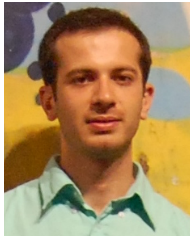
Better Safe than Sorry: Checkpointing In-Memory Data Analytics Applications



Bogdan Ghit and Dick Epema

ACM HPDC 2017
Washington D.C.

Distributed Systems Group
Delft University of Technology
Delft, the Netherlands

TU Delft

About me

PhD degree from TU Delft, advised by Dick Epema

Thesis topic on scheduling data analytics frameworks

This talk is about Panda

Hadoop

Fawkes

Tyrex

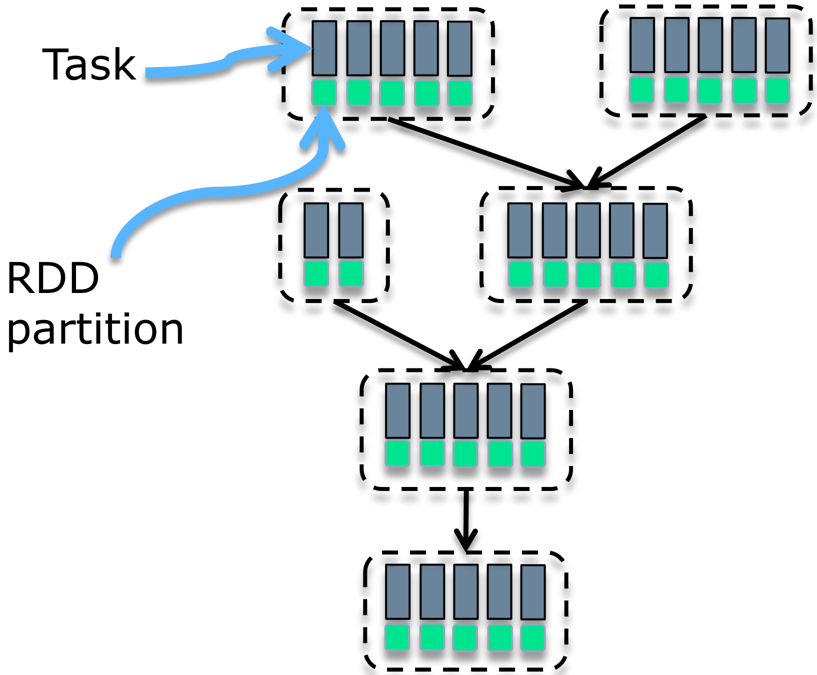
TUDelft

