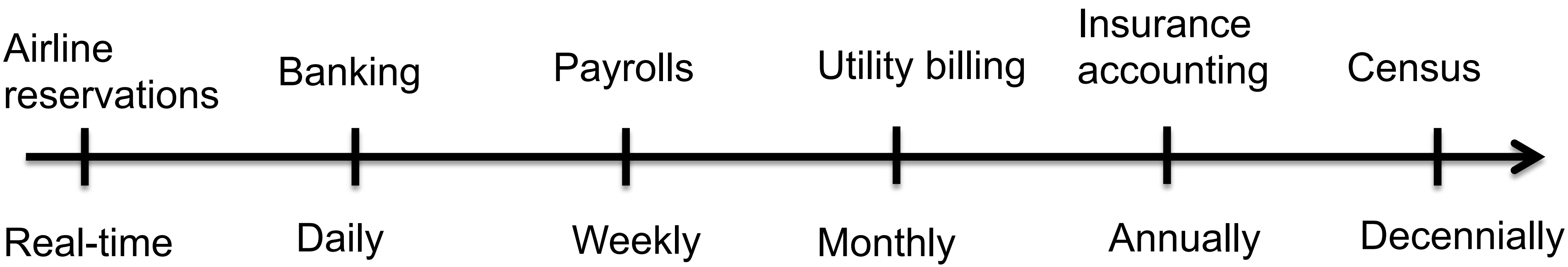
Thesis topic on performance of data analytics frameworks.

Member of the PDS group (see tag cloud below).



A brief history of big data

Vast market since early 1900s:



Variety

"The Yale library in 2040 will have approximately 200,000,000 volumes, which will occupy over 6,000 miles of shelves, requiring a cataloging staff of over 6,000 people." - Fremont Rider, 1944.

Velocity

Volume

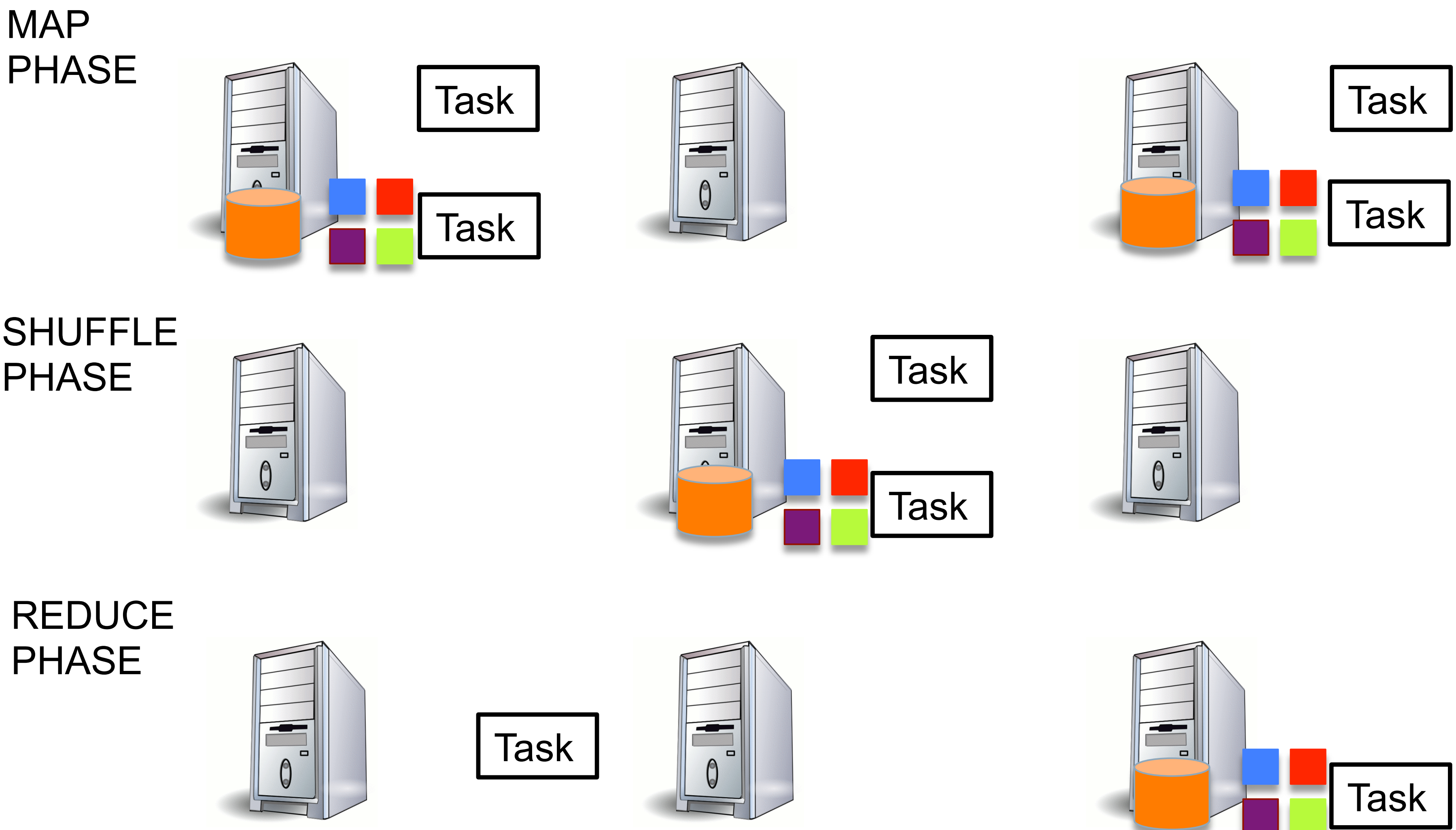
Yesterday’s big data processing systems

Burroughs accounting machine.

Widespread from the early 1900s to 1980s.

Replaced by low-cost computers such as IBM PC.

Today's big data processing systems



Data analytics jobs

Data analytics jobs run many small tasks in parallel on compute slots of different machines.

Frameworks (MapReduce, Spark, Dryad) provide abstractions to construct jobs automatically.

Frameworks hide task parallelism, network communication patterns and fault tolerance.

In production traces short jobs prevail, but large jobs dominate.

Making single jobs faster

Data locality
Delay Scheduler [Eurosys'10], PACMan [NSDI'12], Spark [NSDI'12], Tachyon [SoCC'14].

Straggler mitigation
LATE Scheduler [OSDI'08], Mantri [OSDI'10], Scarlett [EuroSys'11], Dolly [NSDI'13], GRASS [NSDI'14], Hopper [SIGCOMM'15].

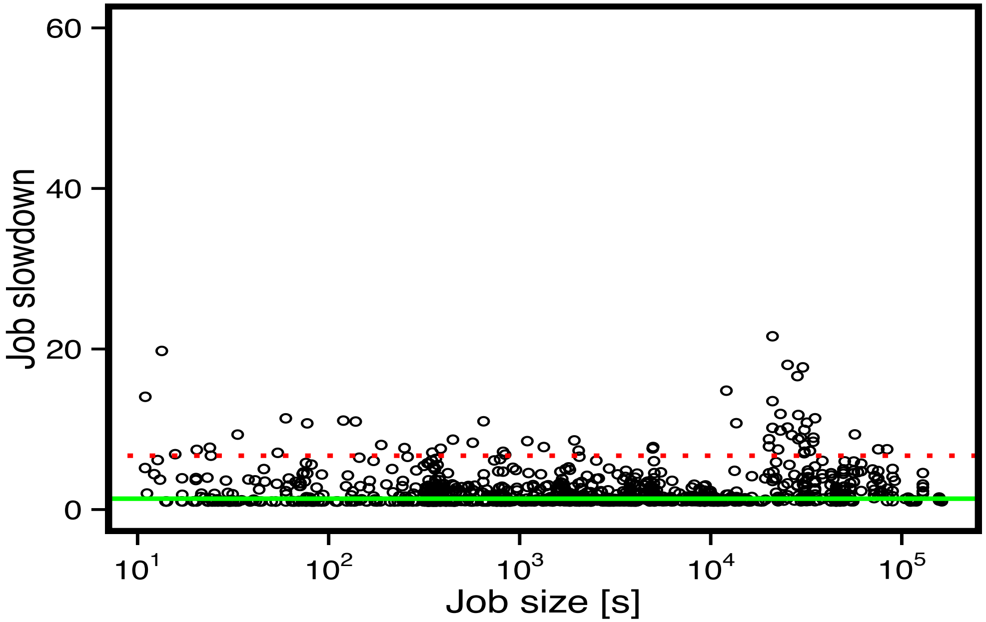
Shuffle optimizations
Coupling Scheduler [INFOCOM'12], Max/SplitSRPT [Performance'13].

Tension between fast service and fairness not addressed.

Missing: policies tailored for workloads with many short jobs.

TUDelft

MapReduce workloads are challenging



Job Size [s]

95th percentile

95th percentile
Median

Job slowdown variability: 95th percentile/median.

Short jobs suffer!

TUDelft

This work

(1) Two main mechanisms to allocate resources.
Four scheduling policies.

(2) Is fairness achieved?
Accurate large-scale simulations of MapReduce.

