Jorge Quiané

Christoph Pinkel

Jörg Schad

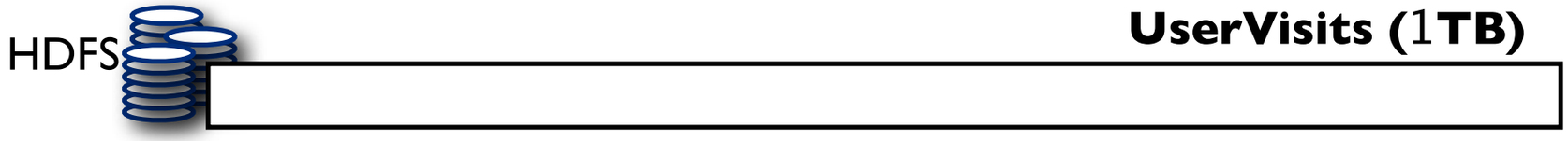
Jens Dittrich

SAARLAND UNIVERSITY

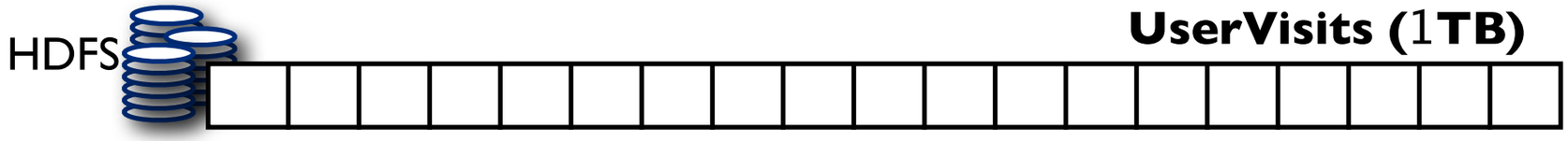
RAFTing MapReduce

Fast Recovery on the Raft

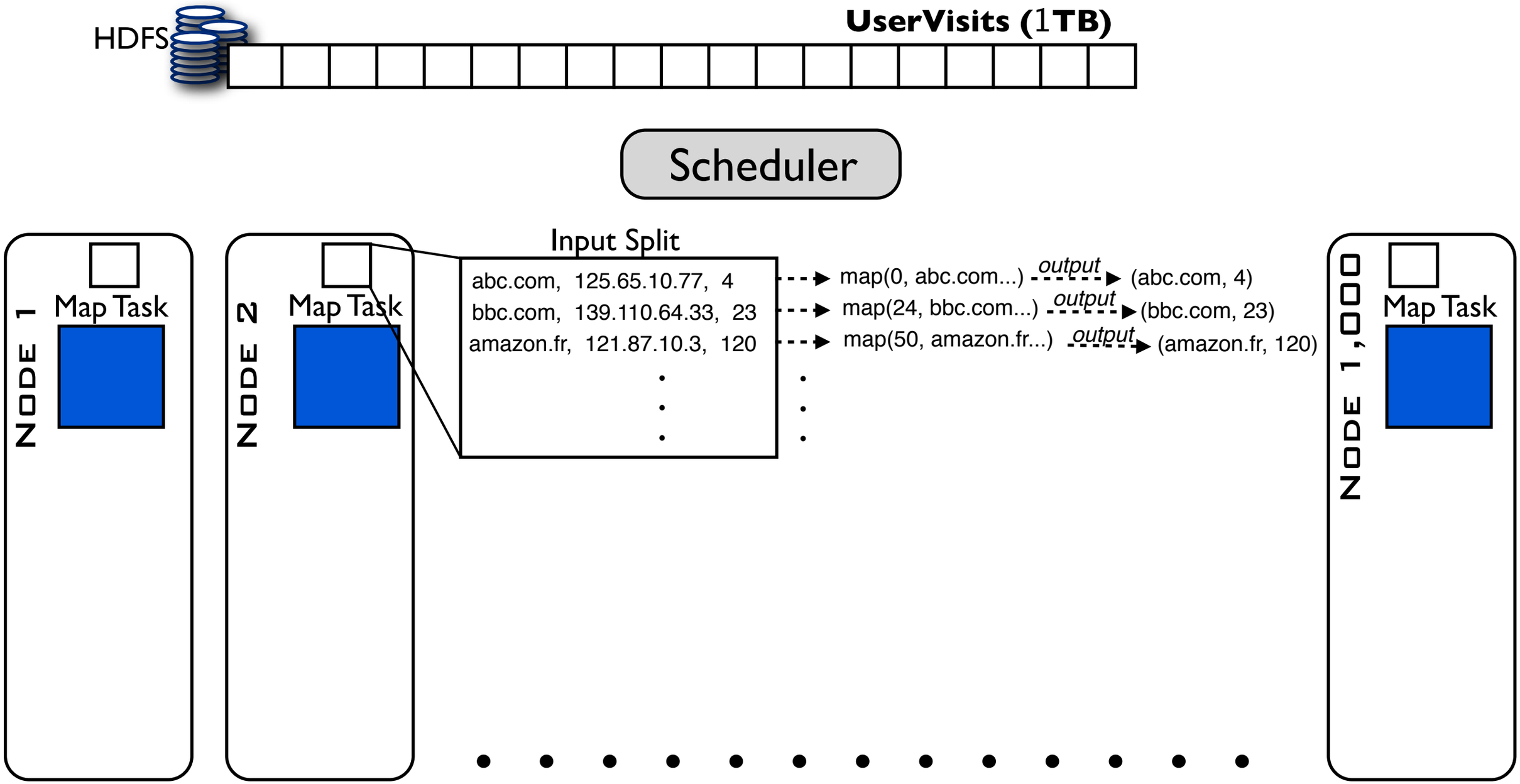
MapReduce (Recall)



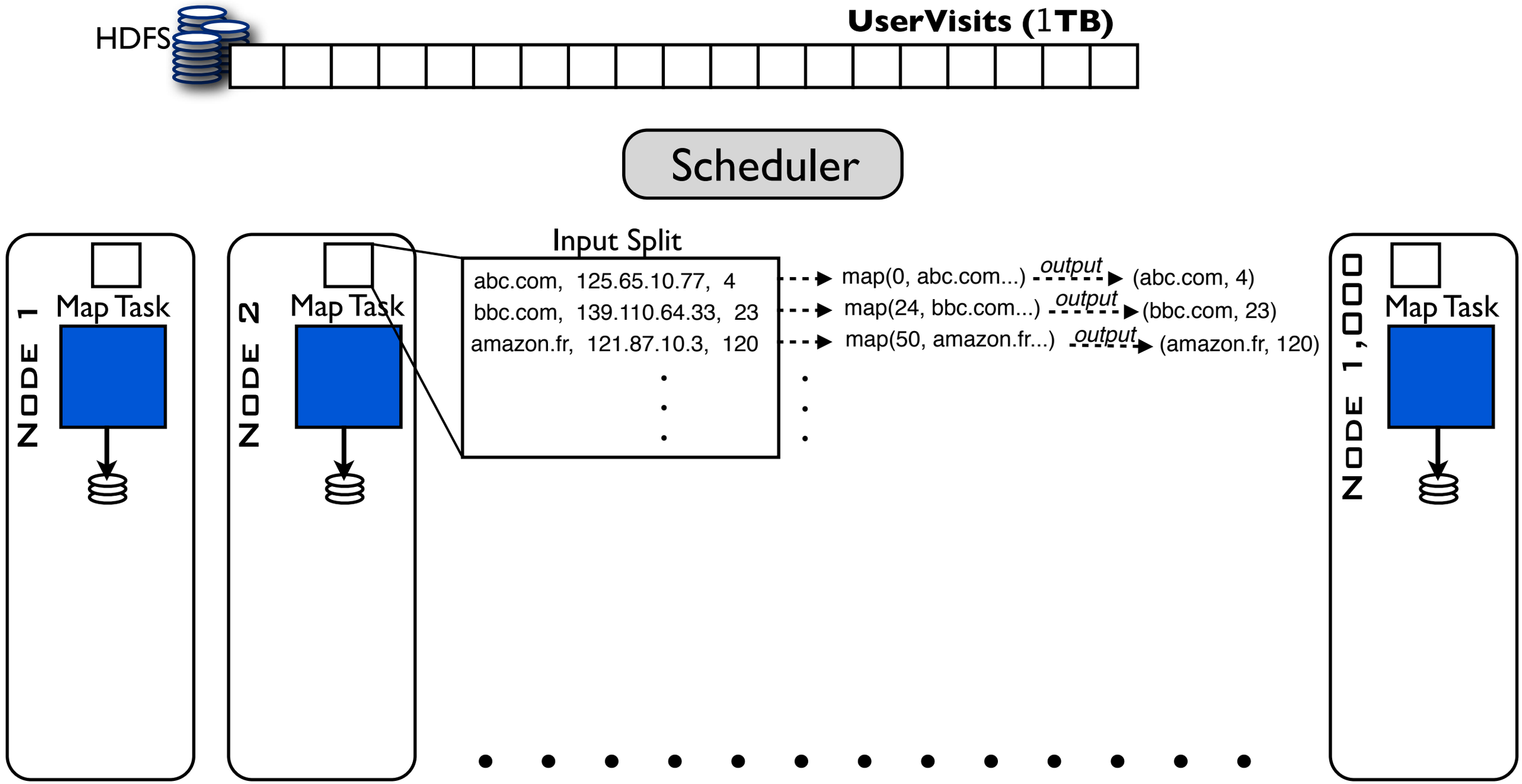
MapReduce (Recall)



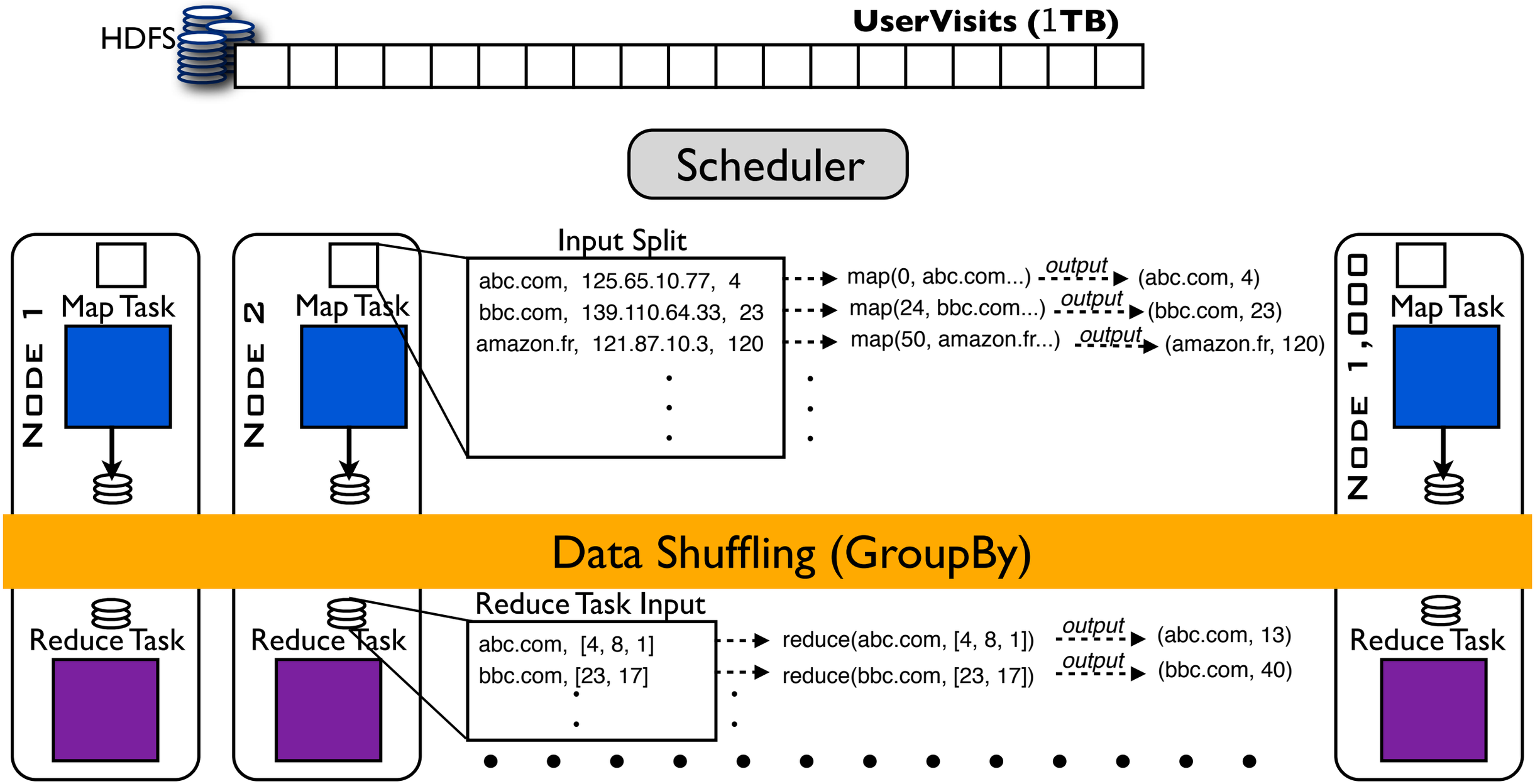
MapReduce (Recall)



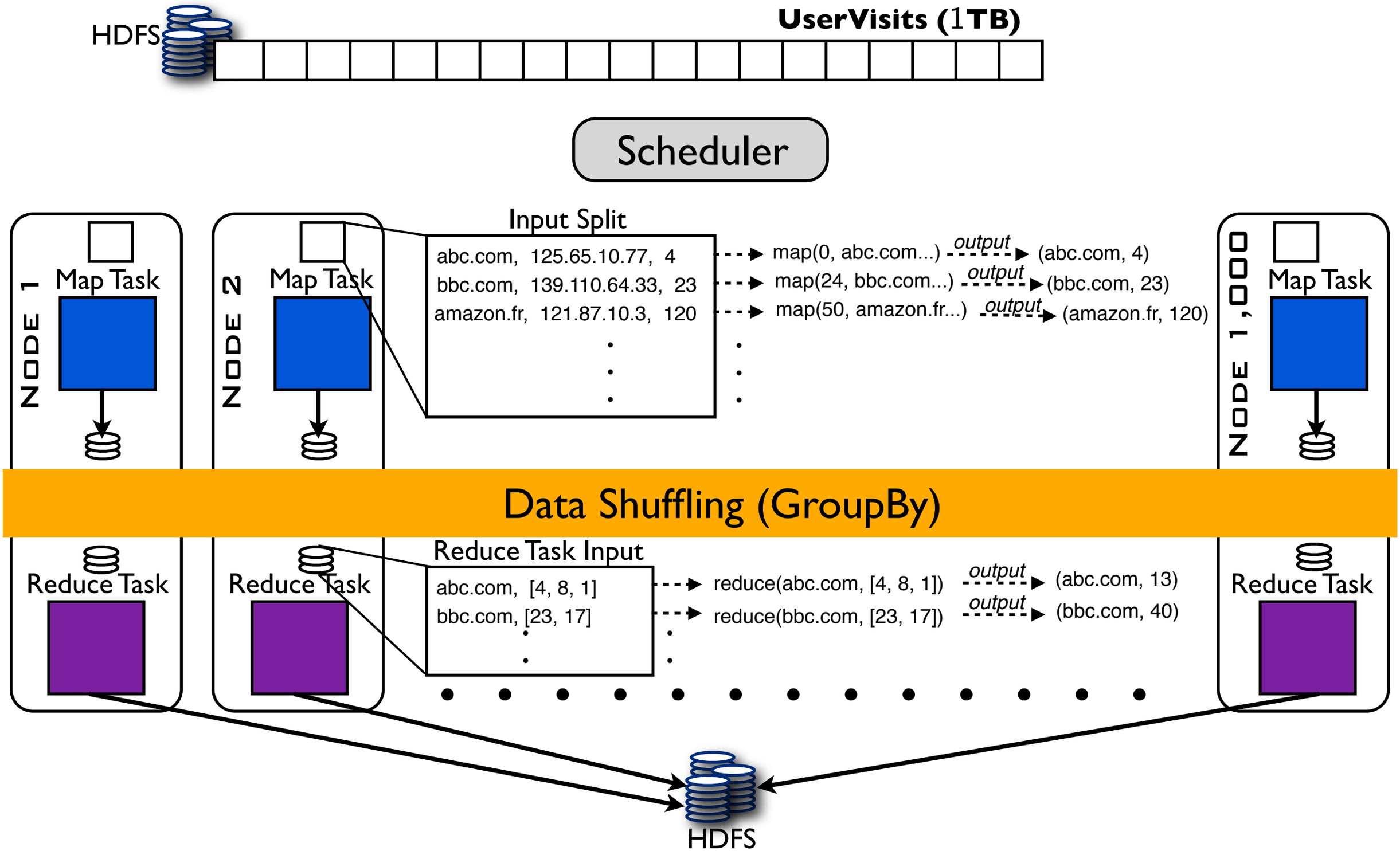
MapReduce (Recall)



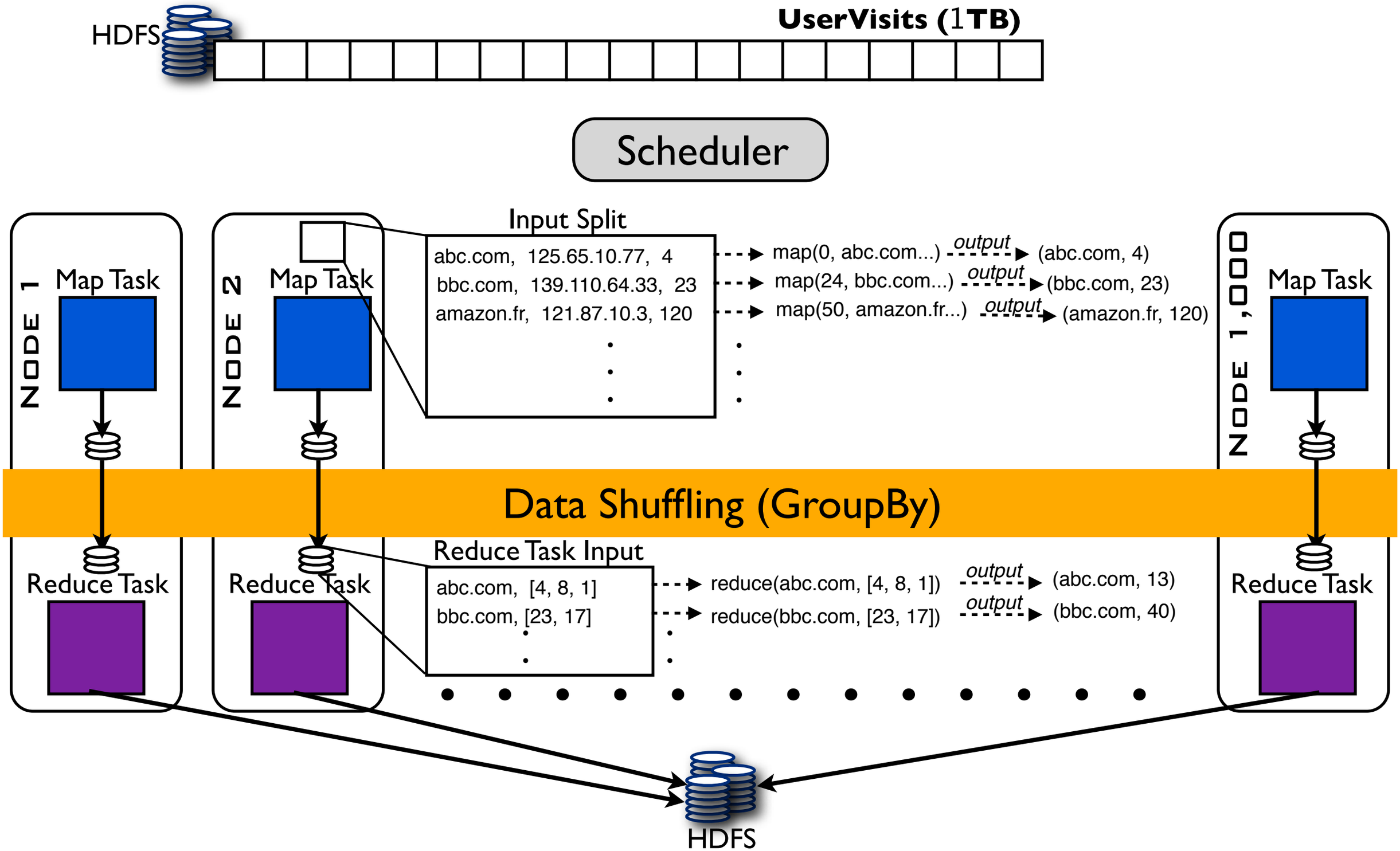
MapReduce (Recall)



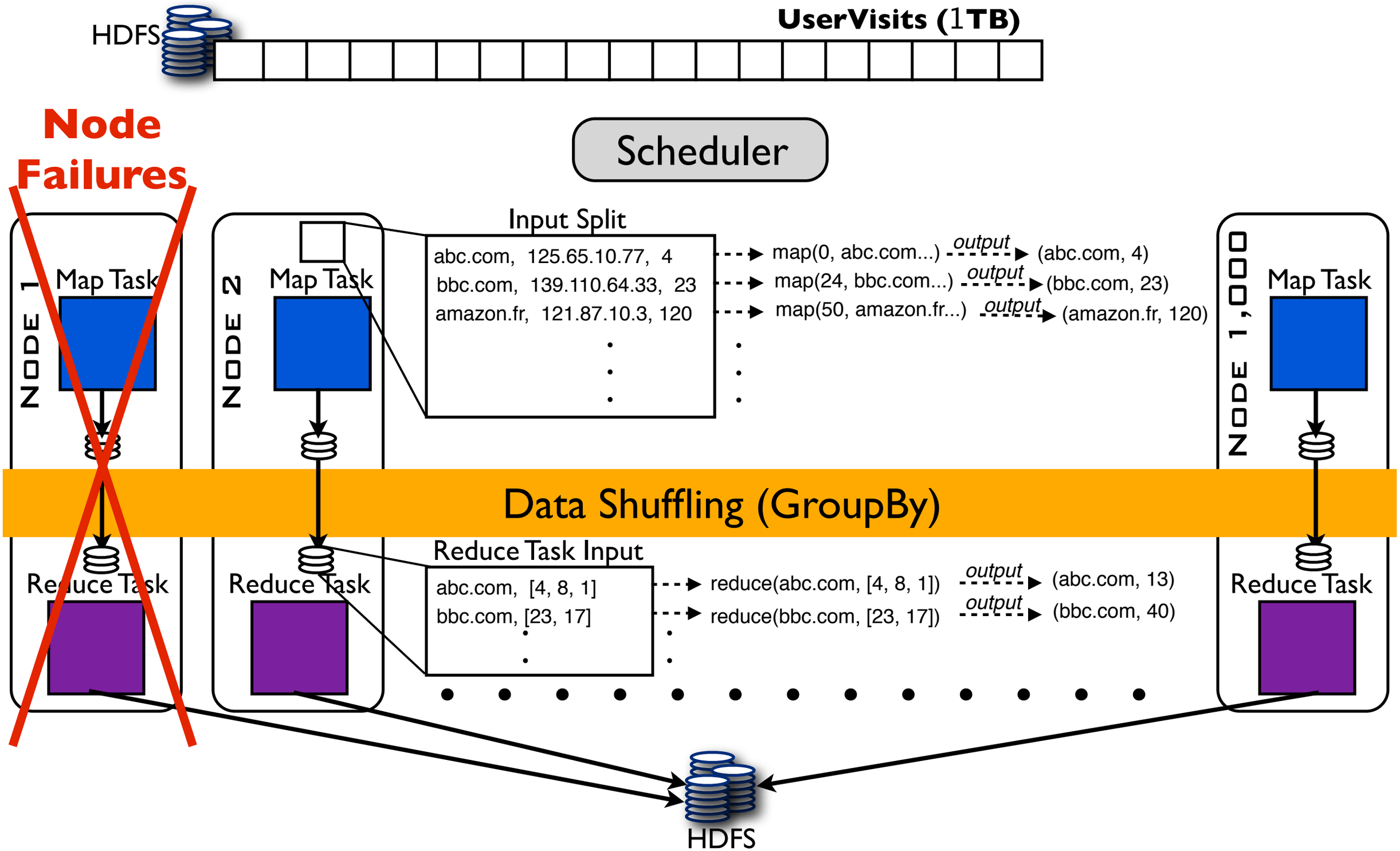
MapReduce (Recall)



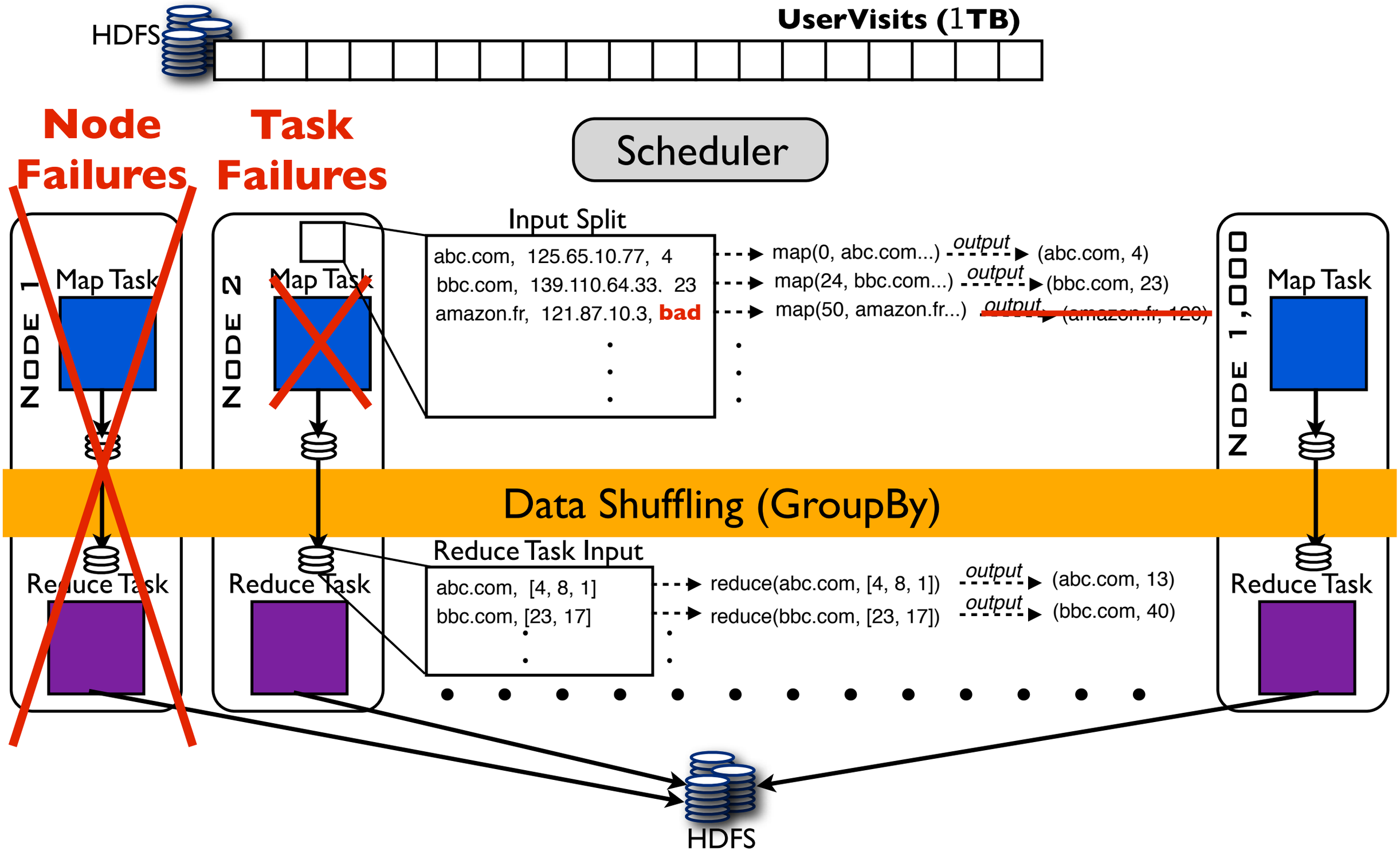
Failures are the Rule!



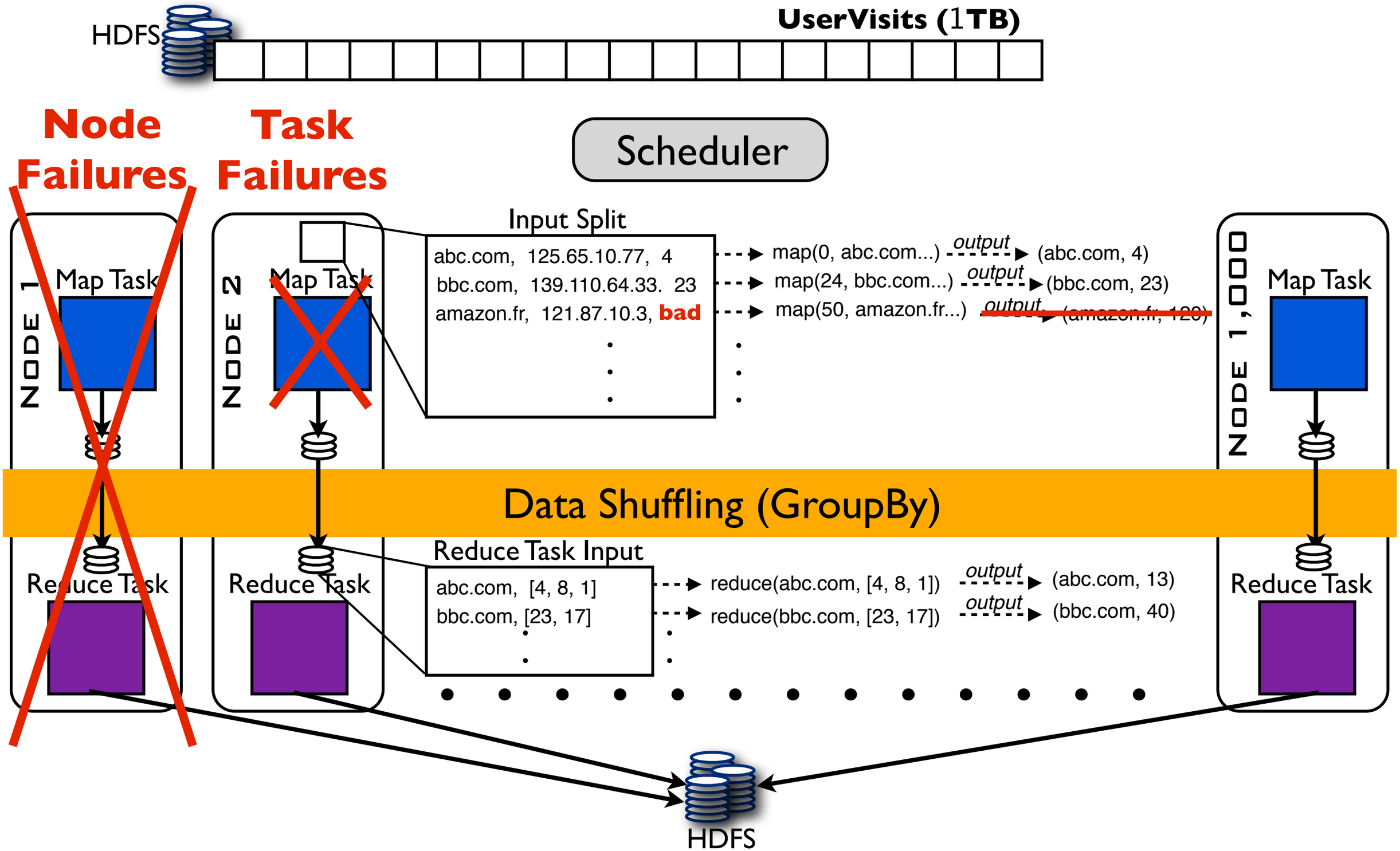
Failures are the Rule!



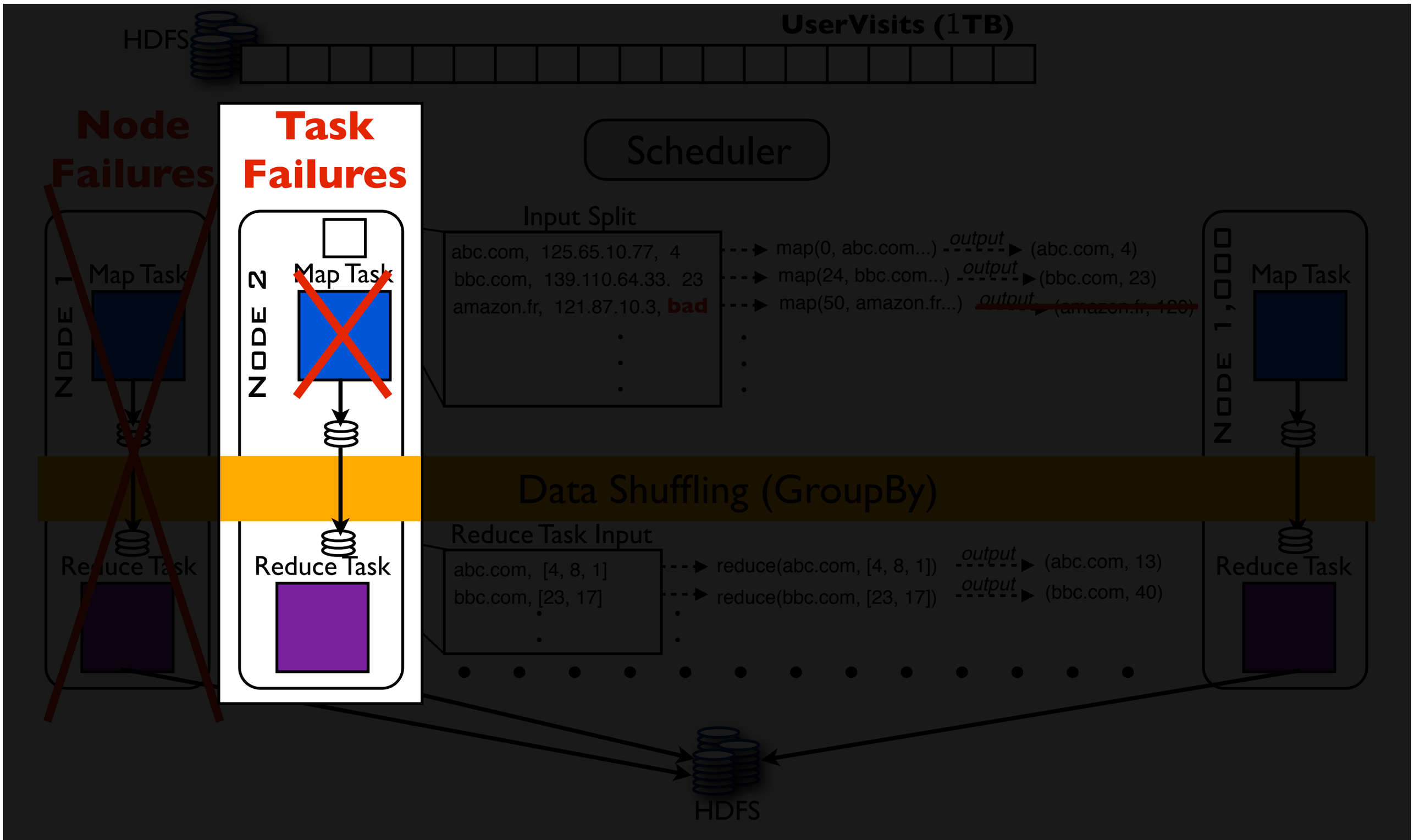
Failures are the Rule!