Call for Efficiency

Large-scale data processing is now widespread

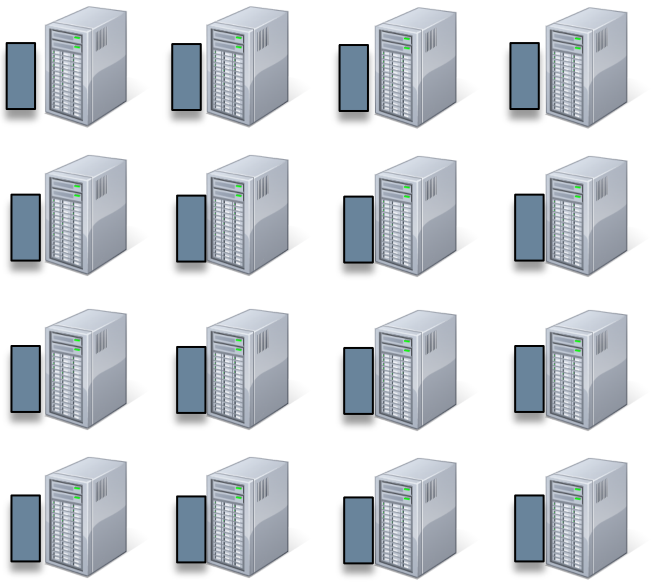
TUDelft

Spark Scheduling Model



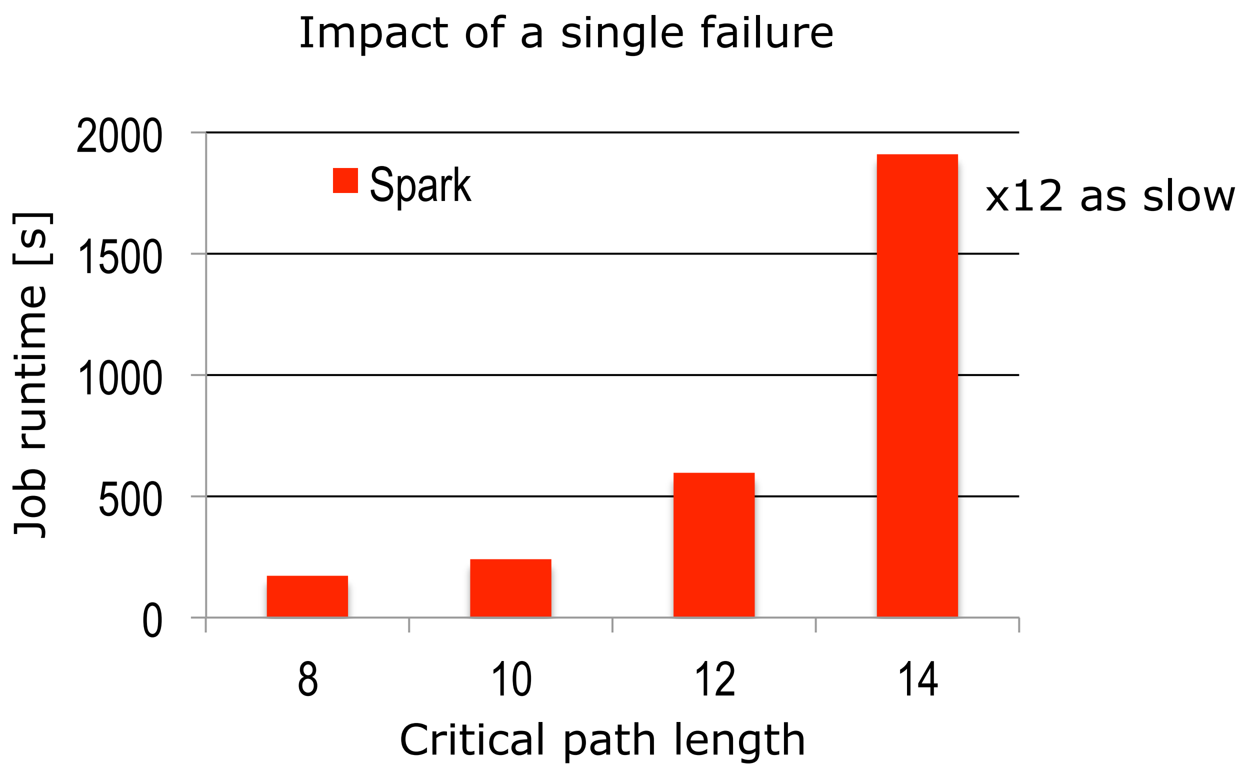
In-memory parallel computation

DAG-aware scheduling

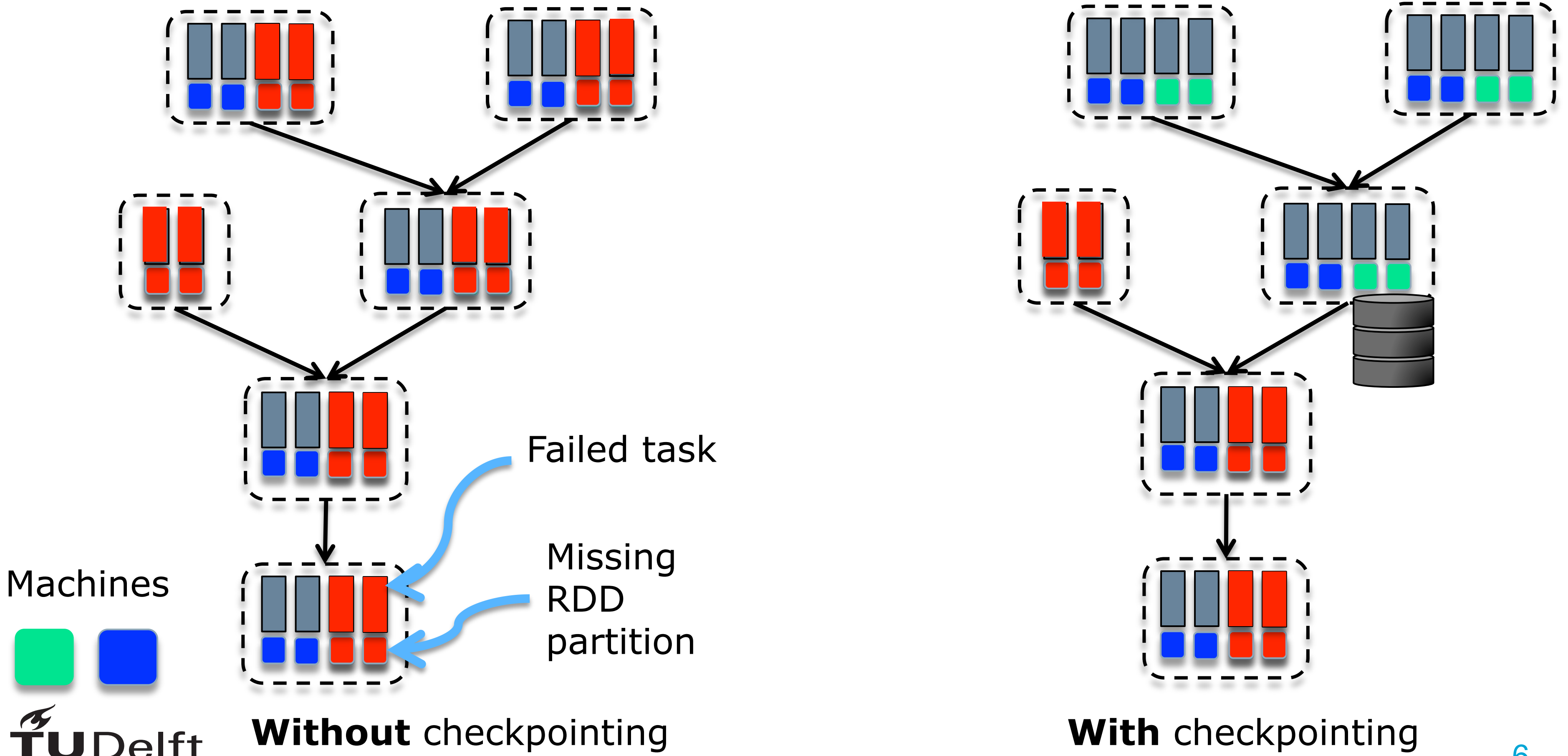


Task to slot allocation

Resilient but Inefficient by Design



Recomputation vs. Checkpointing



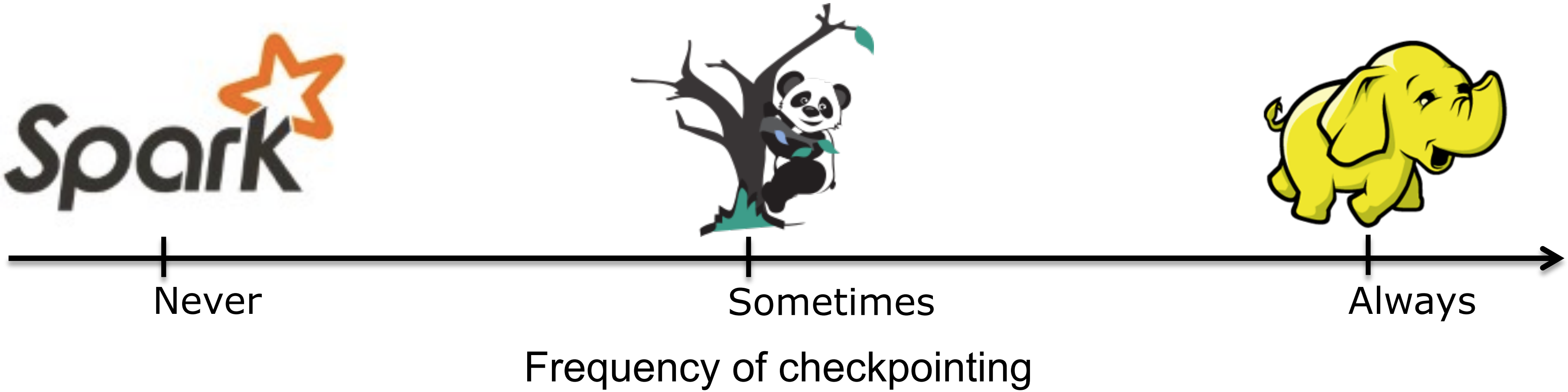
The Case for Checkpointing

Failure type	Rate (per year)	Machines lost	Downtime