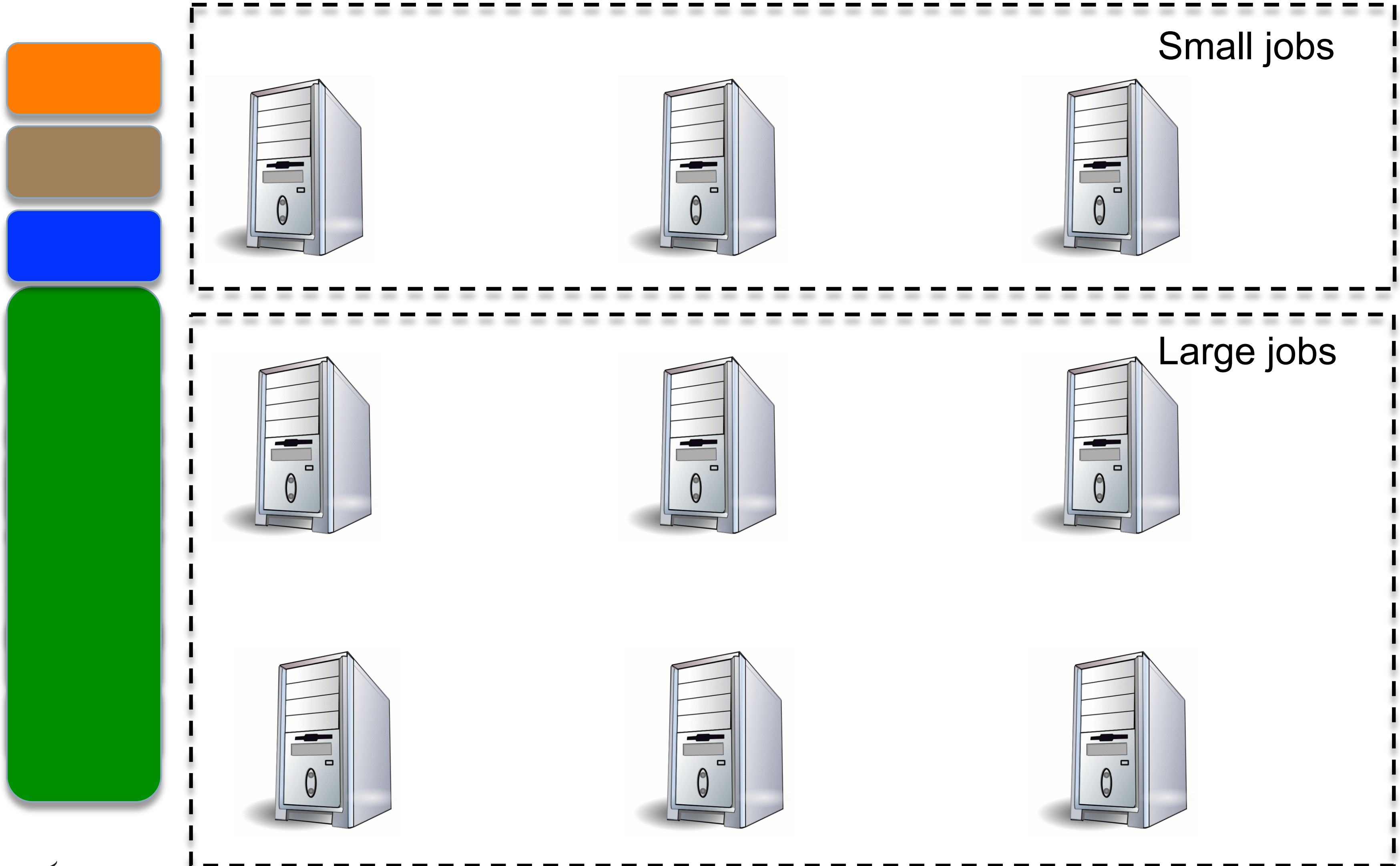
TU Delft

Outline

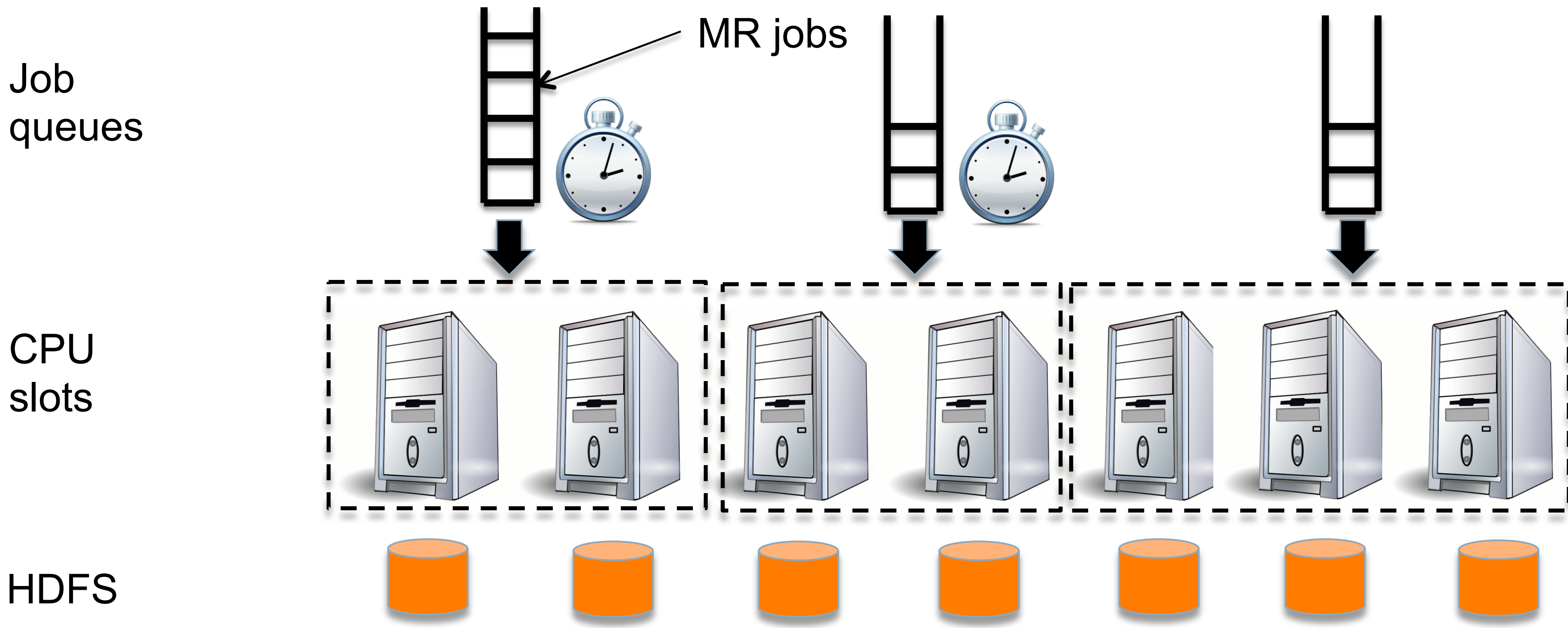
- Main mechanisms
- Scheduling policies
- Experimental setup
- Results