Task Failures



Task Failures



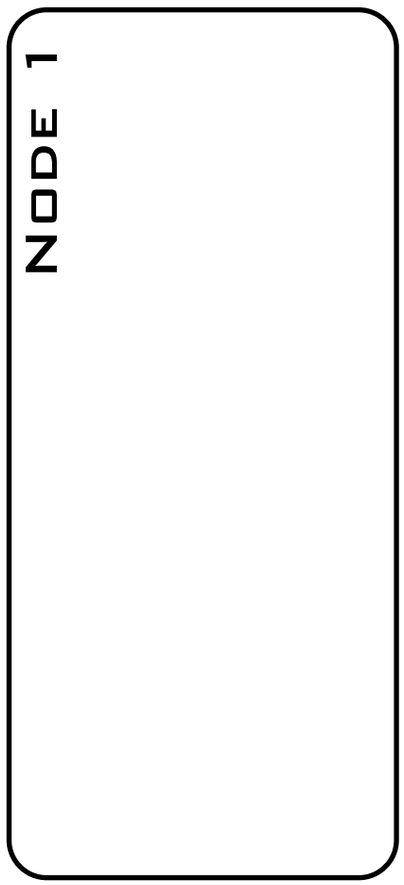
Current Approach

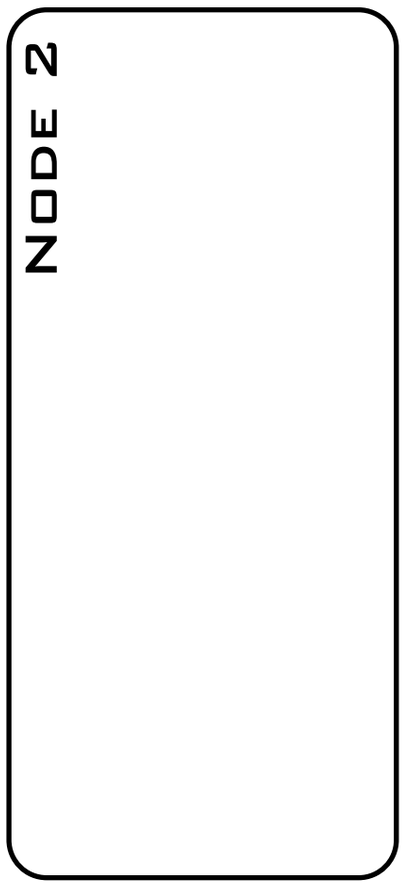
Hadoop without Failures

Map Task: 20 seconds

Time: 0s

Scheduler

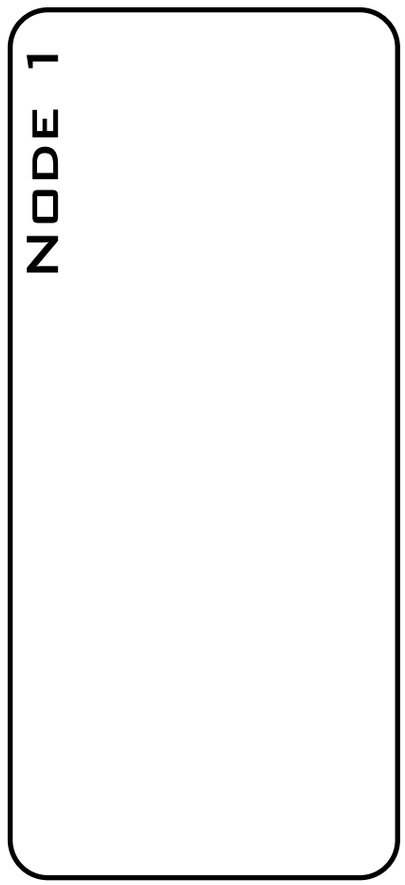


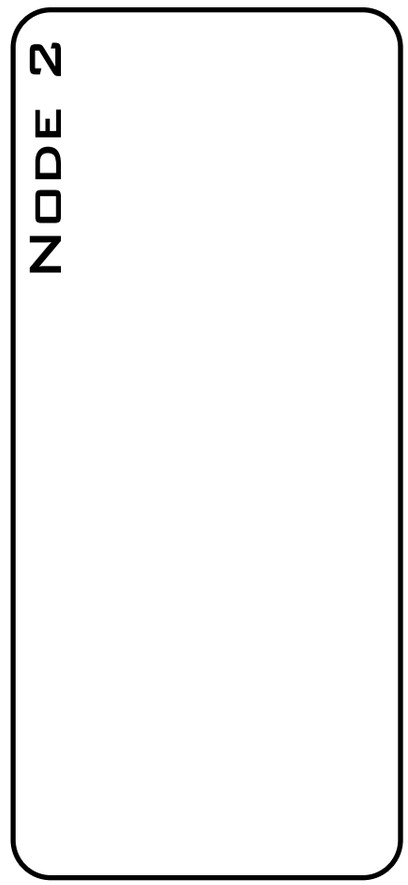


Hadoop with Task Failures

Time: 0s

Scheduler





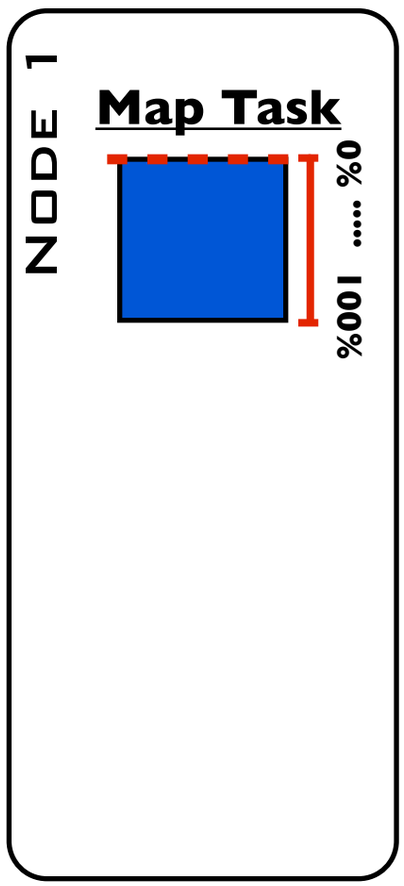
Current Approach

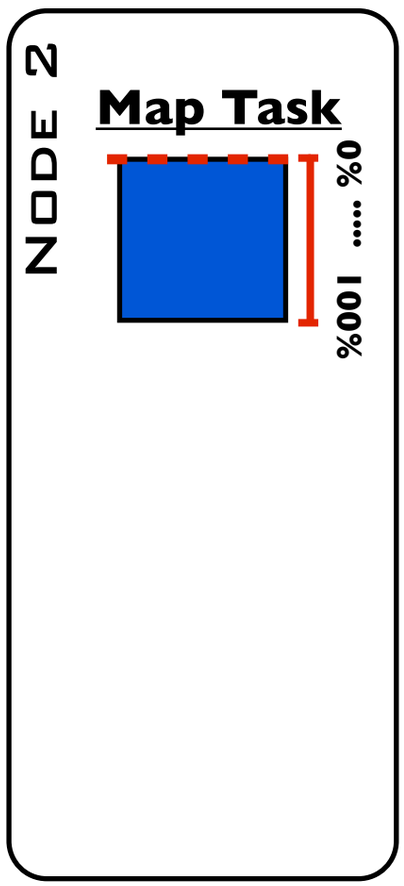
Hadoop without Failures

Map Task: 20 seconds

Time: 0s

Scheduler

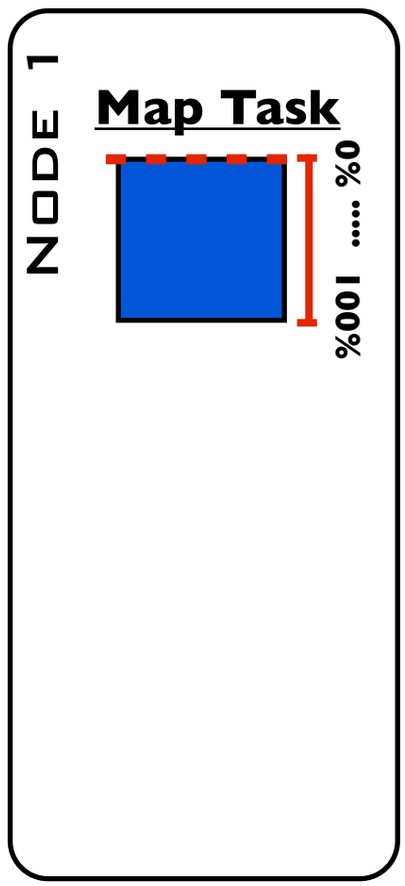


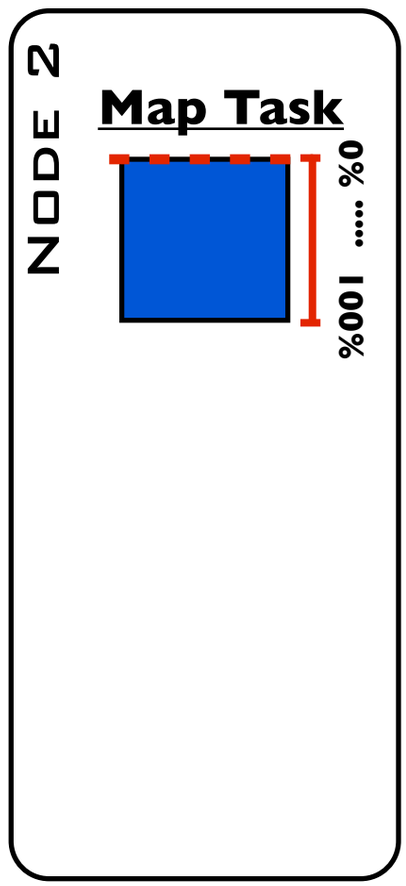


Hadoop with Task Failures

Time: 0s

Scheduler





Current Approach

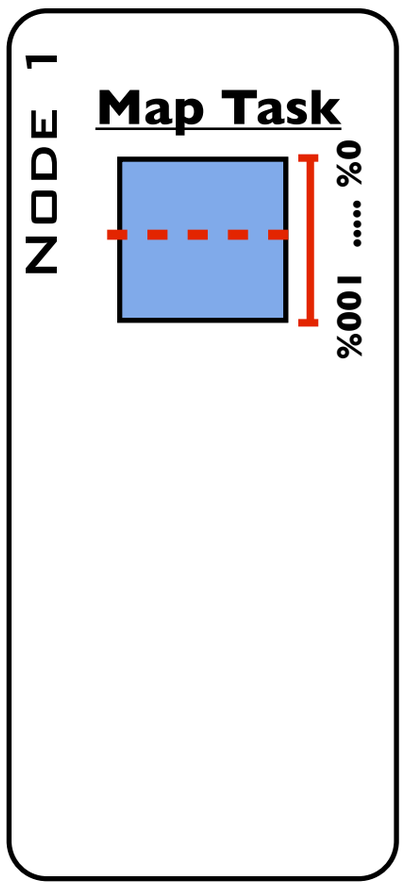
Hadoop without Failures

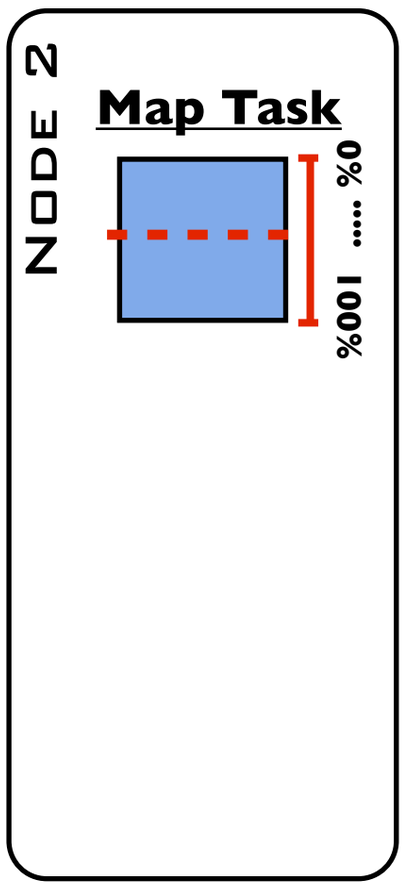
Map Task: 20 seconds

Time:10s

Time:10s

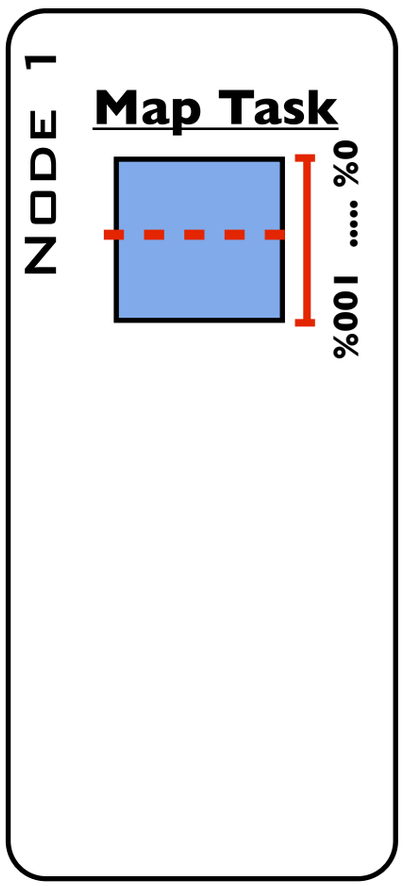
Scheduler

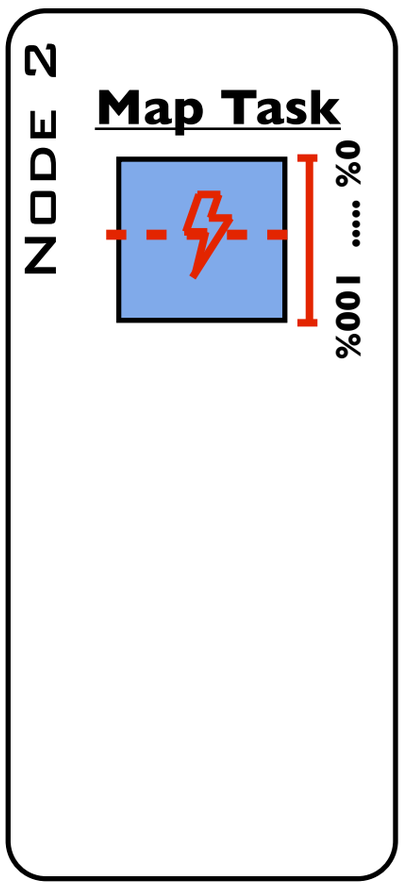




Hadoop with Task Failures

Scheduler





Current Approach

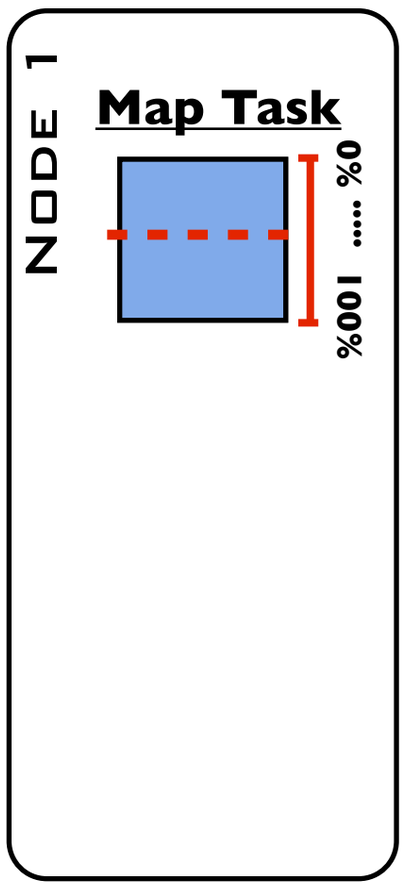
Hadoop without Failures

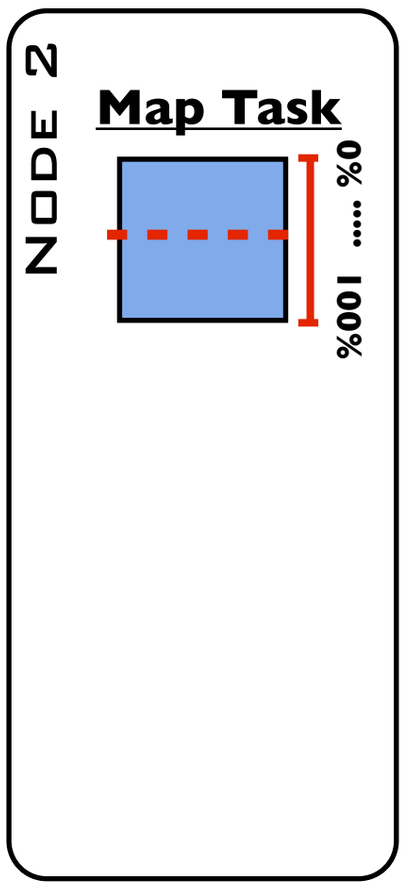
Map Task: 20 seconds

Time:10s

Time:10s

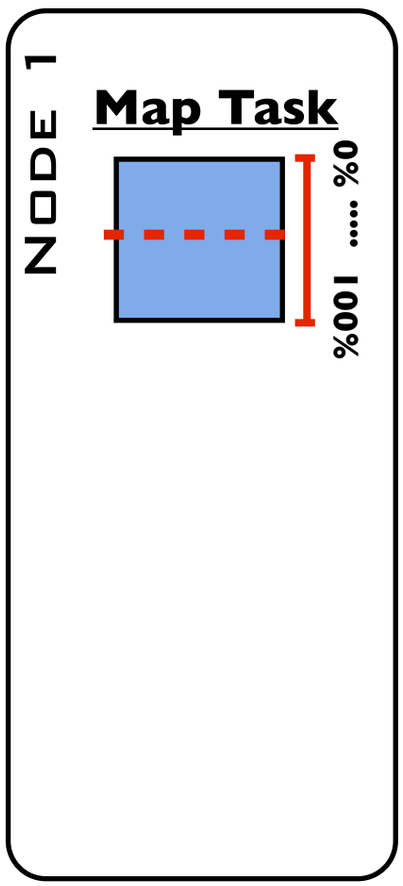
Scheduler

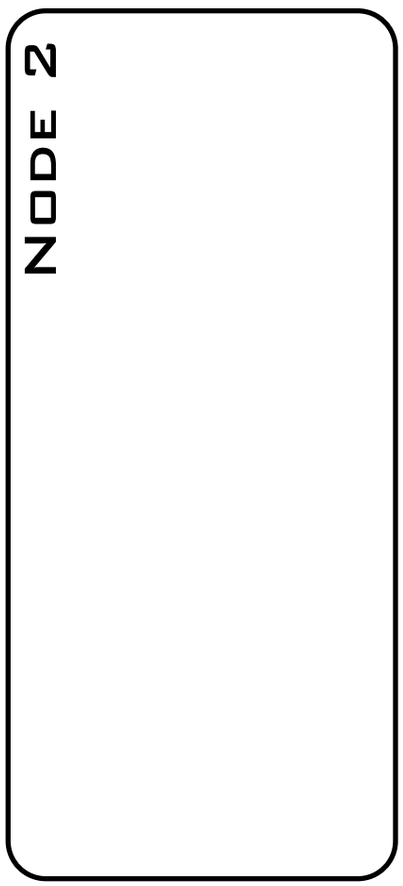




Hadoop with Task Failures

Scheduler





Current Approach

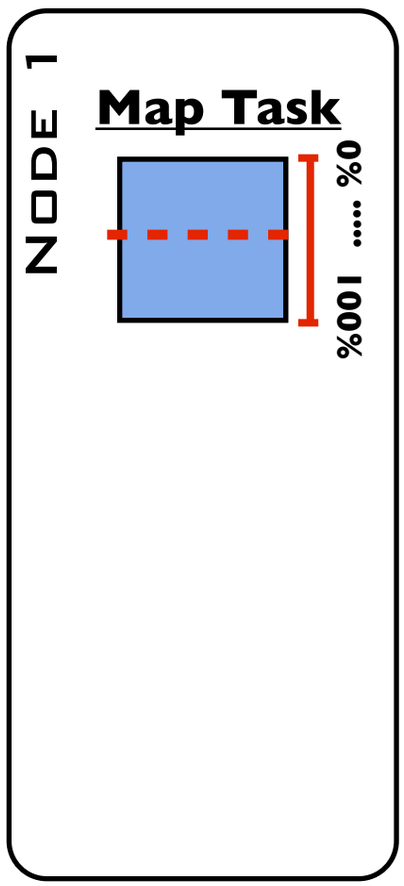
Hadoop without Failures

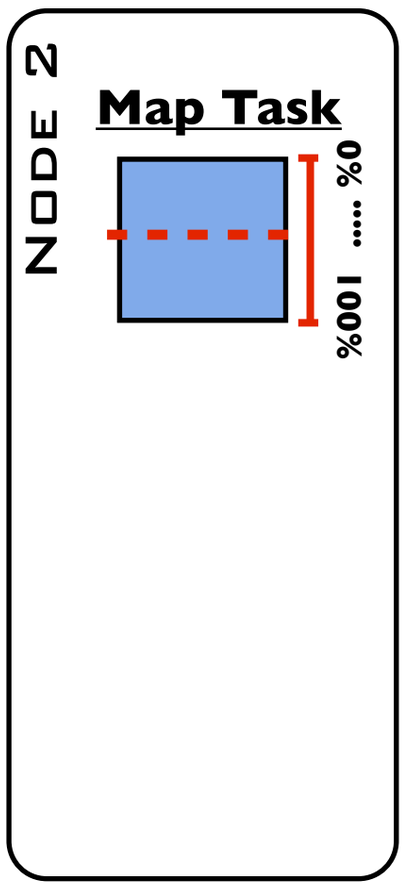
Map Task: 20 seconds

Time:10s

Time:10s

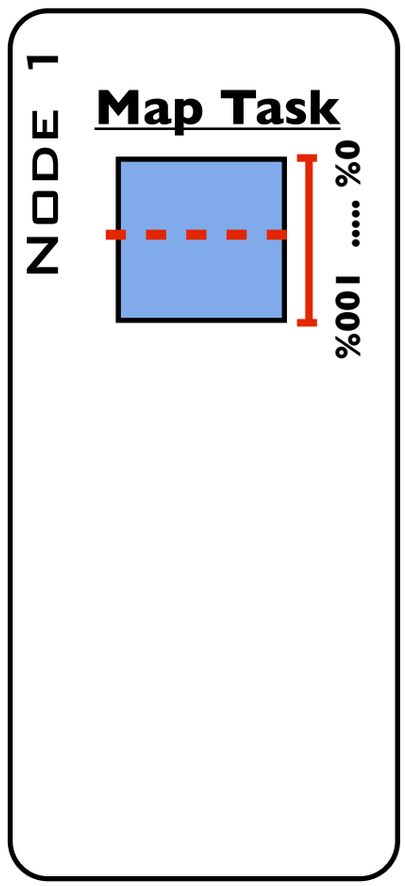
Scheduler

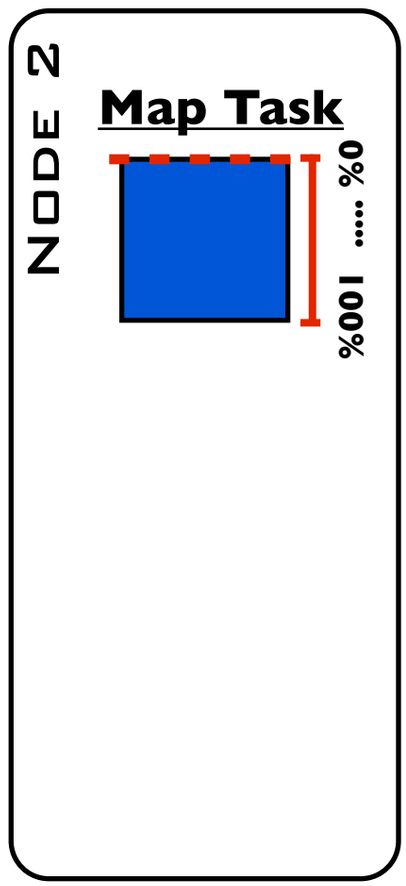




Hadoop with Task Failures

Scheduler





Current Approach

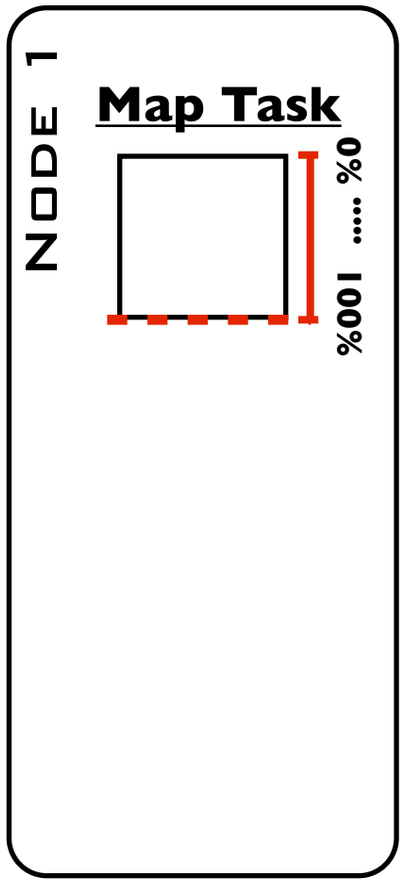
Hadoop without Failures

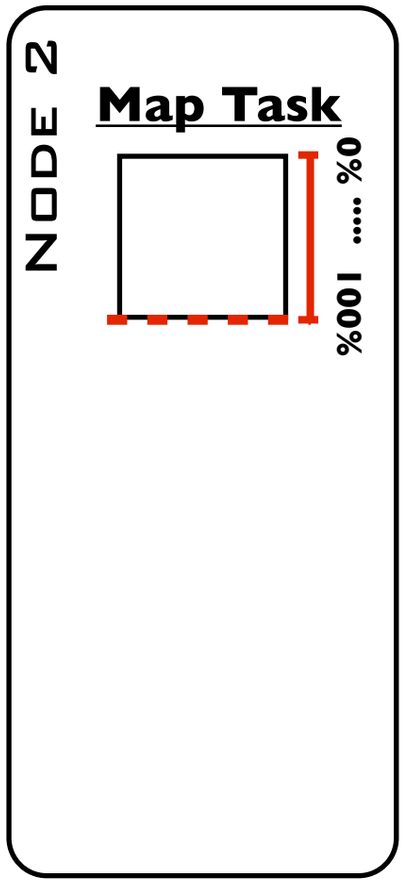
Map Task: 20 seconds

Time:20s

Time:20s

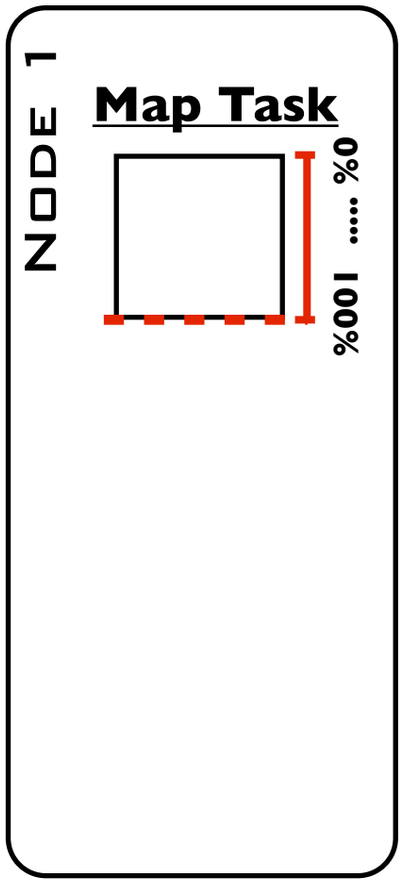
Scheduler

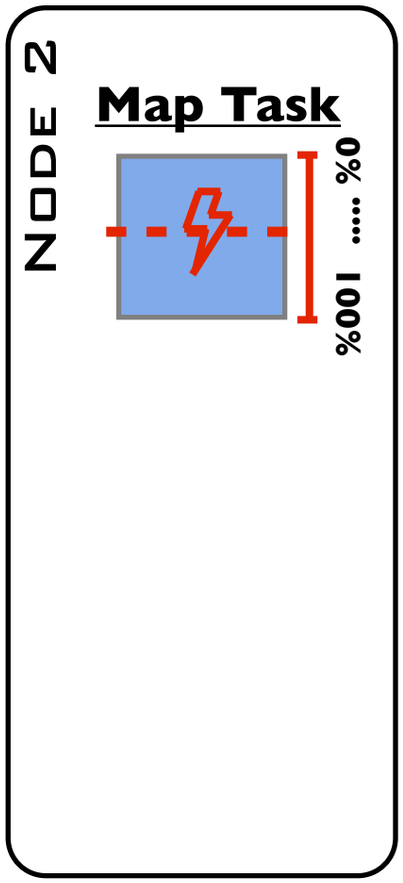




Hadoop with Task Failures

Scheduler





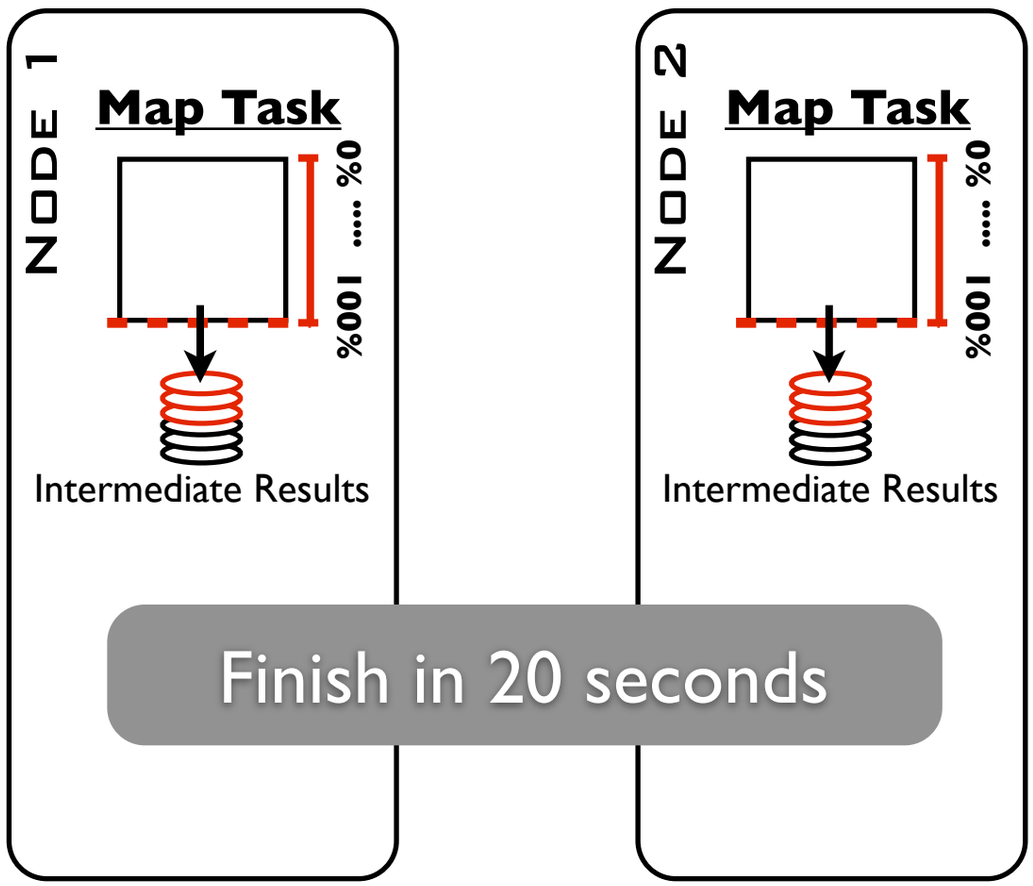
Current Approach

Hadoop without Failures

Map Task: 20 seconds

Time:20s

Scheduler

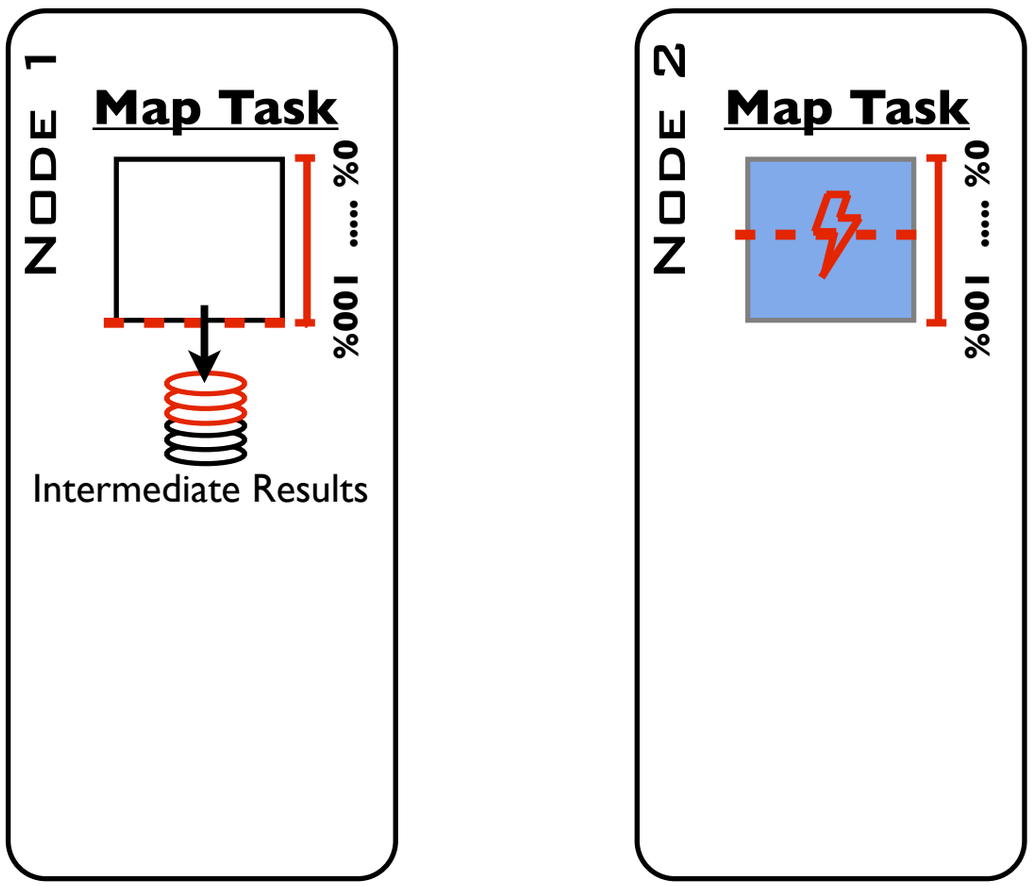


Finish in 20 seconds

Hadoop with Task Failures

Time:20s

Scheduler



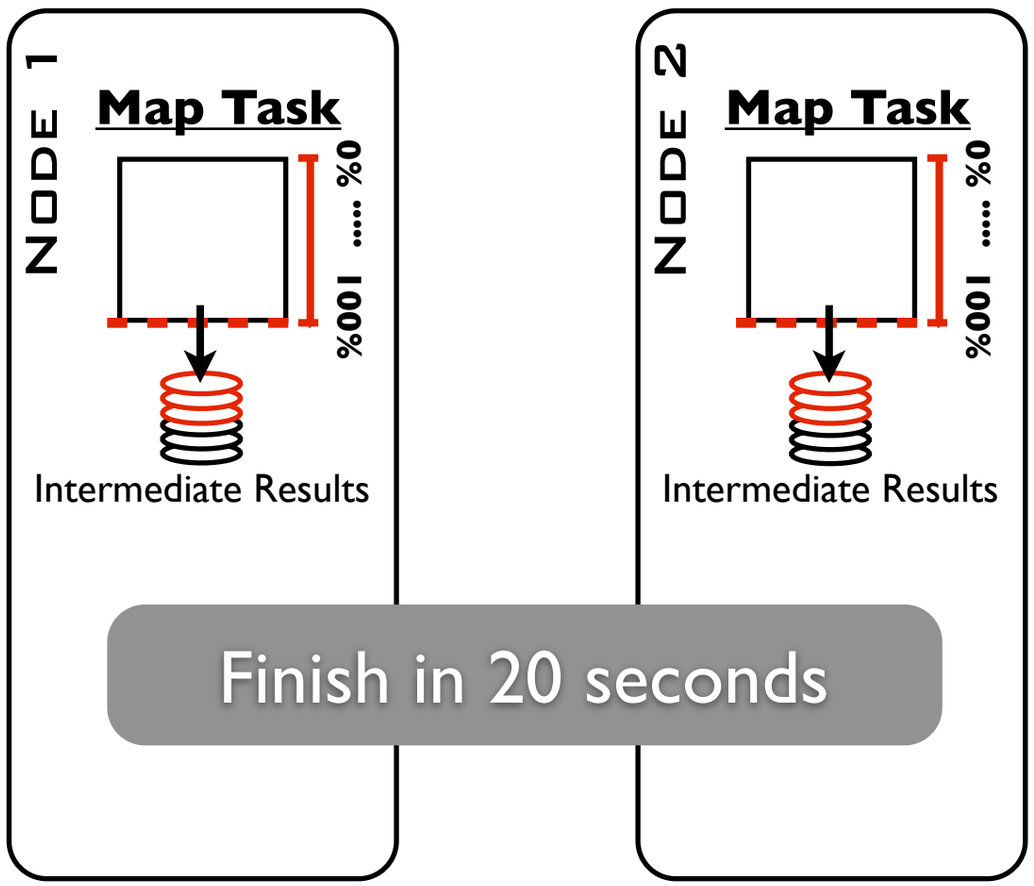
Current Approach

Hadoop without Failures

Map Task: 20 seconds

Time:20s

Scheduler

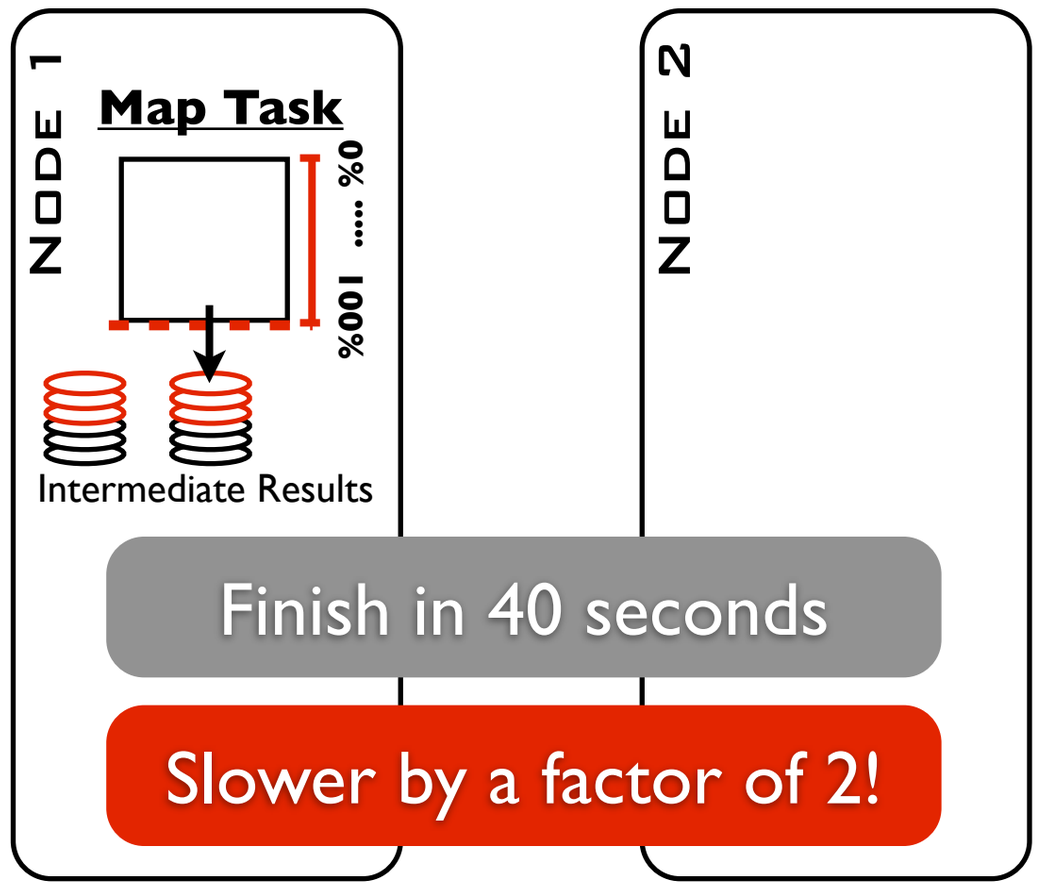


Finish in 20 seconds

Hadoop with Task Failures

Time:40s

Scheduler



Finish in 40 seconds

Slower by a factor of 2!

