CS 744: BIG DATA SYSTEMS

Shivaram Venkataraman

Fall 2018

MOTIVATION

Build Google Web Search !

- Crawl documents, build inverted indexes etc.

Need for

- automatic parallelization
- network, disk optimization
- handling of machine failures

OUTLINE

- Programming Model
- Execution Overview
- Fault Tolerance
- Optimizations

PROGRAMMING MODEL

Data type: Each record is (key, value)

Map function:

$$(K_{in}, V_{in}) \rightarrow \text{list}(K_{inter}, V_{inter})$$

Reduce function:

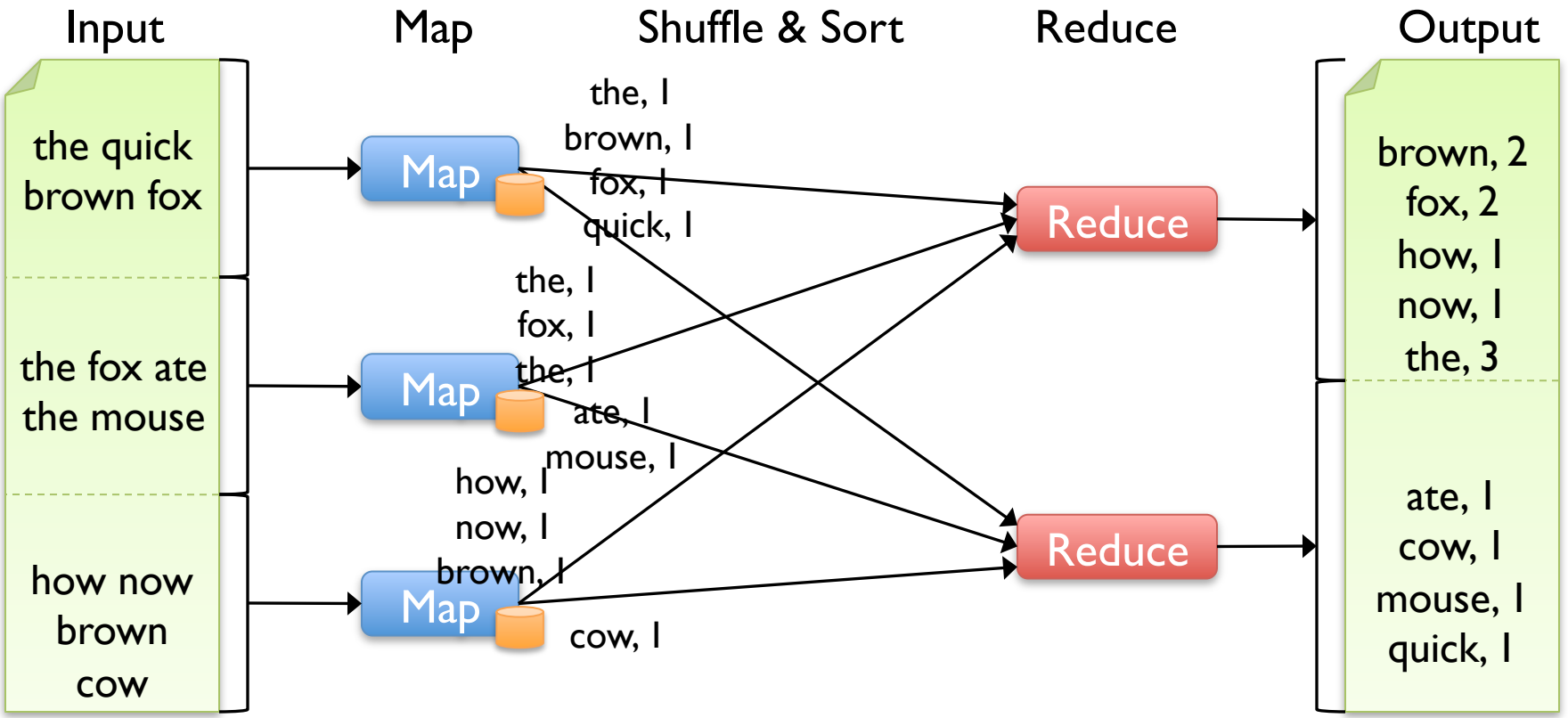
$$(K_{inter}, \text{list}(V_{inter})) \rightarrow \text{list}(K_{out}, V_{out})$$