Large jobs monopolize the cluster

Intuitive solution



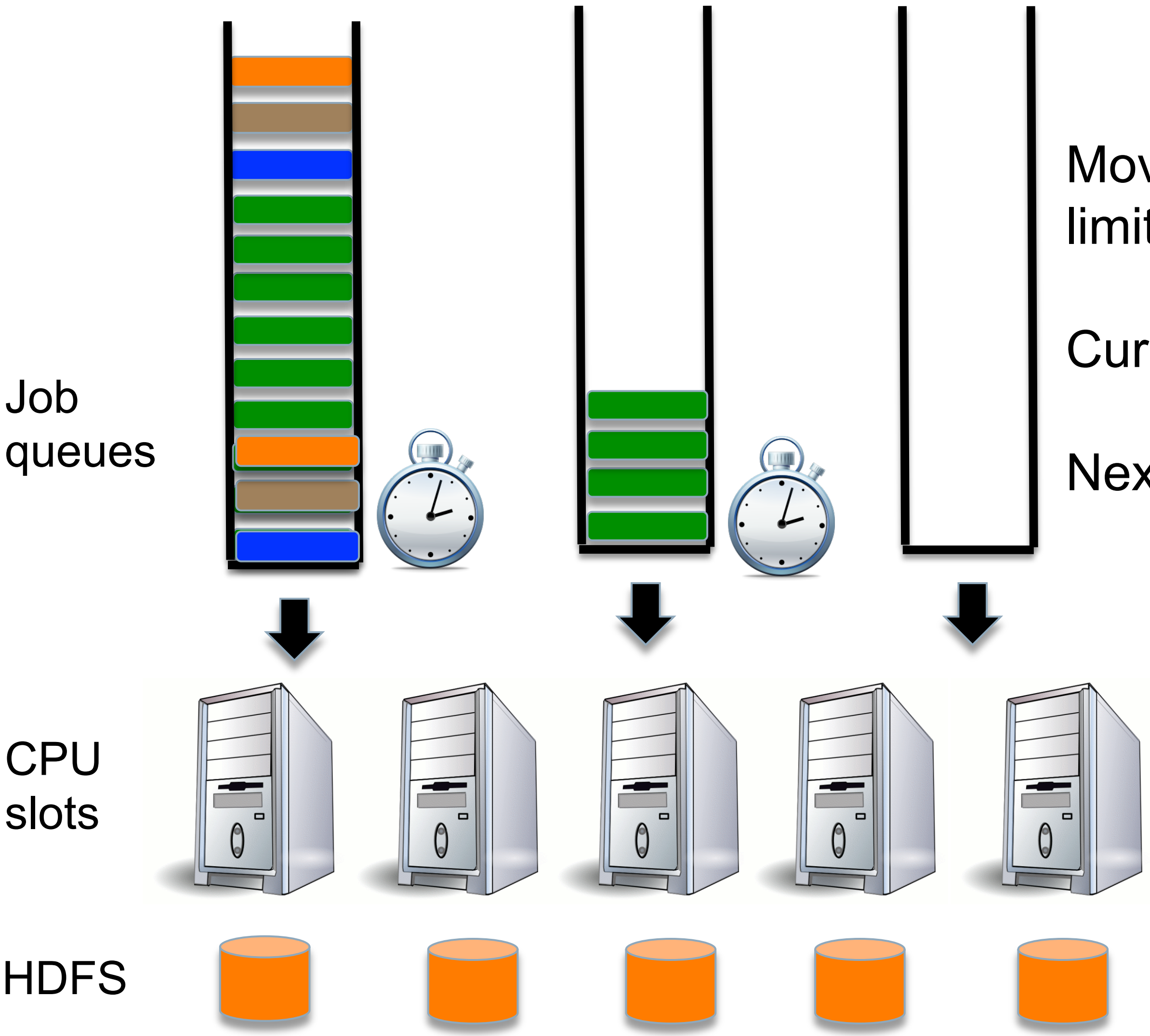
Mechanism 1: Logical resource partitioning



Allocate compute slots across disjoint partitions.

Restrict the amount of service offered to jobs in queues.

Mechanism 2: System feedback



Move jobs that exceed the time limit across queues.

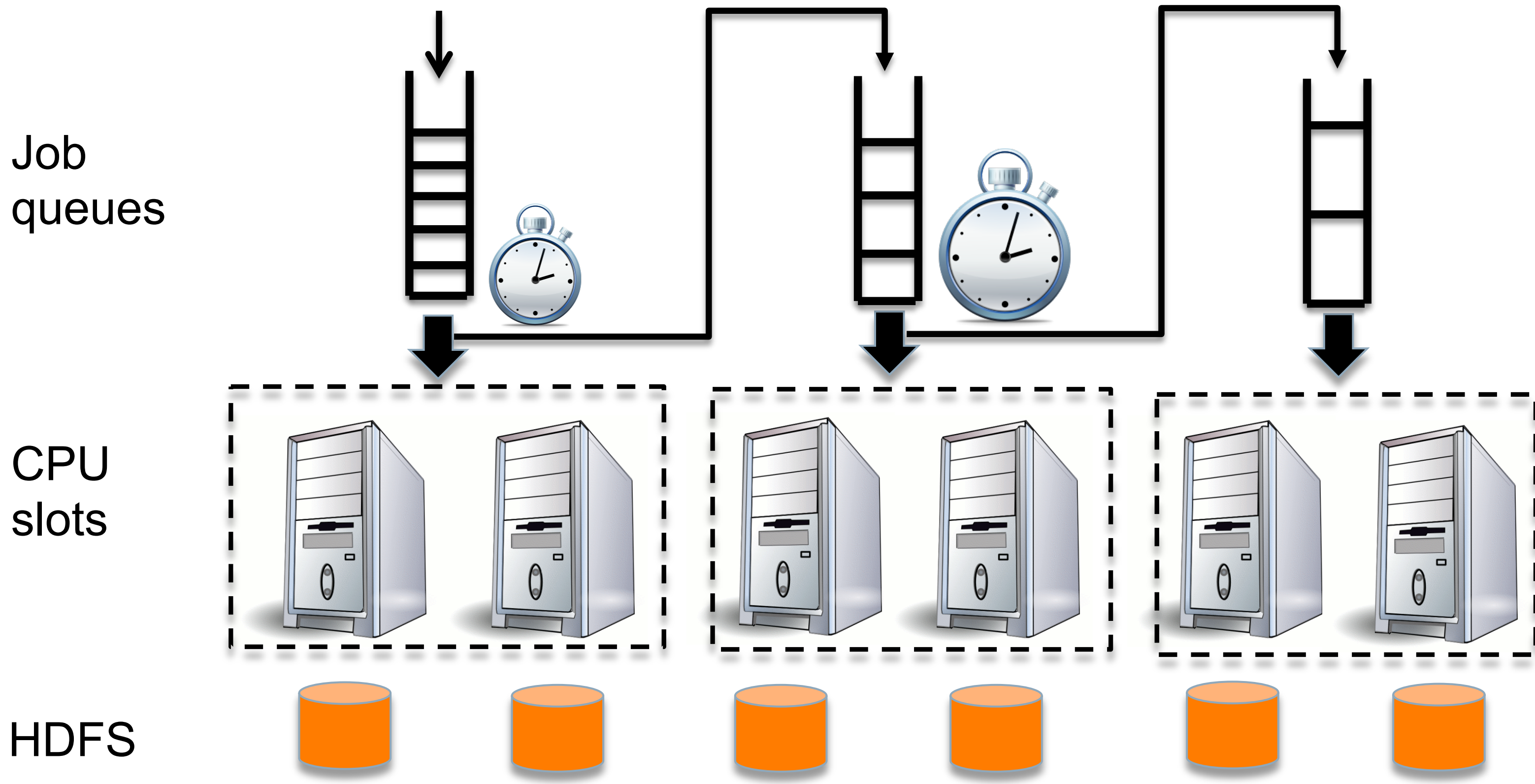
Current tasks run till completion.

Next tasks start in the next queue.

Intermediate data is saved on local disks.

The TAGS policy

Partitions	YES
Feedback	YES



Move job to the **next queue** when it exceeds the timer using capacity from the **current partition**.

(+) Jobs are elastic and they are not killed.

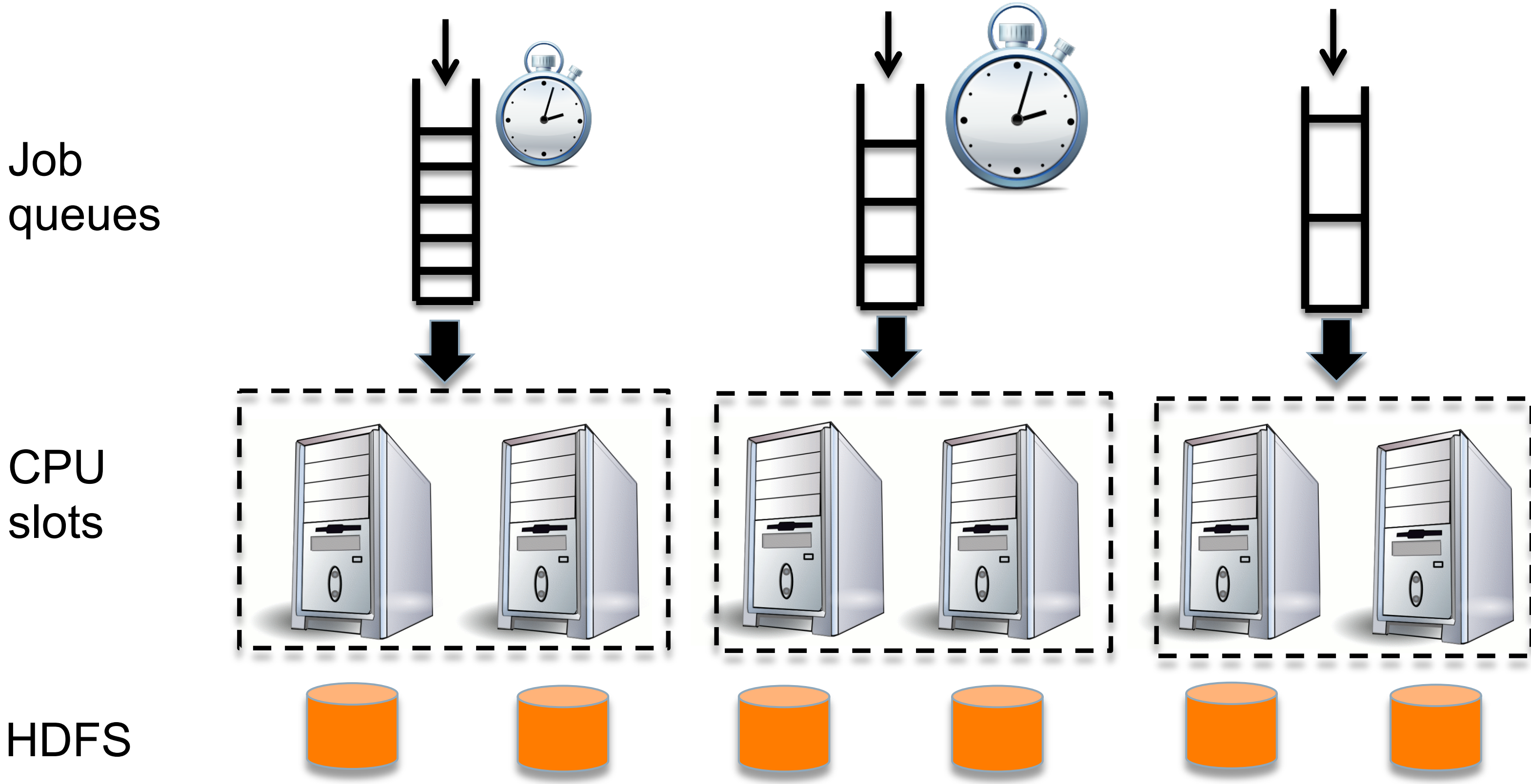
(+) Partition capacities in addition to time cutoffs.