Overheating	0.5	All	1-2 days
PDU	1	500-1000	6 h
Net. rewiring	1	5%	2 days
Racks	20	40-80	1-6 h
Servers	1000	-	-
HDD	1000s	-	-

Cycle scavenging on cheap but unreliable spot instances reduces costs by 50-60%

Jeff Dean, https://www.cs.cornell.edu/projects/ladis2009/talks/dean-keynote-ladis2009.pdf

Where We Want to Go

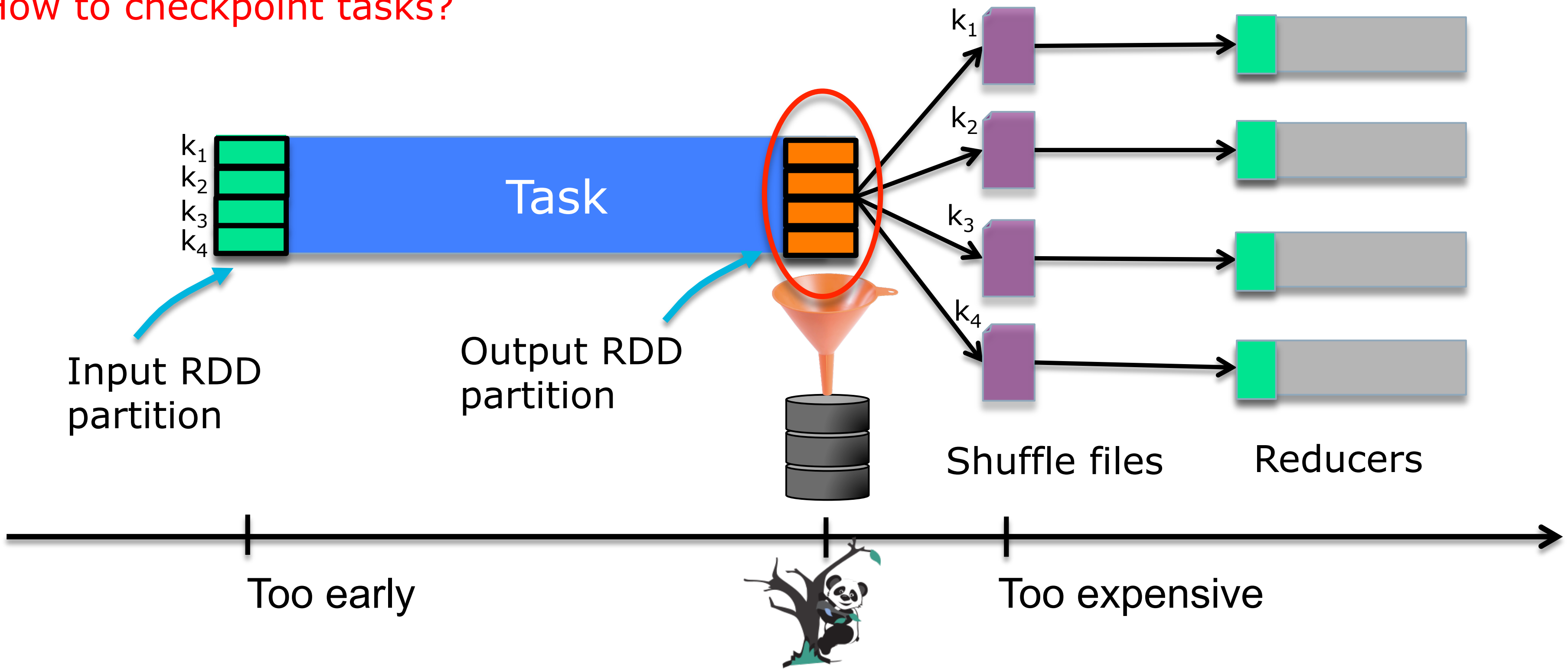


Checkpointing as a task-selection problem:

1) How to checkpoint tasks?
2) Which tasks to checkpoint?