Solution: Local Checkpointing

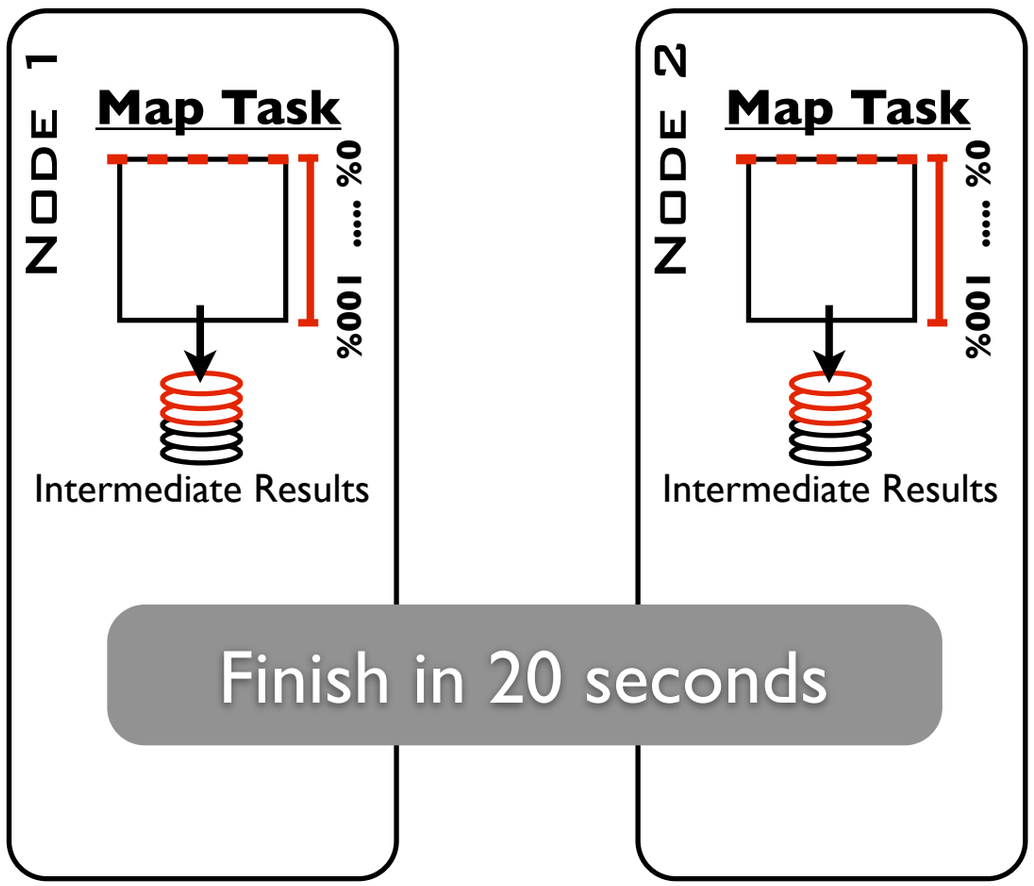
Map Task: 20 seconds

Time:20s

Time:40s

Hadoop without Failures

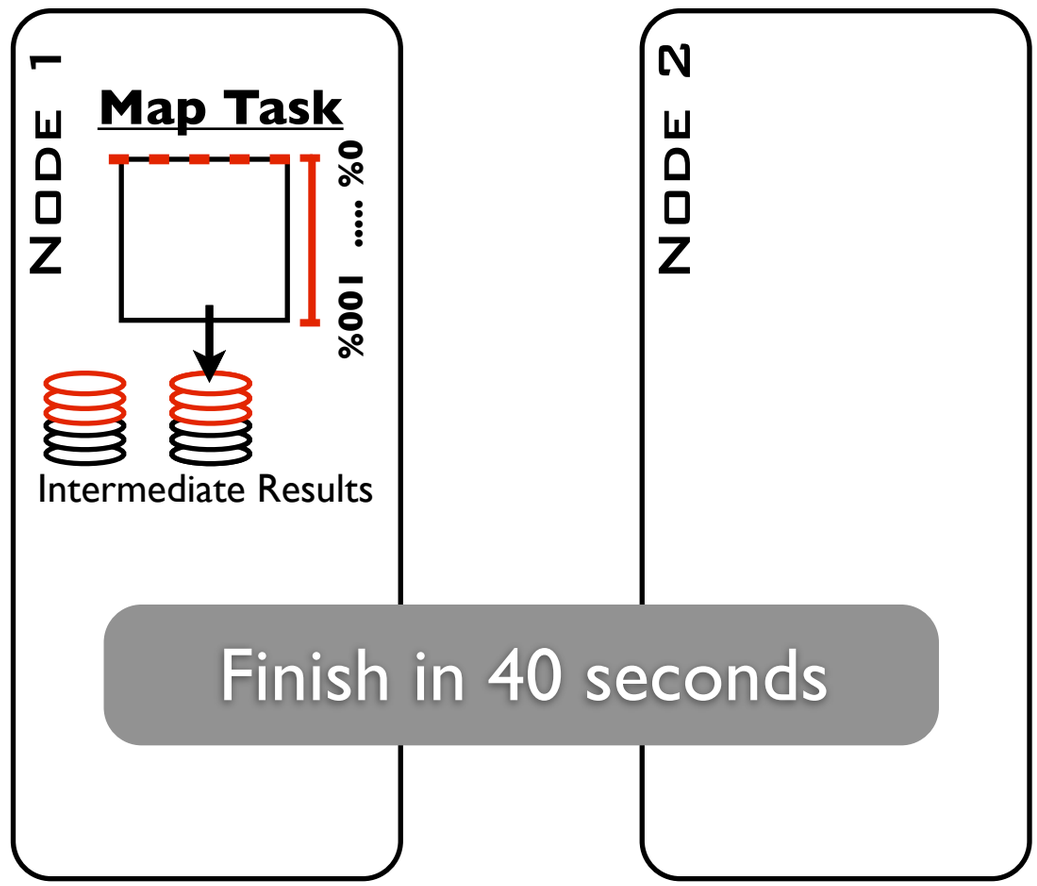
Scheduler



Finish in 20 seconds

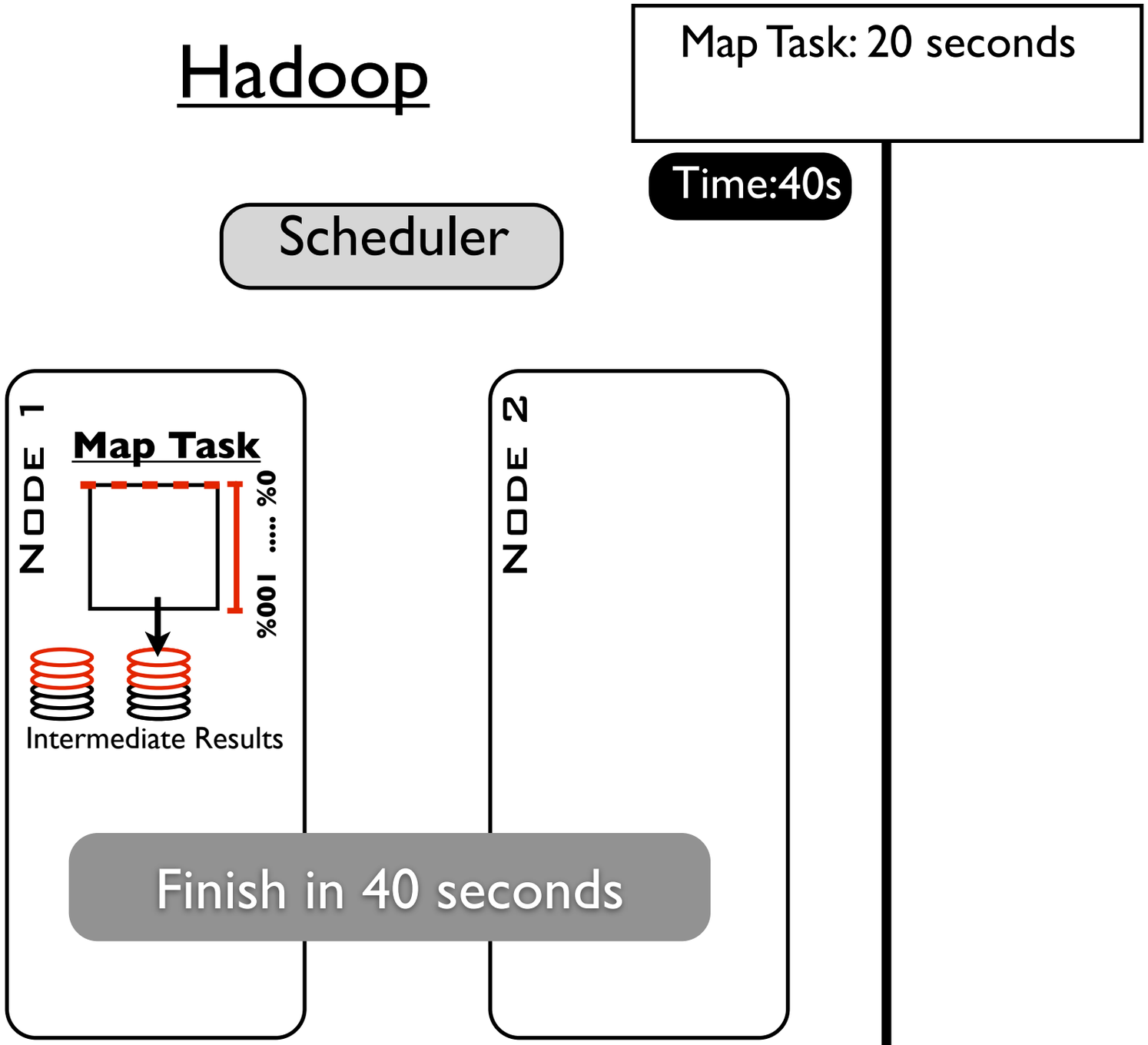
Hadoop with Task Failures

Scheduler



Finish in 40 seconds

Solution: Local Checkpointing



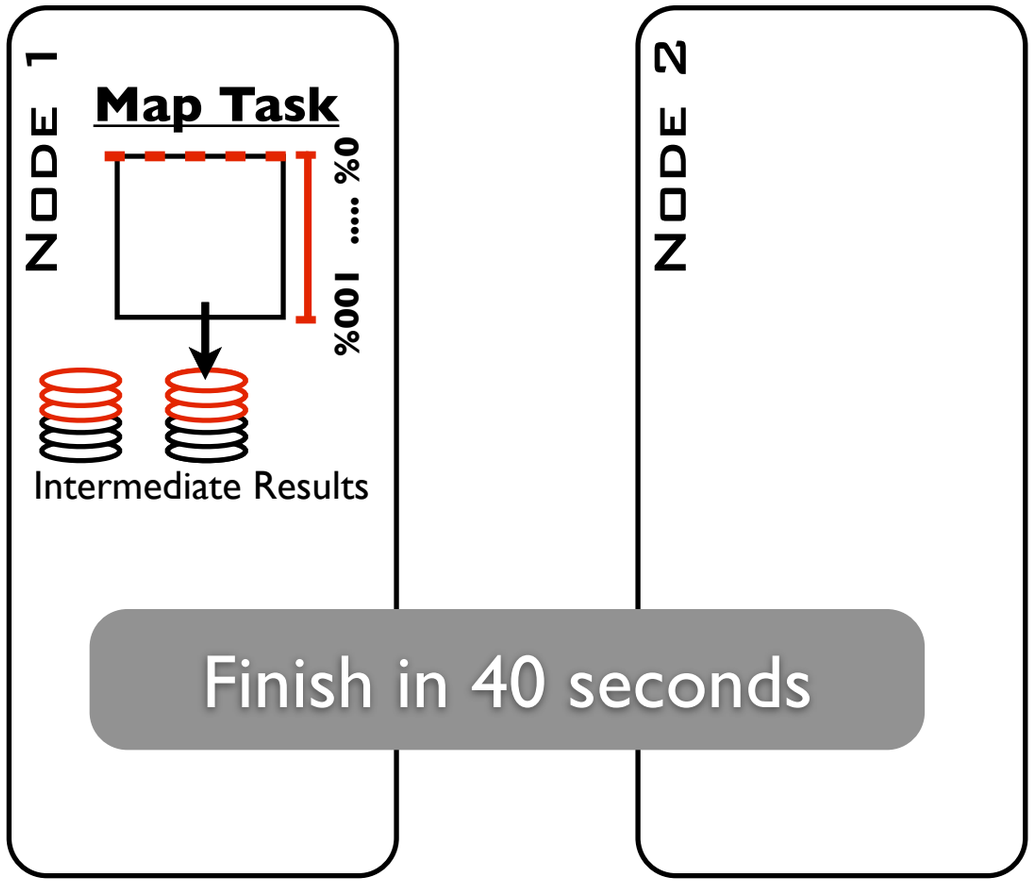
Solution: Local Checkpointing

Map Task: 20 seconds

Hadoop

Time:40s

Scheduler

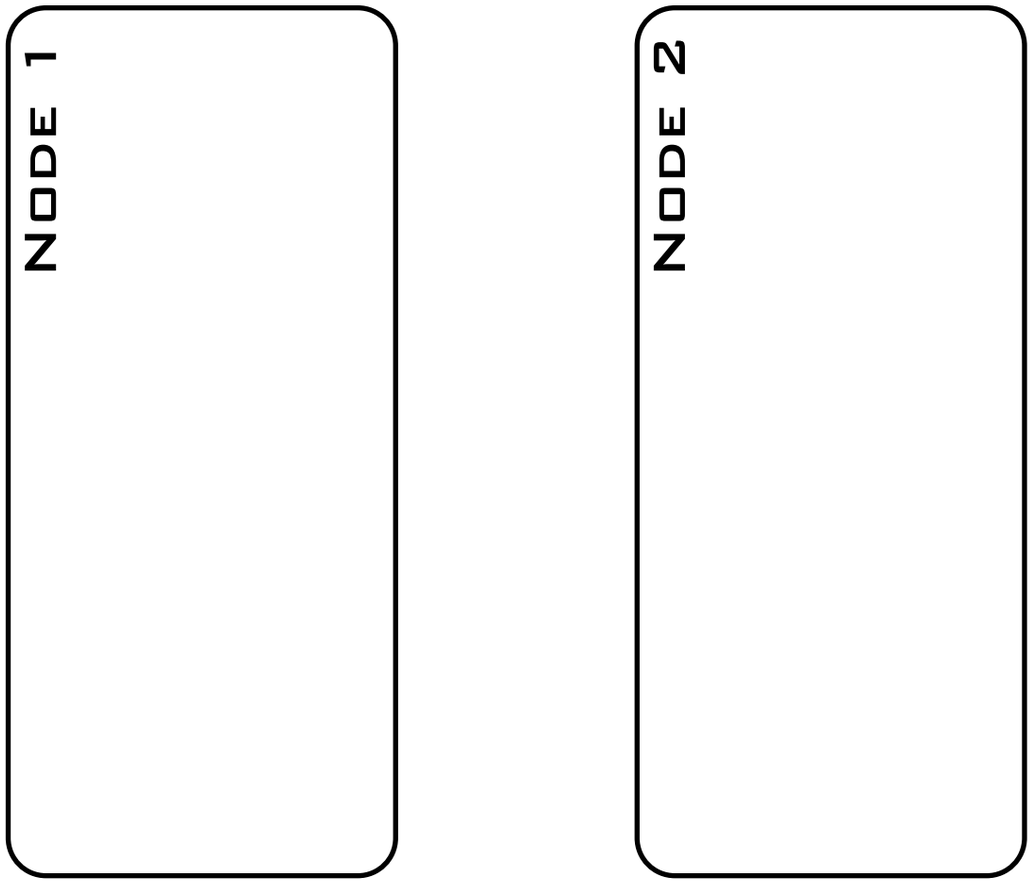


Finish in 40 seconds

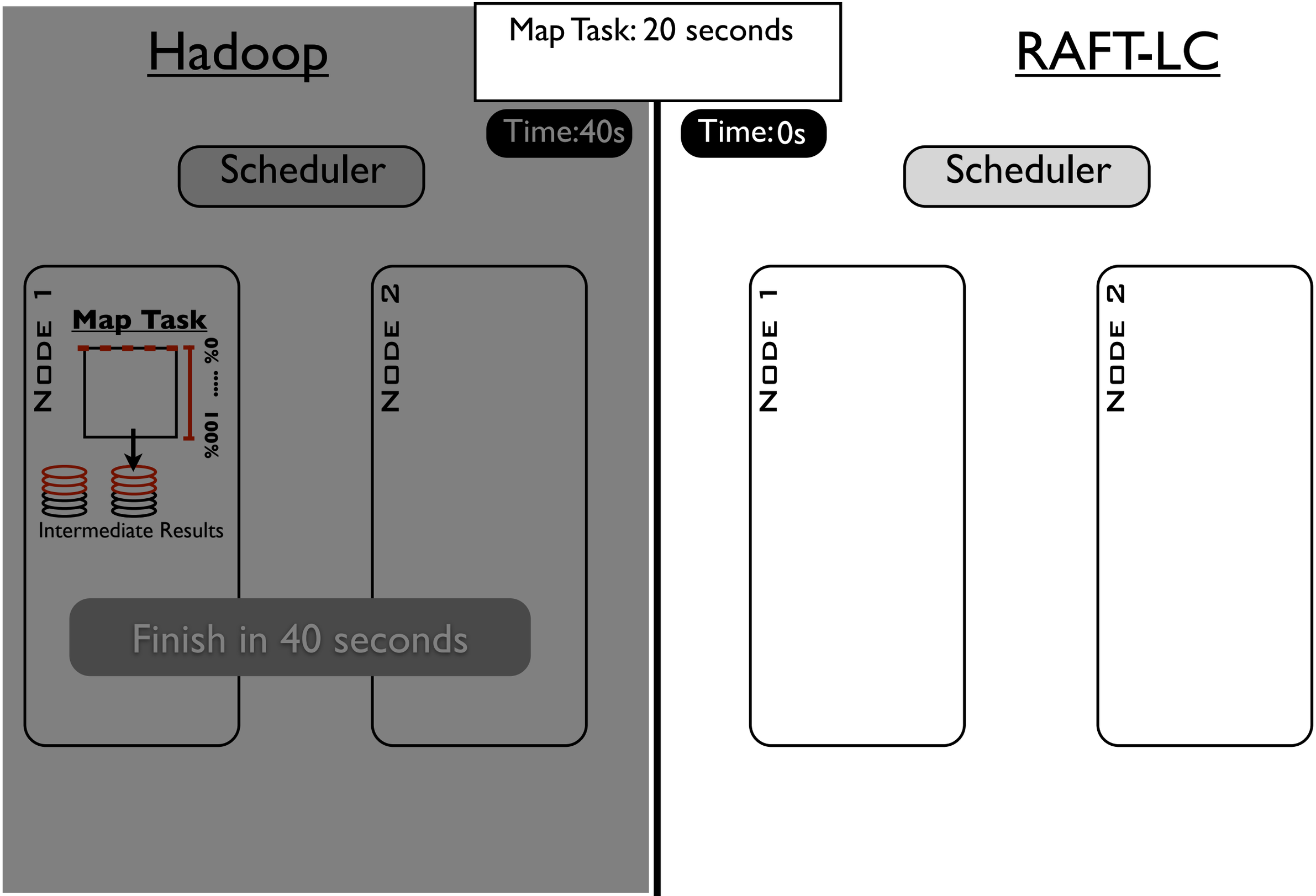
RAFT-LC

Time:0s

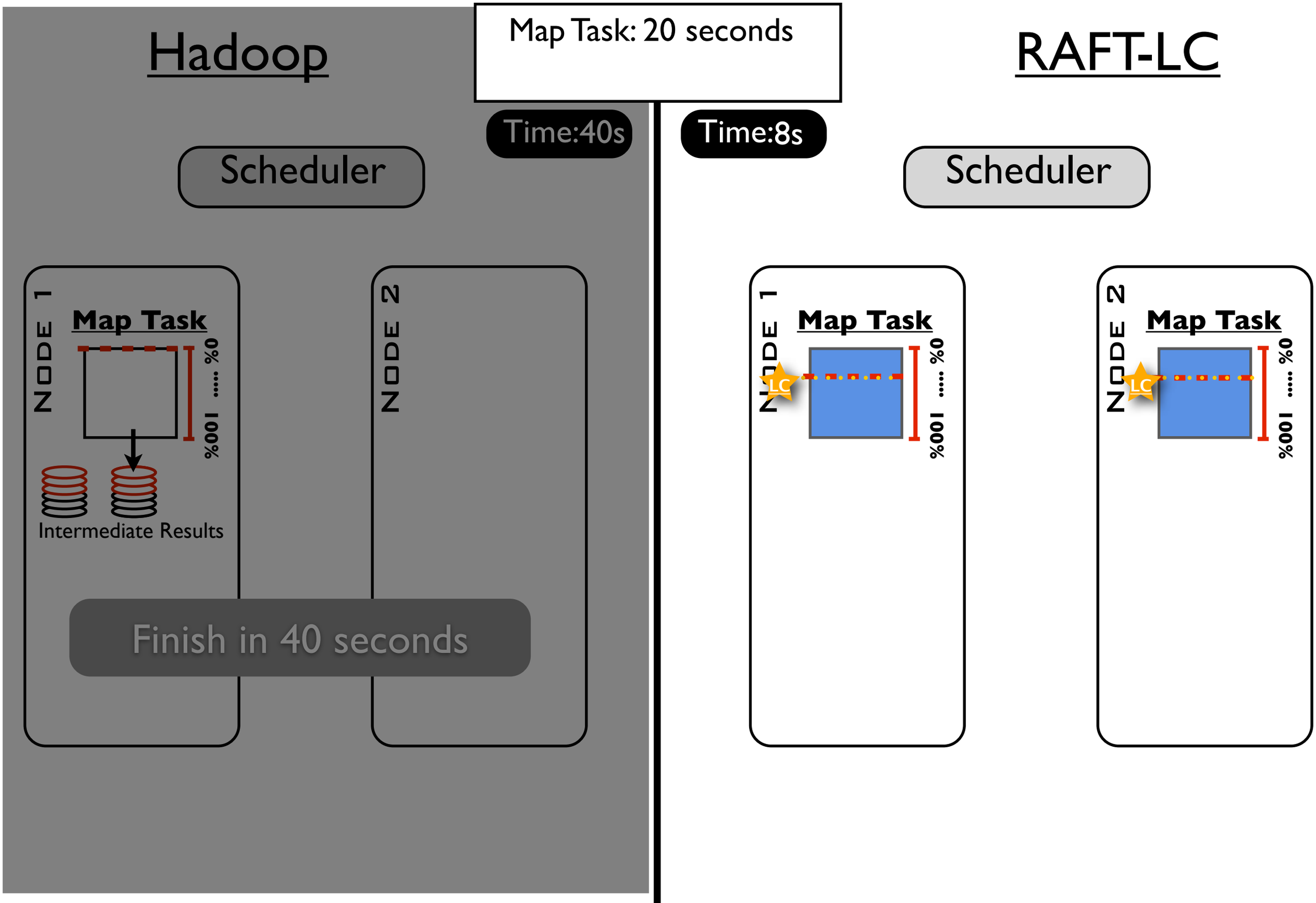
Scheduler



Solution: Local Checkpointing



Solution: Local Checkpointing



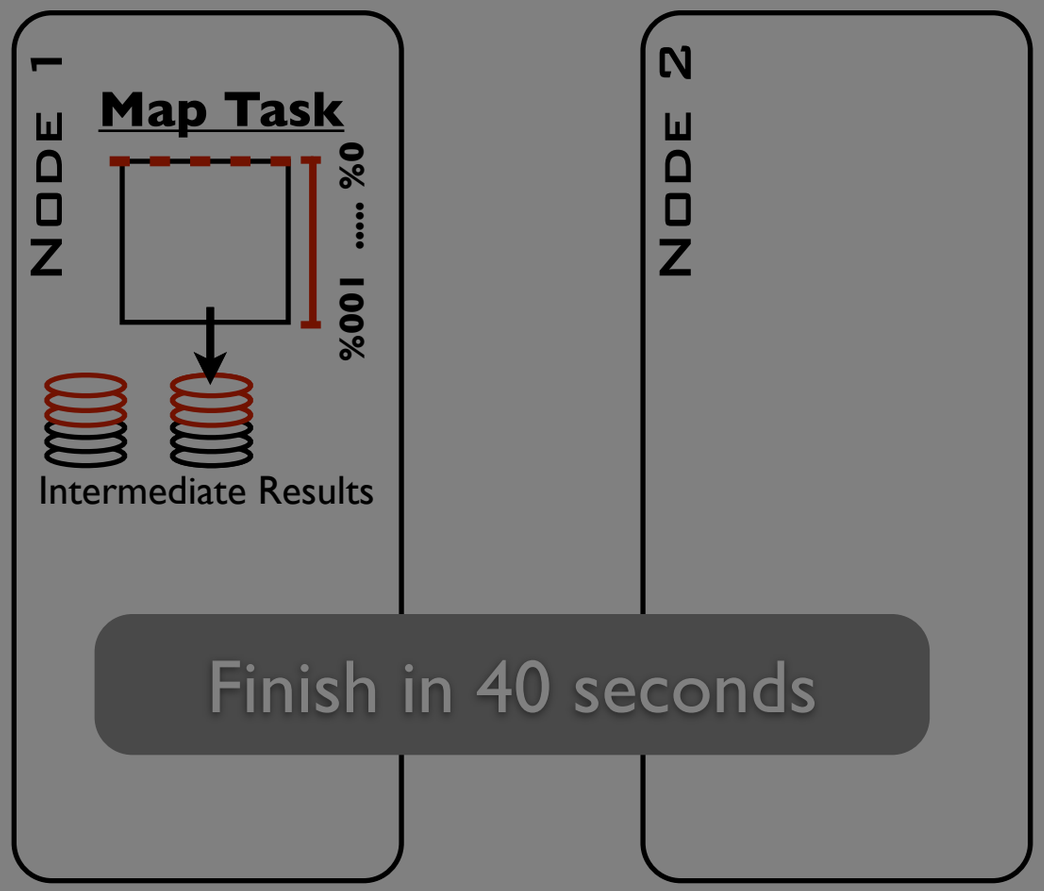
Solution: Local Checkpointing

Map Task: 20 seconds

Hadoop

Time:40s

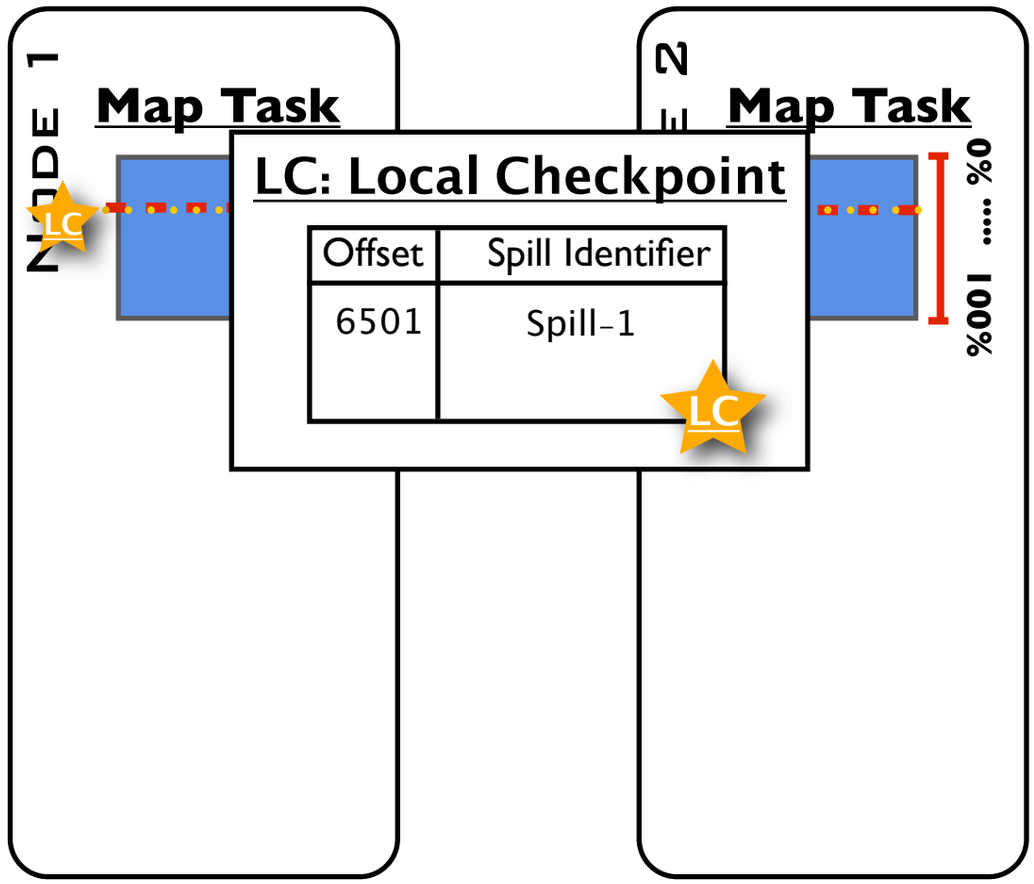
Scheduler



RAFT-LC

Time:8s

Scheduler



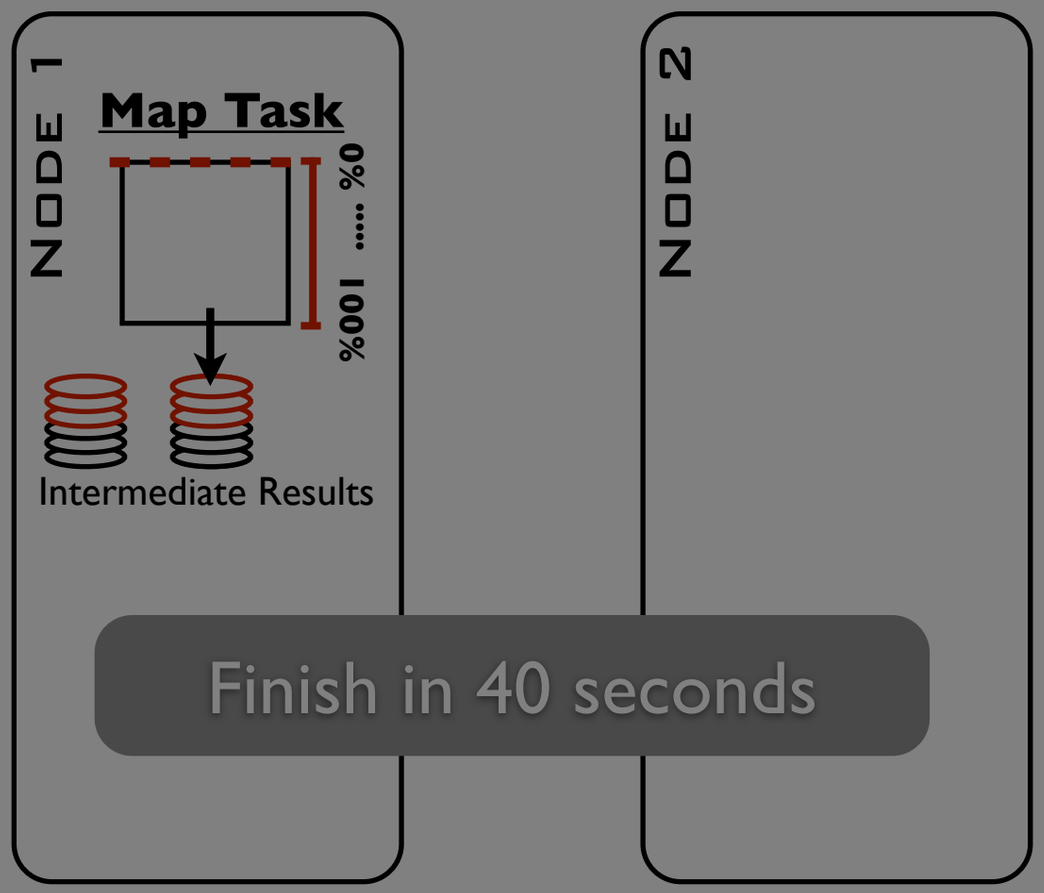
Solution: Local Checkpointing

Map Task: 20 seconds

Hadoop

Time:40s

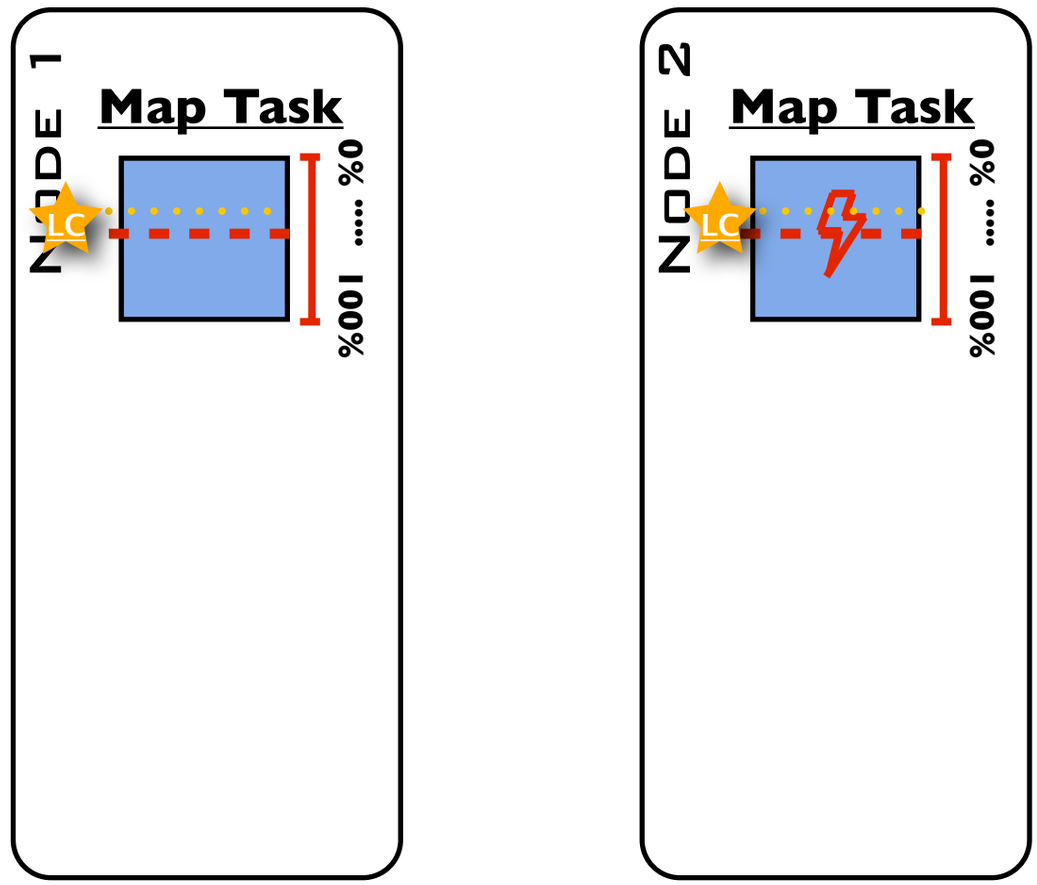
Scheduler



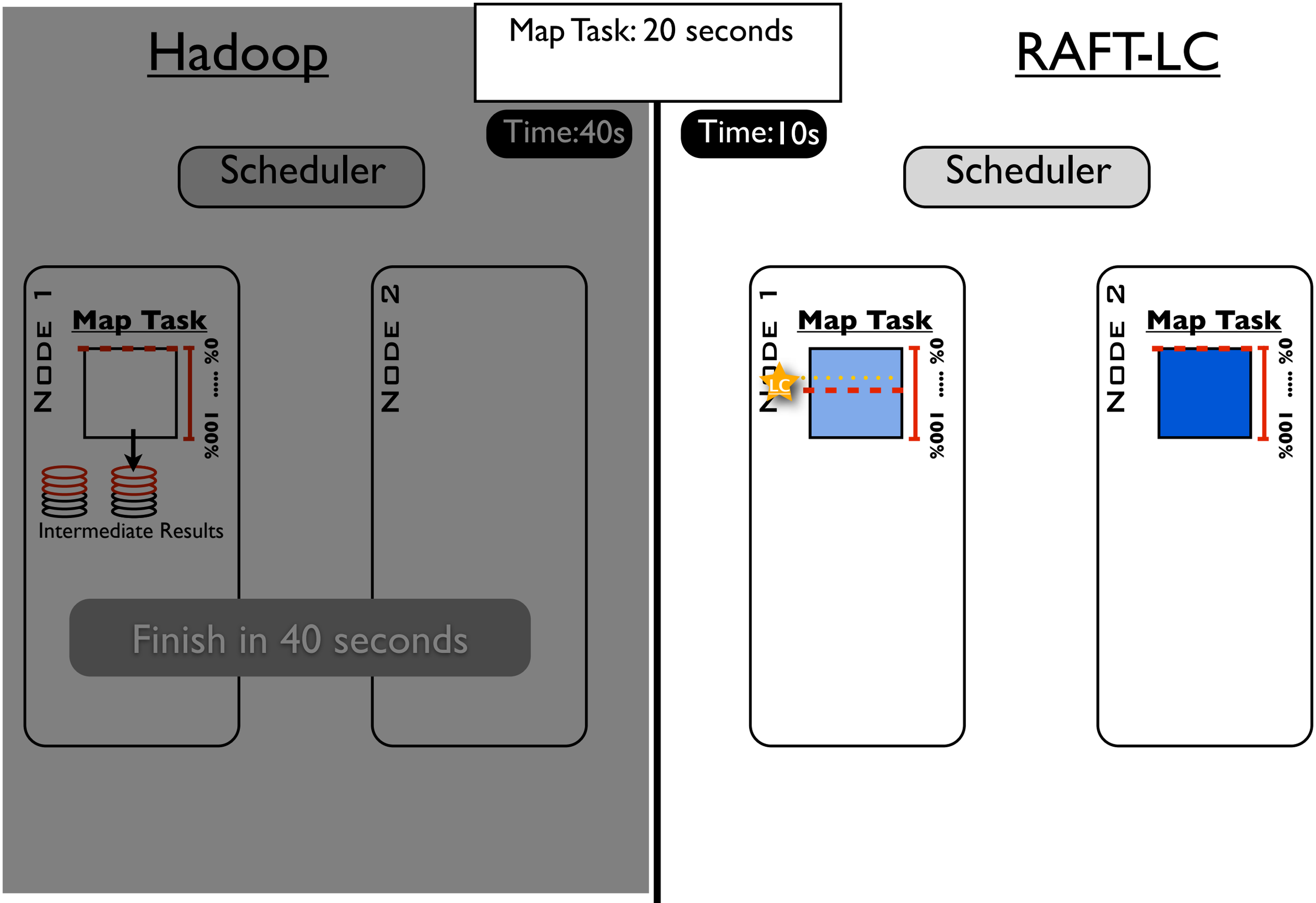
RAFT-LC

Time:10s

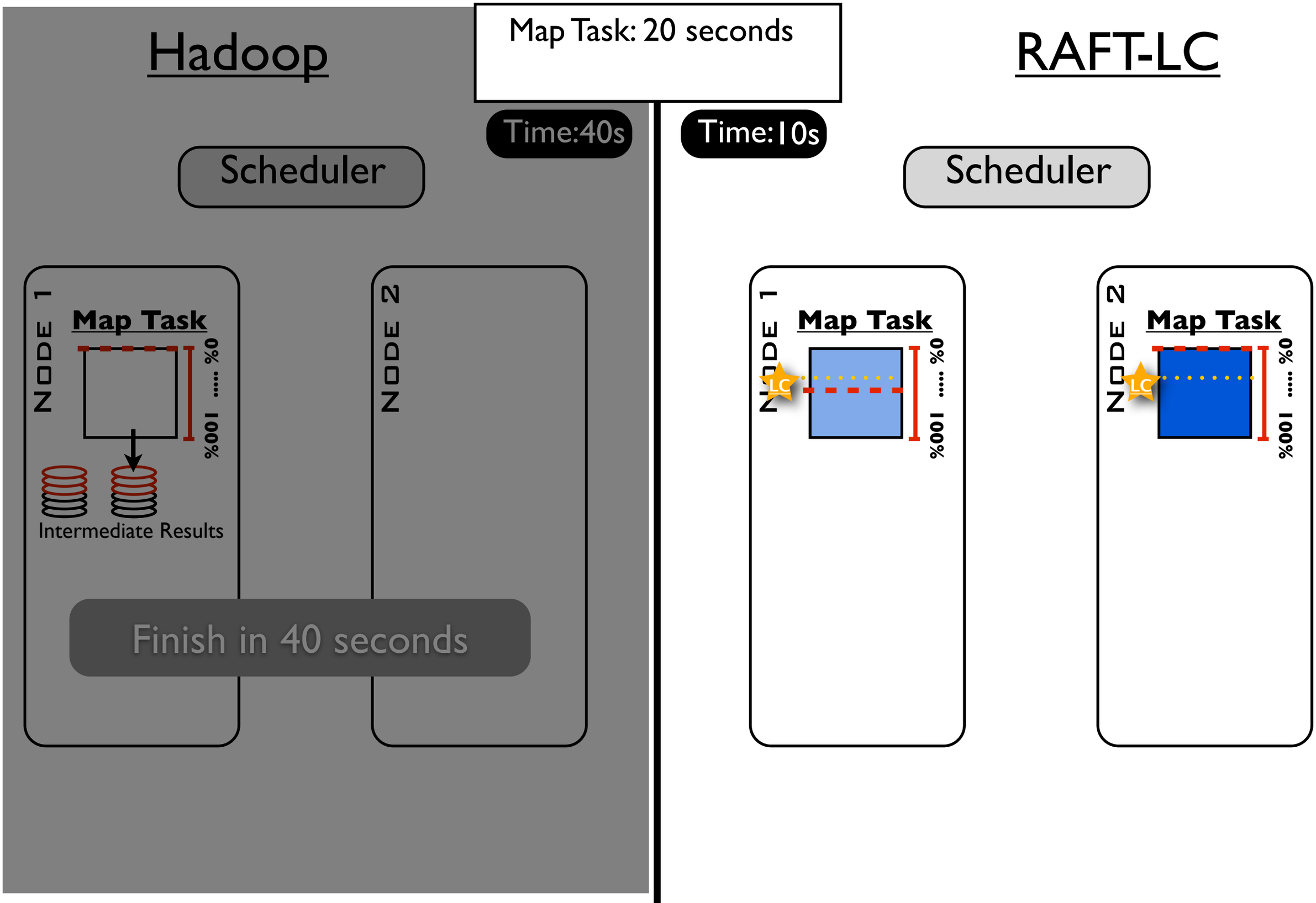
Scheduler



Solution: Local Checkpointing



Solution: Local Checkpointing



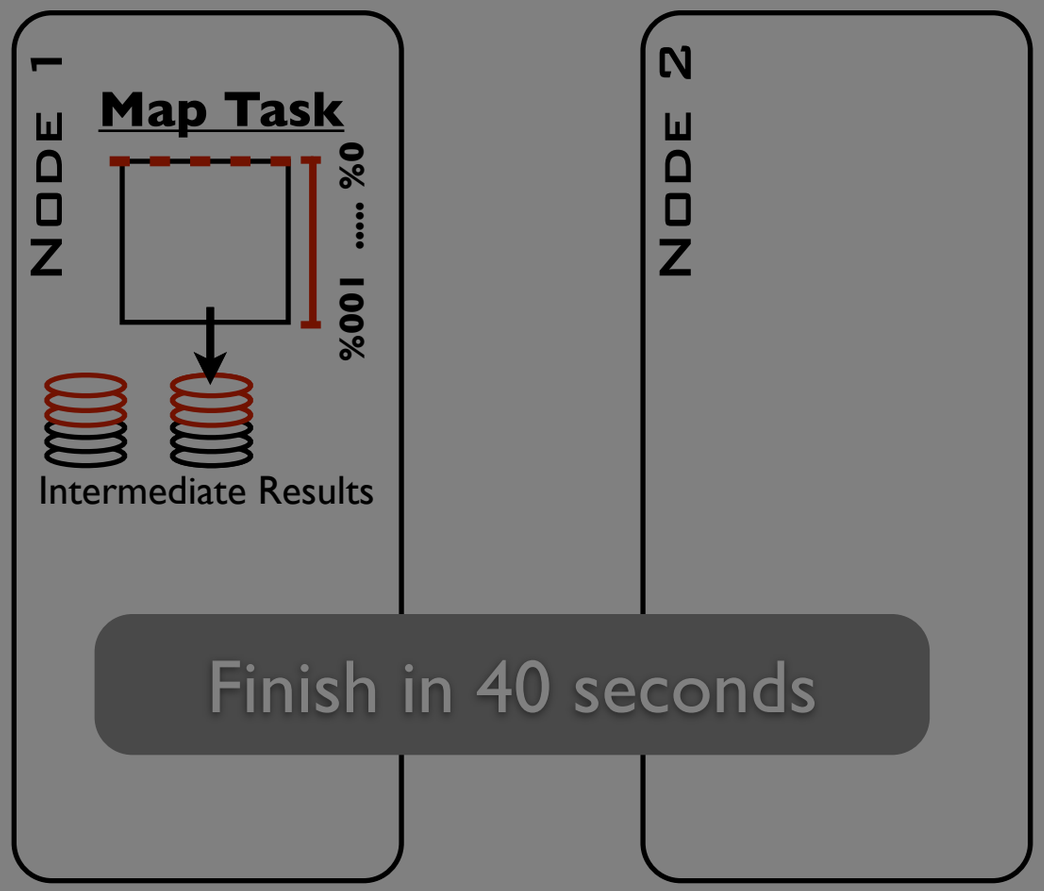
Solution: Local Checkpointing

Map Task: 20 seconds

Hadoop

Time:40s

Scheduler