M. Harchol-Balter, "Task assignment with unknown durations", Distributed Computing Systems, 2001.

The SITA policy

Partitions	YES
Feedback	NO

Job sizes PREDICTED



Dispatch and run jobs **till completion** in their **own partition**.

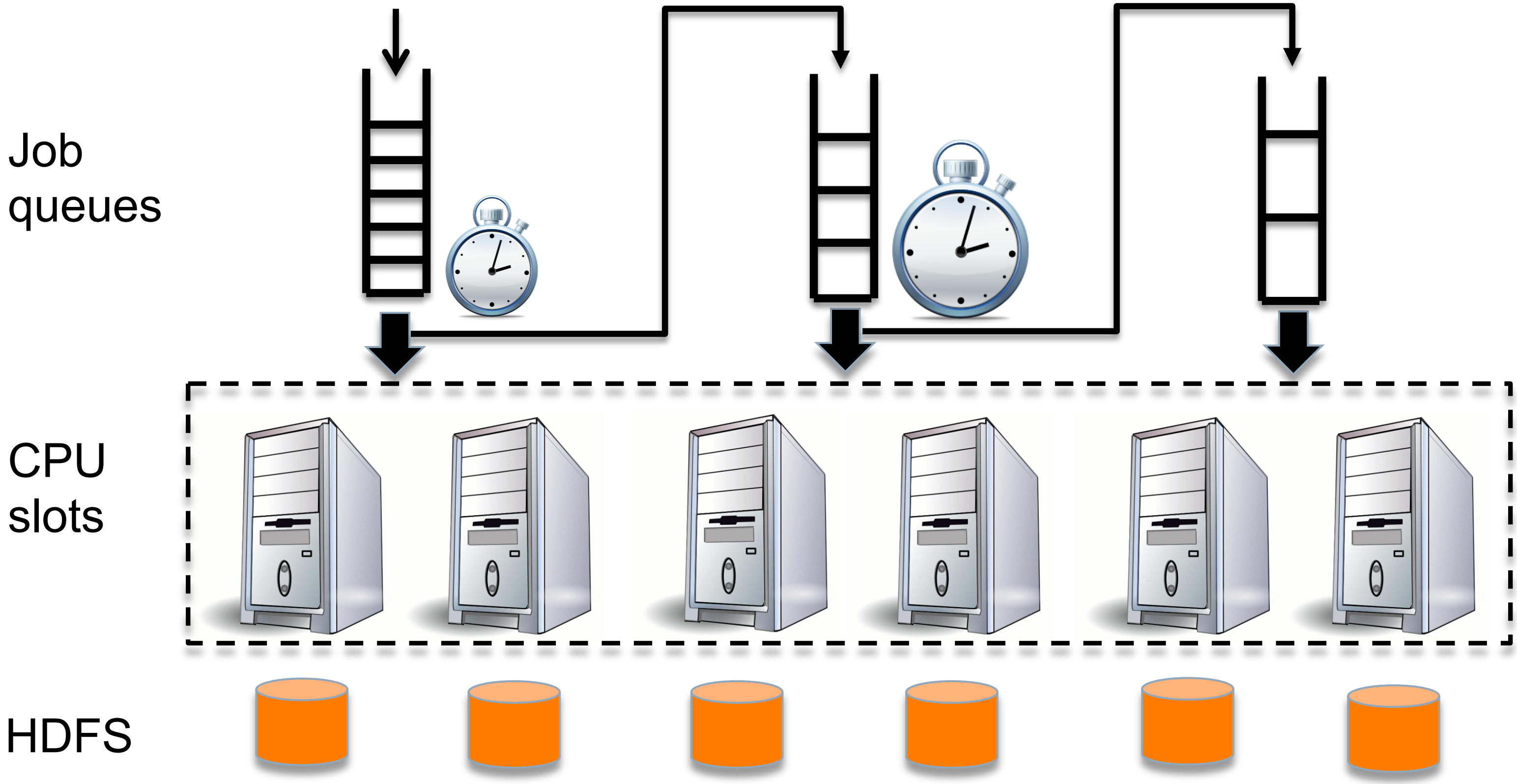
(+) Jobs are elastic and they are not killed.

(+) Partition capacities in addition to time cutoffs.

M. Harchol-Balter et. al., “On choosing a task assignment policy for a distributed server system”, JPDC, 1999.

The FBQ policy

Partitions	NO
Feedback	YES



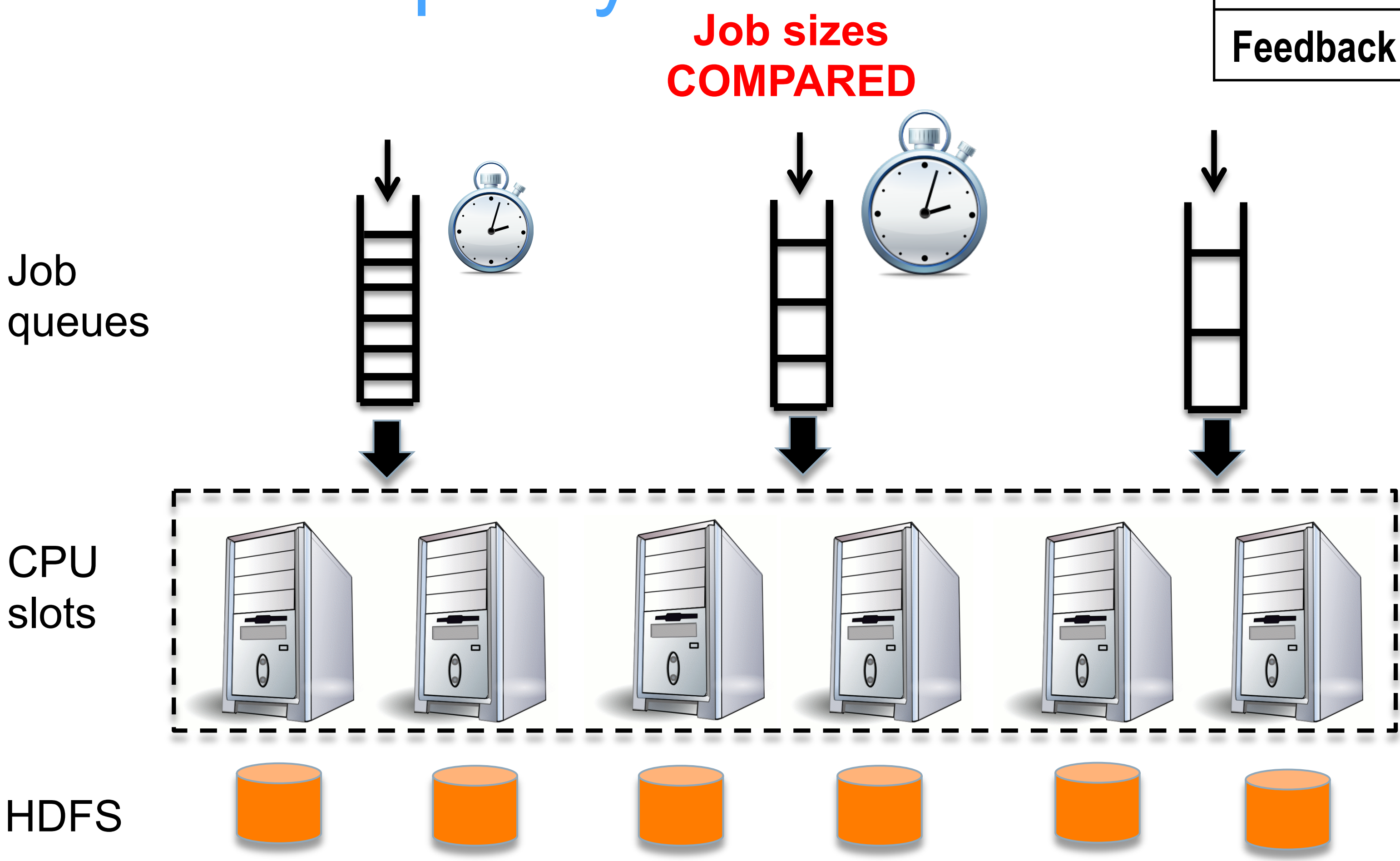
Move job to the **next queue** when it exceeds the timer using capacity from the **complete system**.