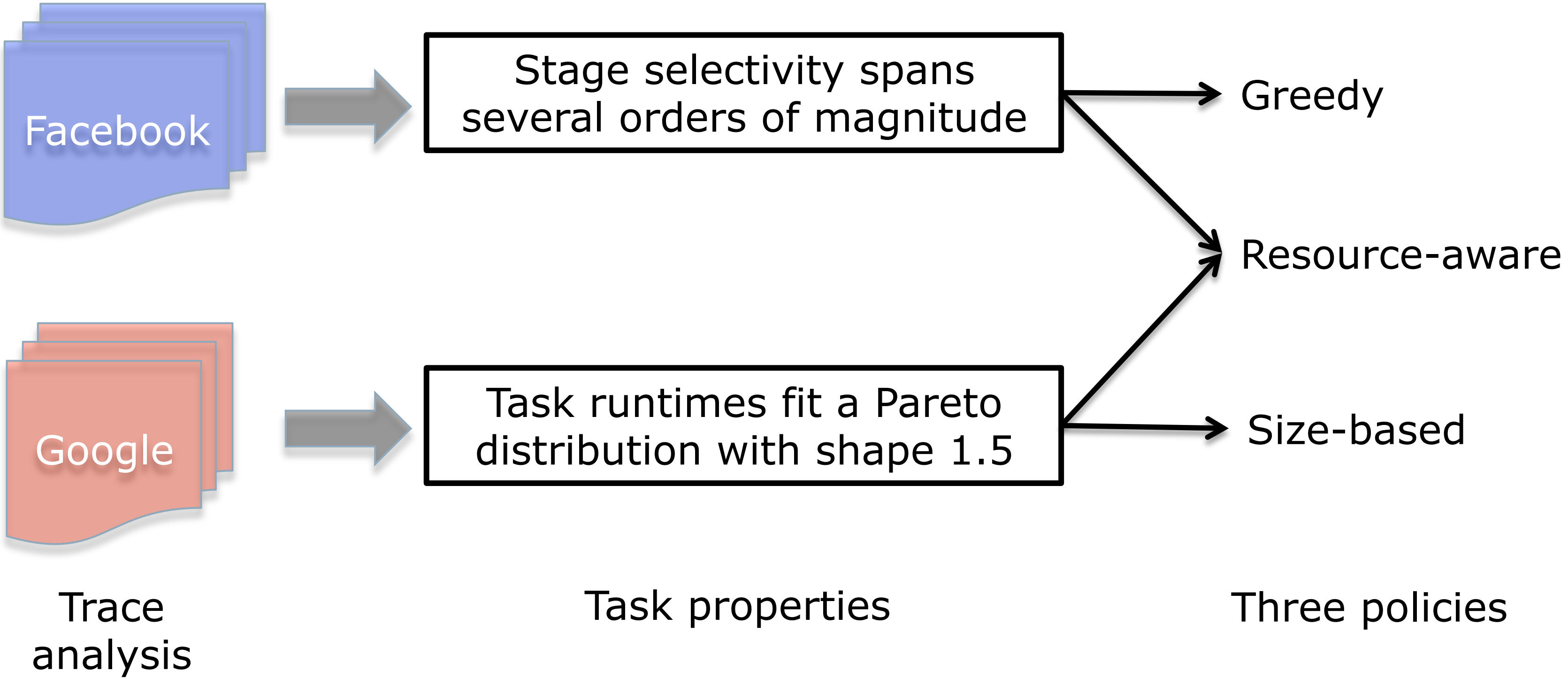
Checkpointing Tasks

How to checkpoint tasks?

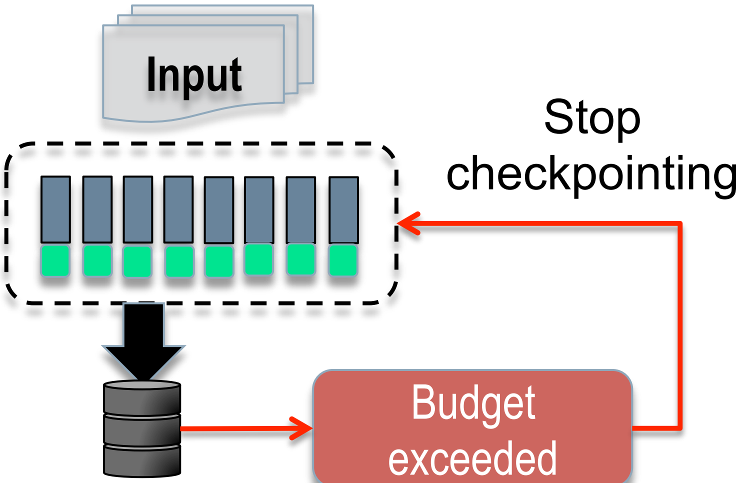


Policy Framework

Which tasks should be checkpointed?



Greedy Checkpointing



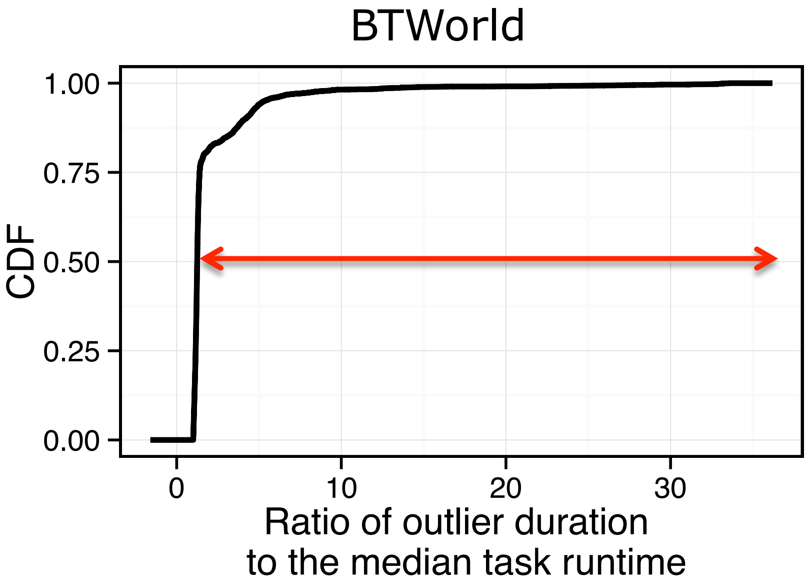
Task selection

- As many tasks as the budget allows
- Inflight checkpointing tasks are allowed to finish

The checkpointing budget

- Limits the checkpointing cost in each stage
- Set to a fraction of the total stage input