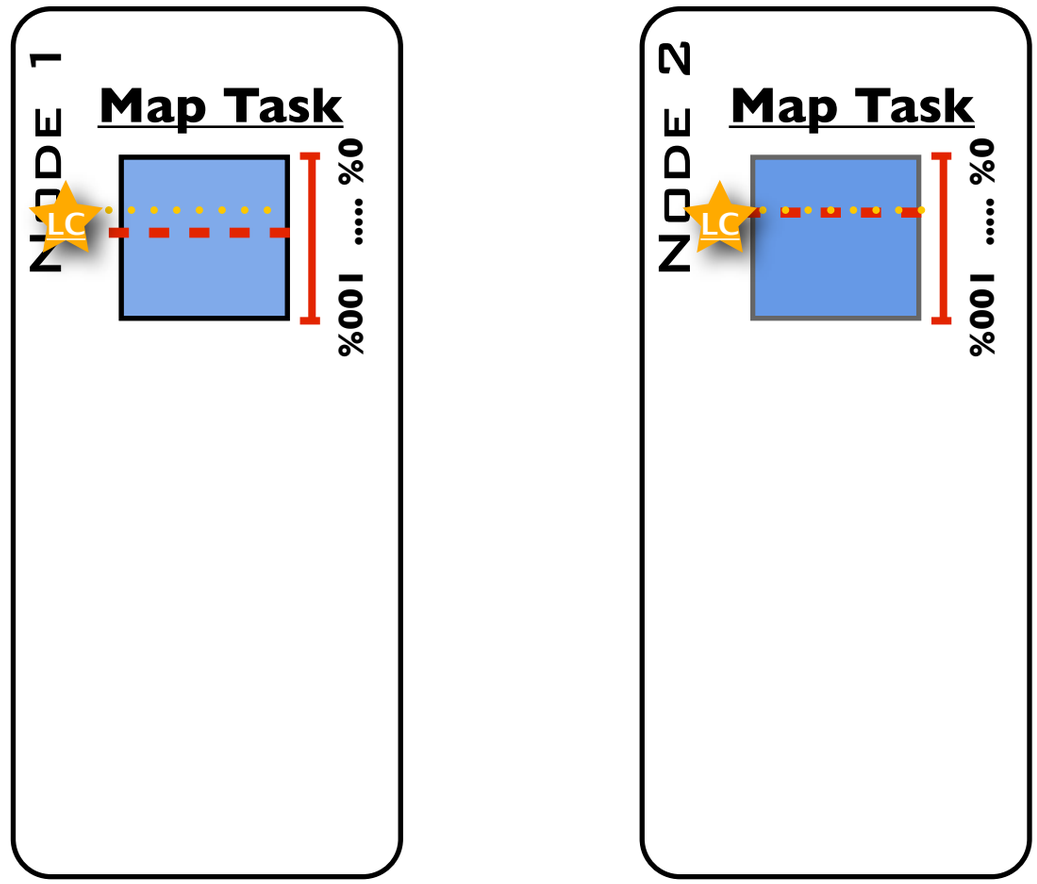


Finish in 40 seconds

RAFT-LC

Time:10s

Scheduler



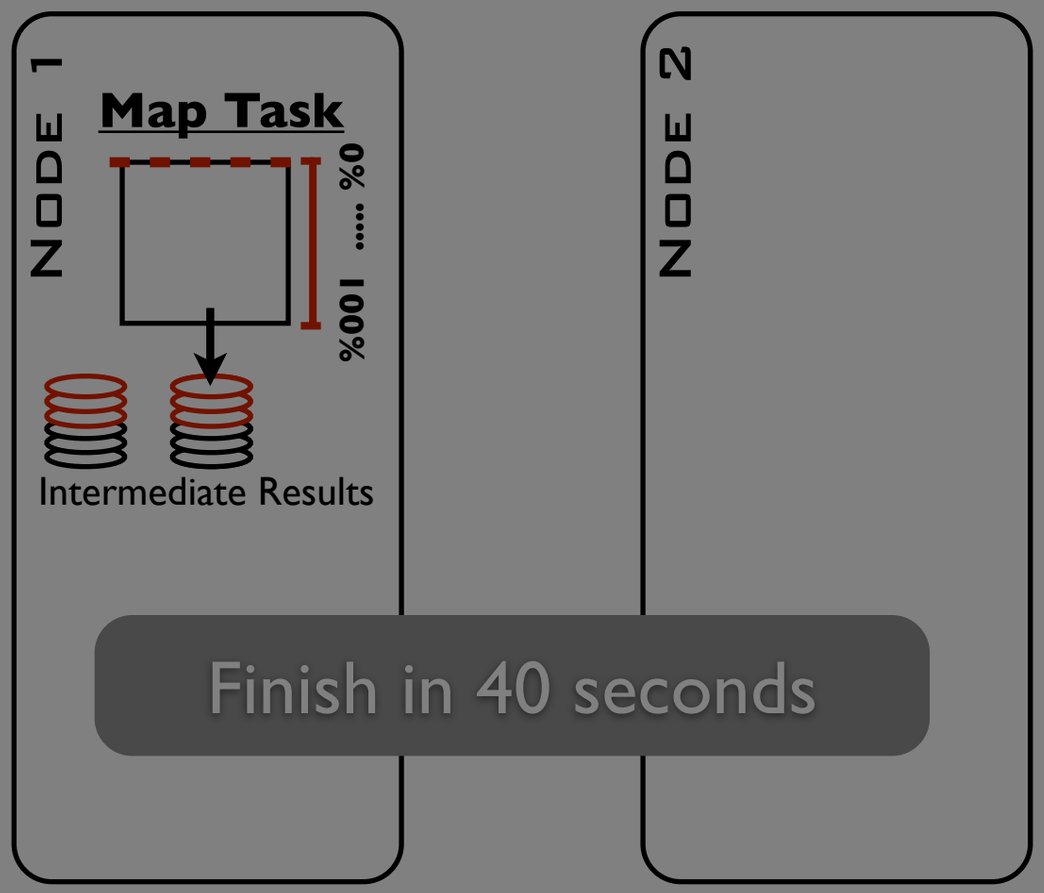
Solution: Local Checkpointing

Map Task: 20 seconds

Hadoop

Time:40s

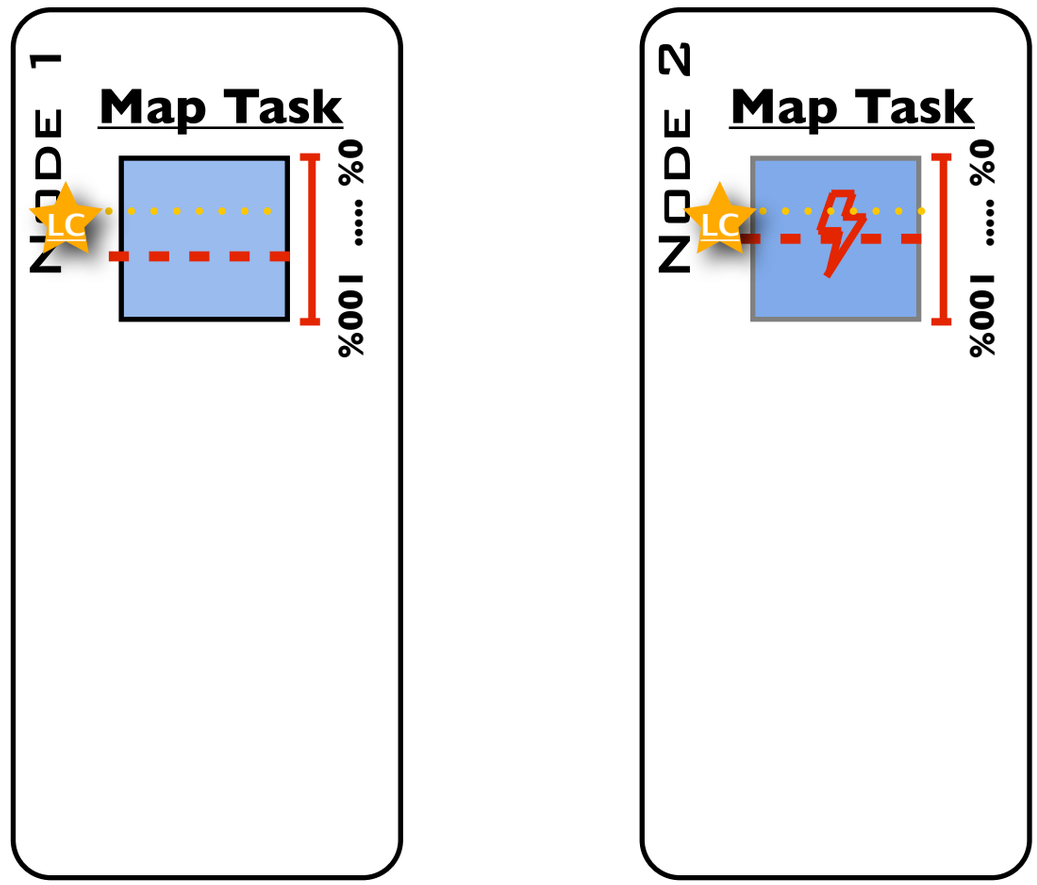
Scheduler



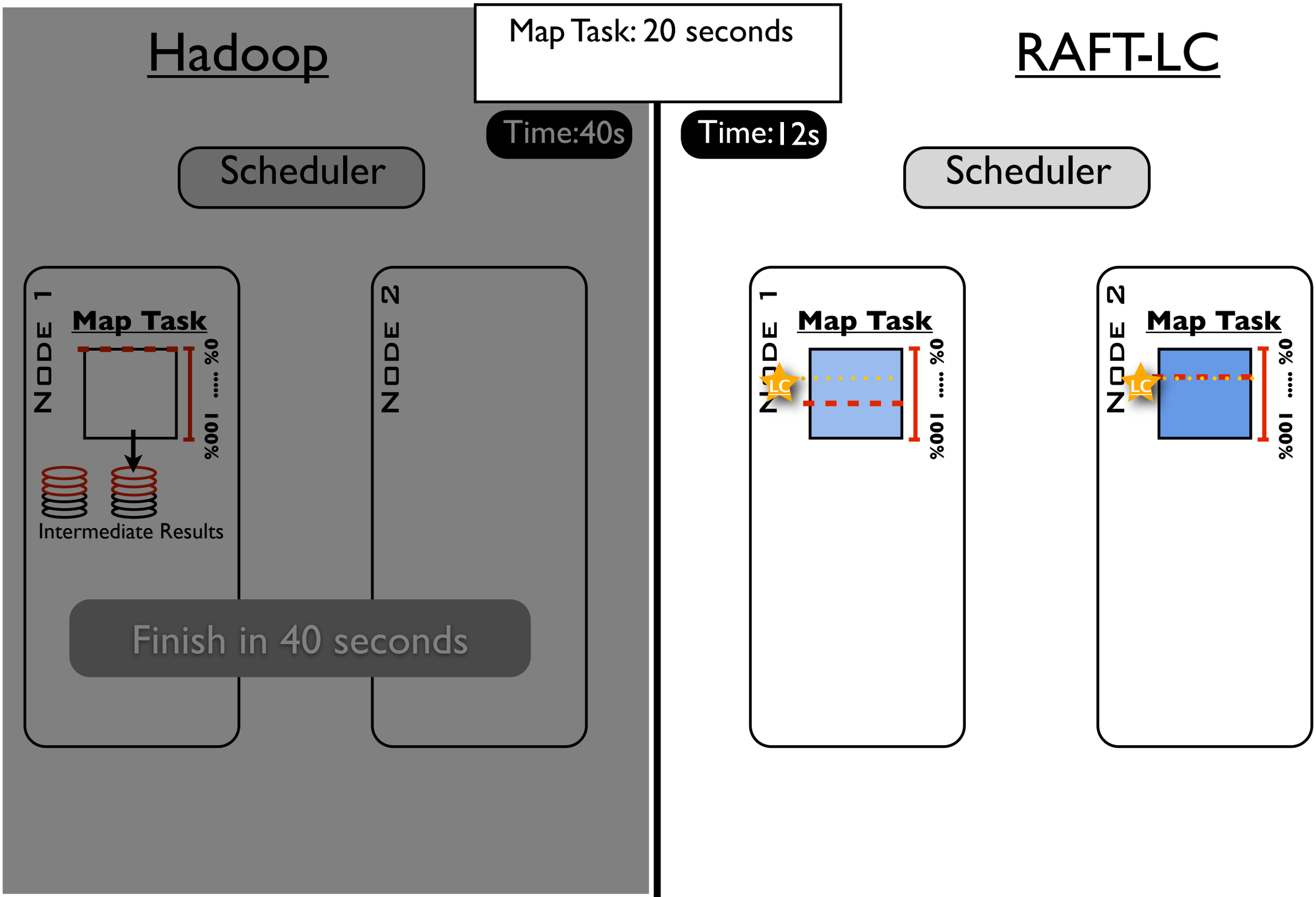
RAFT-LC

Time:12s

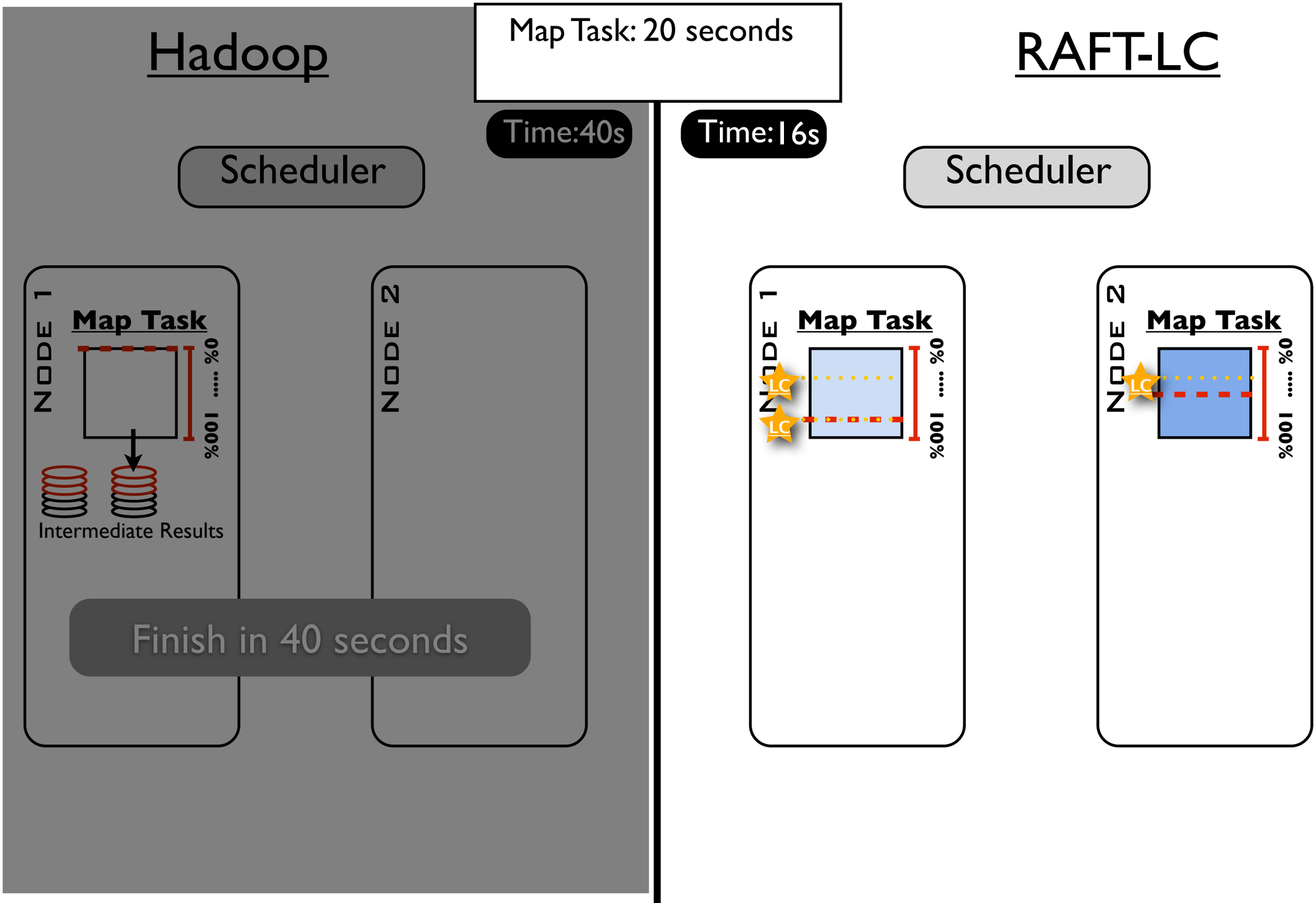
Scheduler



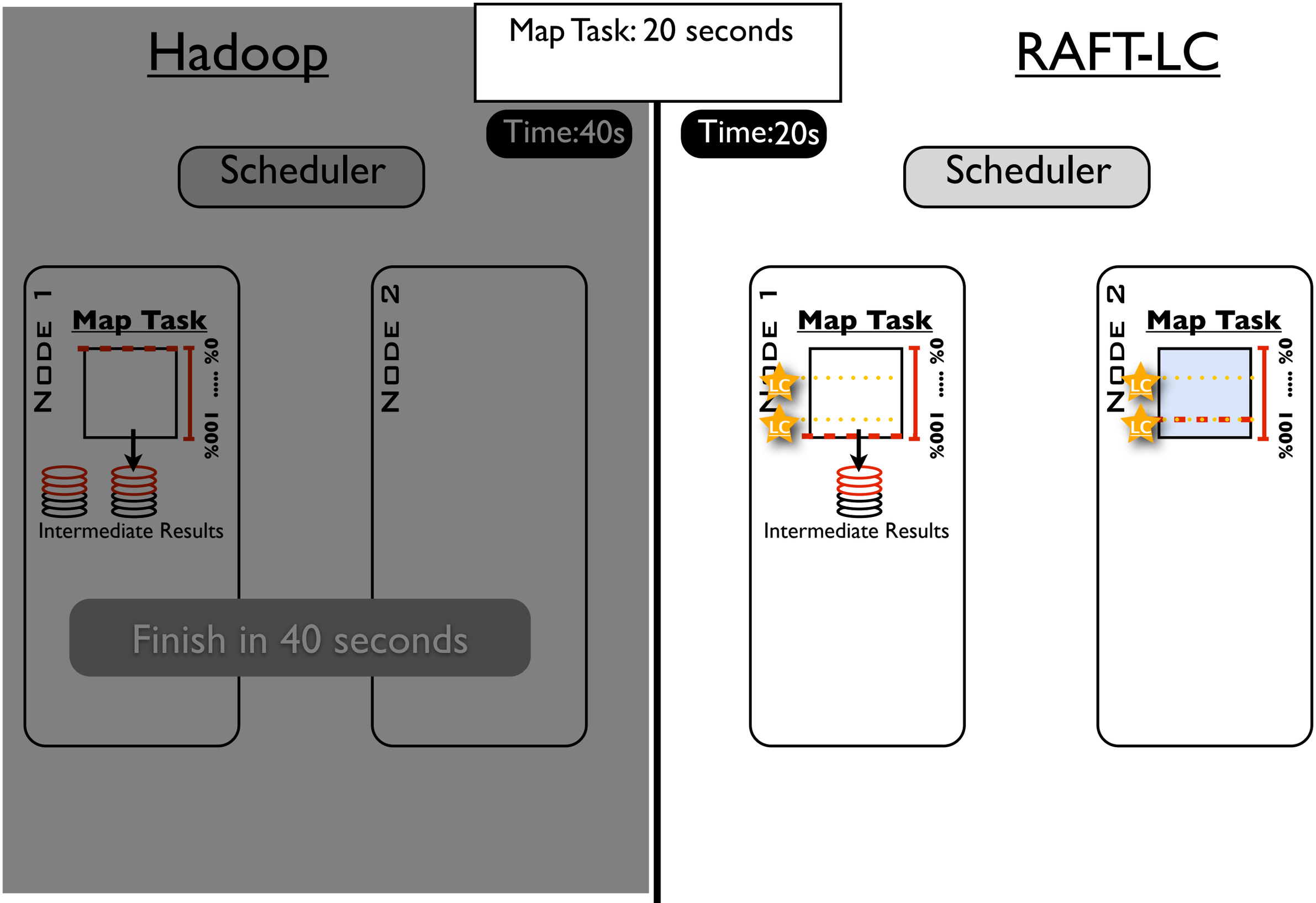
Solution: Local Checkpointing



Solution: Local Checkpointing



Solution: Local Checkpointing



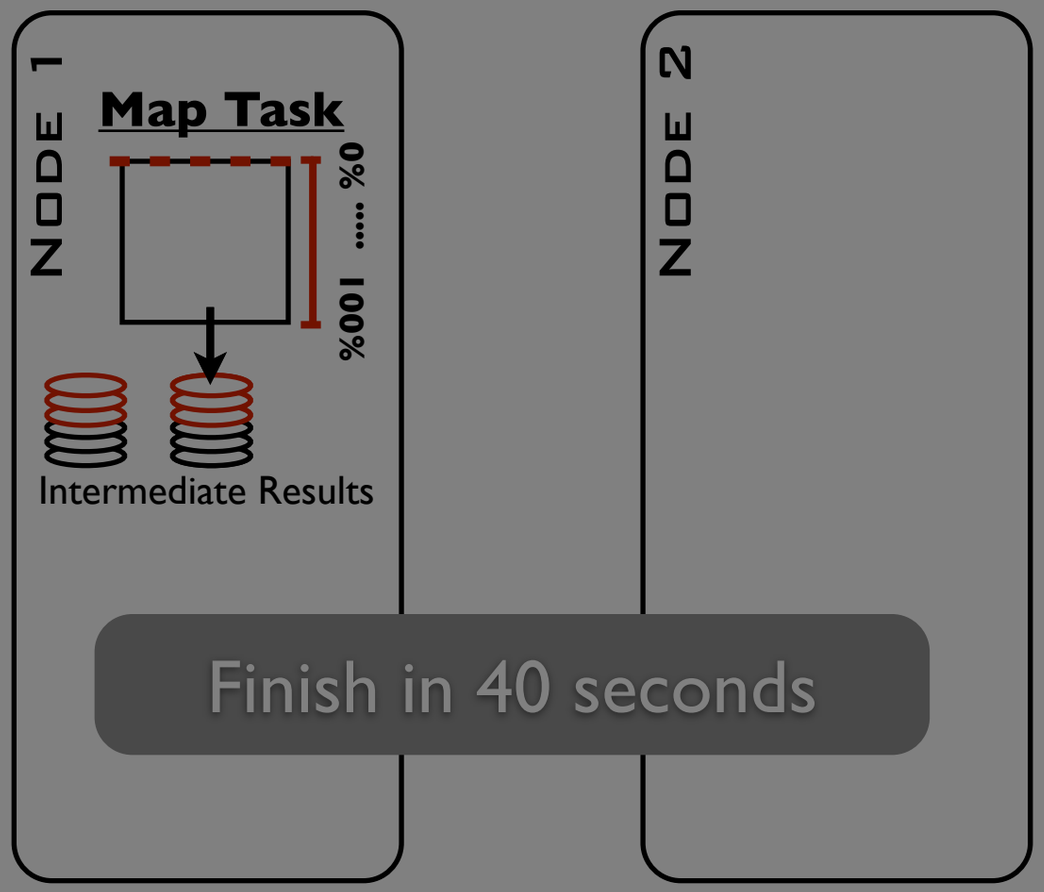
Solution: Local Checkpointing

Map Task: 20 seconds

Hadoop

Time:40s

Scheduler

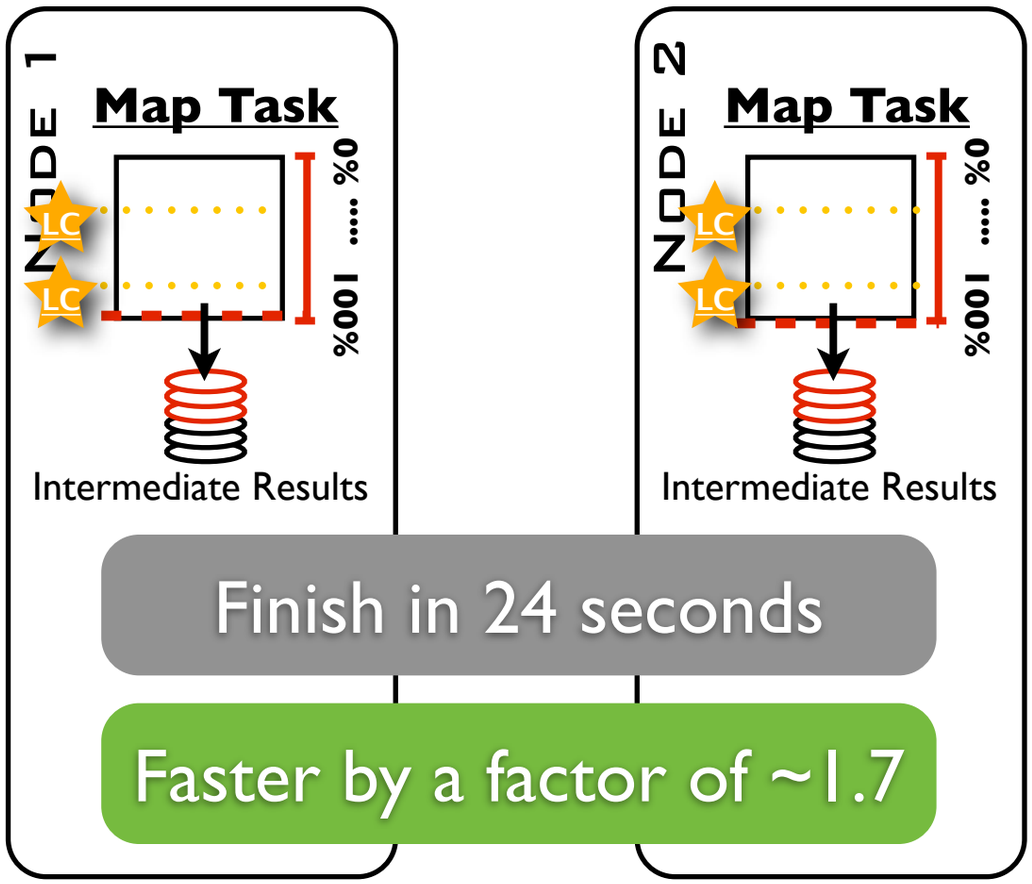


Finish in 40 seconds

RAFT-LC

Time:24s

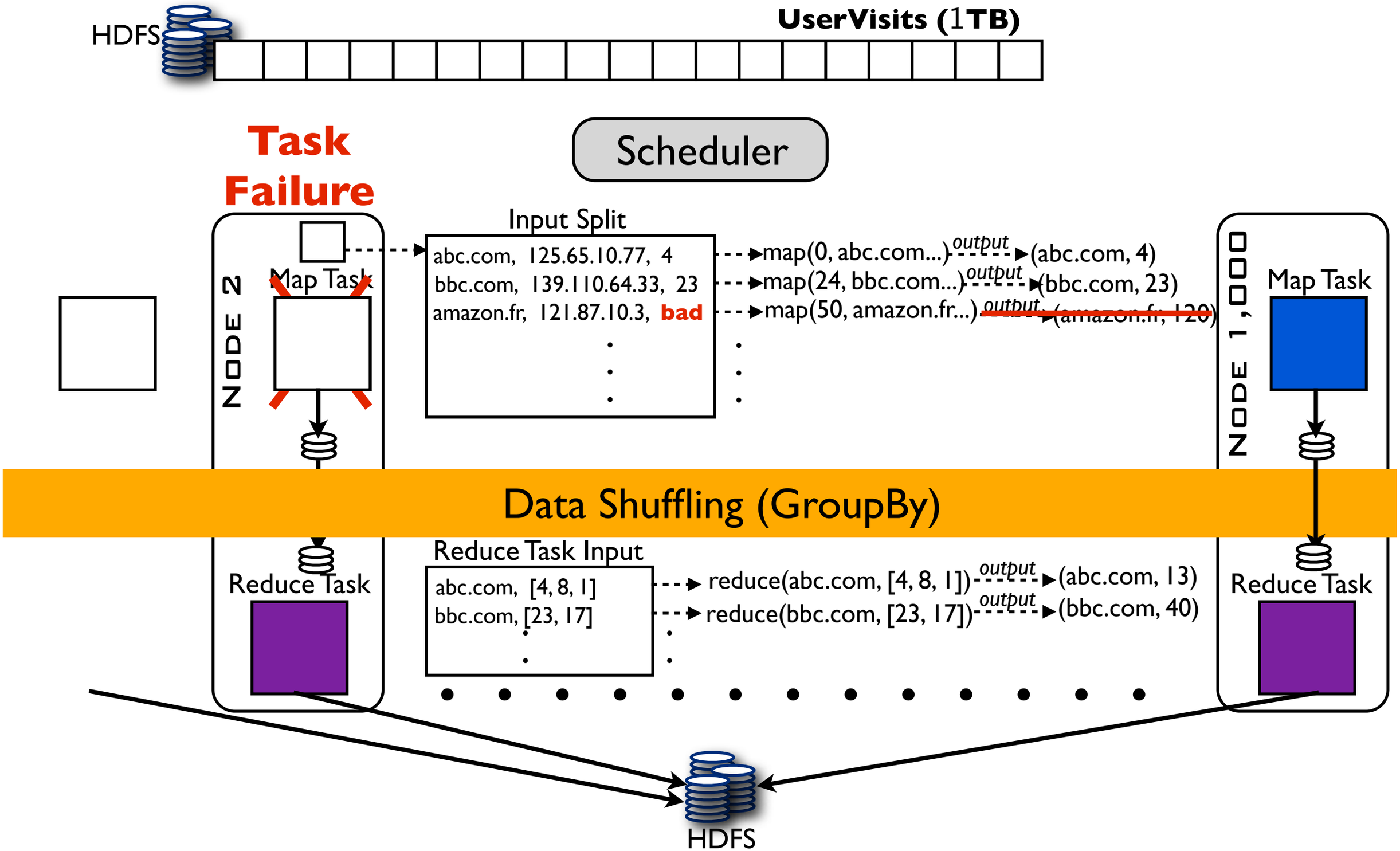
Scheduler



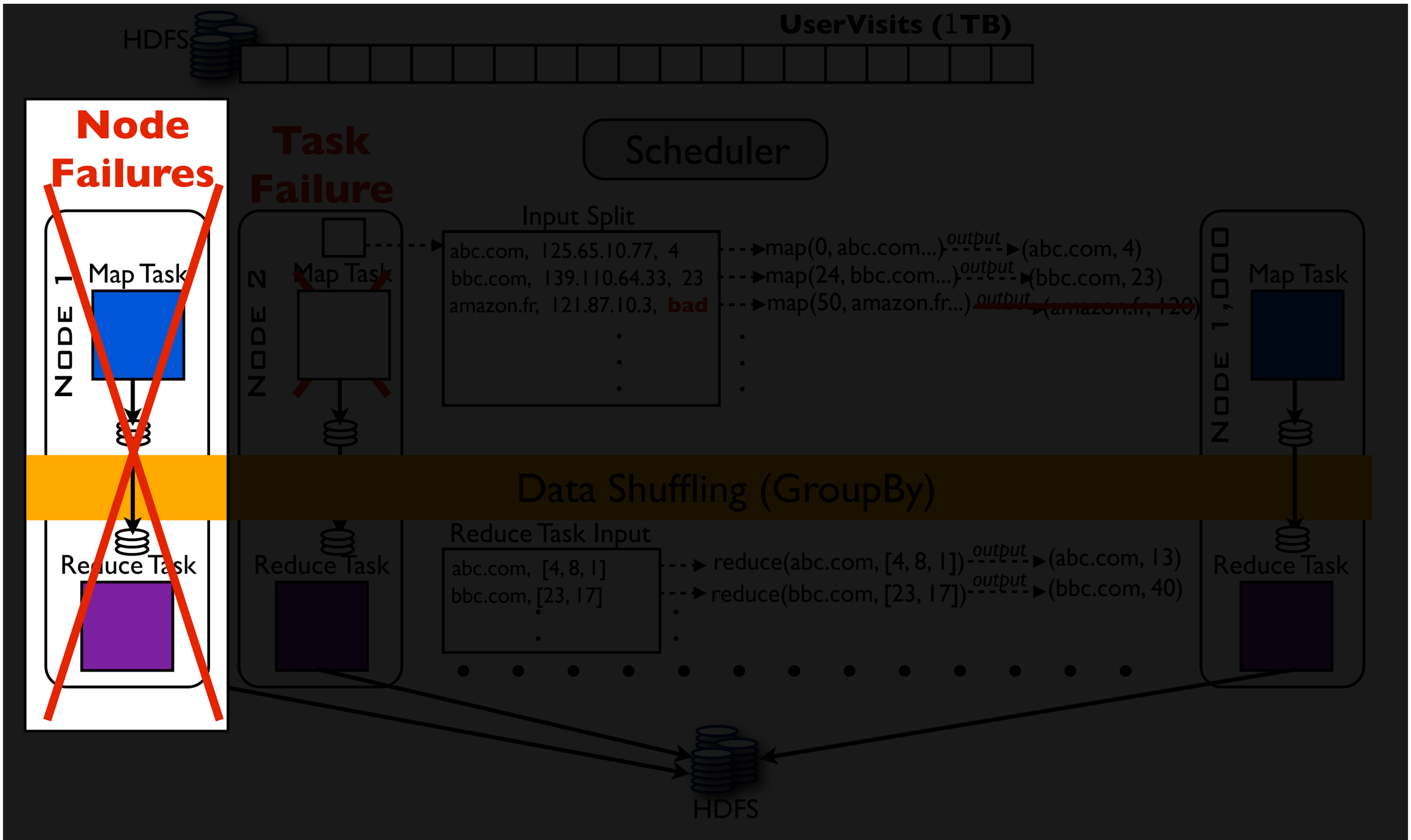
Finish in 24 seconds

Faster by a factor of ~1.7

Node Failures



Node Failures



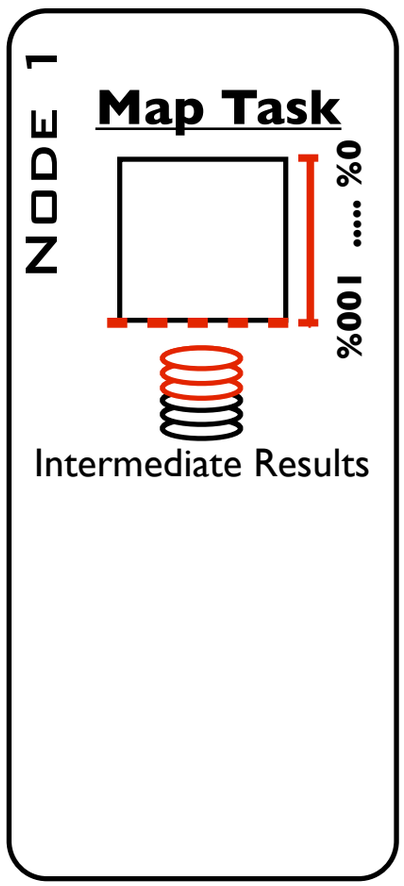
Current Approach

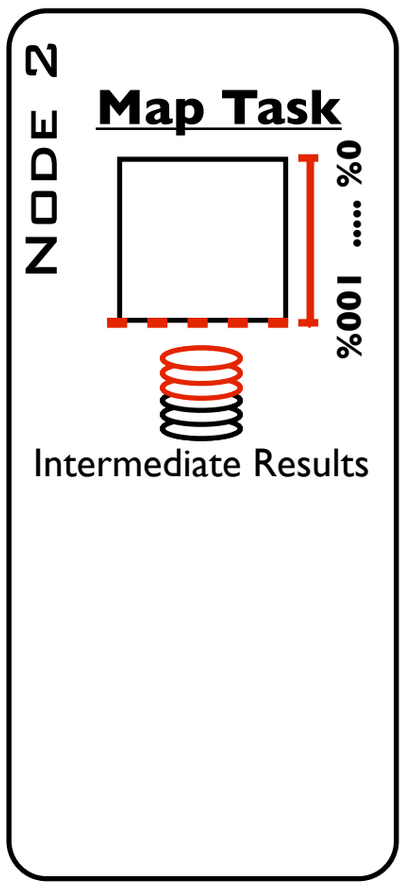
Hadoop without Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:20s

Scheduler

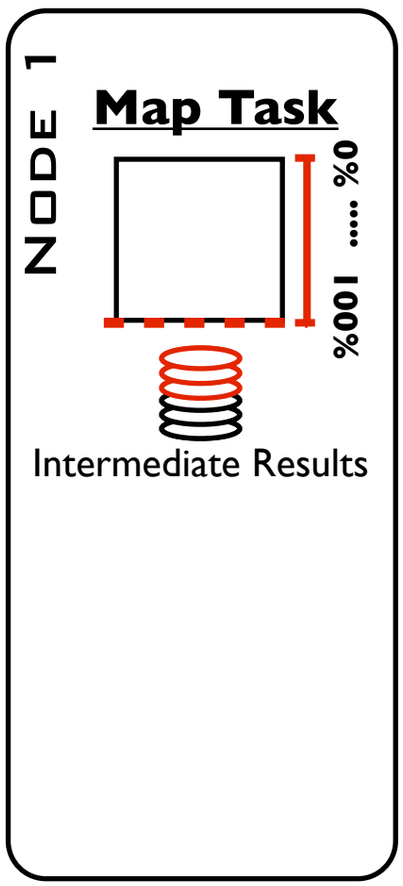


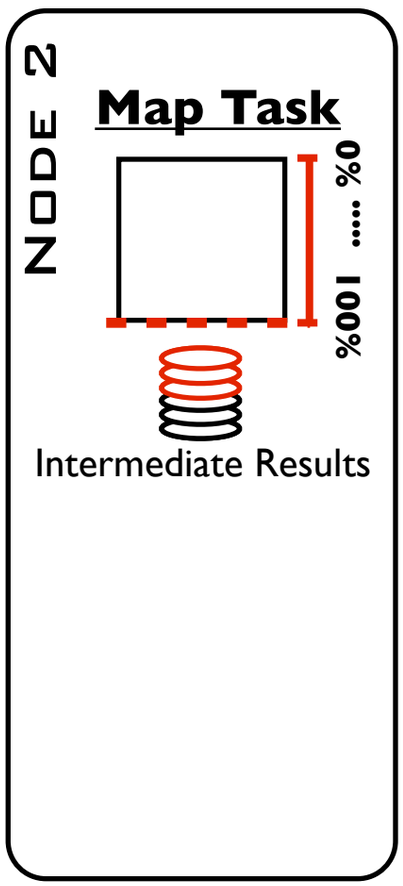


Time:20s

Hadoop with Node Failures

Scheduler





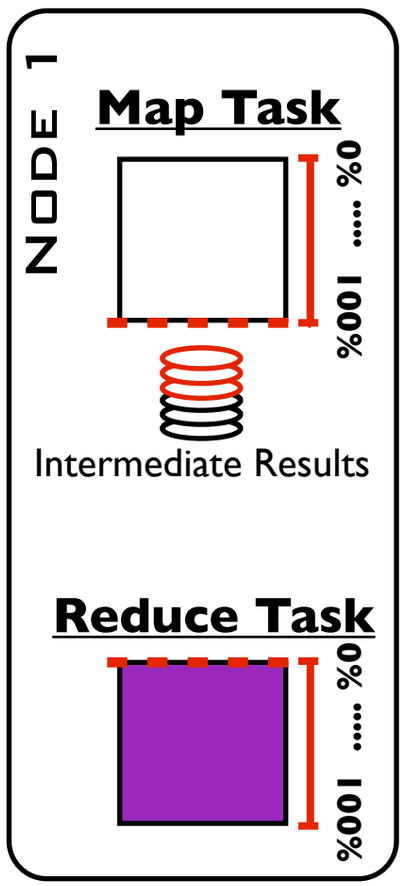
Current Approach

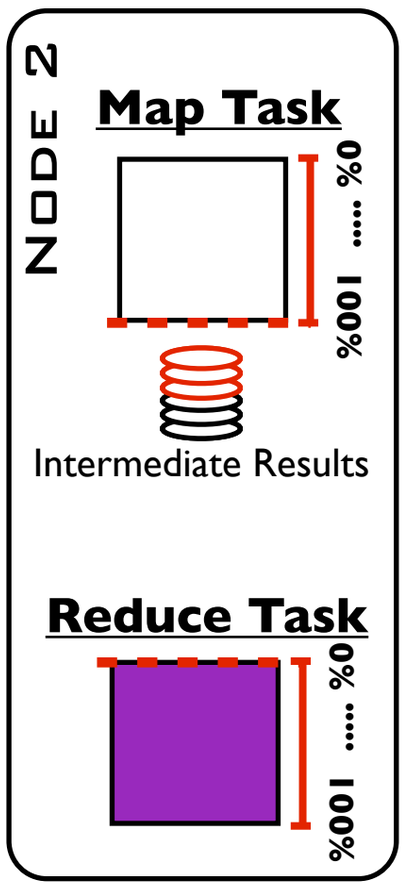
Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop without Failures