(+) Enables resource multiplexing.

L.E. Schrage, “The M/G/1 queue with feedback to lower priority queues”, Management Sciences, 1967.

The COMP policy

Partitions	NO
Feedback	NO



Given K queues, dispatch job to queue m+1 if it is **larger than m out of K-1** previously finished jobs.

(+) Enables resource multiplexing.

Jian Tan et. al., “Adaptive scalable comparison scheduling”, ACM SIGMETRICS, 2007.

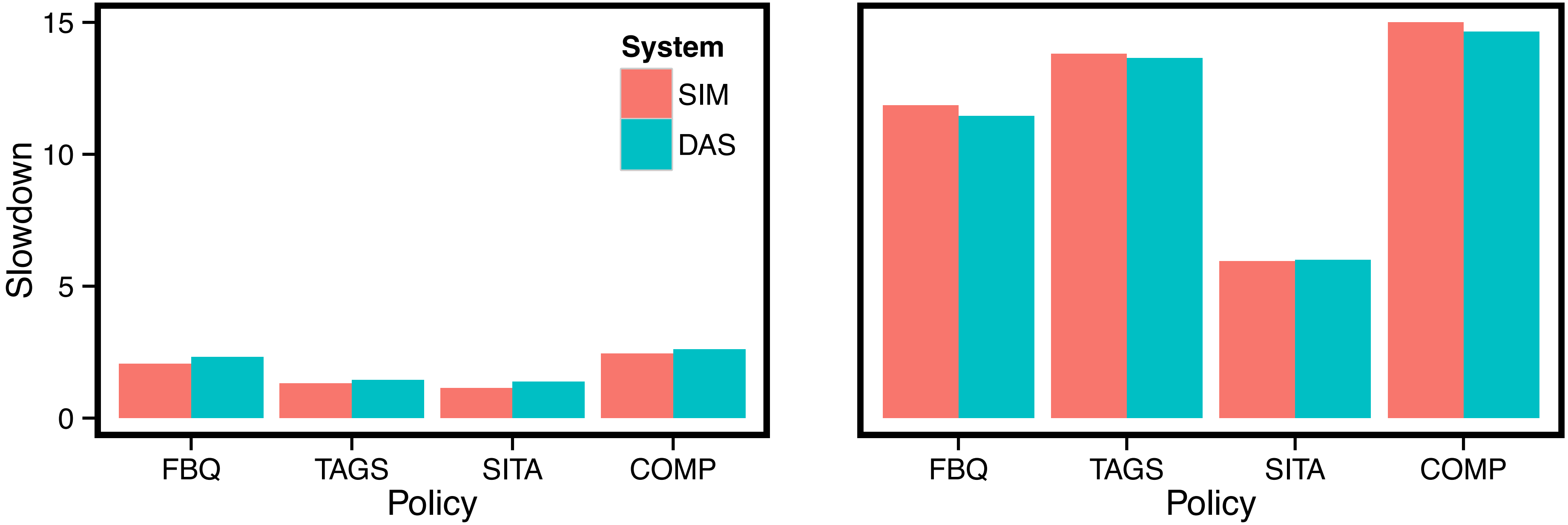
Simulator validation (1/2)

Mumak versus Hadoop on DAS-4:

- 10 nodes with 6 map slots and 2 reduce slots.
- Single jobs: Grep, Sort, Wordcount.

Applications	Maps	Reduces	Job Size [s]	SIM [s]	DAS [s]	Jobs
GREP	2	1	63.14	36.10	43.26	26
SORT	4	1	60.20	32.70	39.97	4
WCOUNT	4	1	126.14	42.04	49.73	4
GREP	50	5	155.32	42.83	53.18	4
WCOUNT	100	10	3,790.46	86.80	93.62	3
SORT	200	20	5,194.64	149.92	156.89	3
GREP	400	40	15,697.18	233.63	239.21	3
WCOUNT	600	60	26,662.53	579.73	589.02	3

Simulator validation (2/2)



Median **95th**

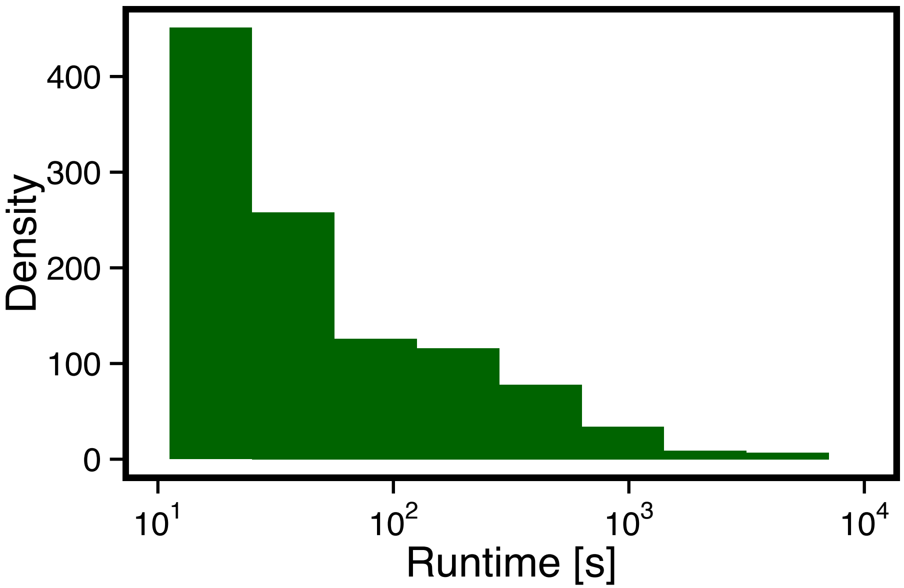
Workload of 50 jobs, system load of 0.7.

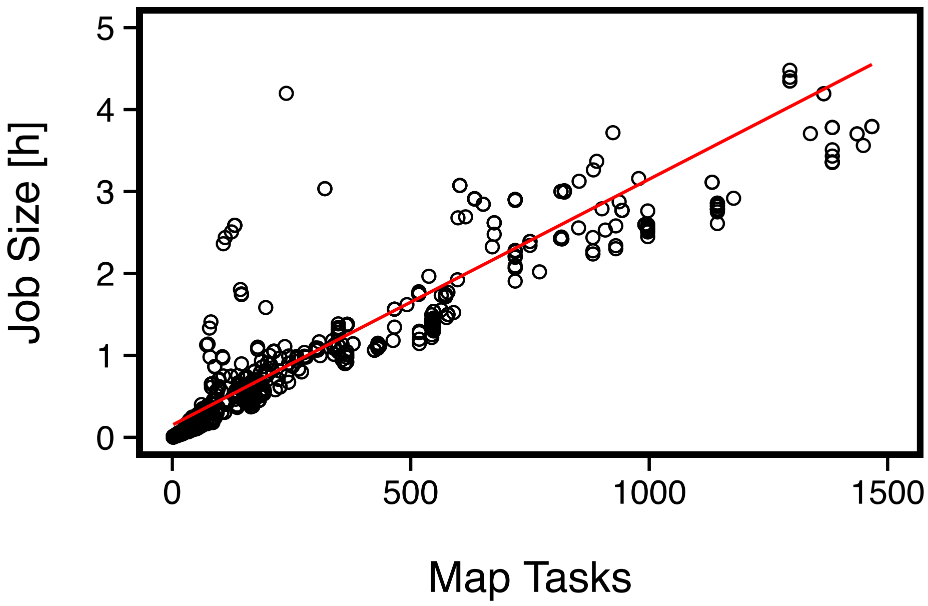
Less than 1% error between SIM and DAS.

TU Delft

Facebook workload

In all experiments: 100 nodes, 60h workload, CV^2=16.35.