Size Checkpointing



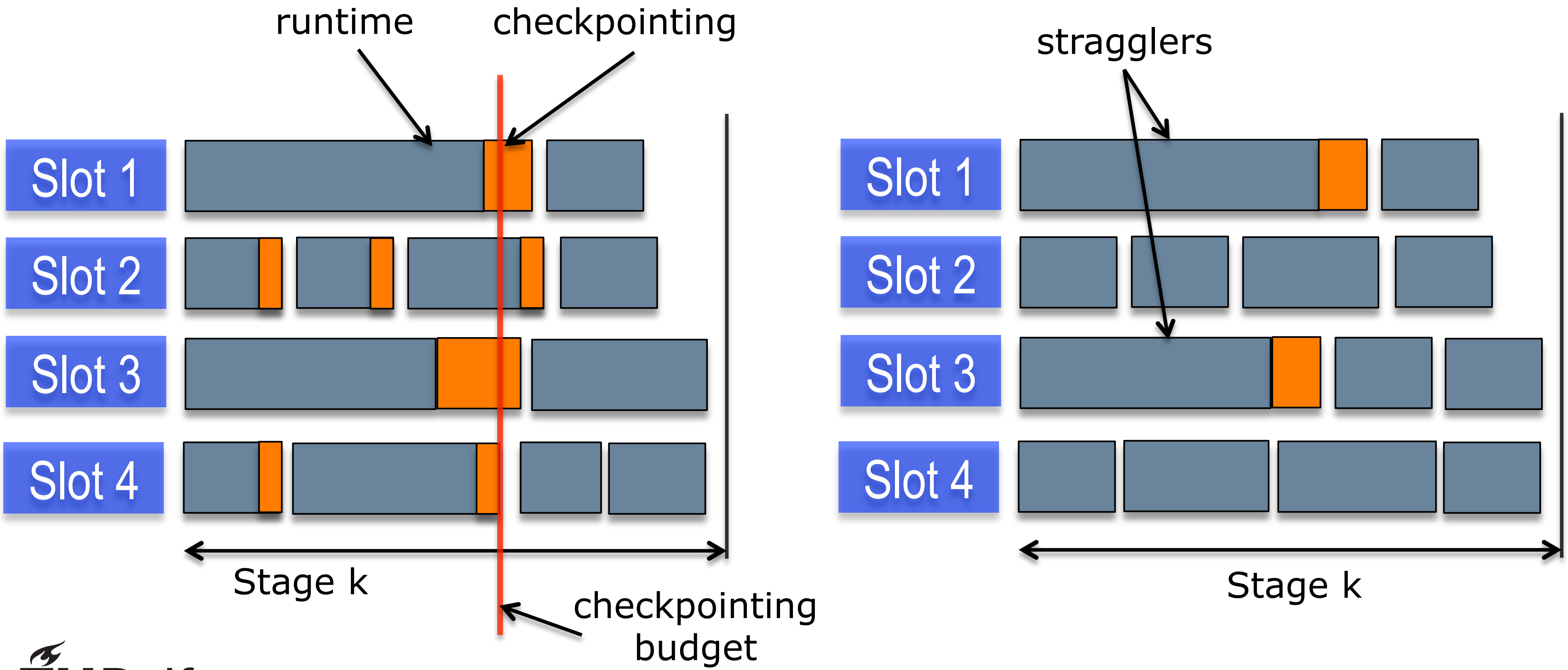
Task selection

- Straggler tasks that run very slow
- Avoid recomputing time-consuming tasks

Identify stragglers

- Build up a history of task runtimes per job
- Tasks **m** times as long as the median runtime