Time:20s

Scheduler

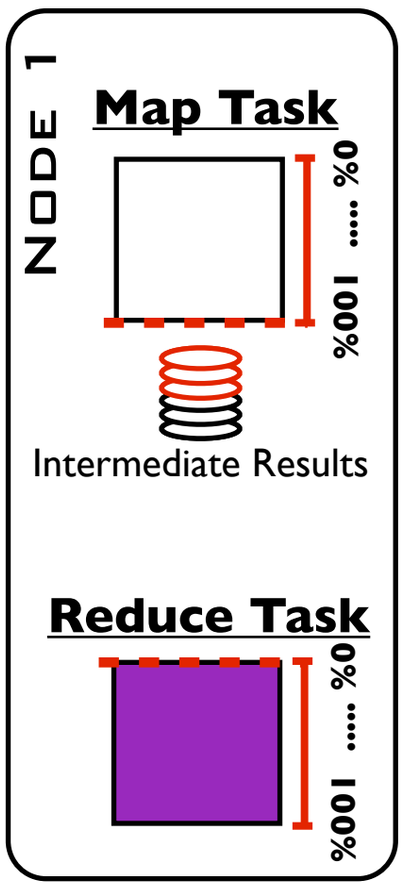


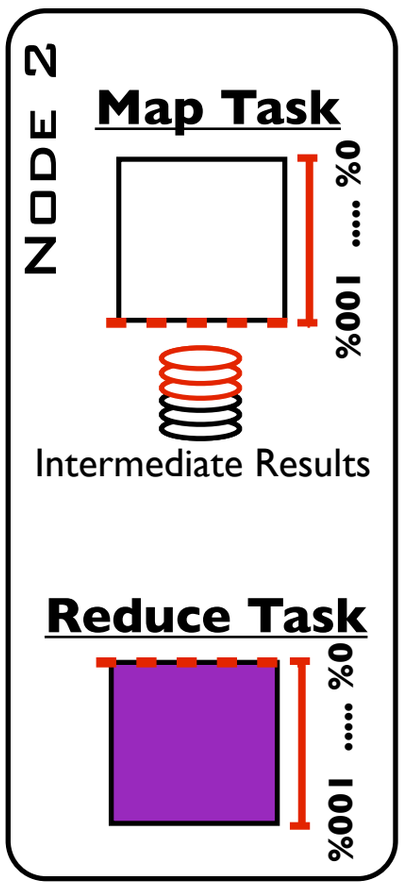


Hadoop with Node Failures

Time:20s

Scheduler





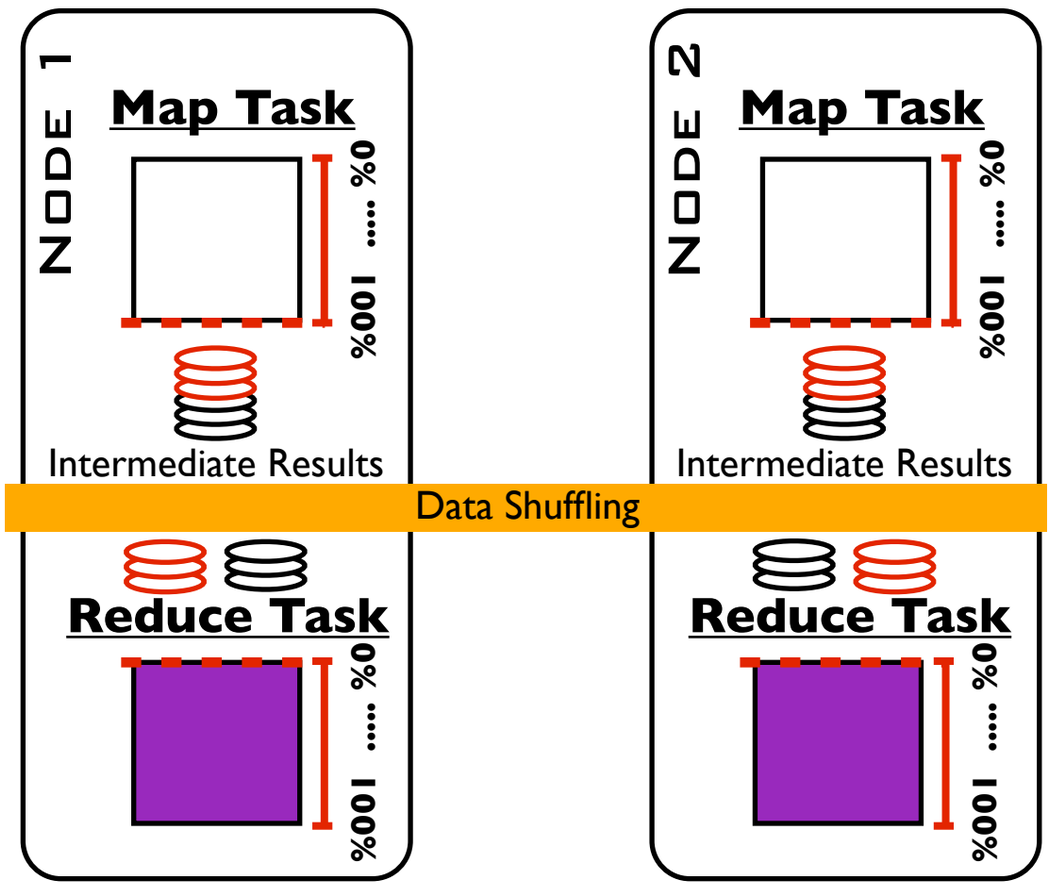
Current Approach

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop without Failures

Time:20s

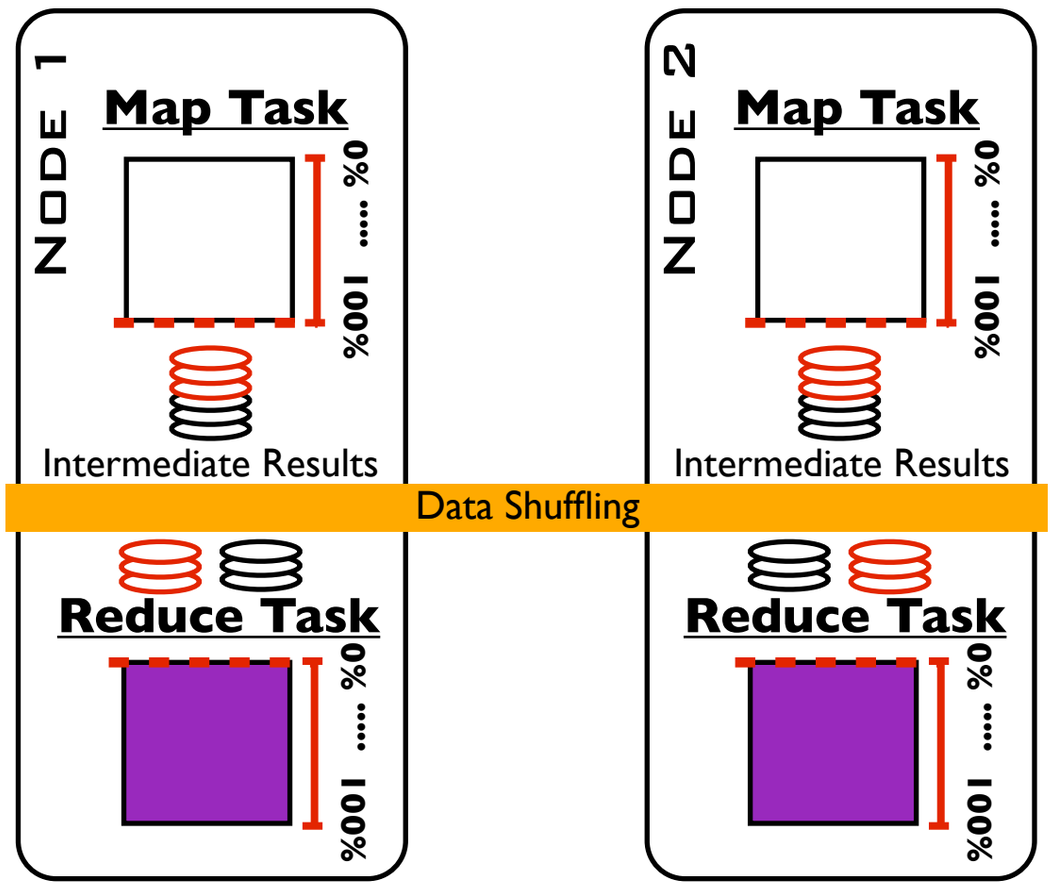
Scheduler



Hadoop with Node Failures

Time:20s

Scheduler

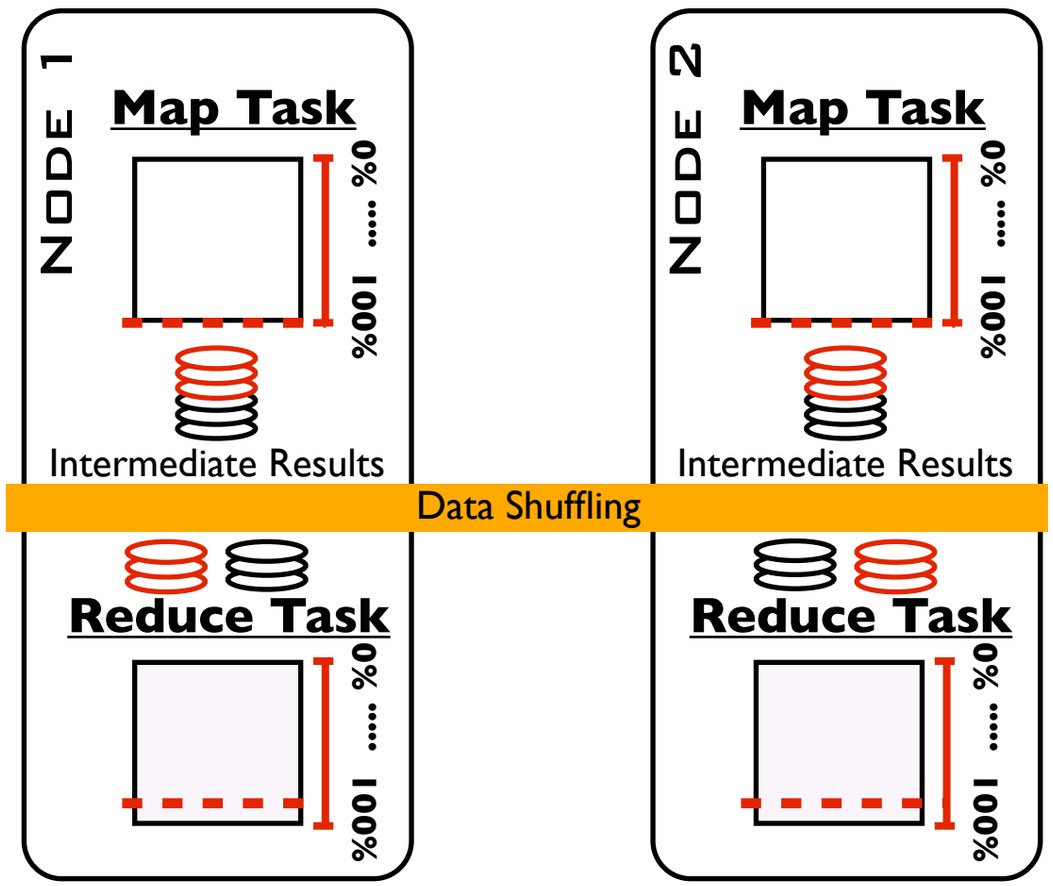


Current Approach

Map Task: 20 seconds
Reduce Task: 30 seconds

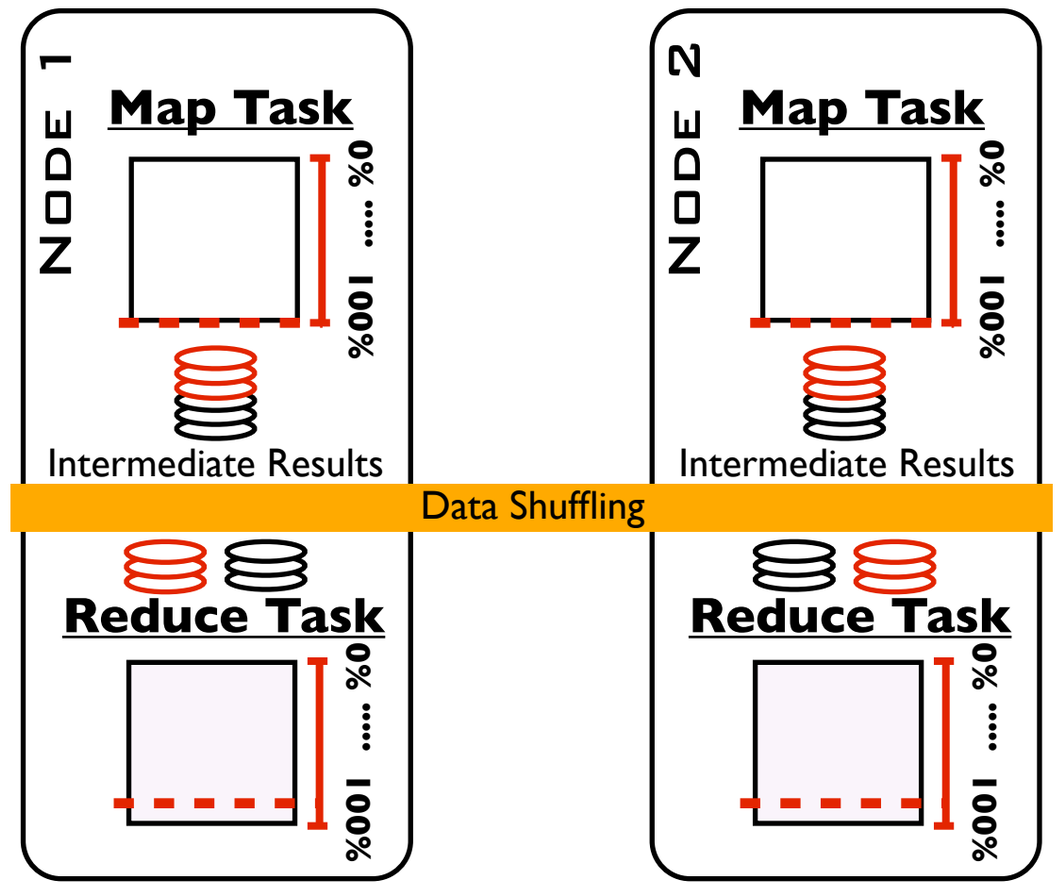
Hadoop without Failures

Time:48s



Hadoop with Node Failures

Time:48s



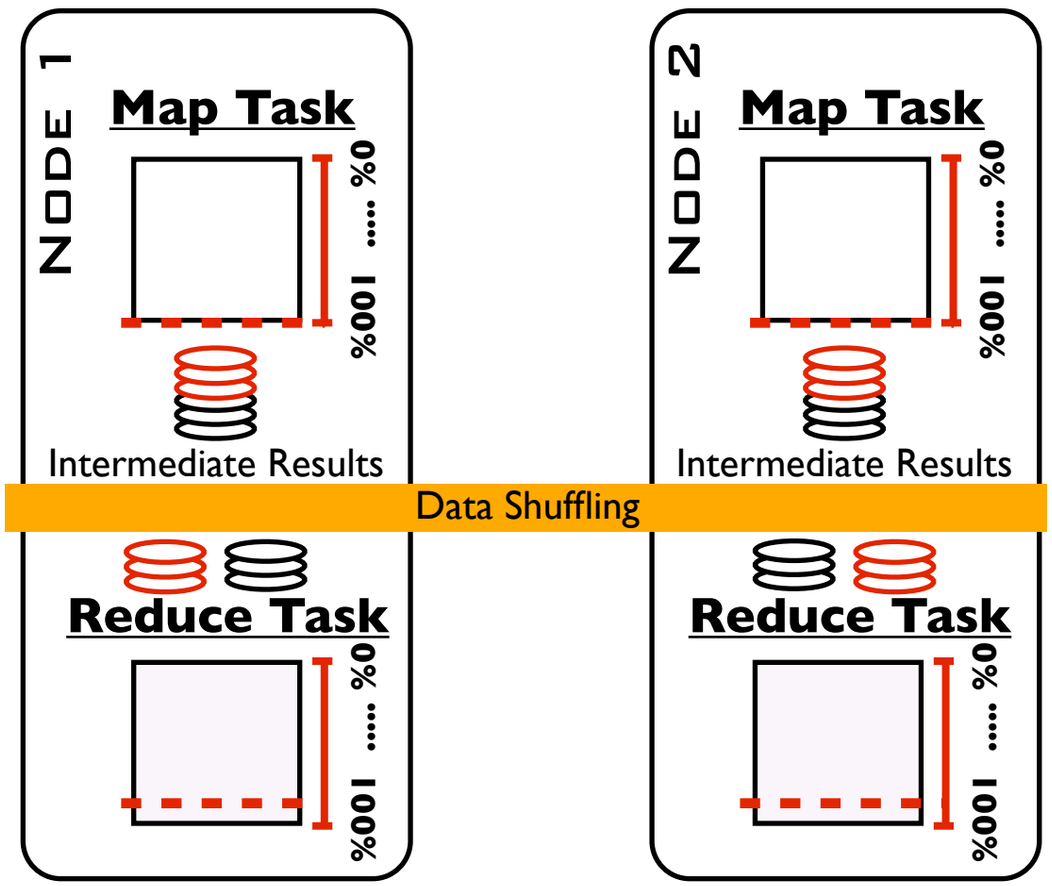
Current Approach

Hadoop without Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:48s

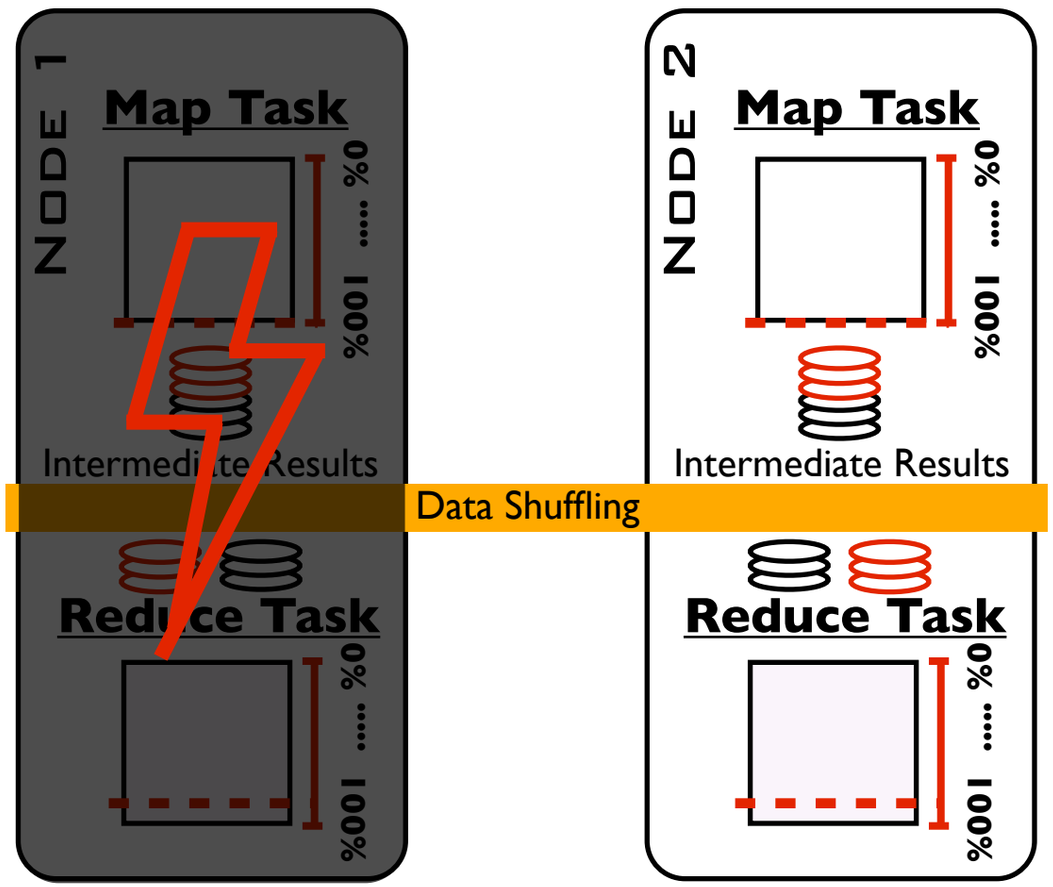
Scheduler



Hadoop with Node Failures

Time:48s

Scheduler



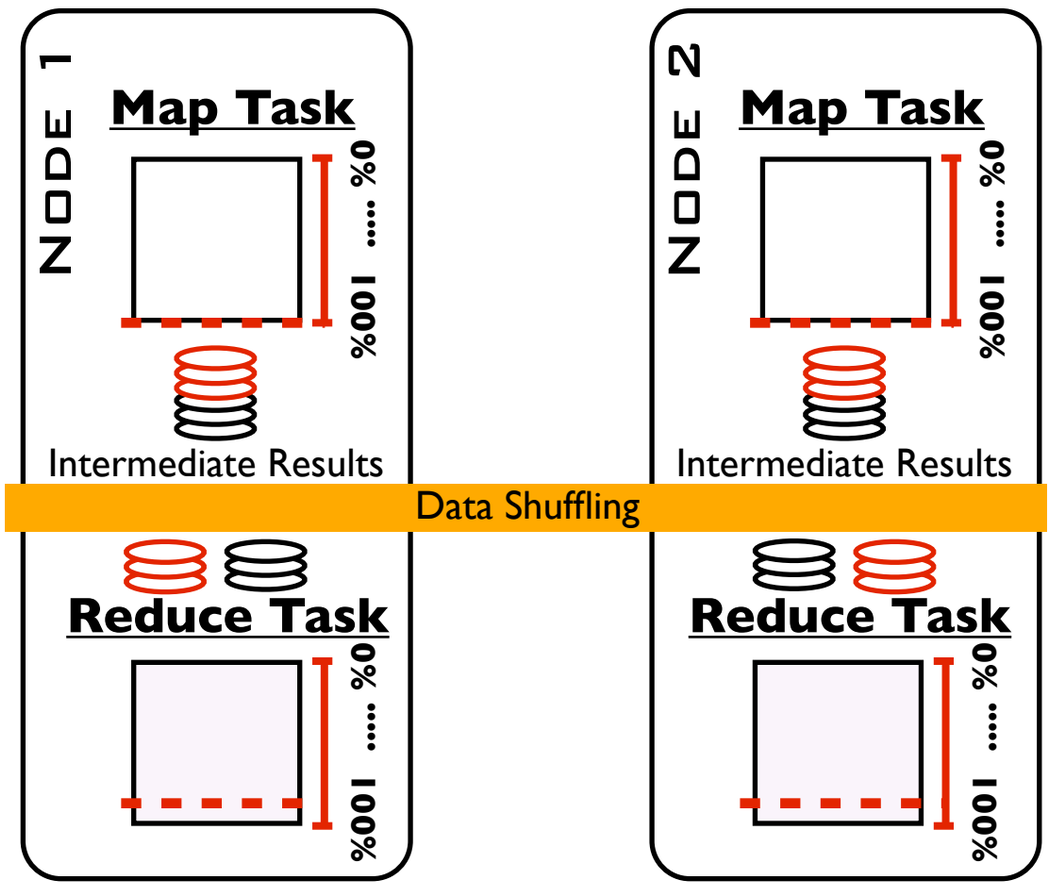
Current Approach

Hadoop without Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:48s

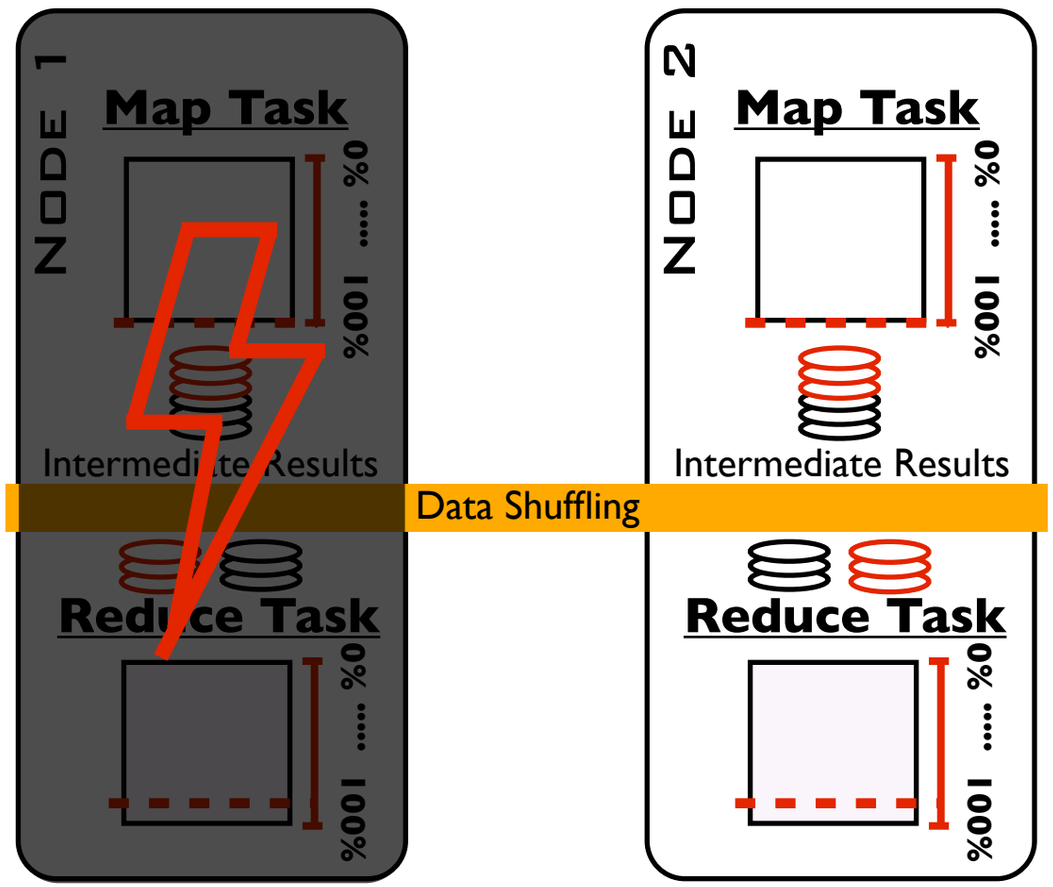
Scheduler



Hadoop with Node Failures

Time:48s

Scheduler

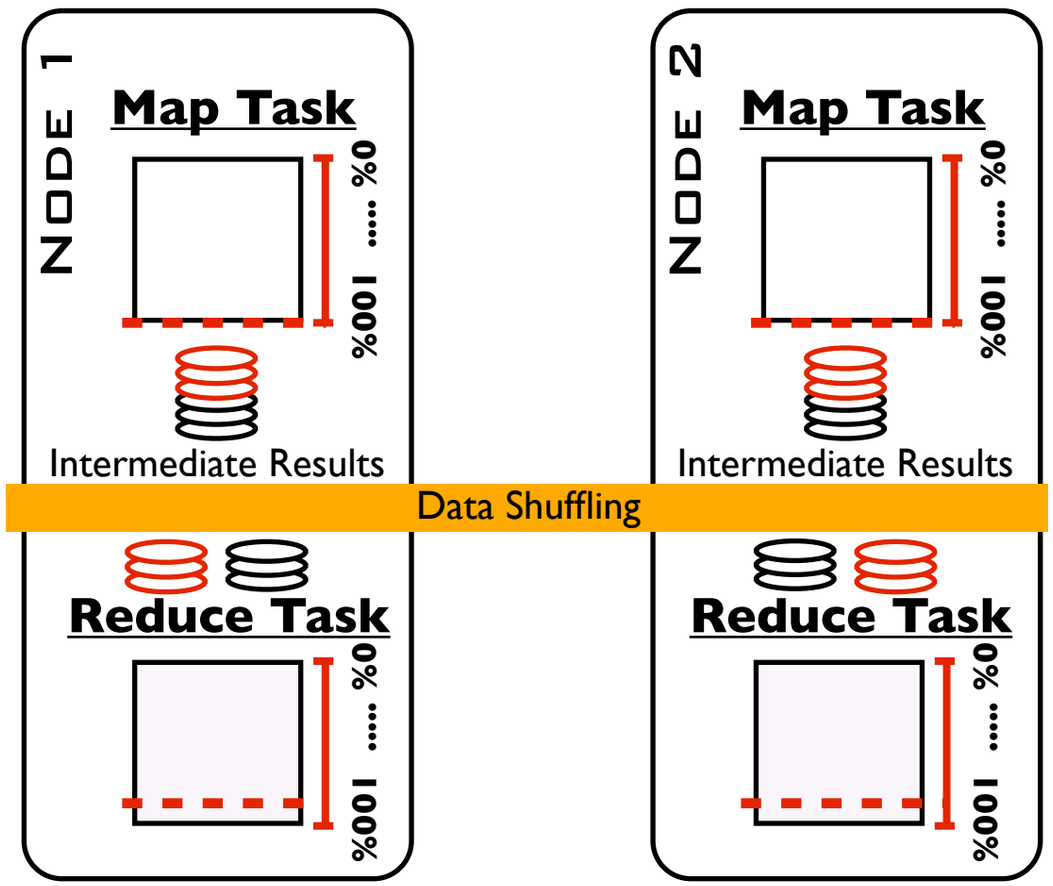


Current Approach

Map Task: 20 seconds
Reduce Task: 30 seconds

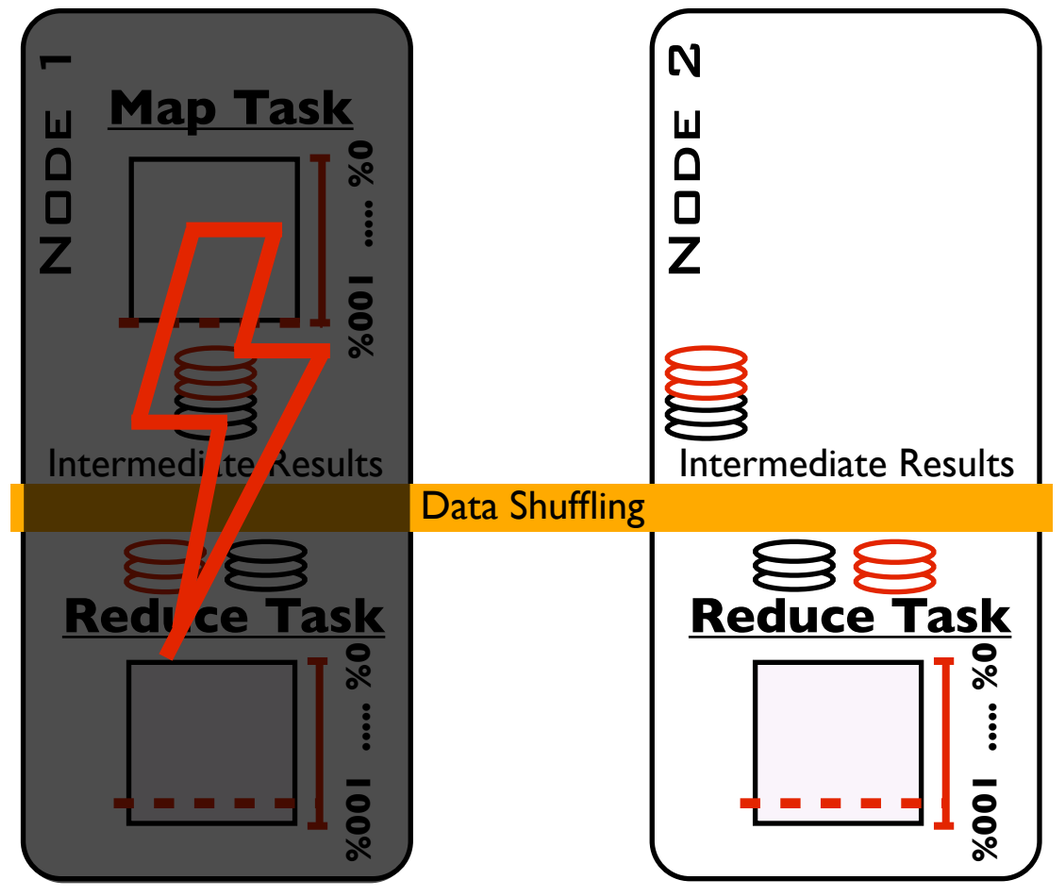
Hadoop without Failures

Time:48s



Hadoop with Node Failures

Time:48s



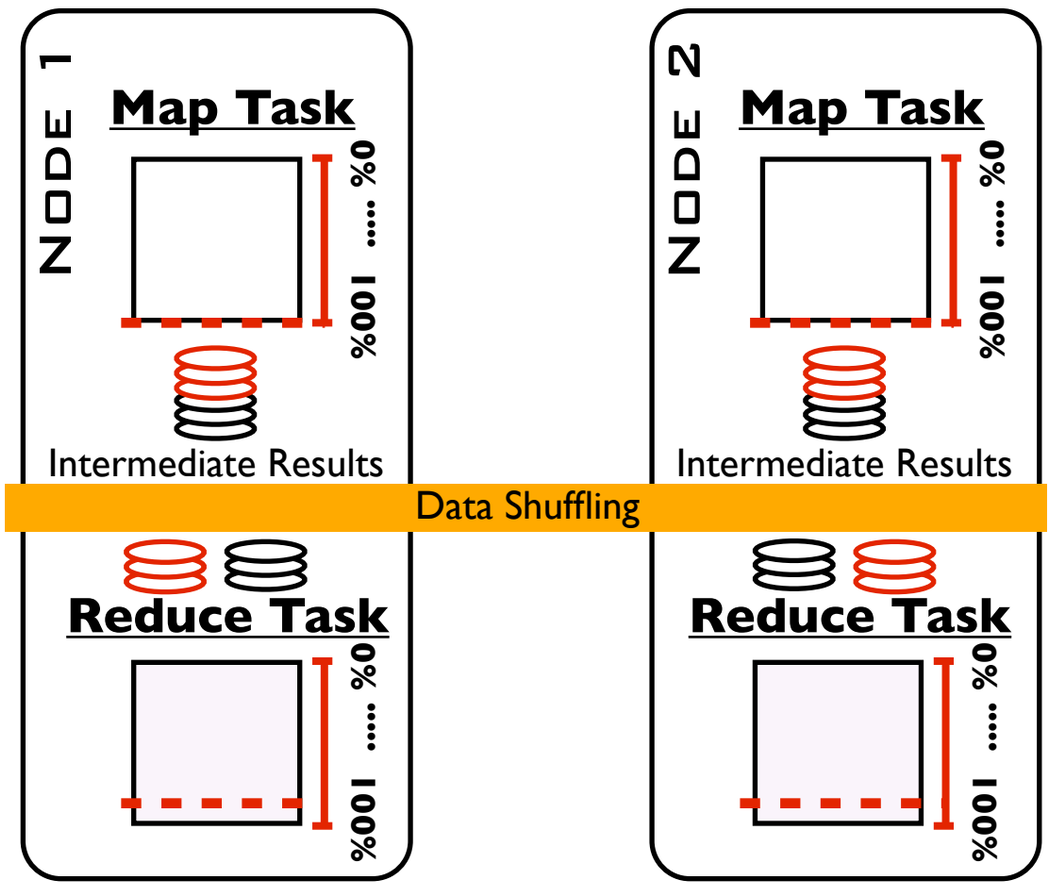
Current Approach

Hadoop without Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:48s

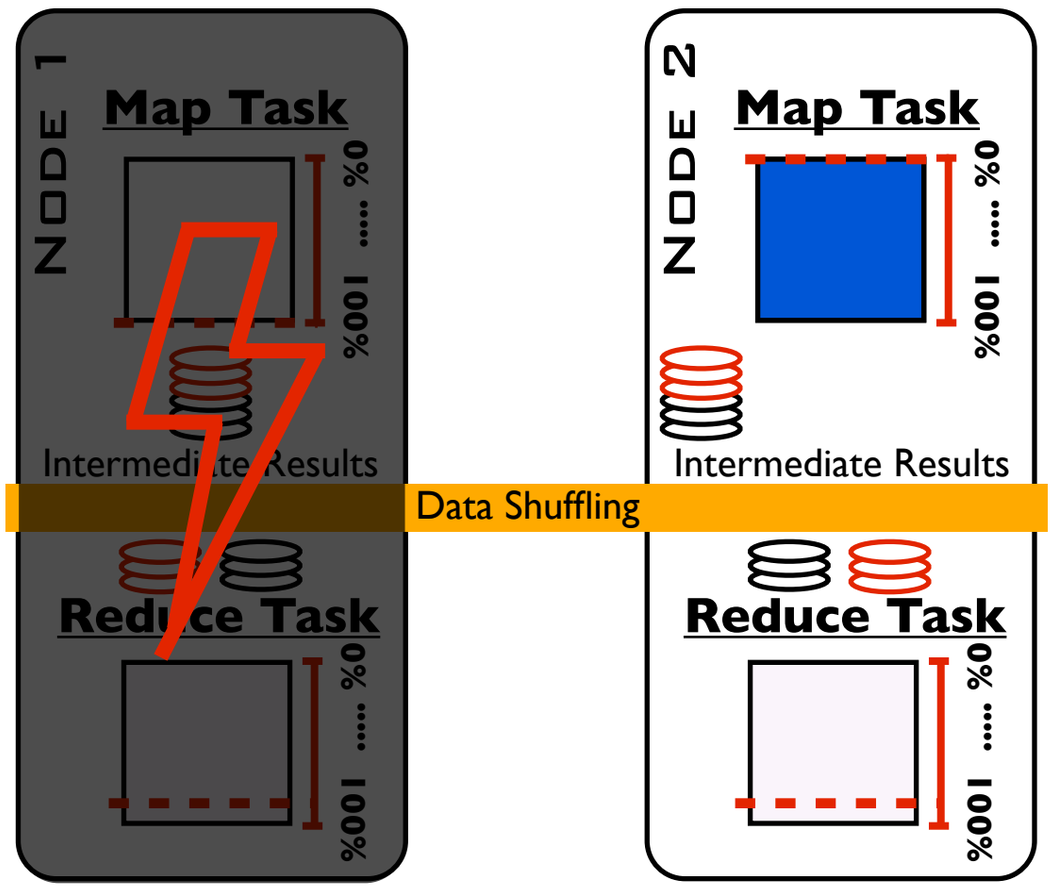
Scheduler



Hadoop with Node Failures

Time:48s

Scheduler



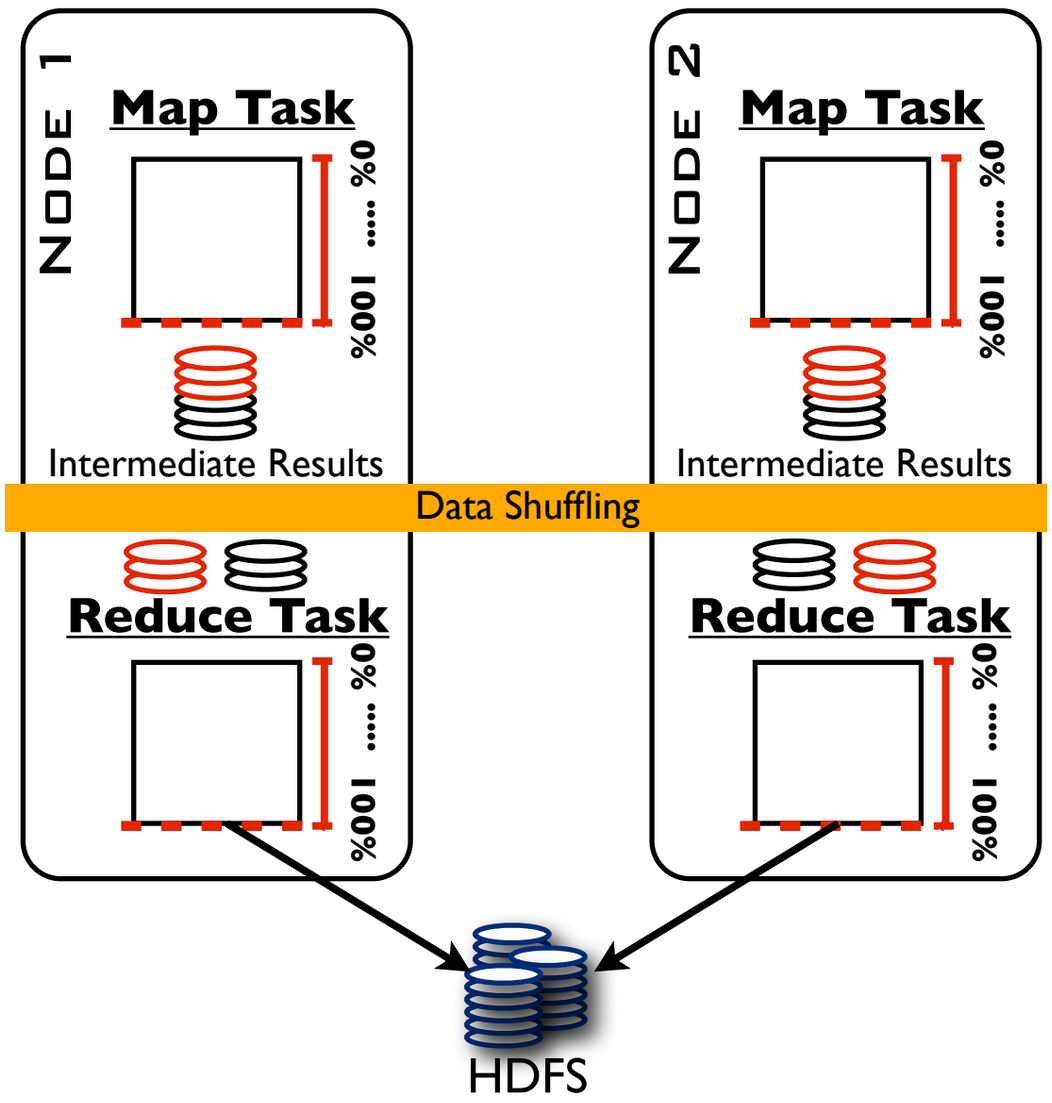
Current Approach

Hadoop without Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:50s

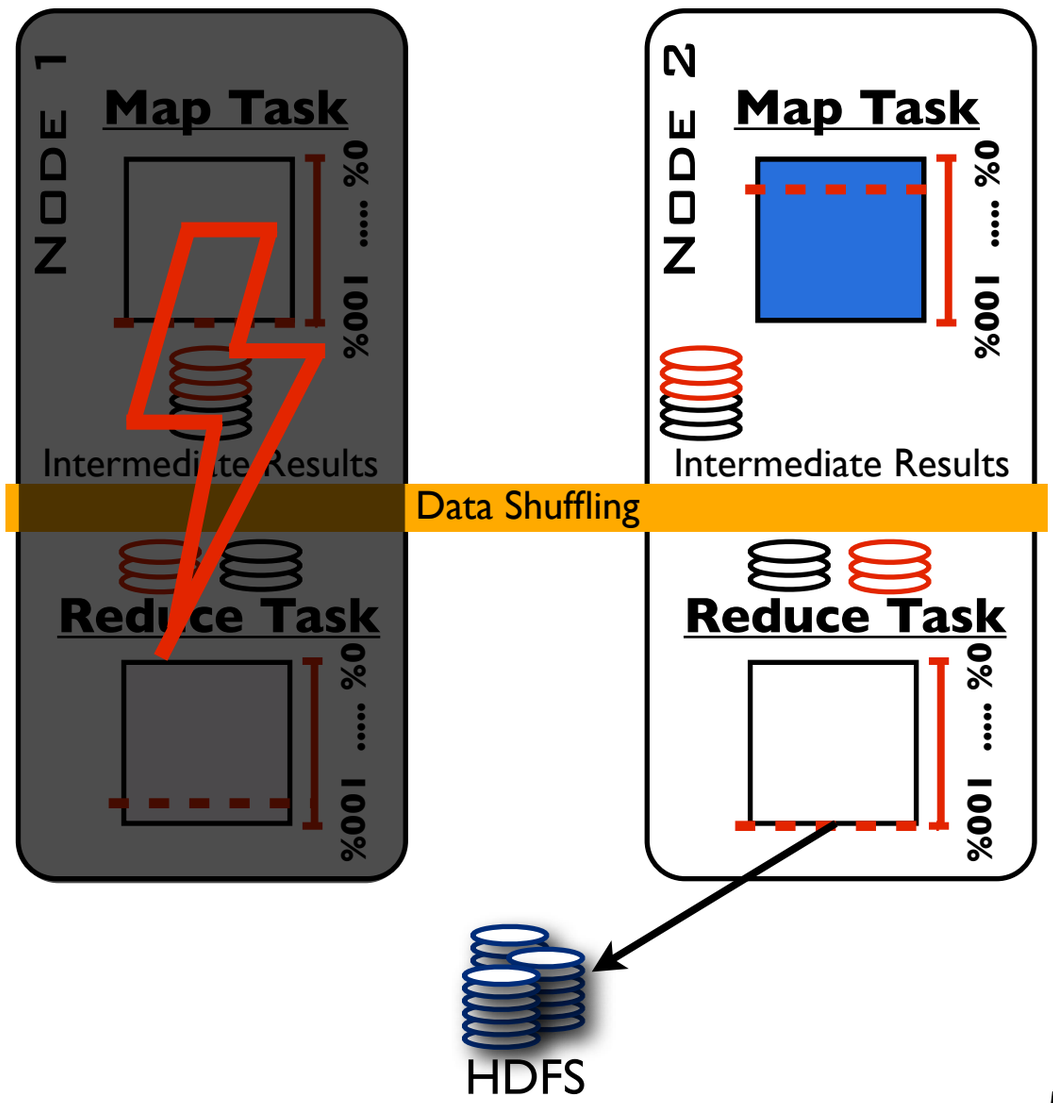
Scheduler



Time:50s

Hadoop with Node Failures

Scheduler



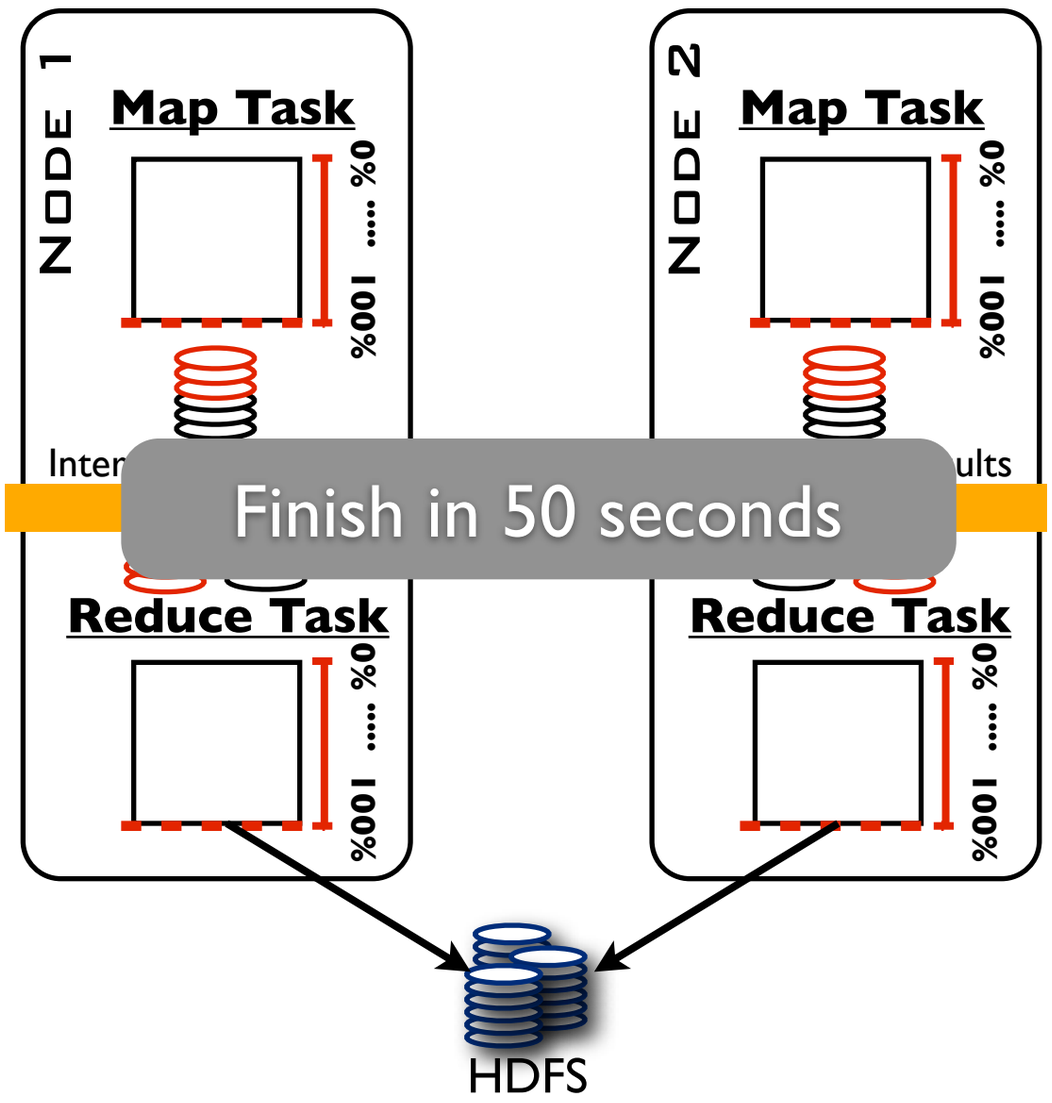
Current Approach

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop without Failures

Time:50s

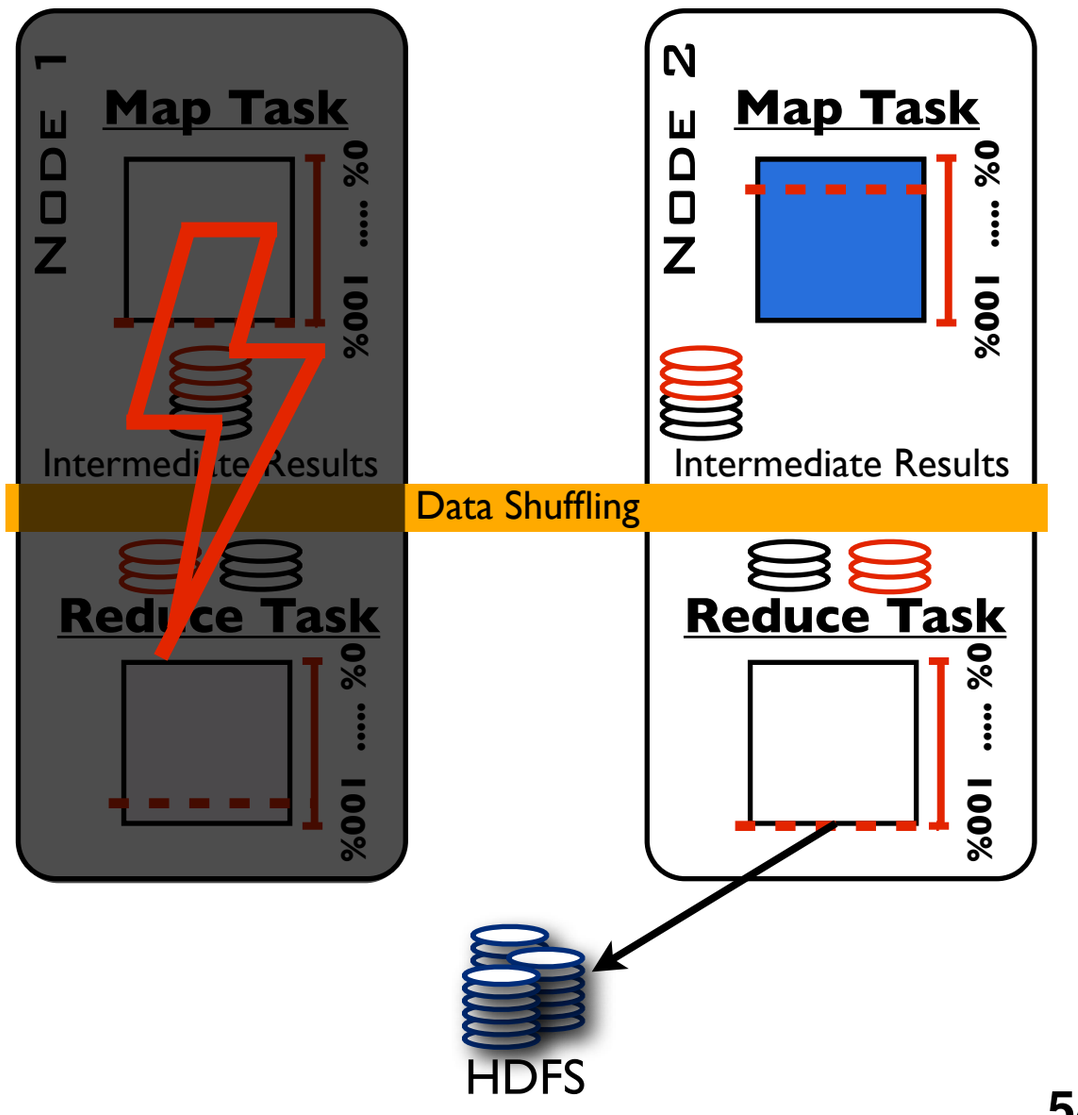
Scheduler



Hadoop with Node Failures

Time:50s

Scheduler



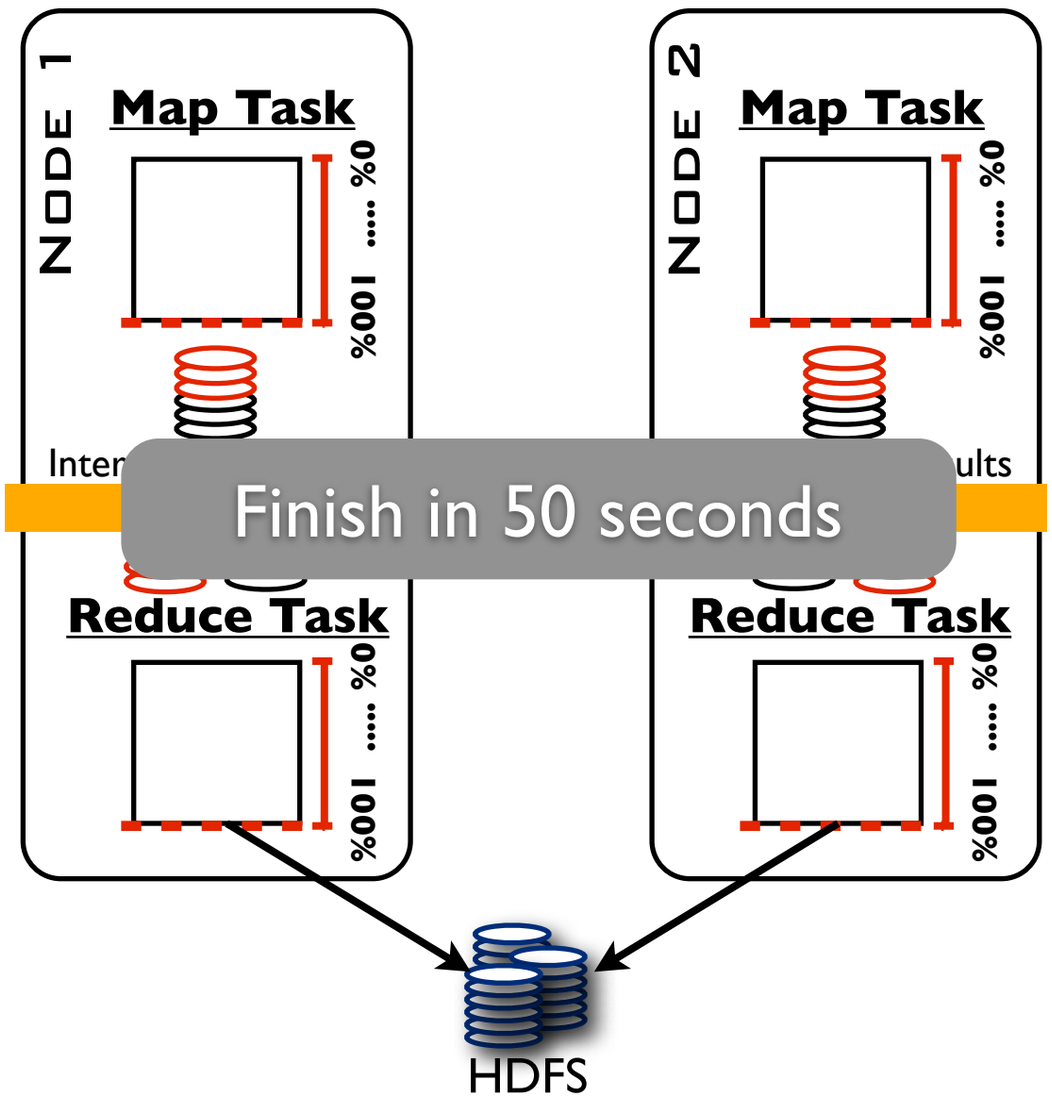
Current Approach