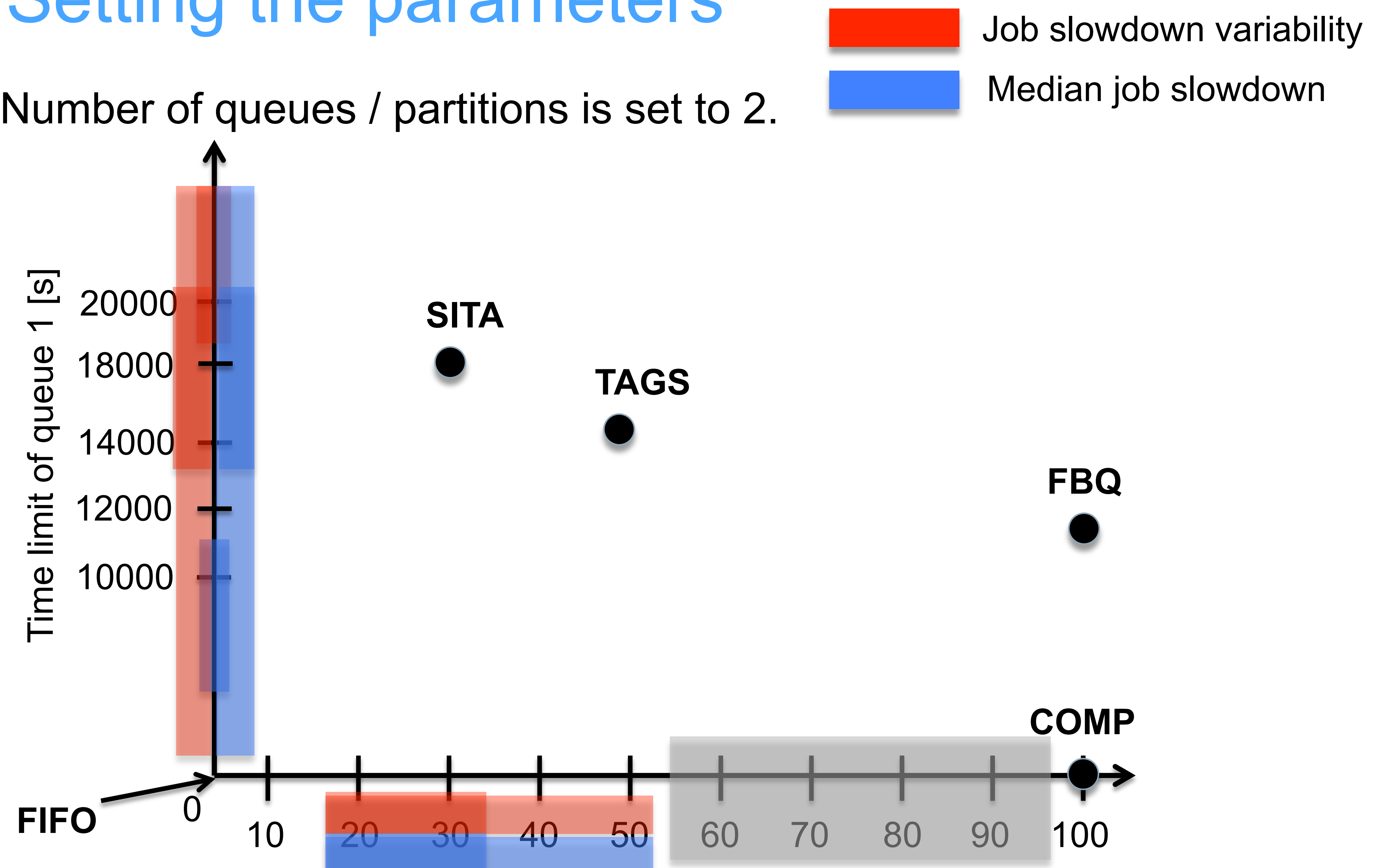
Less than 8% of the jobs account for 50% of the total load.

Strong correlation between input size and job size.

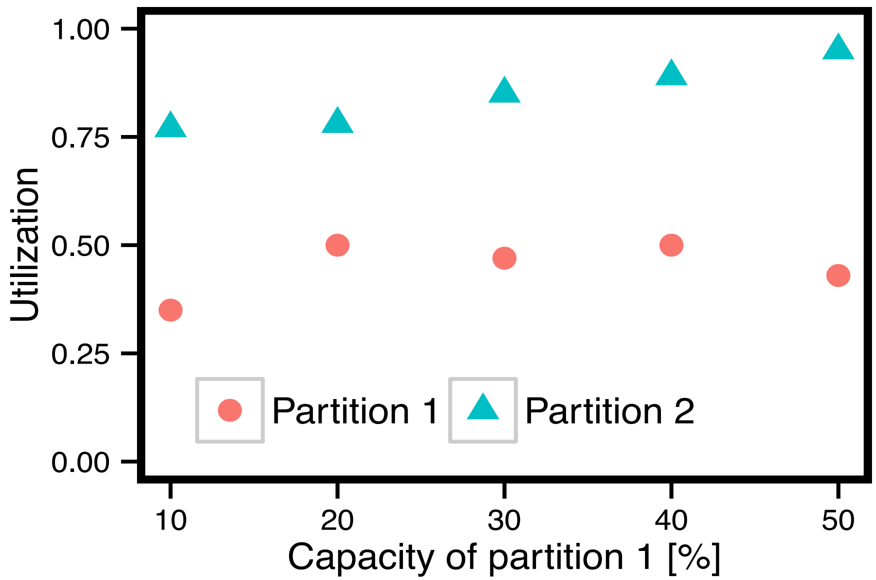
SWIM workloads: https://github.com/SWIMProjectUCB/SWIM/wiki

Setting the parameters

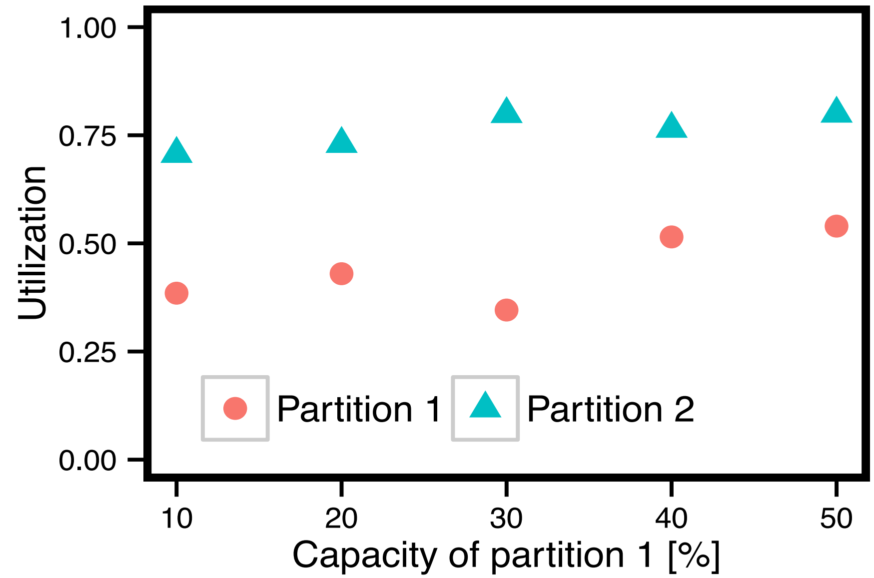
Number of queues / partitions is set to 2.



Load unbalancing



TAGS

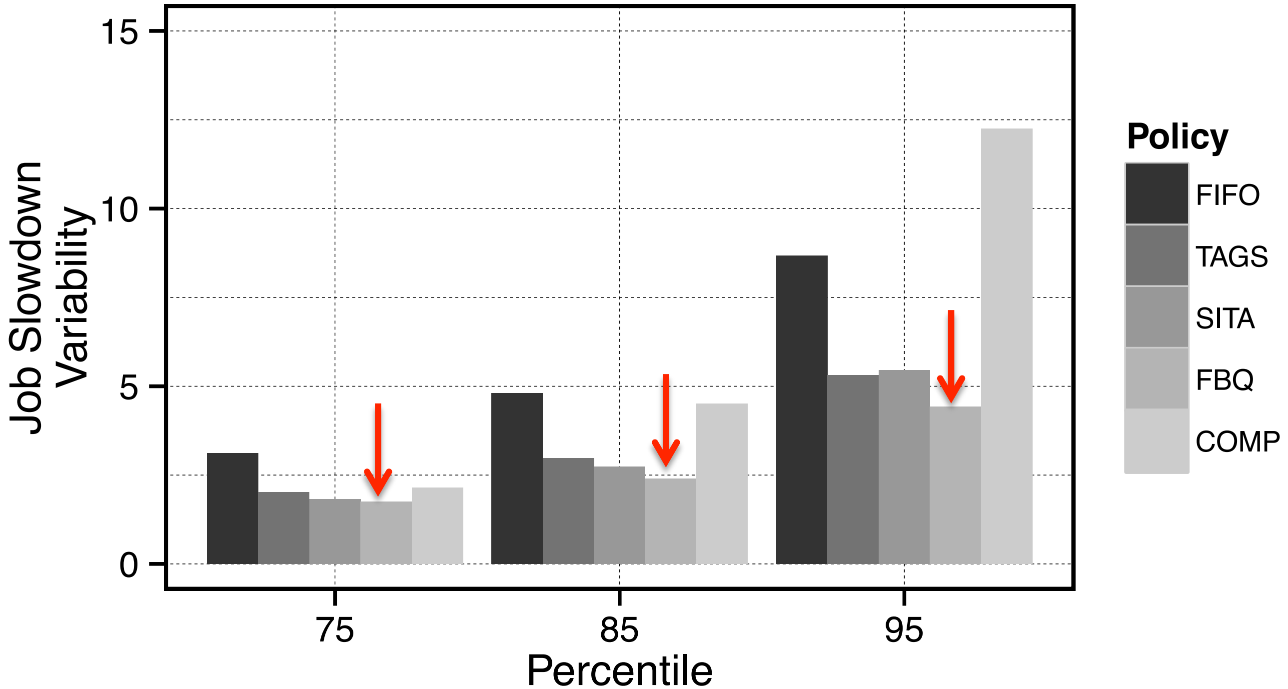


SITA

Partition 1 has significantly lower load than partition 2.

Short jobs run under low load in partition 1.