Greedy versus Size

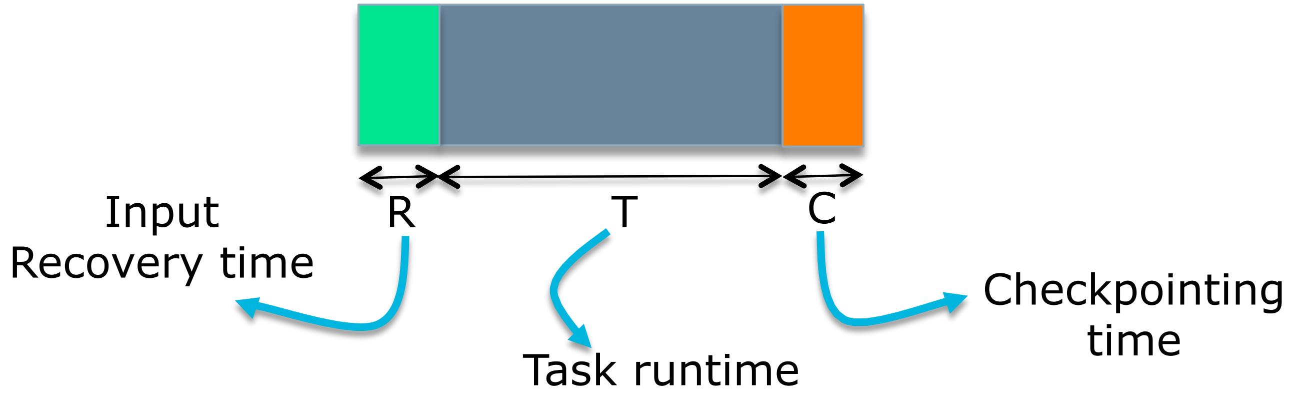


TU Delft

Aware Checkpointing

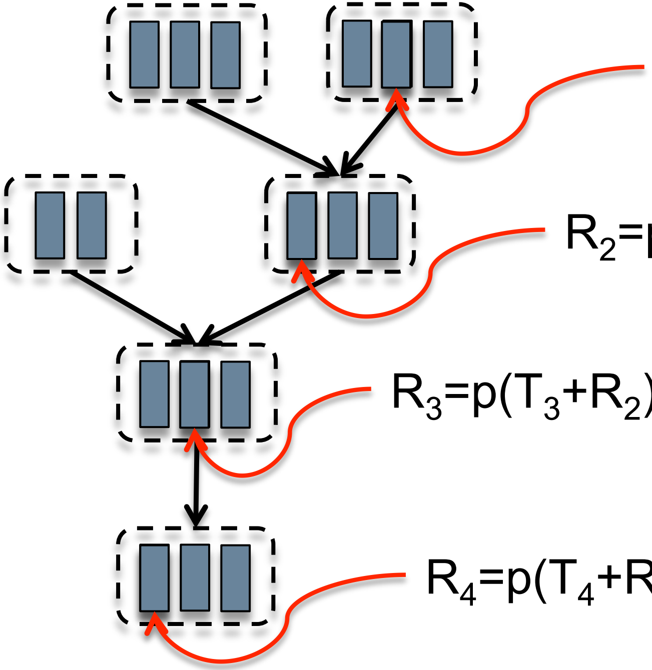
Task selection

- Estimated benefit outweighs the checkpointing cost



- Checkpoint tasks if: p (T + R) > C, p is the likelihood of failure

Recomputation Cost



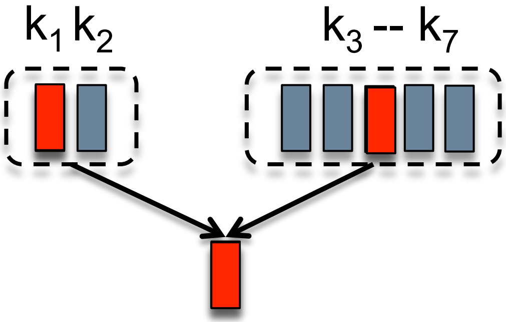
$R_1 = pT_1$

$R_2 = p(T_2 + R_1)$

$R_3 = p(T_3 + R_2)$

$R_4 = p(T_4 + R_3)$

Single recovery path



$R_i = p(T_i + R_m)$

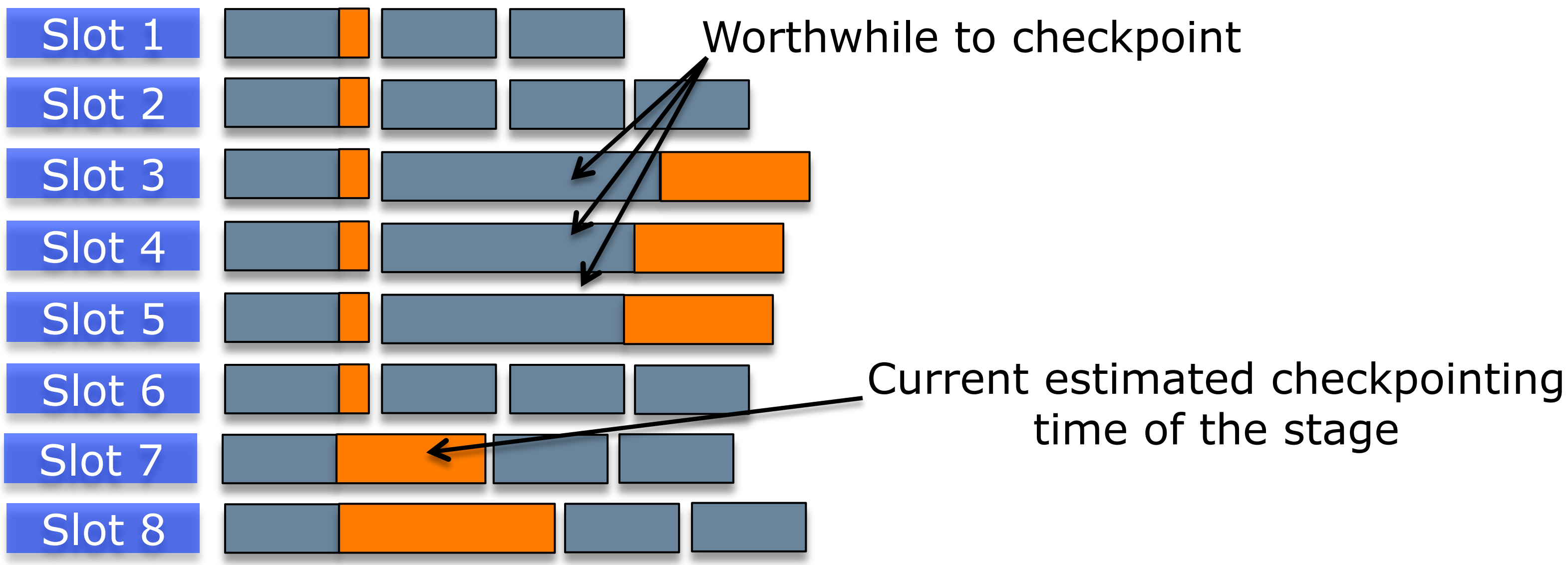
$R_m = \max\{R_{k1}, R_{k5}\}$

Multiple recovery paths

Checkpointing Cost

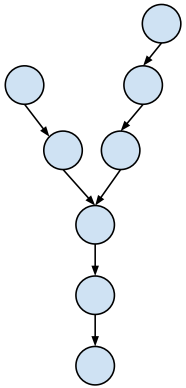
Difficult to anticipate

- Output size and write throughput
- Contention due to other tasks being checkpointed