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop without Failures

Time:50s

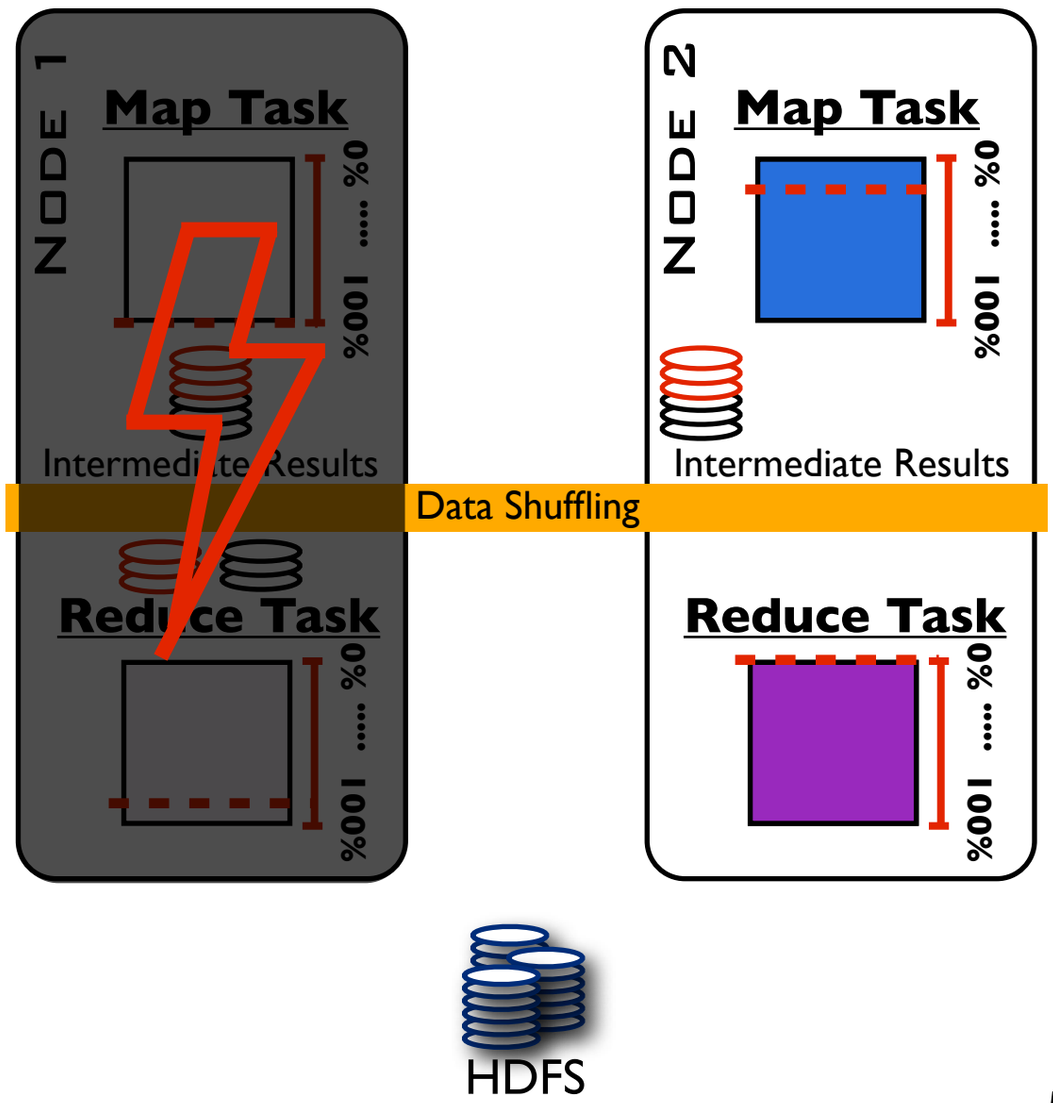
Scheduler



Hadoop with Node Failures

Time:50s

Scheduler



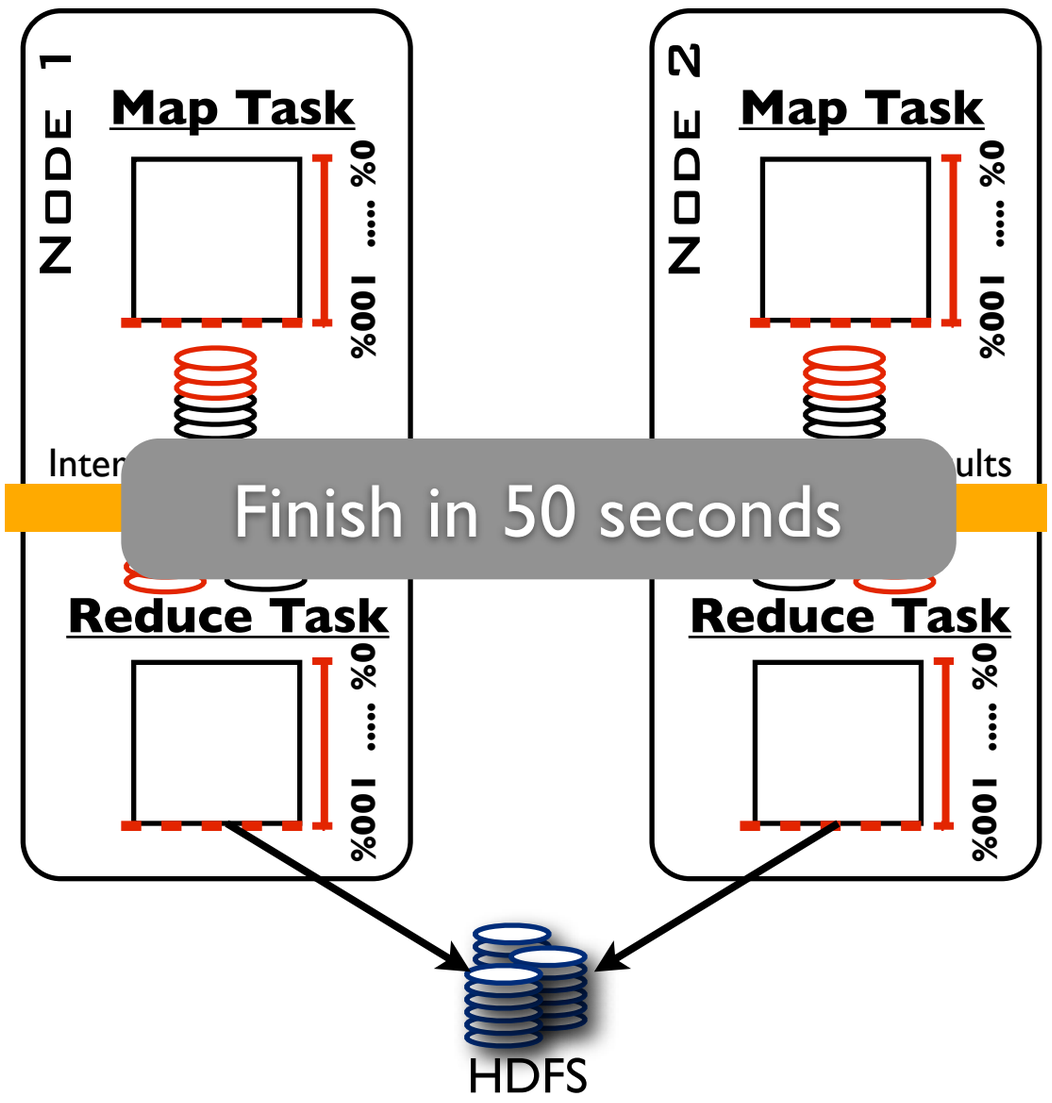
Current Approach

Hadoop without Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:50s

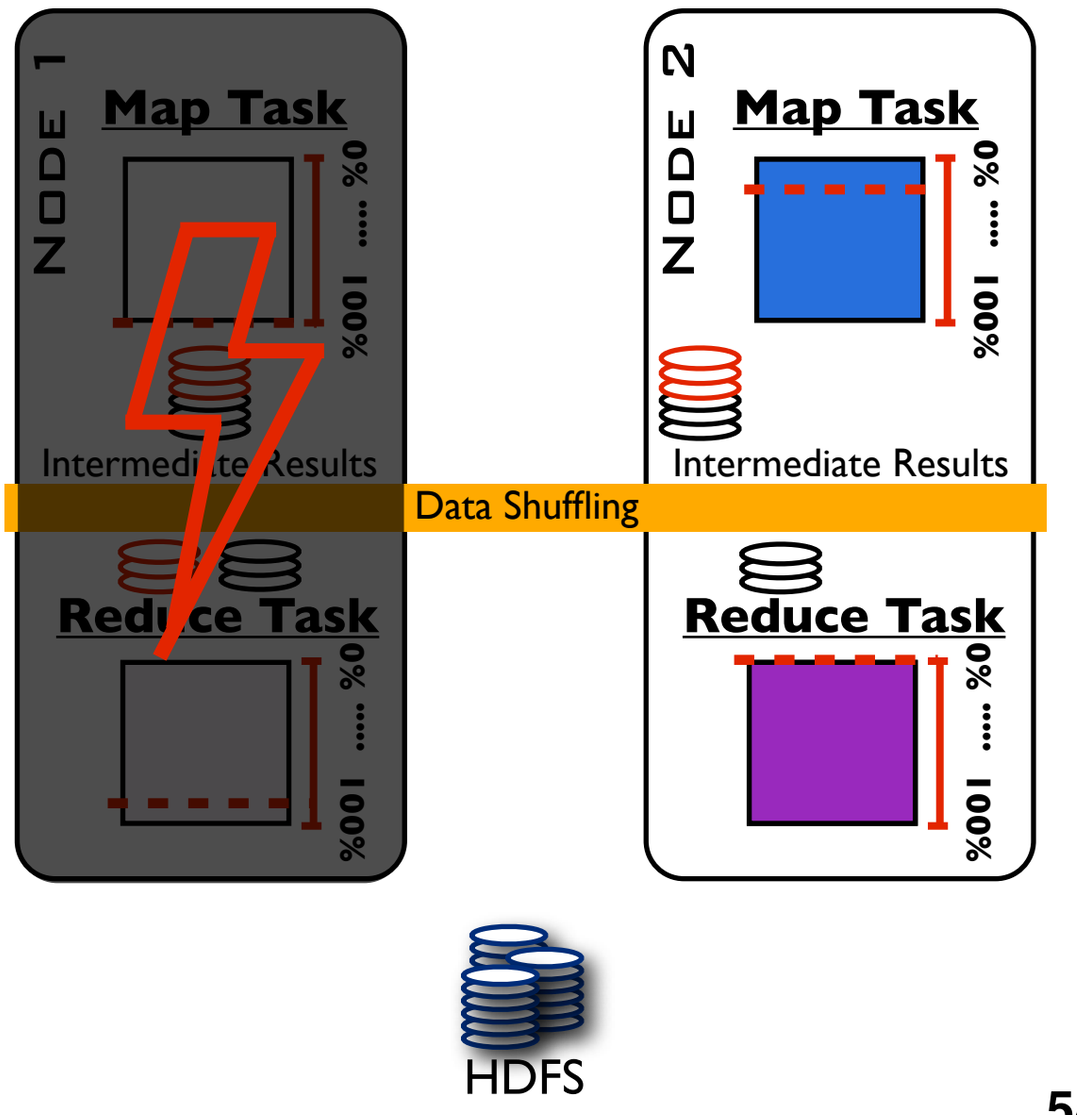
Scheduler



Hadoop with Node Failures

Time:50s

Scheduler



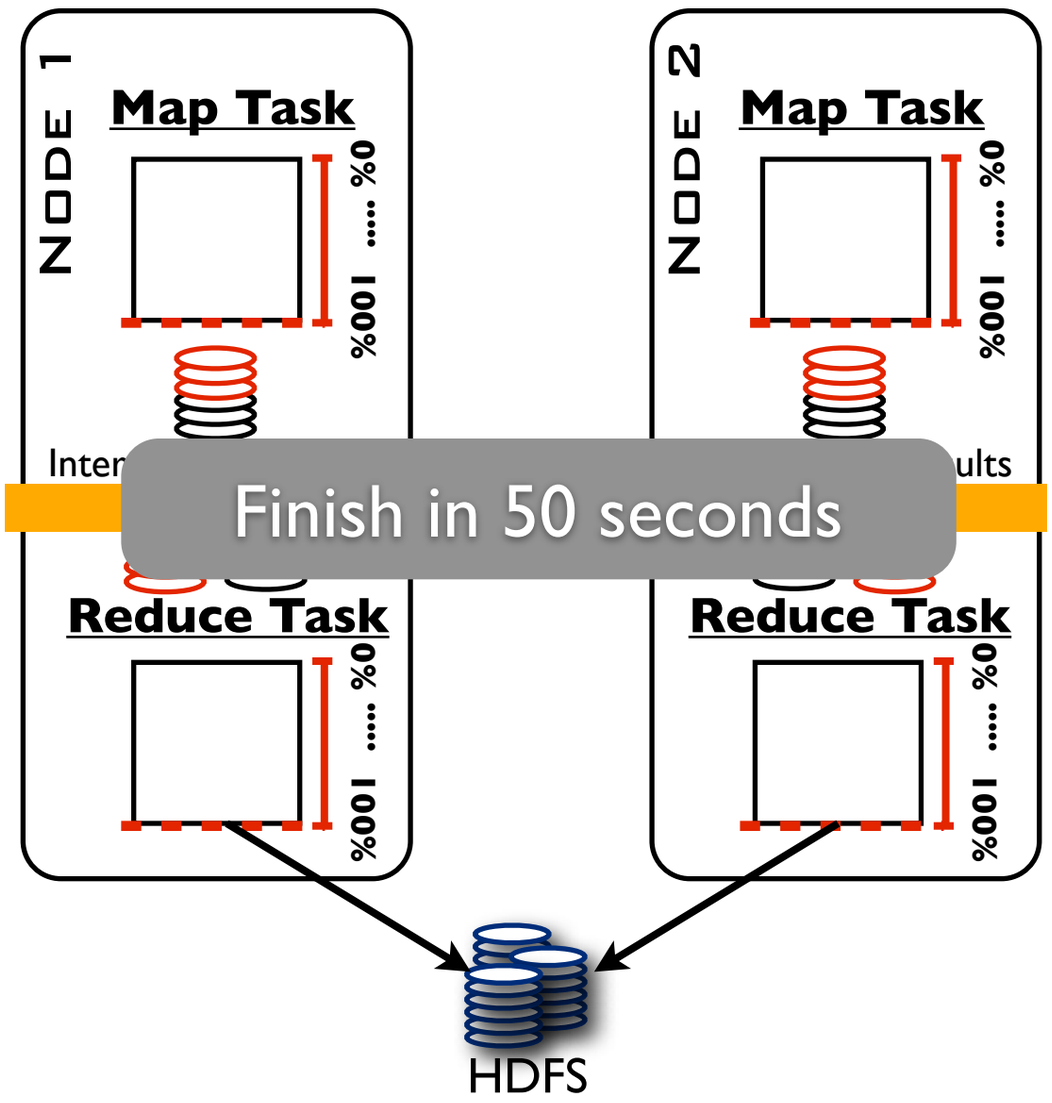
Current Approach

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop without Failures

Time:50s

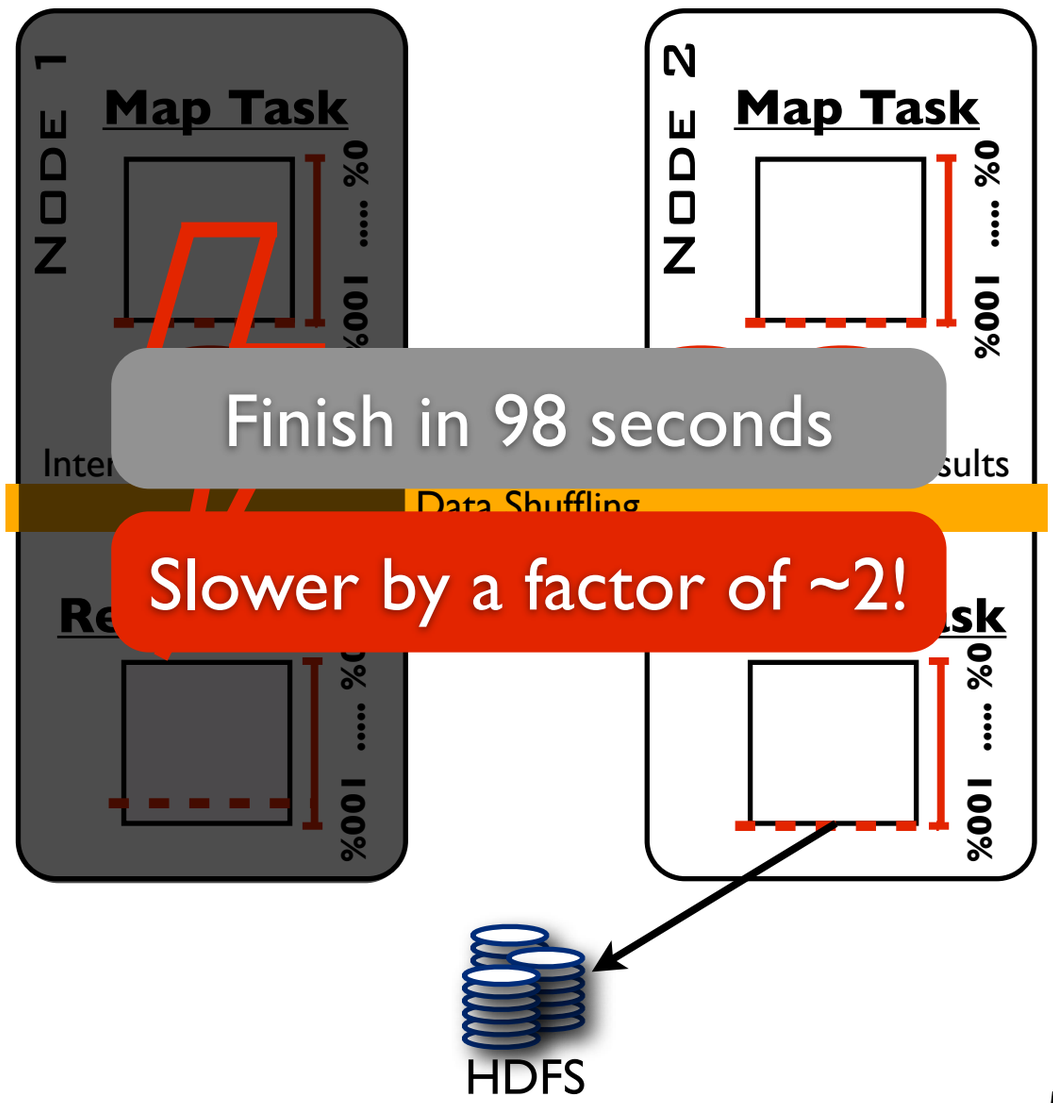
Scheduler



Hadoop with Node Failures

Time:98s

Scheduler



Solution: Query Metadata Checkpointing

Solution: Query Metadata Checkpointing

Ideas: (1) forward tracing for each key-value pair,

Solution: Query Metadata Checkpointing

Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

Solution: Query Metadata Checkpointing

Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

QMC: Query Metadata Checkpoint

Offset	Reduce Task ID

Solution: Query Metadata Checkpointing

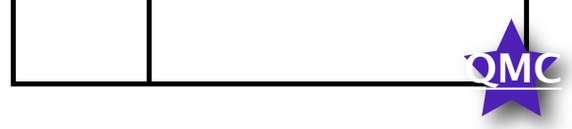
Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

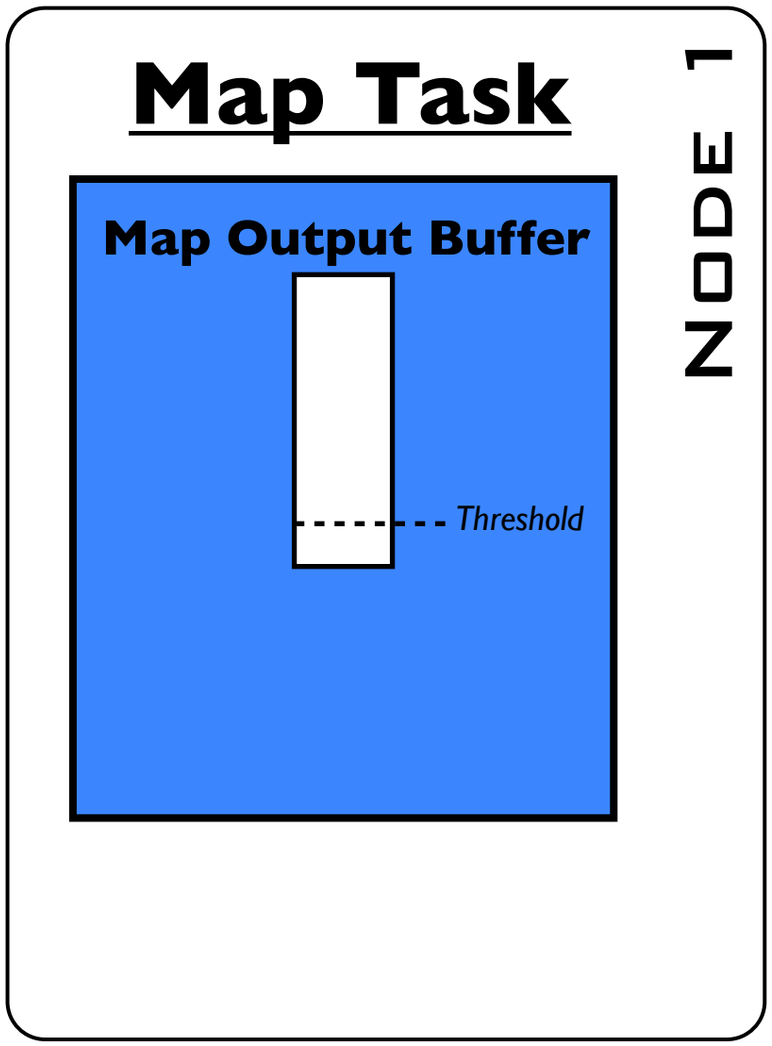
Input Split

Offset	Contents
⋮	⋮
6500	www.bbc.com, 125.65.10.77, ...
6501	skysports.com, 125.65.10.77, ...
⋮	⋮
10000	www.msn.com, 125.65.10.77, ...
10001	www.abc.co.uk, 125.65.10.77, ...
⋮	⋮

QMC: Query Metadata Checkpoint

Offset	Reduce Task ID





Solution: Query Metadata Checkpointing

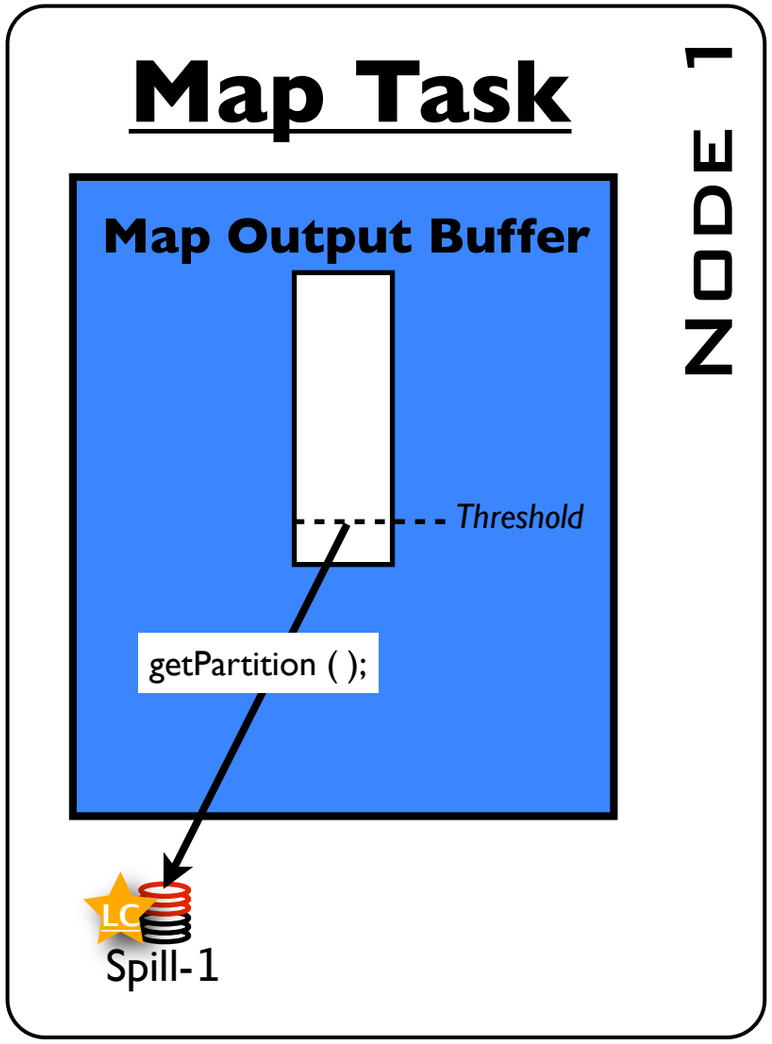
Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

Input Split

Offset	Contents
6500	www.bbc.com, 125.65.10.77, ...
6501	skysports.com, 125.65.10.77, ...
10000	www.msn.com, 125.65.10.77, ...
10001	www.abc.co.uk, 125.65.10.77, ...

QMC: Query Metadata Checkpoint

Offset	Reduce Task ID



Solution: Query Metadata Checkpointing

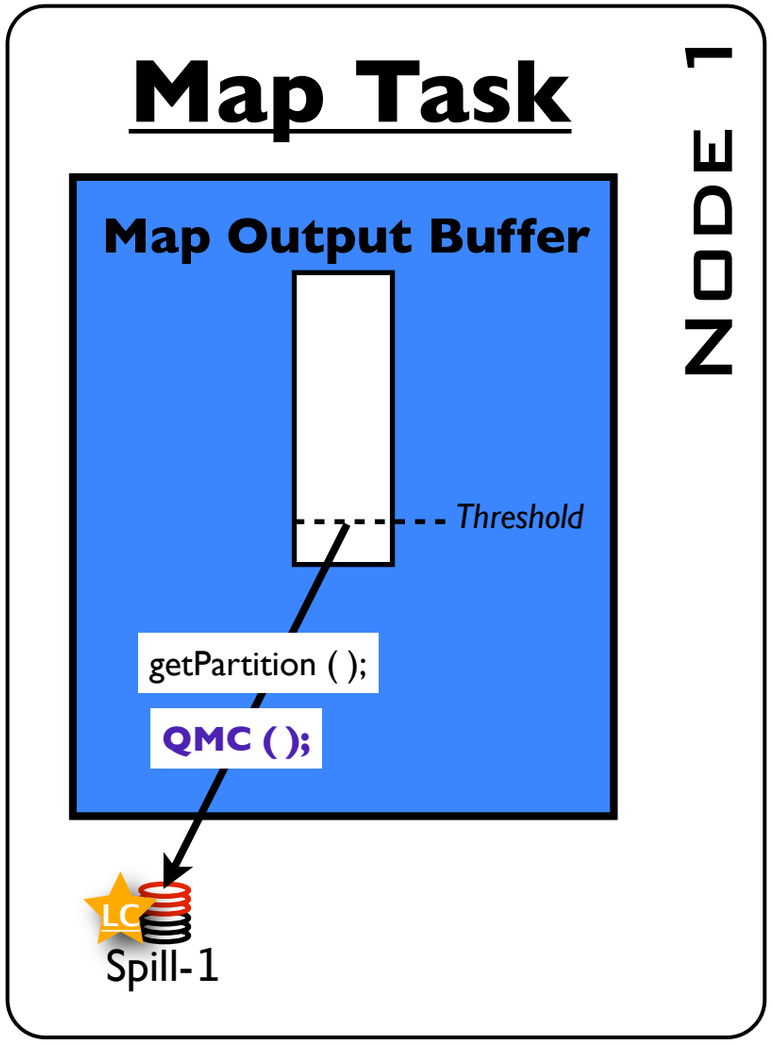
Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

Input Split

Offset	Contents
6500	www.bbc.com, 125.65.10.77, ...
6501	skysports.com, 125.65.10.77, ...
10000	www.msn.com, 125.65.10.77, ...
10001	www.abc.co.uk, 125.65.10.77, ...