Fairness analysis (1/2)



TAGS and SITA shift variability to partition 2.

FBQ reduces slowdown variability by a factor of 2.

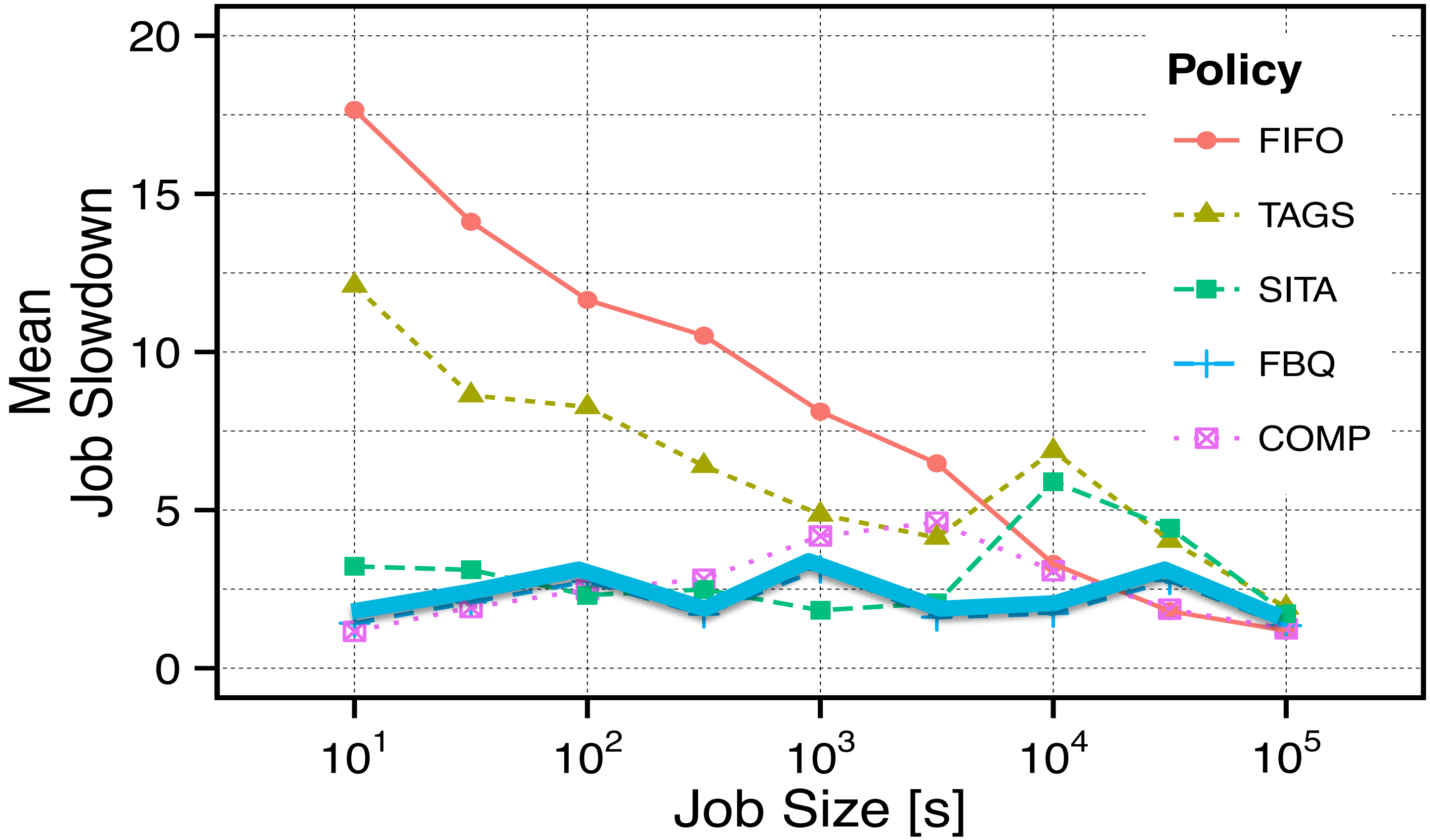
FBQ < SITA < TAGS < COMP < FIFO

Best

Worst

TU Delft

Fairness analysis (2/2)



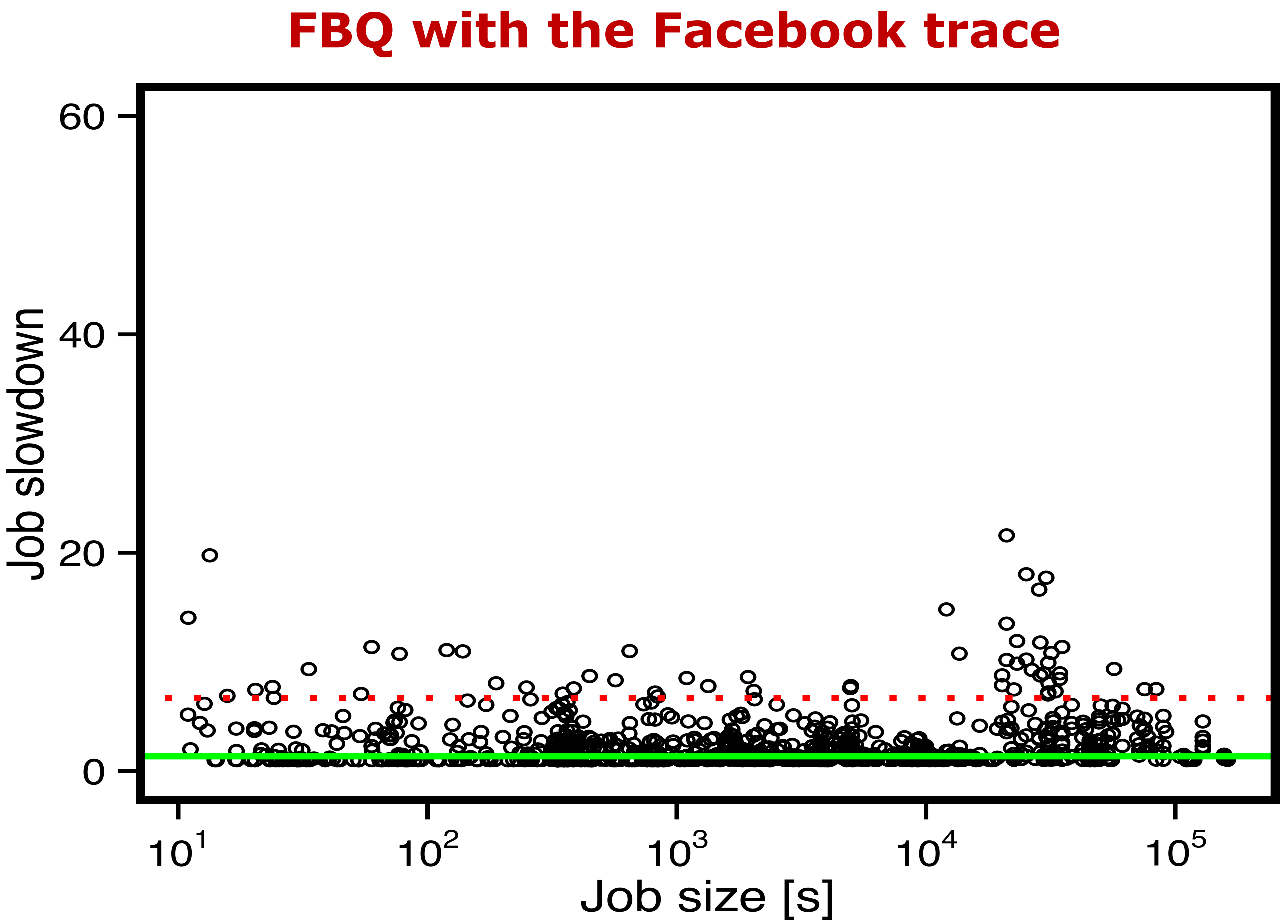
FBQ is stable across all job size ranges.

FBQ < SITA < COMP < TAGS < FIFO

Best

Worst

Goal achieved



Conclusions

There is much job slowdown in data analytics frameworks.

We use logical partitioning and system feedback to prevent short jobs suffering too much.

Out of the four policies, the FBQ policy is the best.