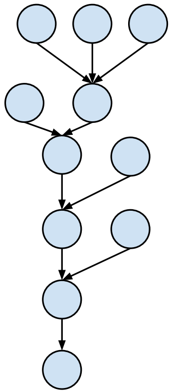


Stage 0 - input read from disk, so the recovery time is 0

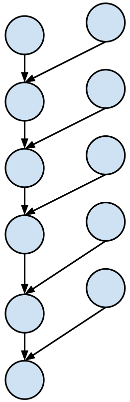
Application DAG Layouts



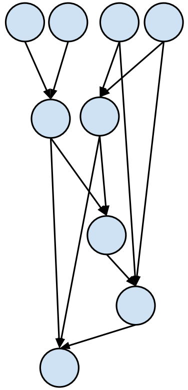
BTWorld
600 GB



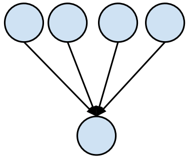
PPPQ
500 GB



NMSQ
500 GB



PageRank
1 GB



Kmeans
10 GB

Experimental Setup

Experiments using single real-world applications

Simulation using empirical workload

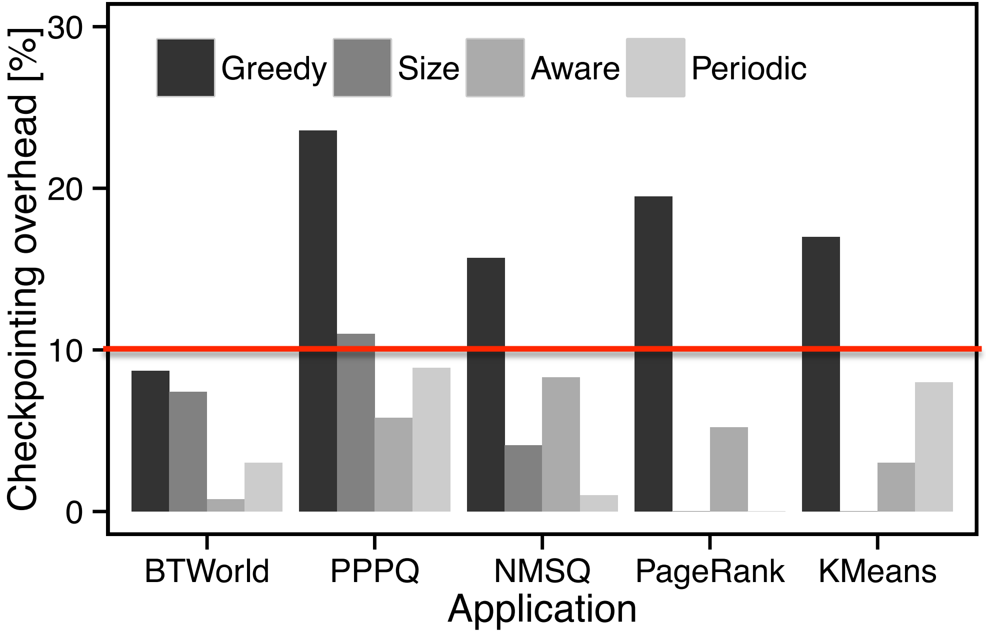
- *Greedy*: the **budget** is set to 10% based on the median selectivity of all tasks
- *Size*: **stragglers** are tasks running 1.5 times as slow as the median task runtime
- *Periodic*: Young's **optimal** checkpointing interval

Experiment 1

What is the checkpointing overhead in our policies?

Experiment details:
- 20-machine cluster
- All applications
- All policies

Takeaway: Size and Aware are very selective in checkpointing and have relatively low overheads.



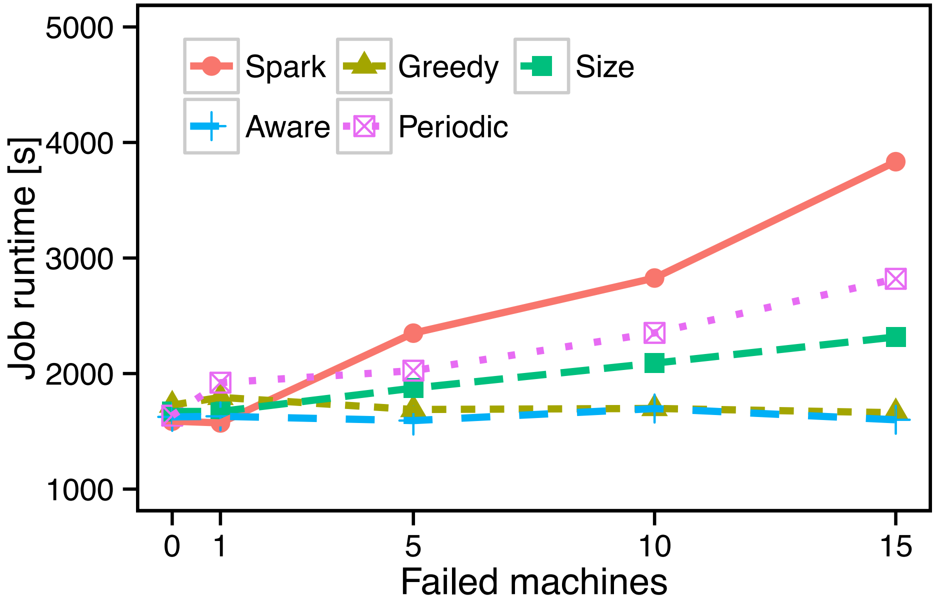
Experiment 2

How does the performance of our policies compare with periodic checkpointing?

Experiment details:

- 20-machine cluster
- BTWorld application
- All policies

Takeaway: Greedy and Aware deliver constant job runtimes for the complete range of failures.



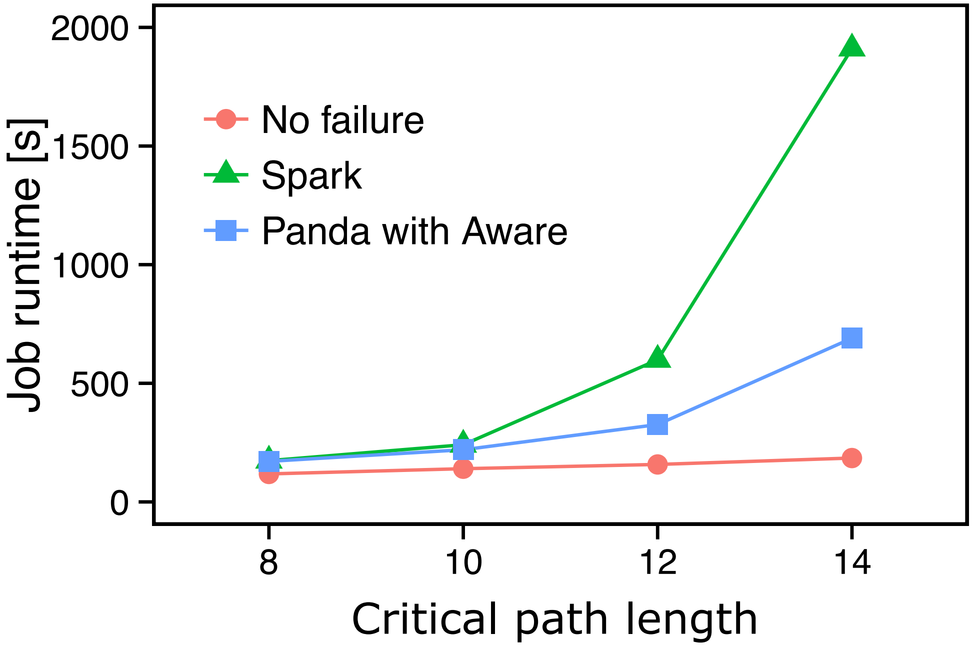
Experiment 3

What is the impact of the lineage length?