QMC: Query Metadata Checkpoint

Offset	Reduce Task ID
6500	Red-1
6501	Red-2



Solution: Query Metadata Checkpointing

Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

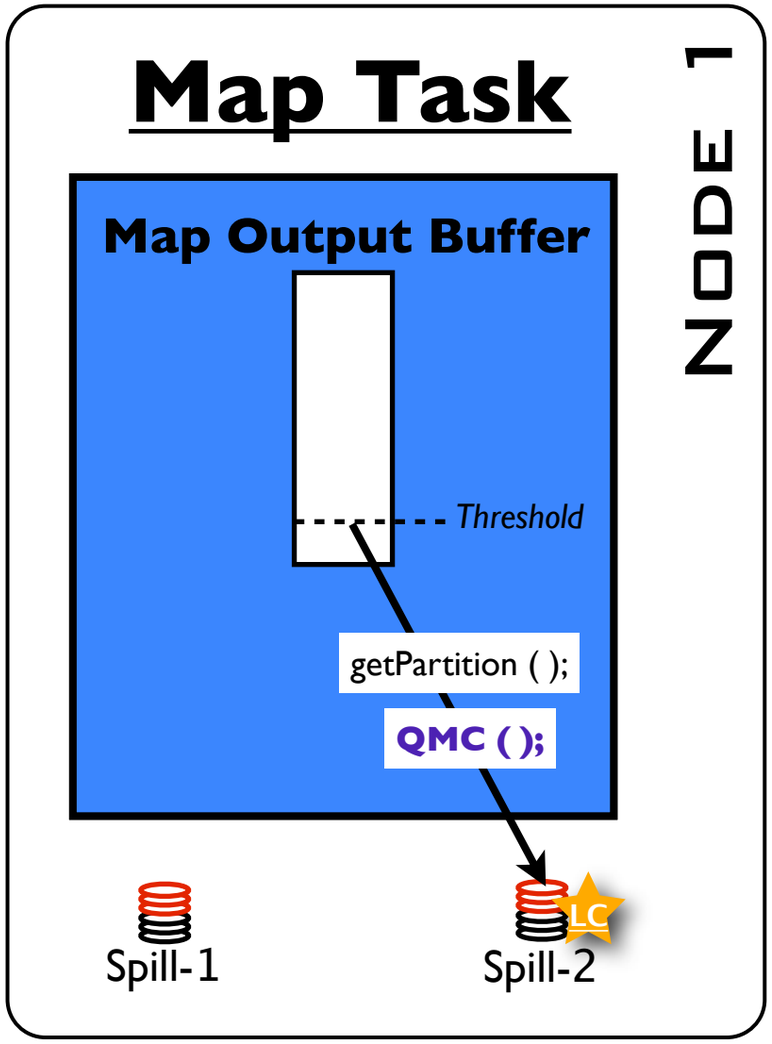
Input Split

Offset	Contents
6500	www.bbc.com, 125.65.10.77, ...
6501	skysports.com, 125.65.10.77, ...
10000	www.msn.com, 125.65.10.77, ...
10001	www.abc.co.uk, 125.65.10.77, ...

QMC: Query Metadata Checkpoint

Offset	Reduce Task ID
6500	Red-1
6501	Red-2
10000	Red-2
10001	Red-1

QMC



Solution: Query Metadata Checkpointing

Ideas: (1) forward tracing for each key-value pair,
(2) push intermediate results to all reducers

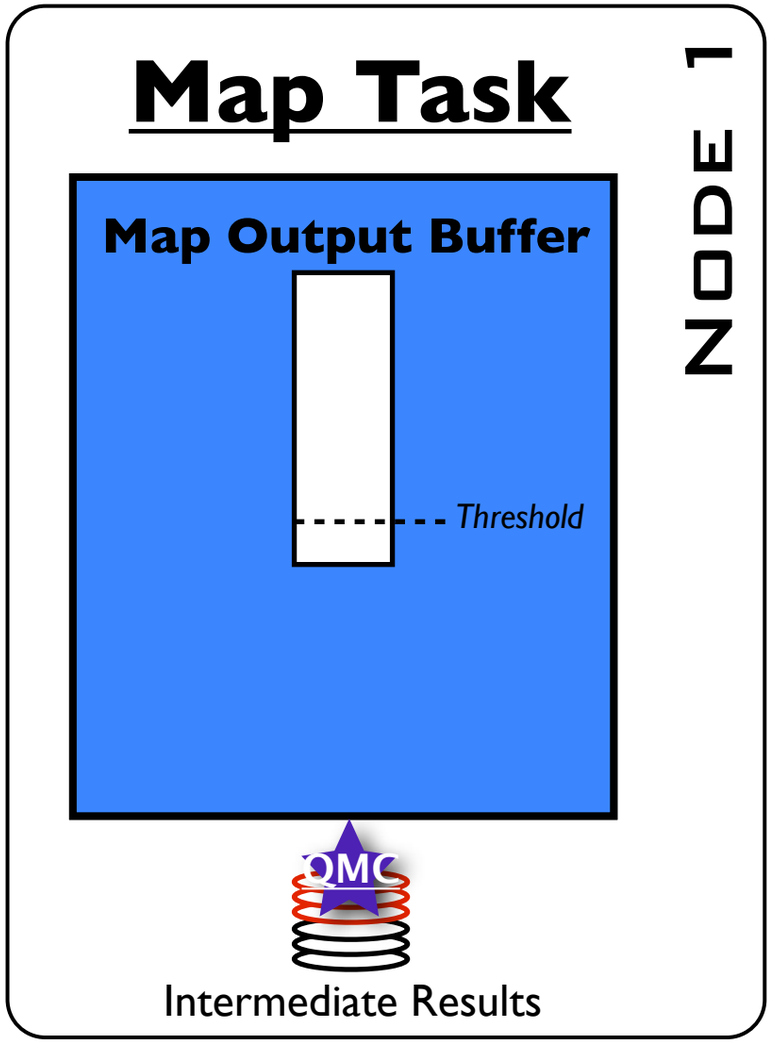
Input Split

Offset	Contents
6500	www.bbc.com, 125.65.10.77, ...
6501	skysports.com, 125.65.10.77, ...
10000	www.msn.com, 125.65.10.77, ...
10001	www.abc.co.uk, 125.65.10.77, ...

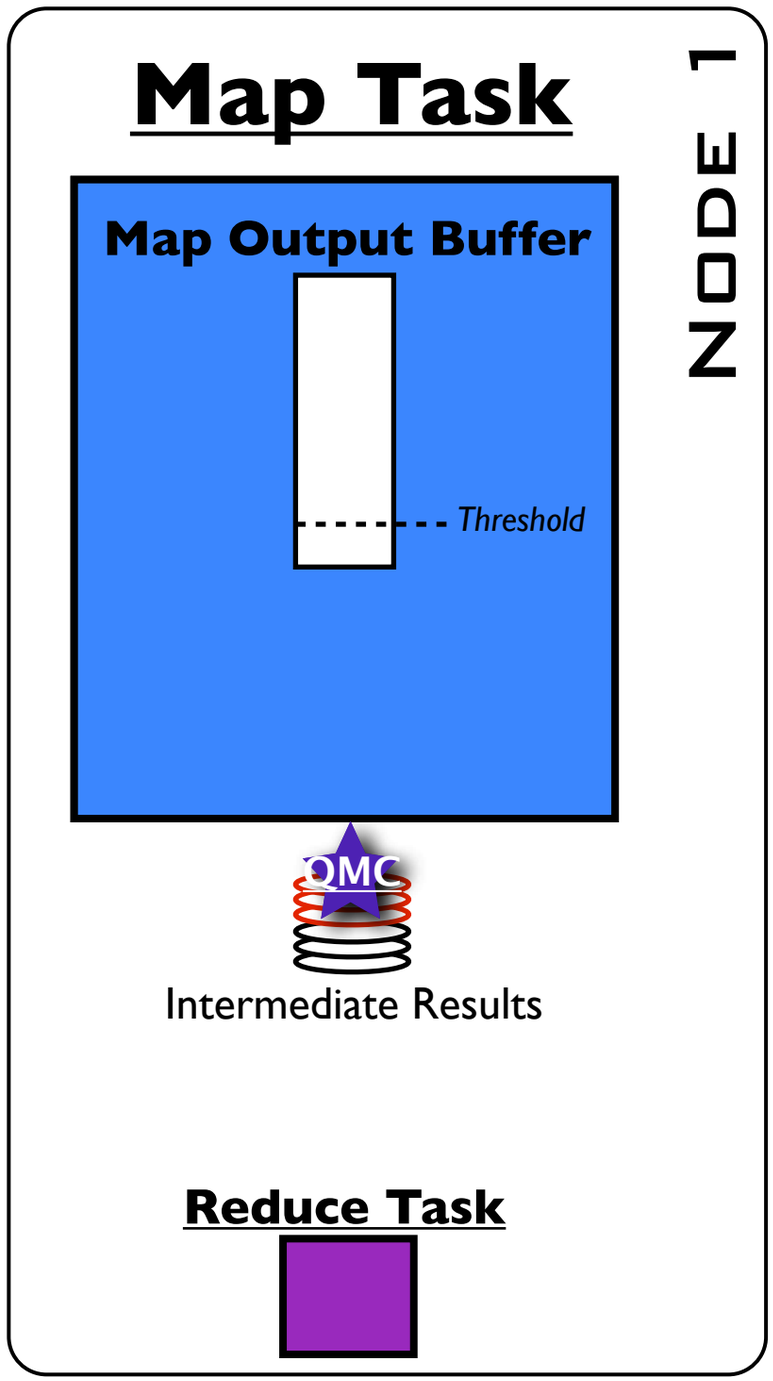
QMC: Query Metadata Checkpoint

Offset	Reduce Task ID
6500	Red-1
6501	Red-2
10000	Red-2
10001	Red-1

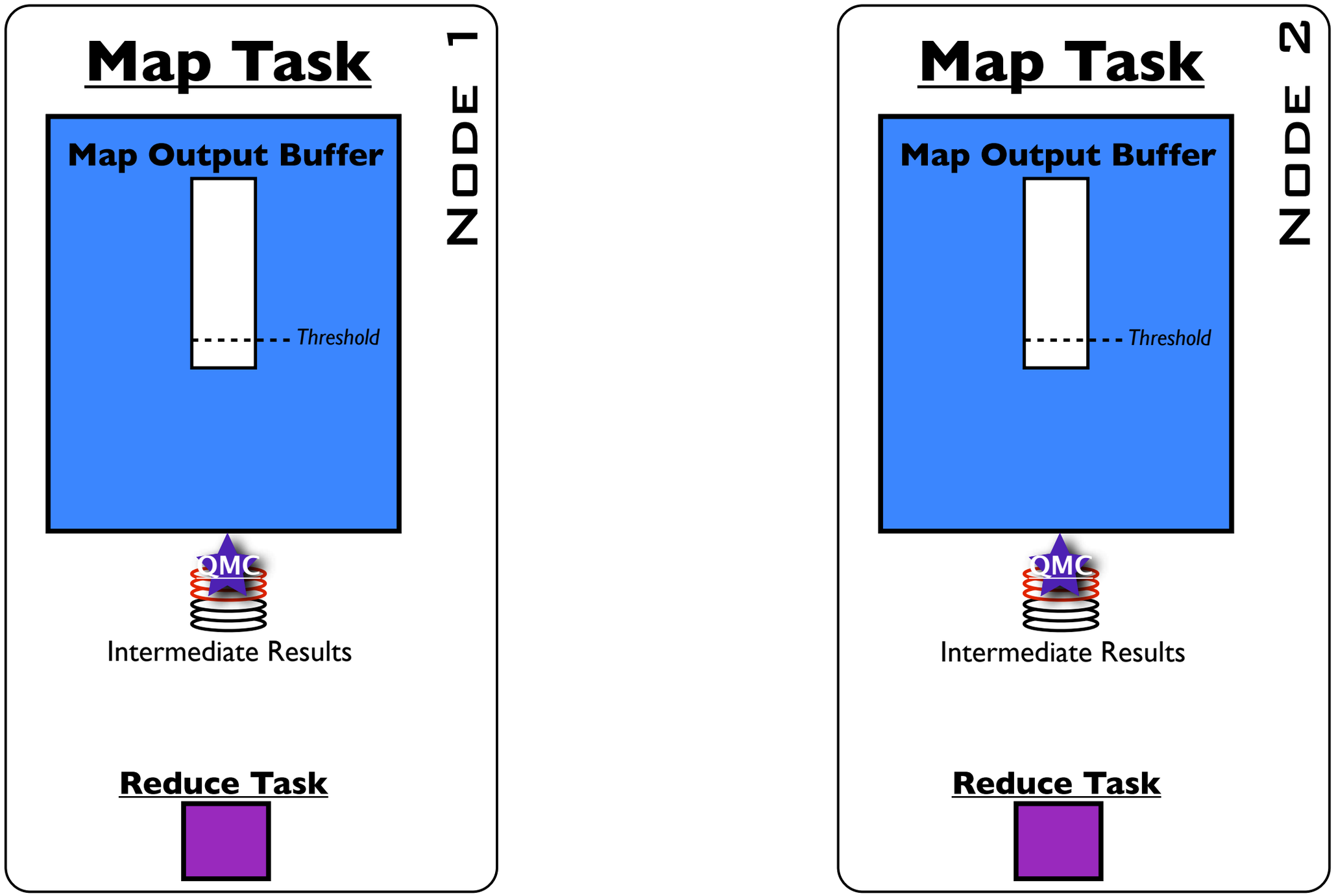
QMC



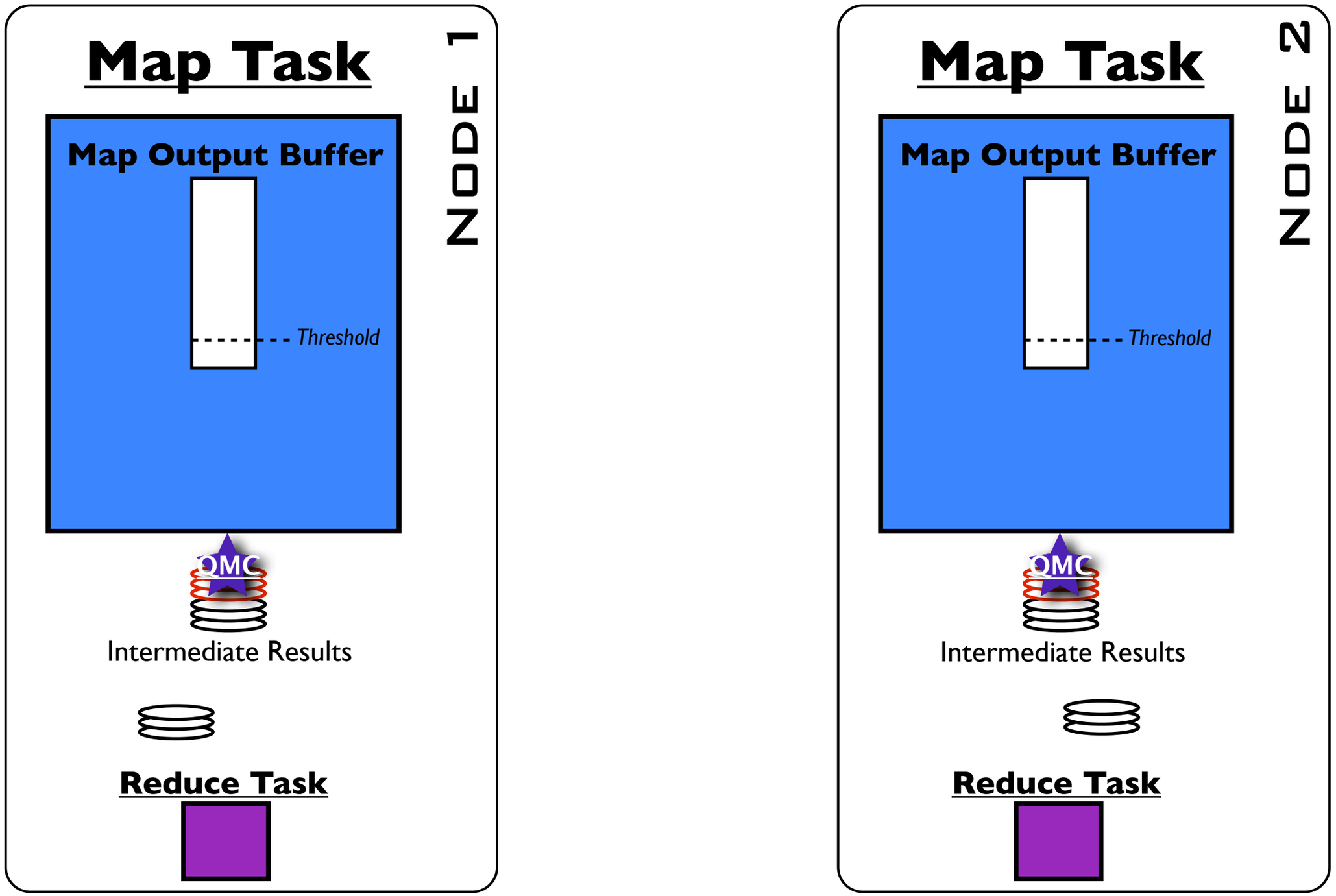
Replicating QMC vs Local Partition



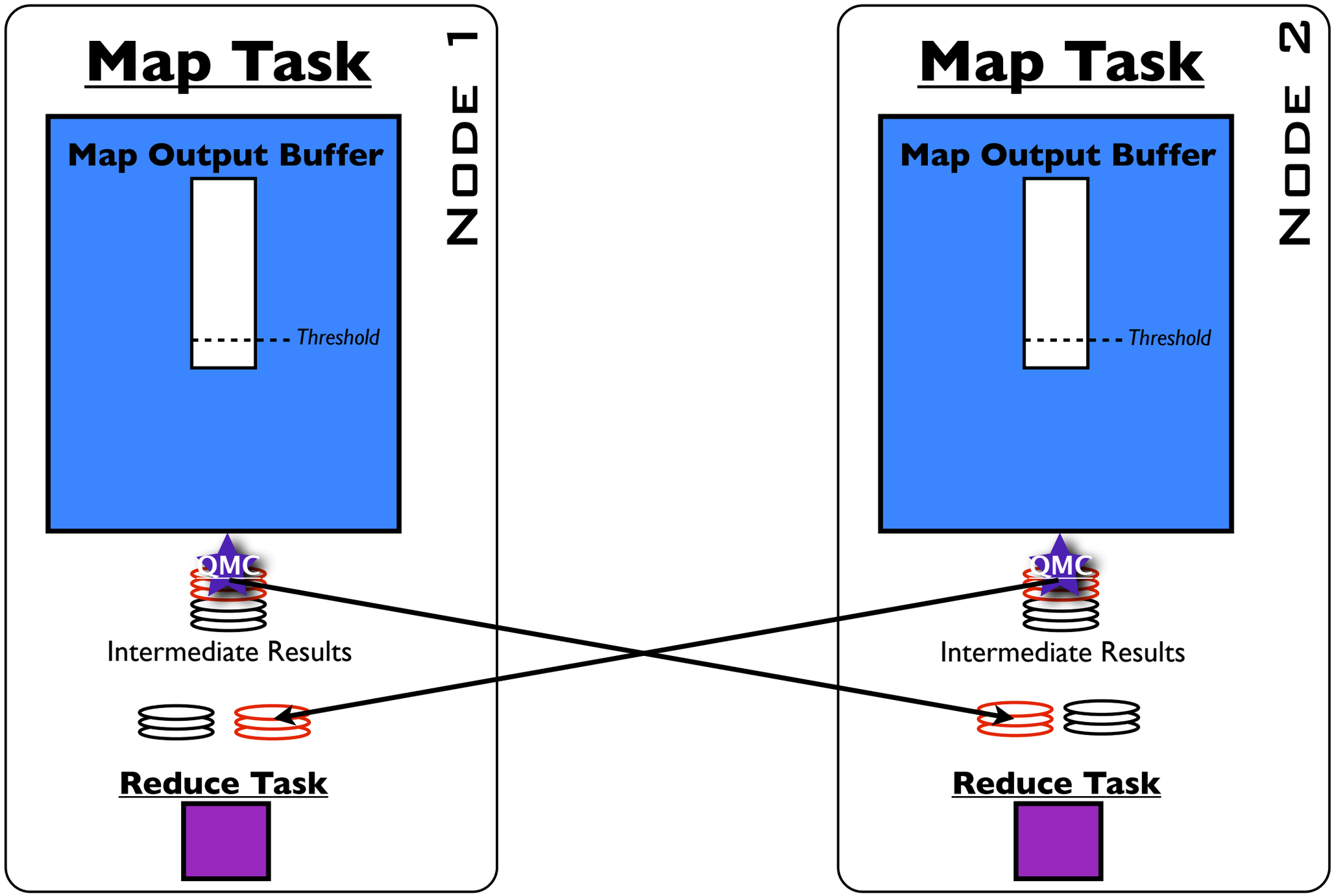
Replicating QMC vs Local Partition



Replicating QMC vs Local Partition

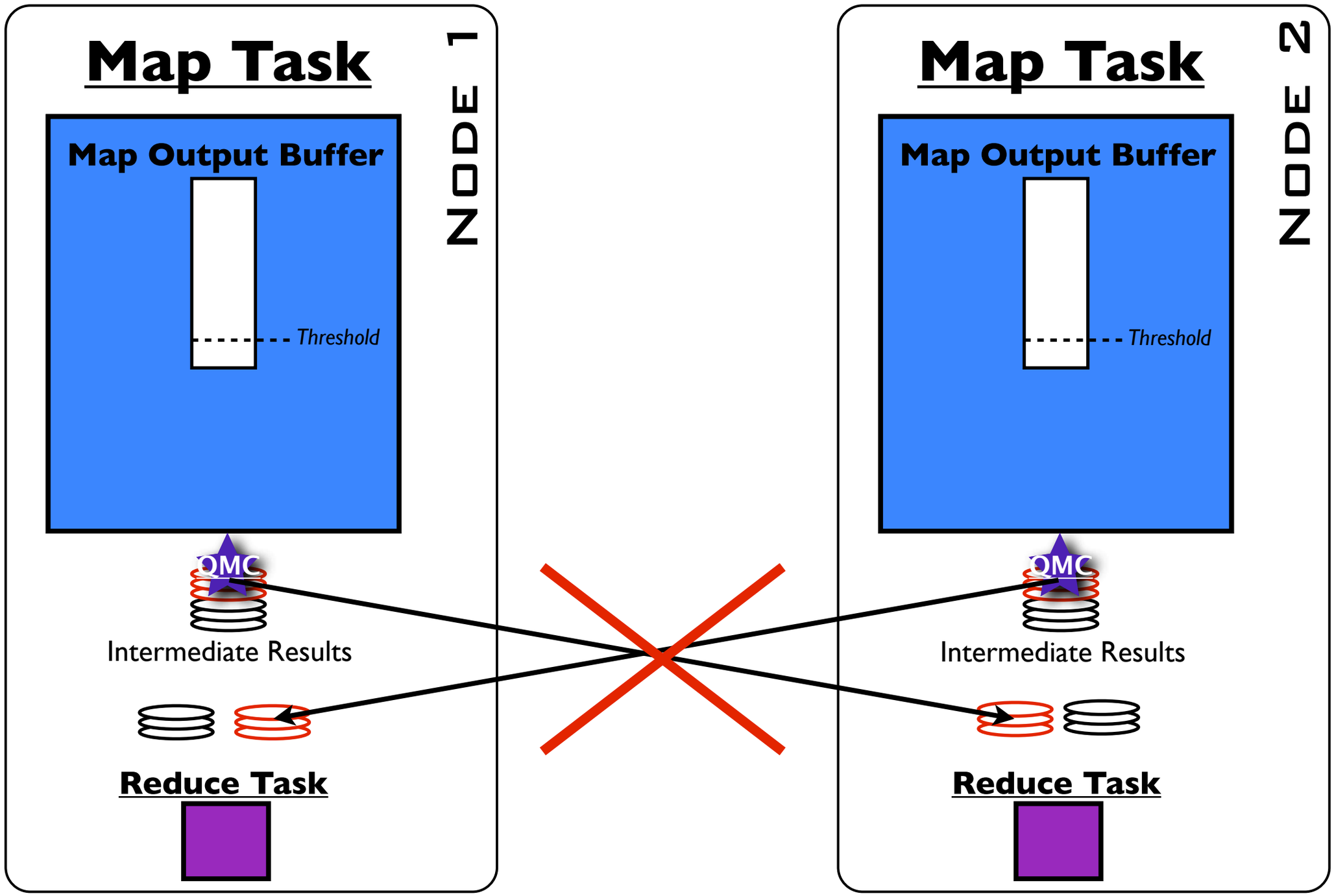


Replicating QMC vs Local Partition



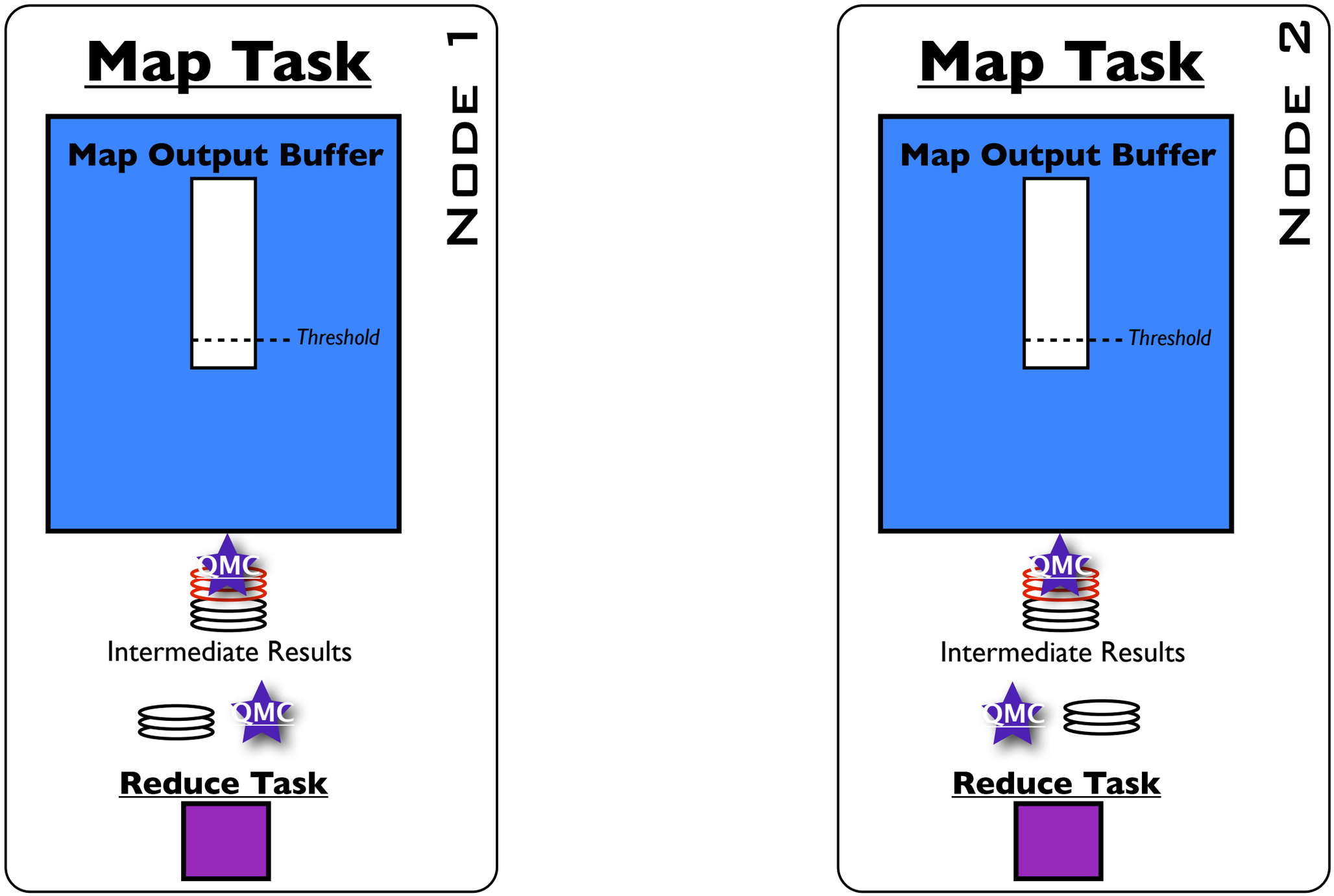
[MapReduce Online, NSDI'10]

Replicating QMC vs Local Partition



[MapReduce Online, NSDI'10]

Replicating QMC vs Local Partition



[MapReduce Online, NSDI'10]

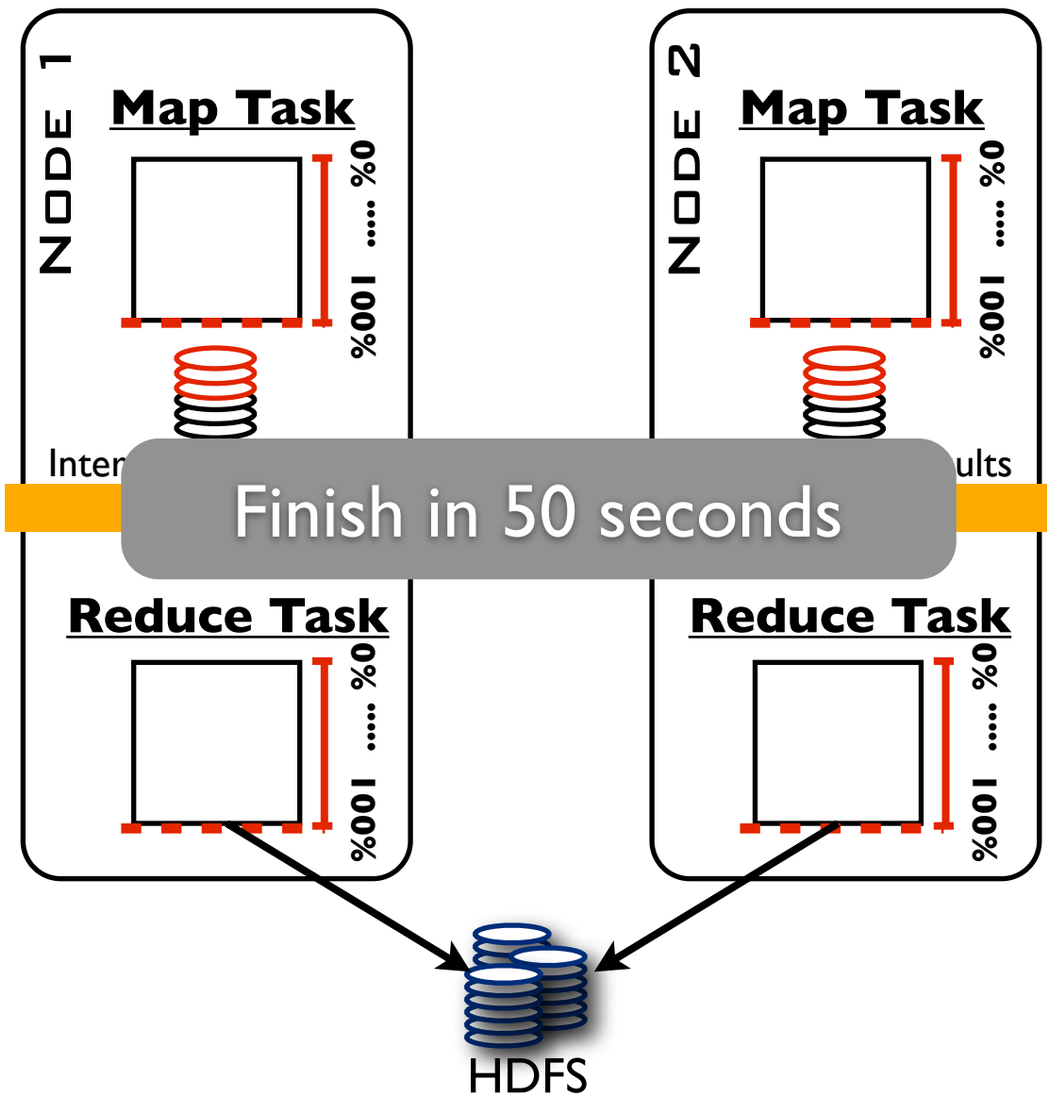
Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop without Failures

Time:50s

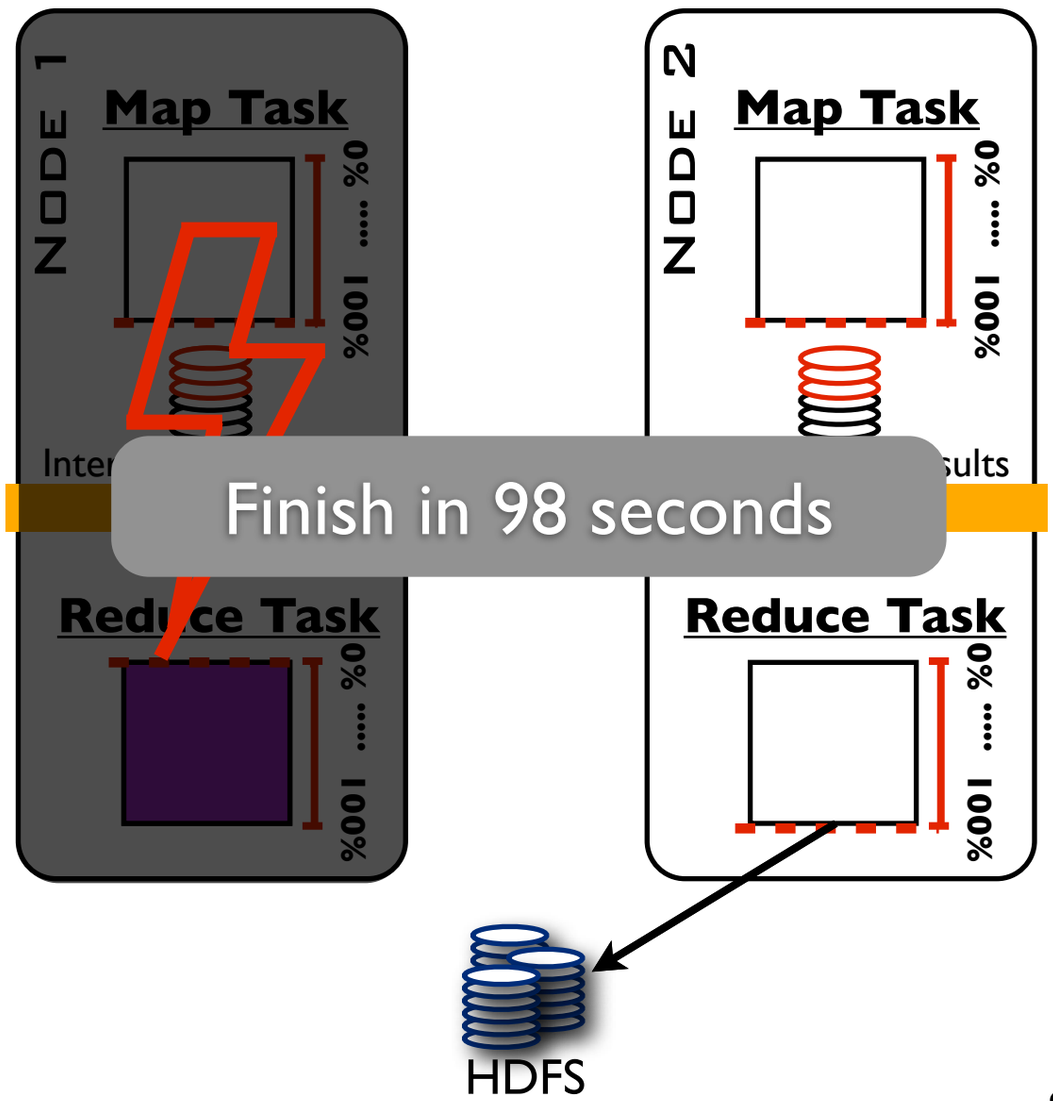
Scheduler



Hadoop with Node Failures

Time:98s

Scheduler

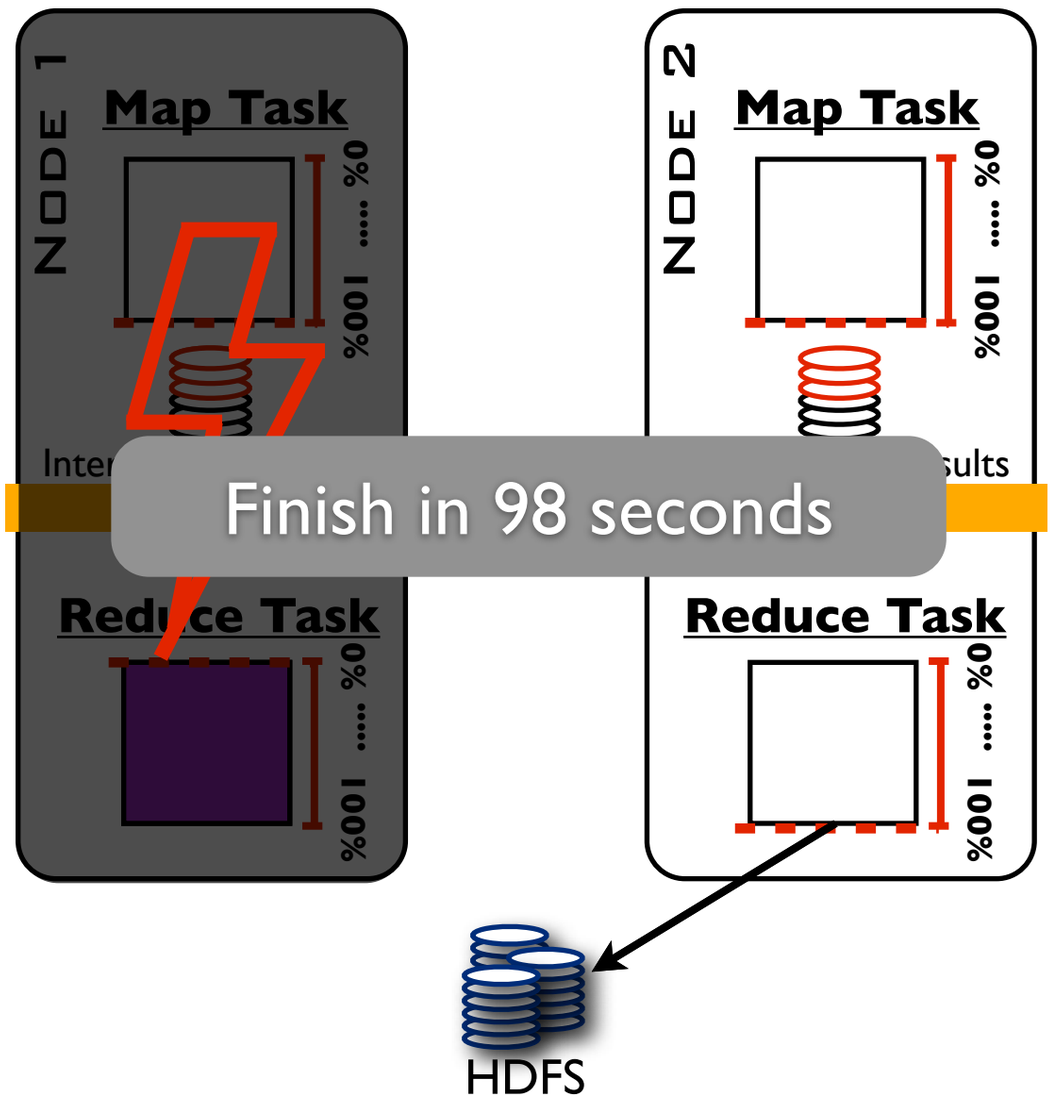


Recovery from Node Failures

Hadoop

Map Task: 20 seconds
Reduce Task: 30 seconds

Time:98s



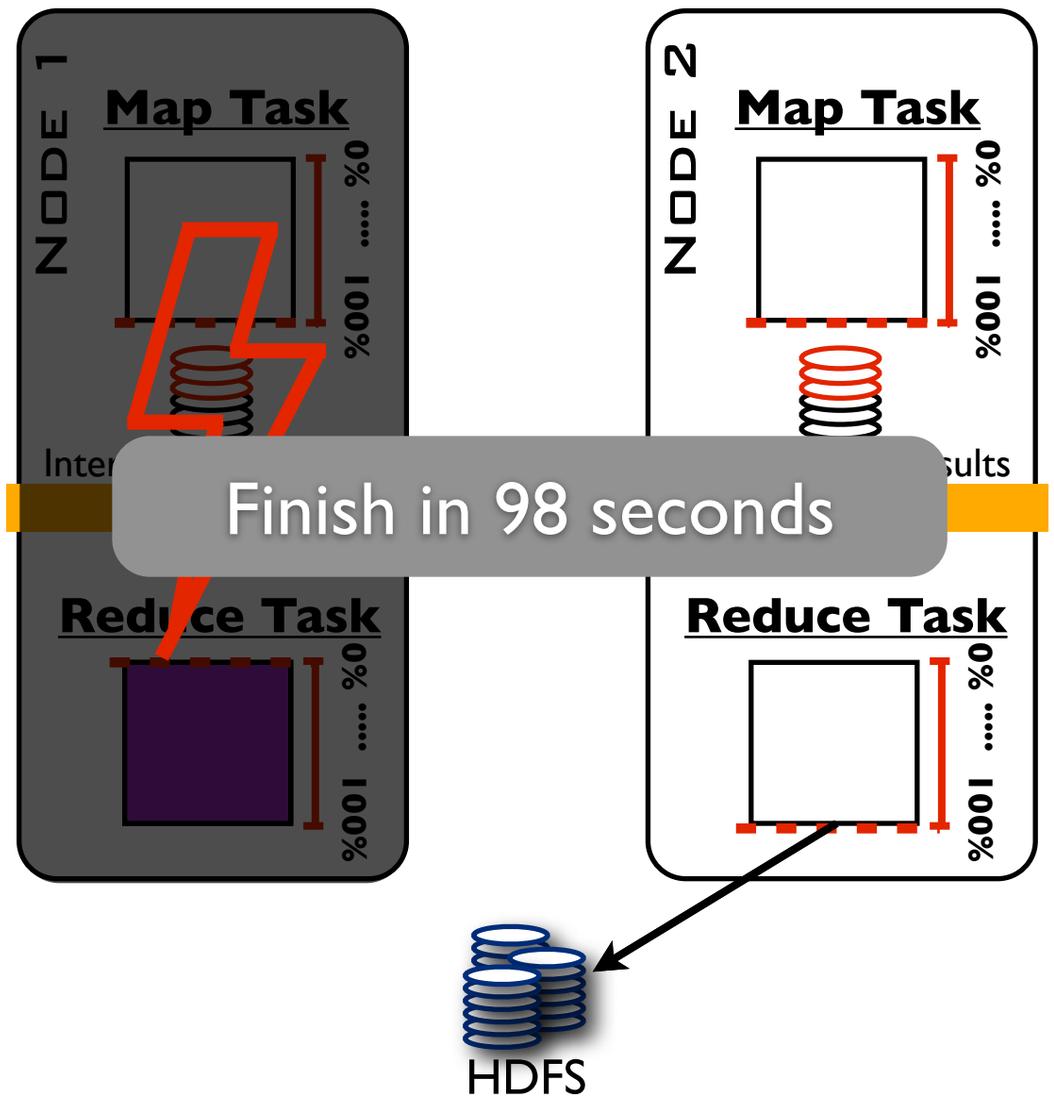
Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