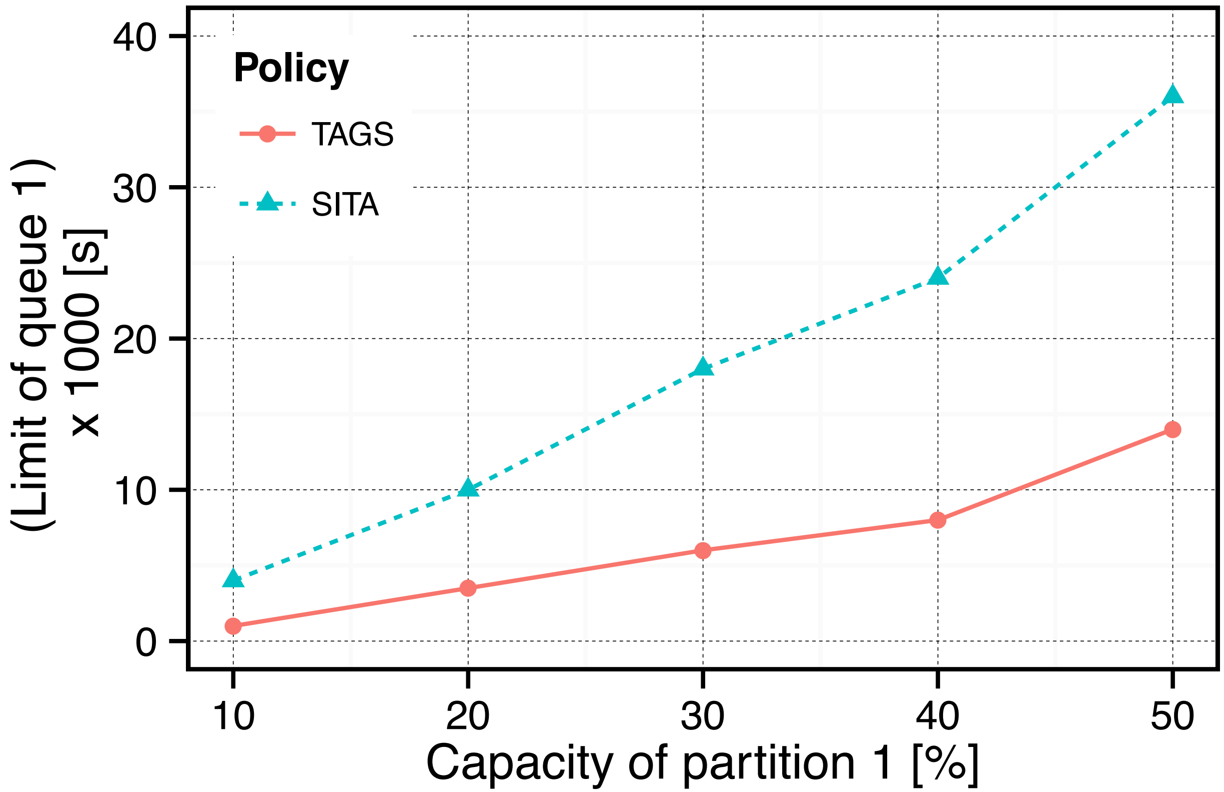
Backup slides

Contrasting the policies

Previous work	Our work
Single or distributed-server model	Datacenters with very large capacity
Simple, rigid non-preemptive jobs	Malleable MapReduce jobs
Wasted work by killing jobs	Work-conserving approach

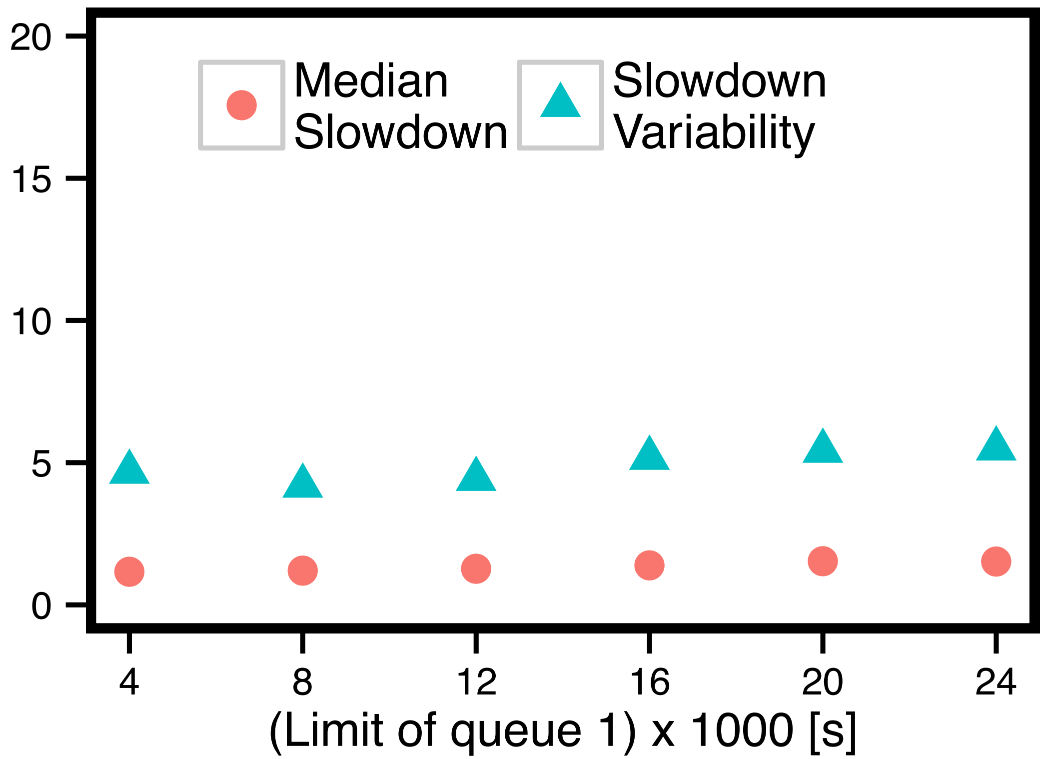
Policy	Queues	Partitions	Feedback	Job Size	Param.
FIFO	single	no	no	unknown	0
FBQ	multiple	no	yes	unknown	K
TAGS	multiple	yes	yes	unknown	$2K-1$
SITA	multiple	yes	no	predicted	$2K-1$
COMP	multiple	no	no	compared	1

Optimal time limits



With SITA jobs run to completion, hence the higher time limit of partition 1.

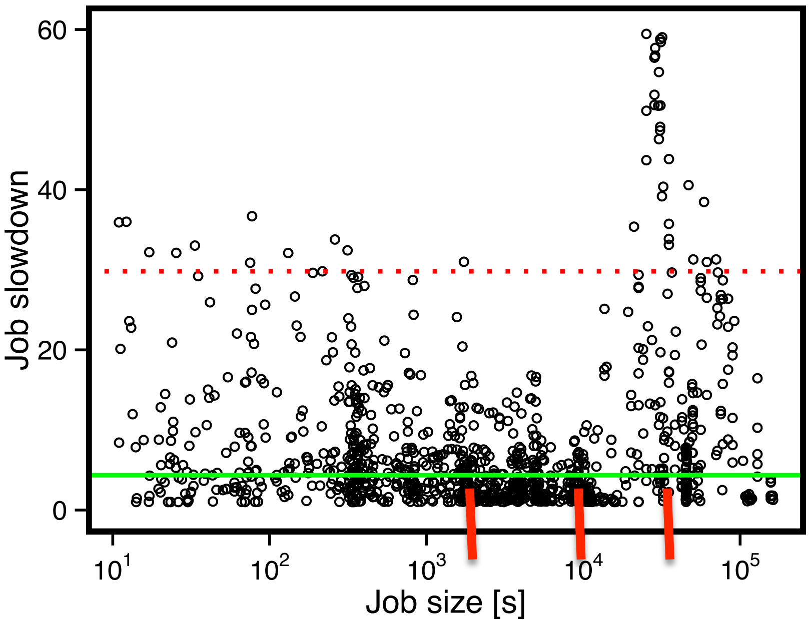
Performance of FBQ



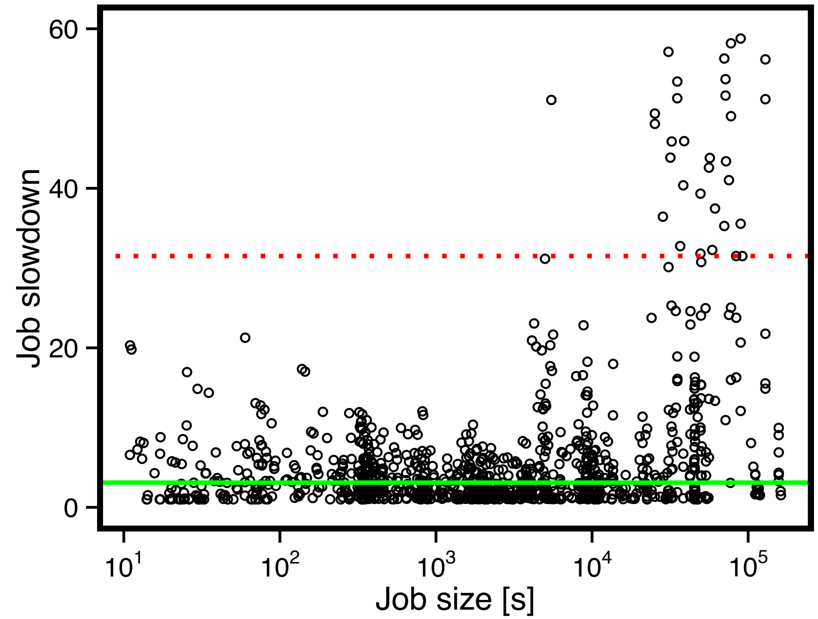
FBQ is very insensitive to the queue time limit.

More than two queues

FBQ with K=2, sys. load 0.9.



FBQ with K=4, sys. load 0.9.



Improves median slowdown by 30%.

No short job is over the 95th percentile.

TUDelft