Experiment details:

- 5-machine cluster
- PageRank
- Aware policy

Takeaway: The Aware policy performs very well irrespective of the lineage length of the application.



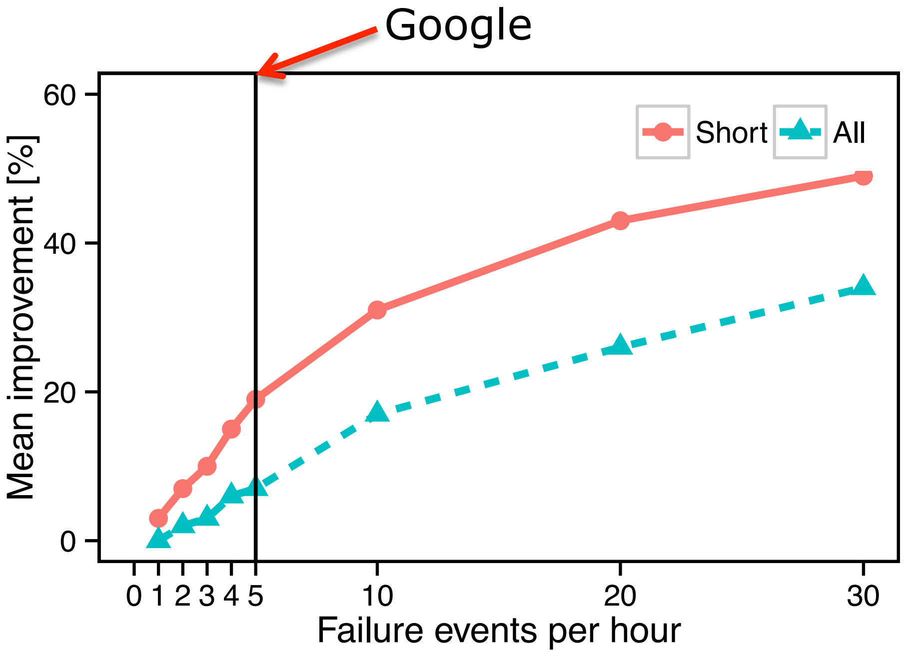
Simulation

What is the impact of the failure pattern?

Simulation details:

- 10,000-machine cluster
- Job profiles from experiments
- Short (< 30min) and long (>2h)

Takeaway: Panda stops being beneficial when the cluster experiences less than one failure per hour.



TUDelft

Takeaways

In-memory data analytics require ***checkpointing****,*

checkpointing is worthwhile ***if you do it right****,*

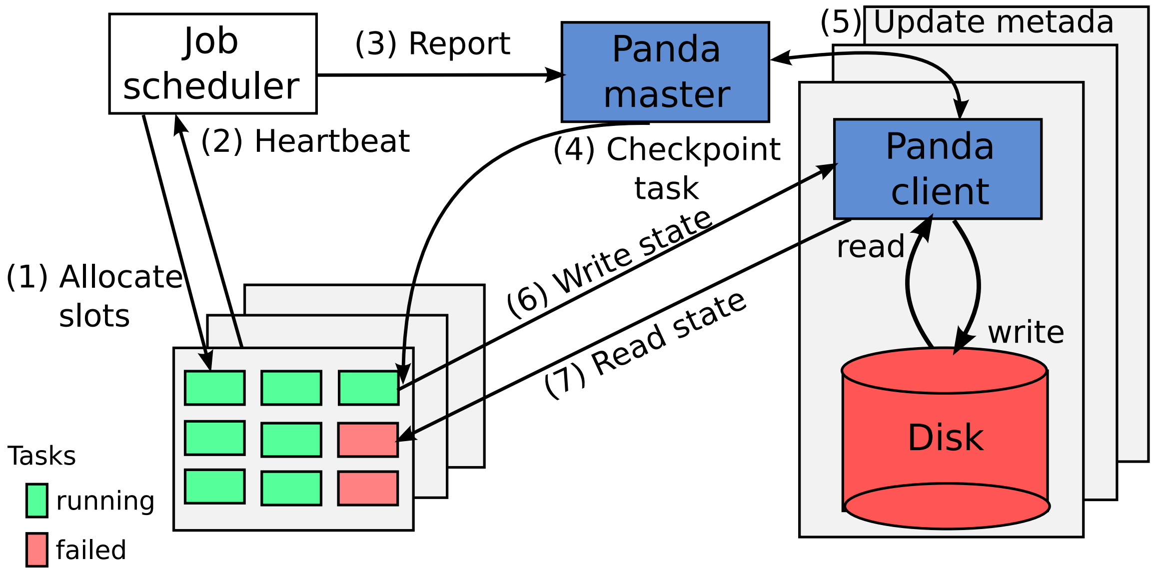
using Panda is the right way to do it!

TU Delft

Backup slides

TU Delft

Integration with Spark



- Fault-tolerance
- Scalability

Checkpointing is Important

Flint [EuroSys’16]

- Variation of periodic checkpointing
- Datasets fit in the cluster memory
- Failure of the complete cluster

TR-Spark [SoCC’16]

- Requires the distribution of task runtimes
- Requires the distribution of VM lifetimes

Panda [HPDC’17]

- More policies, more applications
- Both experiments and simulations

Remote Storage

Remote bulk object store (S3)

- 30-40 MB/s r/w performance per core
- Scales to 60-80 GB/s across 2800 simultaneous calls

E. Jonas et. al. “Occupy the Cloud: Distributed Computing for the 99%”