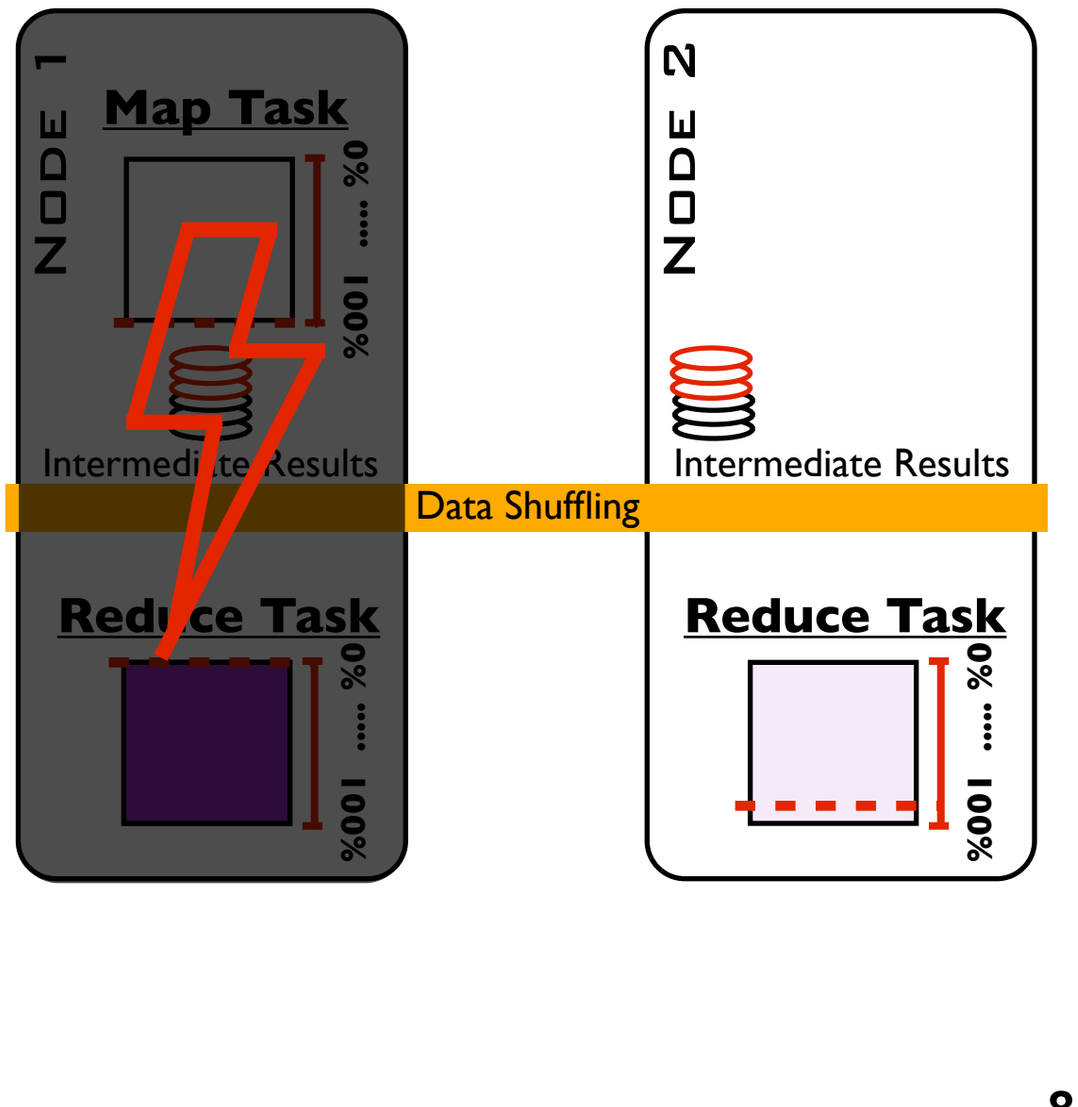
Scheduler



RAFT-QMC

Time:48s

Scheduler



Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

Scheduler

NODE 1

Map Task

100% 0%

Intermediate Results

Reduce Task

100% 0%

NODE 2

Map Task

100% 0%

Reduce Task

100% 0%

Finish in 98 seconds

HDFS

RAFT-QMC

Time:48s

Scheduler

NODE 1

Map Task

100% 0%

Intermediate Results

Reduce Task

100% 0%

NODE 2

Intermediate Results

Reduce Task

100% 0%

Data Shuffling

Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

Scheduler

NODE 1

Map Task

100% 0%

Intermediate Results

Reduce Task

100% 0%

NODE 2

Map Task

100% 0%

Reduce Task

100% 0%

Finish in 98 seconds

HDFS

RAFT-QMC

Time:48s

Scheduler

NODE 1

Map Task

100% 0%

Intermediate Results

Reduce Task

100% 0%

NODE 2

100% 0%

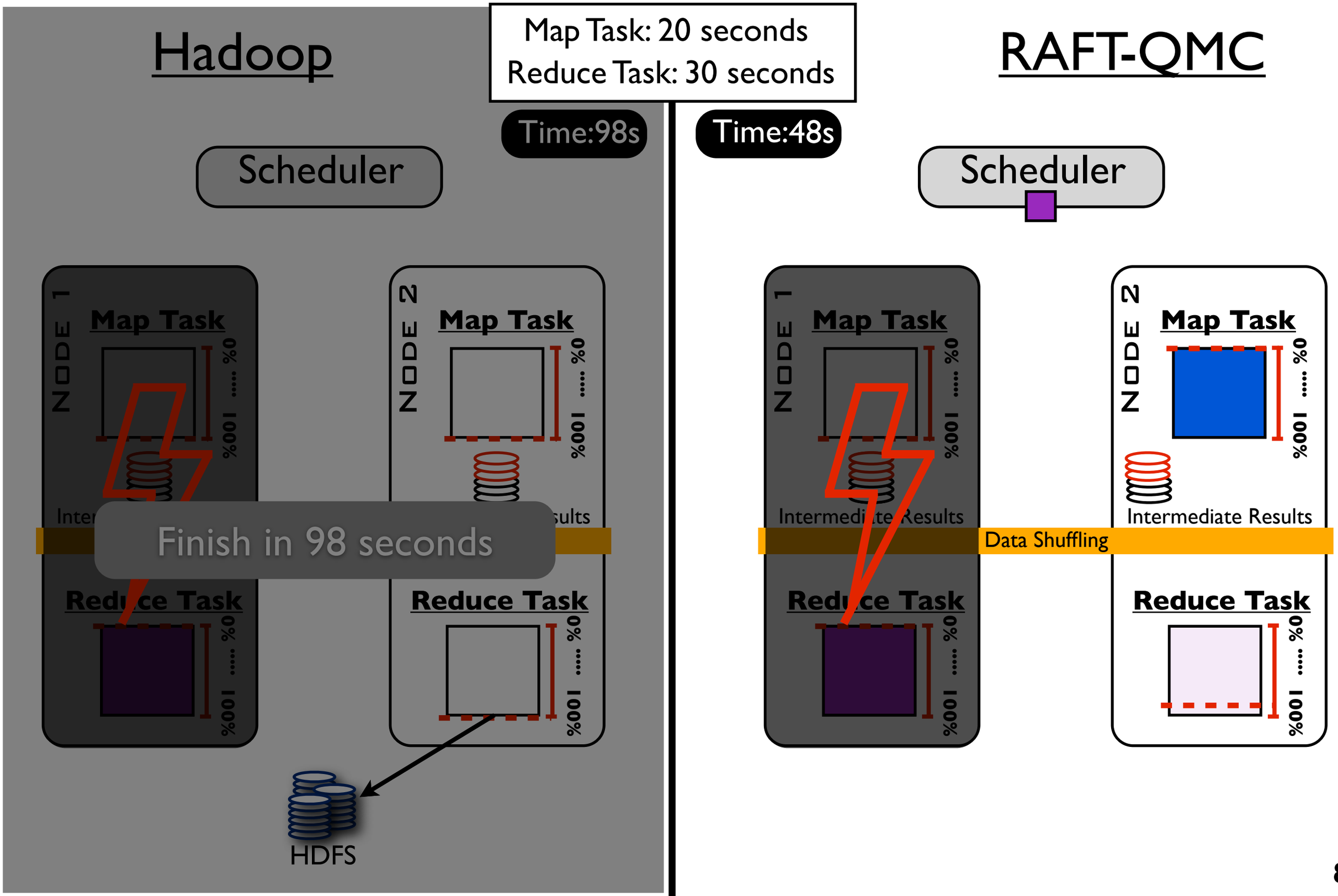
Intermediate Results

Data Shuffling

Reduce Task

100% 0%

Recovery from Node Failures



Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

Scheduler

NODE 1

Map Task

100%

Intermediate Results

Reduce Task

100%

NODE 2

Map Task

100%

Intermediate Results

Reduce Task

100%

Finish in 98 seconds

HDFS

RAFT-QMC

Time:48s

Scheduler

NODE 1

Map Task

100%

Intermediate Results

Reduce Task

100%

NODE 2

Map Task

QMC

100%

Intermediate Results

Reduce Task

100%

Data Shuffling

Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

Scheduler

NODE 1

Map Task

100%

Intermediate Results

Reduce Task

100%

NODE 2

Map Task

100%

Intermediate Results

Reduce Task

100%

Finish in 98 seconds

HDFS

RAFT-QMC

Time:50s

Scheduler

NODE 1

Map Task

100%

Intermediate Results

Reduce Task

100%

NODE 2

Map Task

QMC

100%

Intermediate Results

Reduce Task

100%

Data Shuffling

HDFS

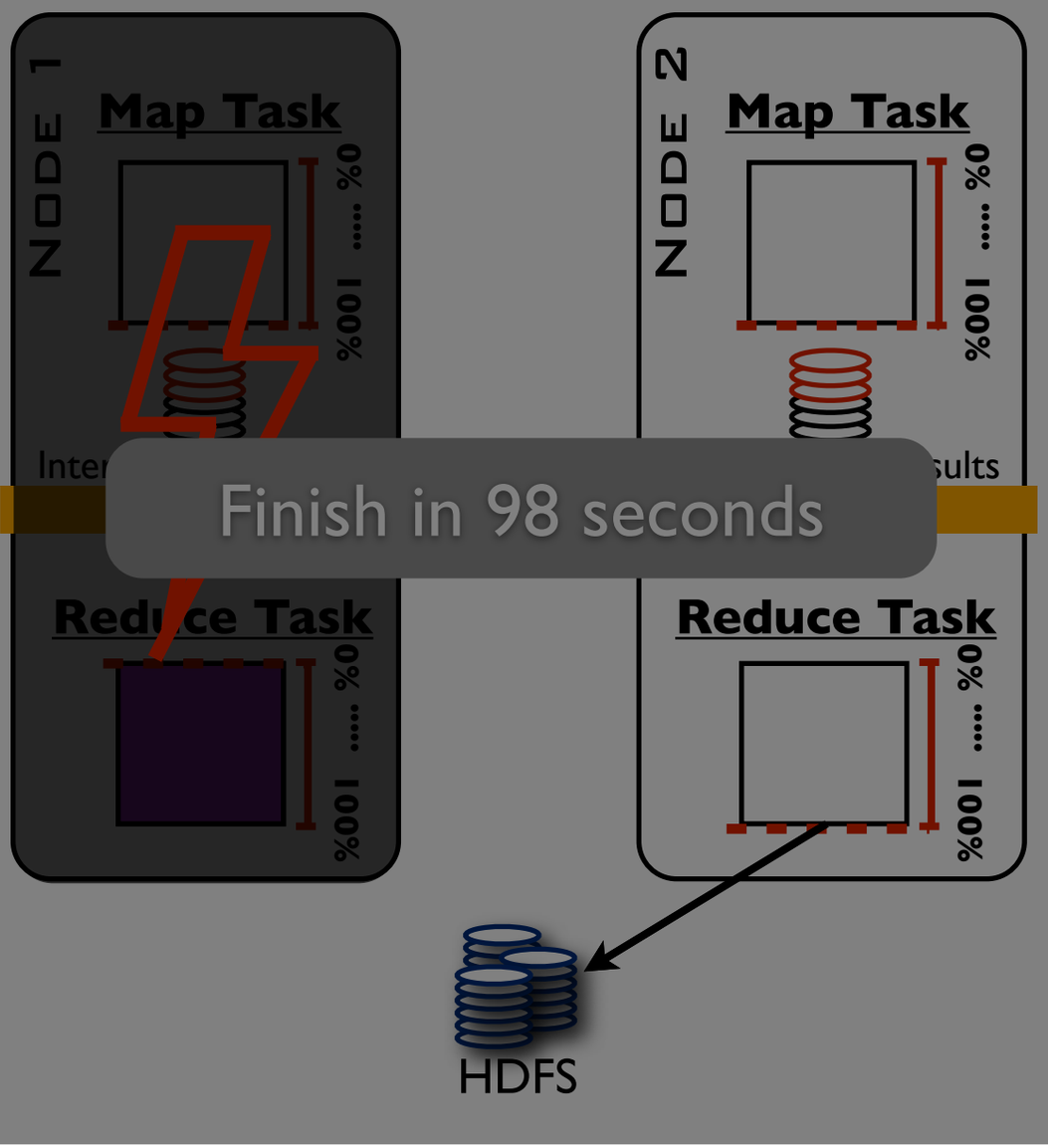
Recovery from Node Failures

Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

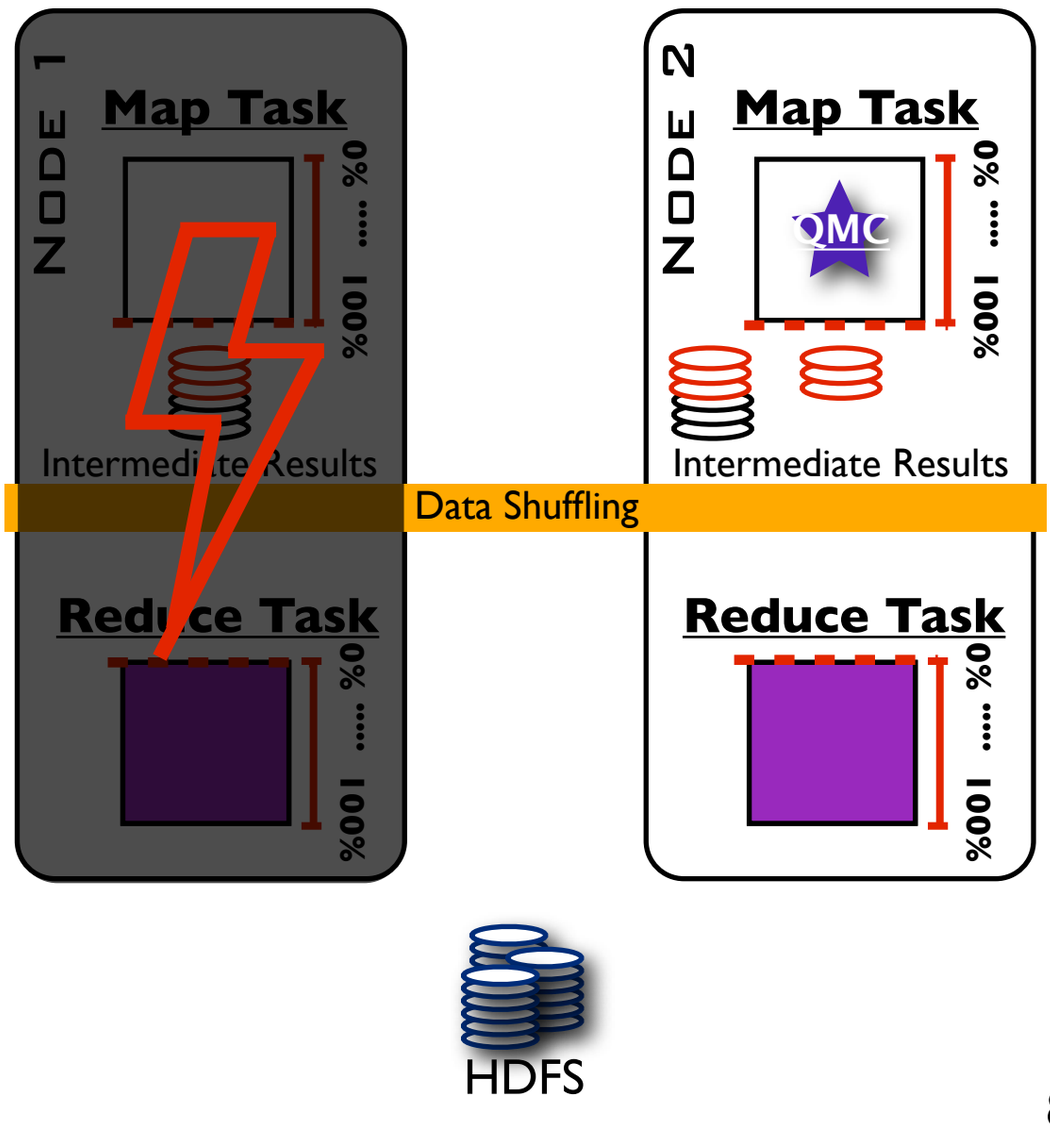
Scheduler



RAFT-QMC

Time:50s

Scheduler



Recovery from Node Failures

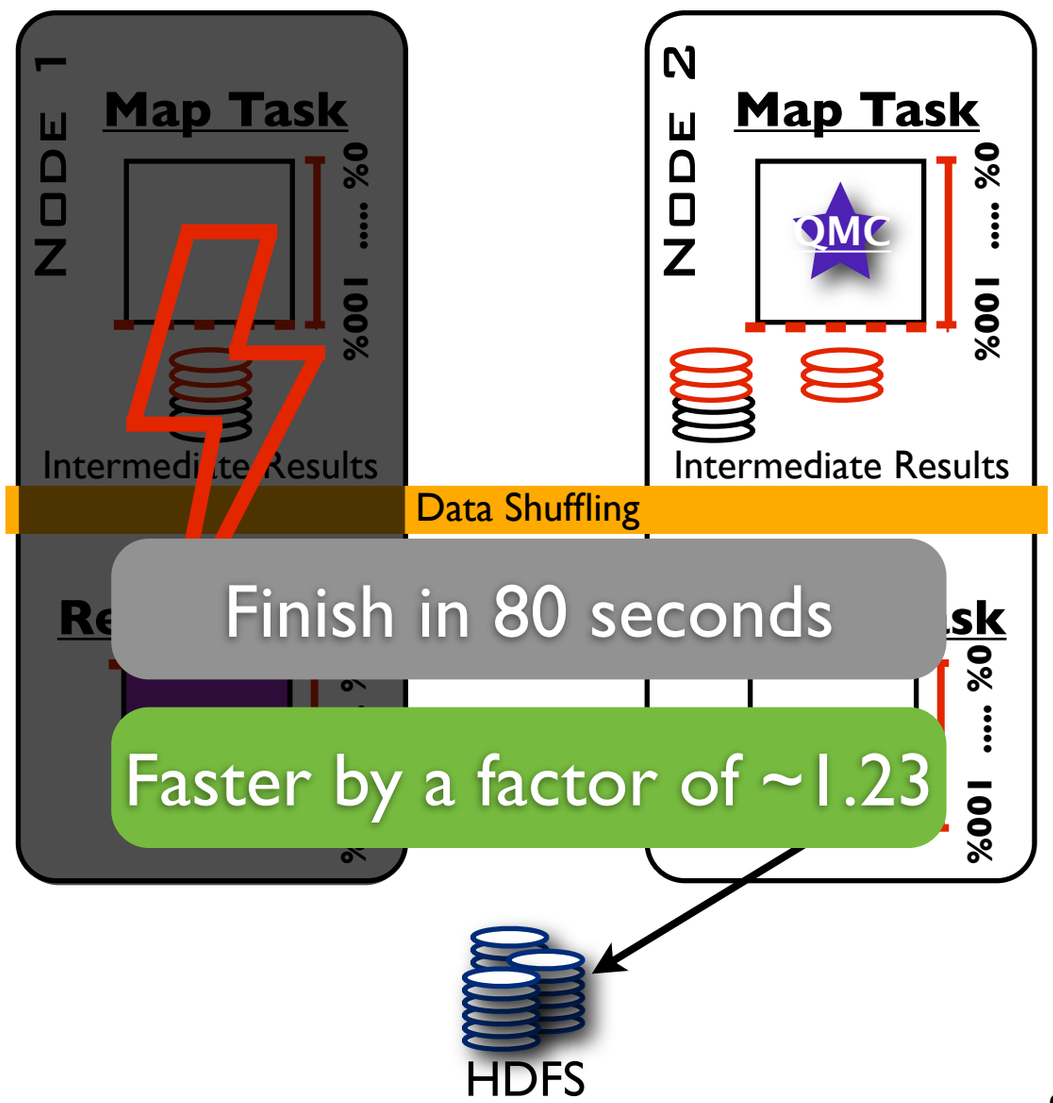
Map Task: 20 seconds
Reduce Task: 30 seconds

Hadoop

Time:98s

RAFT-QMC

Time:80s



Experiments Setup

Experiments Setup

- Tested methods:
  - RAFT (LC + QMC)
  - RAFT-QMC
  - RAFT-RC
  - RAFT-LC

In this talk

Experiments Setup

- Tested methods:
  - RAFT (LC + QMC)
  - RAFT-QMC
  - RAFT-RC

  In this talk
  - RAFT-LC
- Benchmark:
  - **Q1**: Selection Task
  - **Q2**: Aggregation Task
  - **Q3**: Selective Aggregation Task

Experiments Setup

- Tested methods:
  - RAFT (LC + QMC)
  - RAFT-QMC
  - RAFT-RC

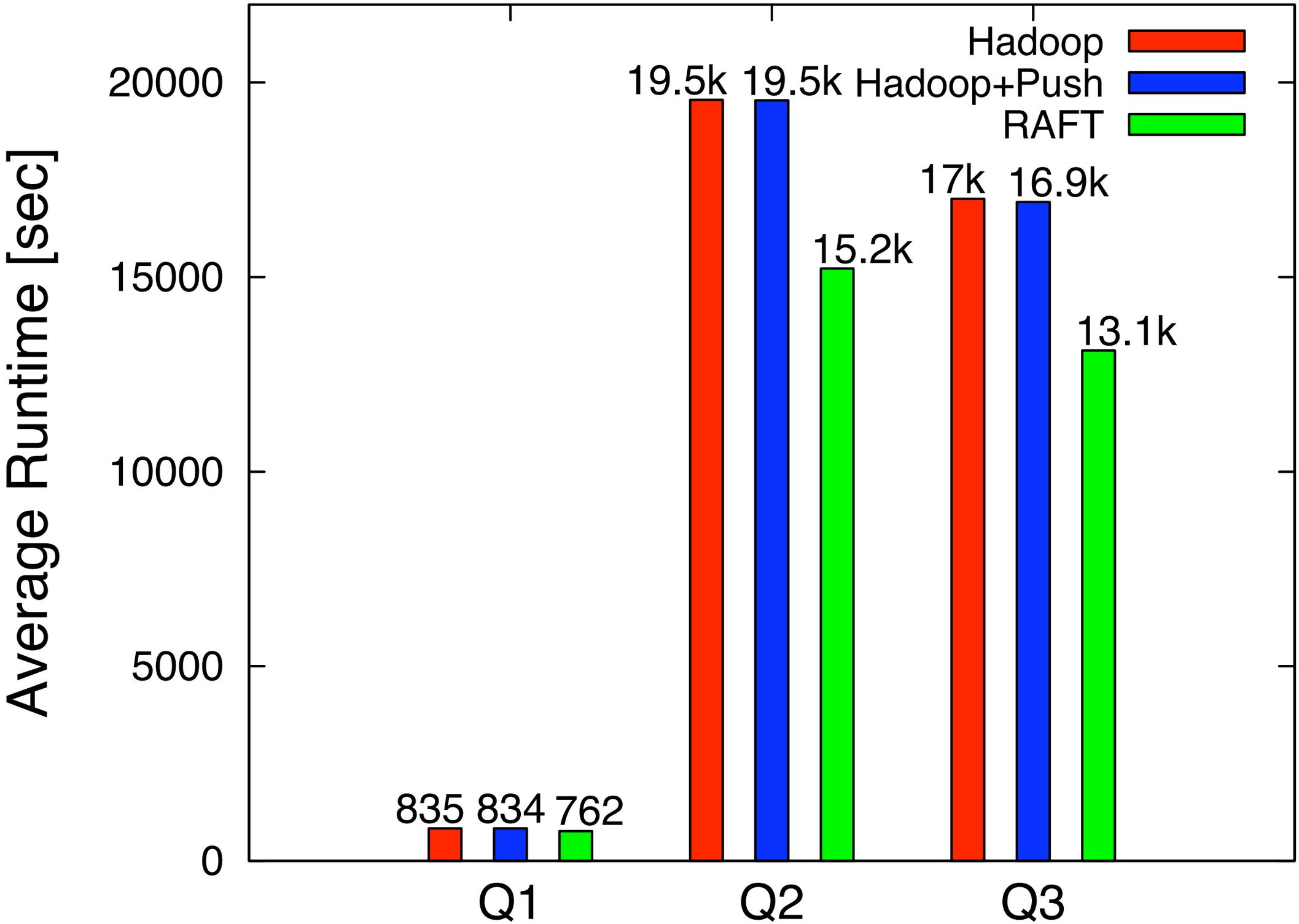
  In this talk
  - RAFT-LC
- Benchmark:
  - **Q1**: Selection Task
  - **Q2**: Aggregation Task
  - **Q3**: Selective Aggregation Task
- **1** TB data set (web logs data), [Pavlo et al., SIGMOD 2009]

Experiments Setup

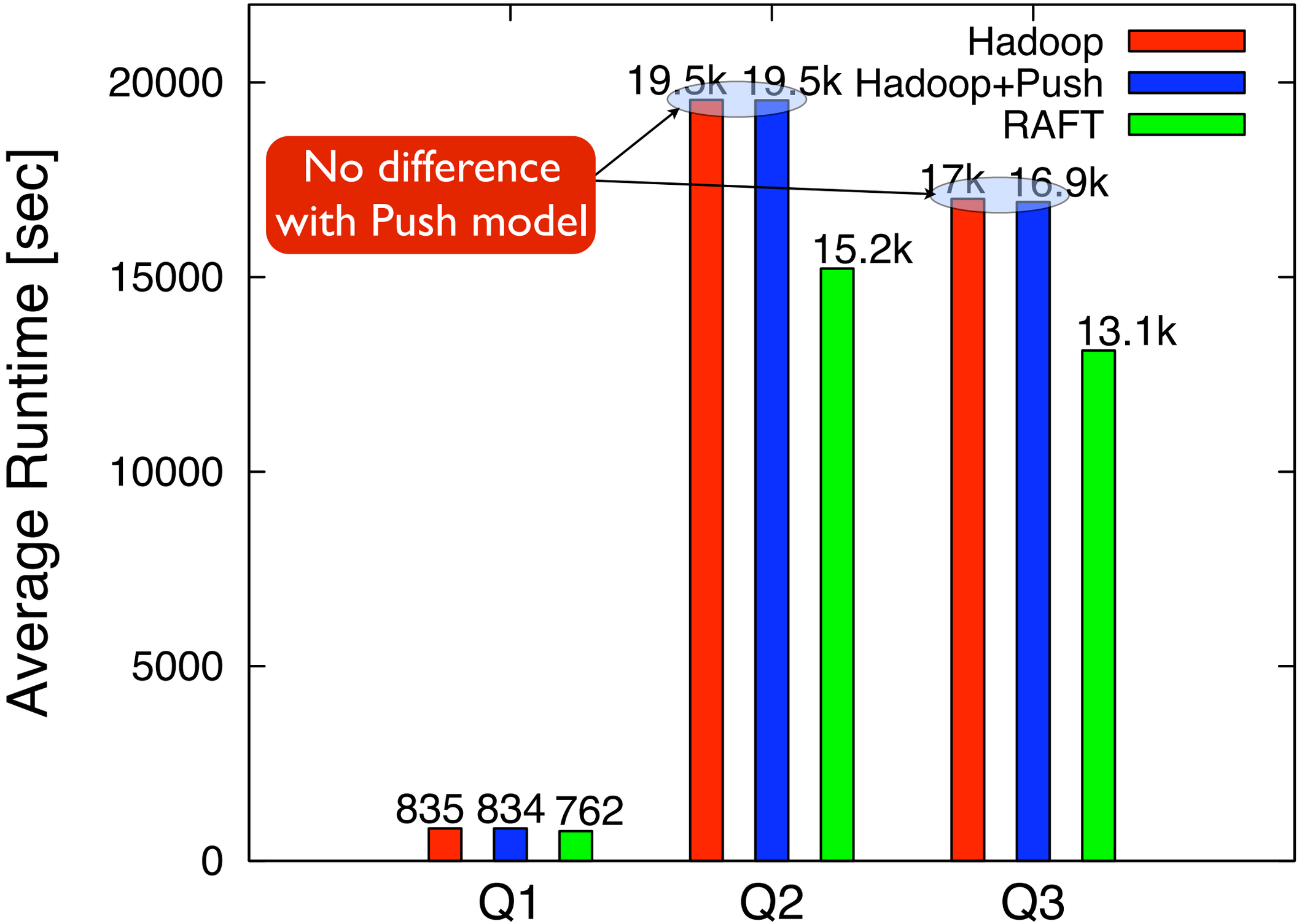
- Tested methods:
  - RAFT (LC + QMC)
  - RAFT-QMC
  - RAFT-RC

  In this talk
  - RAFT-LC
- Benchmark:
  - **Q1**: Selection Task
  - **Q2**: Aggregation Task
  - **Q3**: Selective Aggregation Task
- **1** TB data set (web logs data), [Pavlo et al., SIGMOD 2009]
- **45**-virtual node cluster

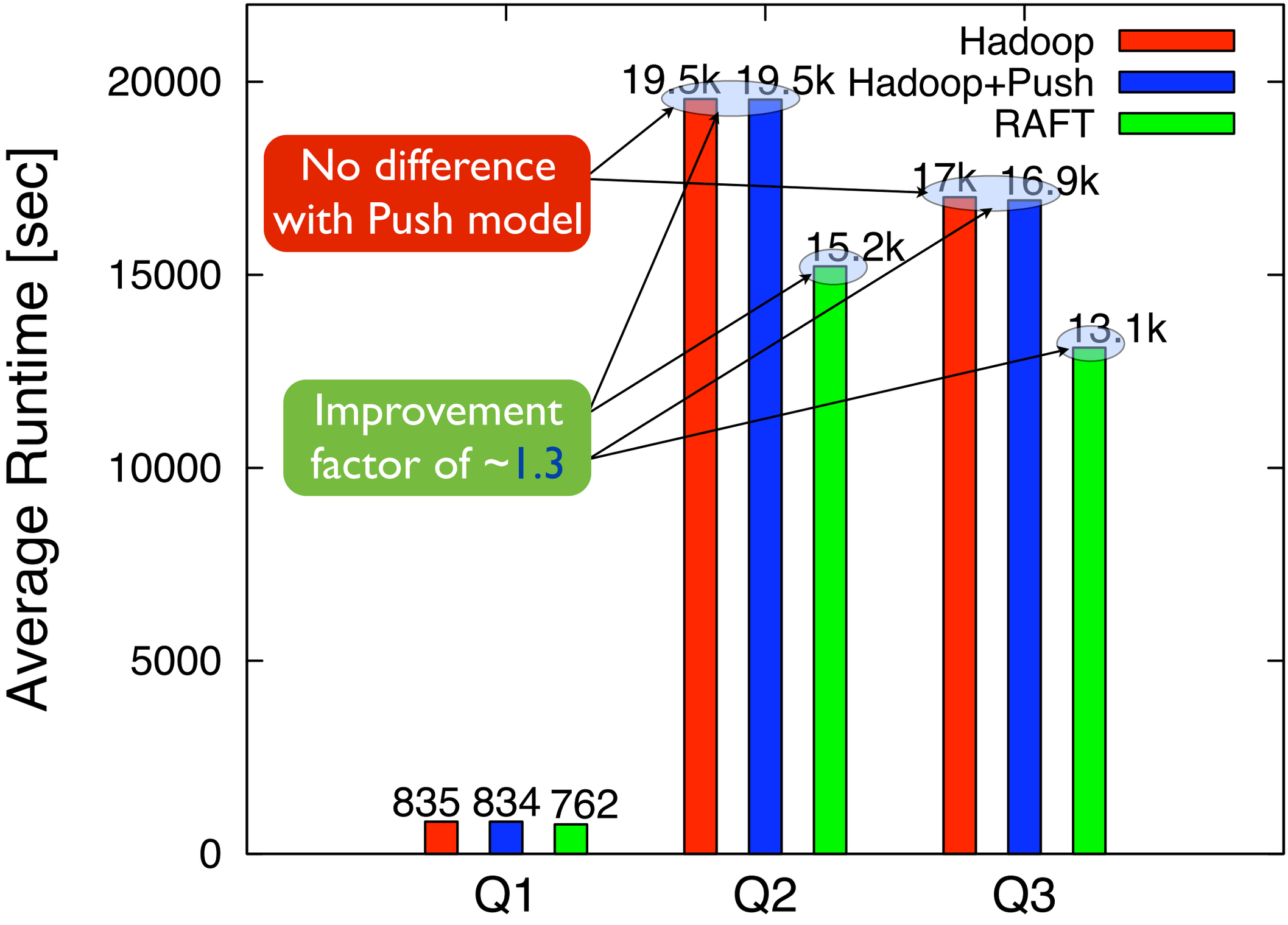
RAFT Performance



RAFT Performance



RAFT Performance



RAFT Overhead

	RAFT
Q1	5.7%
Q2	6.6%
Q3	8.7%

RAFT Overhead

	RAFT
Q1	5.7%
Q2	6.6%
Q3	8.7%

RAFT Overhead

	RAFT	RAFT-QMC	RAFT-RC
Q_1	5.7%	3.3%	13.7%
Q_2	6.6%	4.3%	9.5%
Q_3	8.7%	4.7%	9.8%

RAFT Overhead

	RAFT	RAFT-QMC	RAFT-RC
Q_1	5.7%	3.3%	13.7%
Q_2	6.6%	4.3%	9.5%
Q_3	8.7%	4.7%	9.8%

RAFT Overhead

	RAFT	RAFT-QMC	RAFT-RC
Q_1	5.7%	3.3%	13.7%
Q_2	6.6%	4.3%	9.5%
Q_3	8.7%	4.7%	9.8%

RAFT Overhead

	RAFT	RAFT-QMC	RAFT-RC
Q_1	5.7%	3.3%	13.7%
Q_2	6.6%	4.3%	9.5%
Q_3	8.7%	4.7%	9.8%

RAFT performance
come almost for free!

Conclusion

Issue: failures decrease performance significantly!

Conclusion

Issue: failures decrease performance significantly!

Proposal: a family of Recovery Algorithms for Fast-Tracking (RAFT) MapReduce:

* Local Checkpointing,
* Query Metadata Checkpointing,
* Remote Checkpointing

Conclusion

Issue: failures decrease performance significantly!

Proposal: a family of Recovery Algorithms for Fast-Tracking (RAFT) MapReduce:

* Local Checkpointing,
* Query Metadata Checkpointing,
* Remote Checkpointing

In this talk

Conclusion

Issue: failures decrease performance significantly!

Proposal: a family of Recovery Algorithms for Fast-Tracking (RAFT) MapReduce:

* Local Checkpointing,
* Query Metadata Checkpointing,
* Remote Checkpointing

In this talk

RAFT algorithms are **fast** and **inexpensive**

Conclusion

Issue: failures decrease performance significantly!

Proposal: a family of Recovery Algorithms for Fast-Tracking (RAFT) MapReduce:

* Local Checkpointing,
* Query Metadata Checkpointing,
* Remote Checkpointing

In this talk

RAFT algorithms are **fast** and **inexpensive**

Results: RAFT outperforms Hadoop by a factor of ~1.3 on average, still a negligible runtime overhead!

Conclusion

Issue: failures decrease performance significantly!

Proposal: a family of Recovery Algorithms for Fast-Tracking (RAFT) MapReduce:

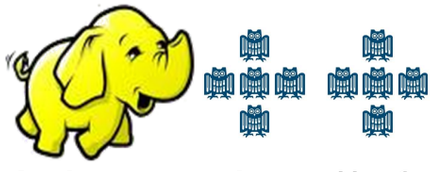
* Local Checkpointing,
* Query Metadata Checkpointing,
* Remote Checkpointing

In this talk

RAFT algorithms are **fast** and **inexpensive**

Results: RAFT outperforms Hadoop by a factor of ~1.3 on average, still a negligible runtime overhead!

PART OF A BIGGER PROJECT:



Hadoop++, http://infosys.cs.uni-saarland.de/hadoop++.php

Conclusion

Issue: failures decrease performance significantly!

Proposal: a family of Recovery Algorithms for Fast-Tracking (RAFT) MapReduce:

* Local Checkpointing,
* Query Metadata Checkpointing,
* Remote Checkpointing

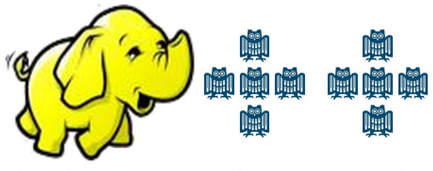
In this talk

RAFT algorithms are **fast** and **inexpensive**

Results: RAFT outperforms Hadoop by a factor of ~1.3 on average, still a negligible runtime overhead!

SEE YOU AT SIGMOD' 2011 FOR A RAFT DEMONSTRATION

PART OF A BIGGER PROJECT:



Hadoop++, http://infosys.cs.uni-saarland.de/hadoop++.php