EXAMPLE: WORD COUNT

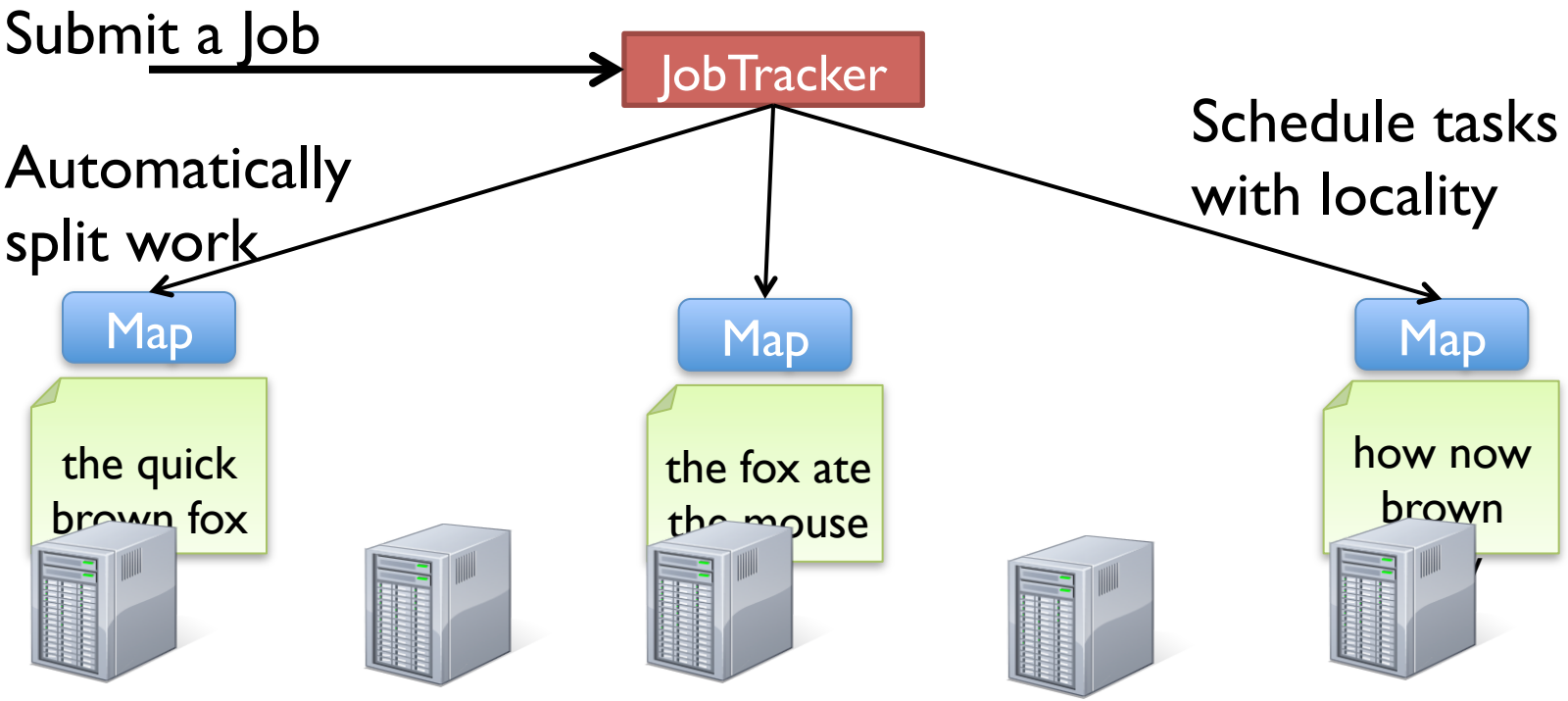
```
def mapper(line):
    for word in line.split():
        output(word, 1)


def reducer(key, values):
    output(key, sum(values))
```

WORD COUNT EXECUTION



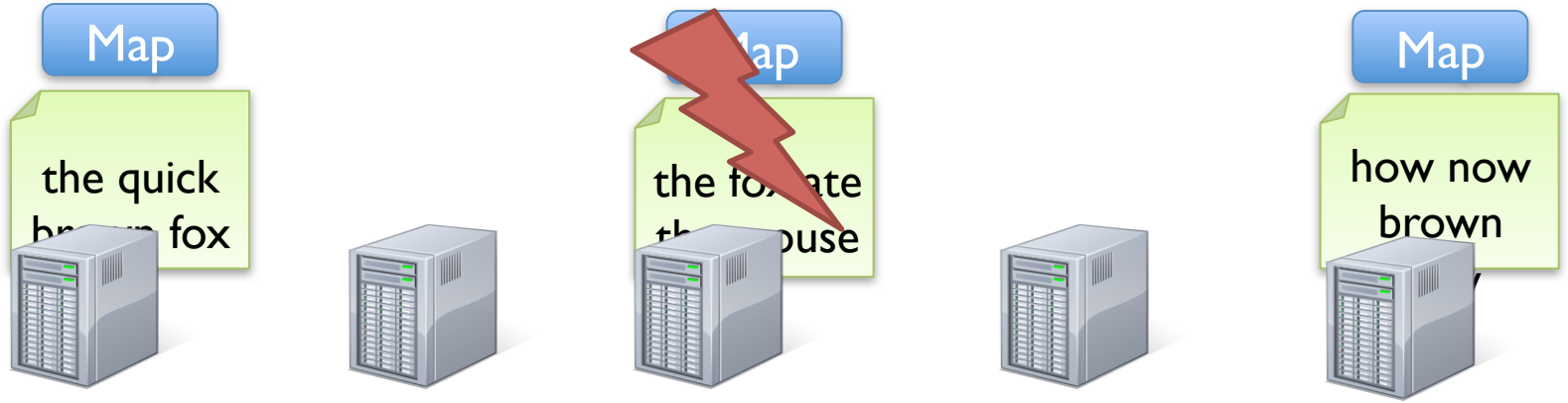
WORD COUNT EXECUTION



FAULT RECOVERY

If a task crashes:

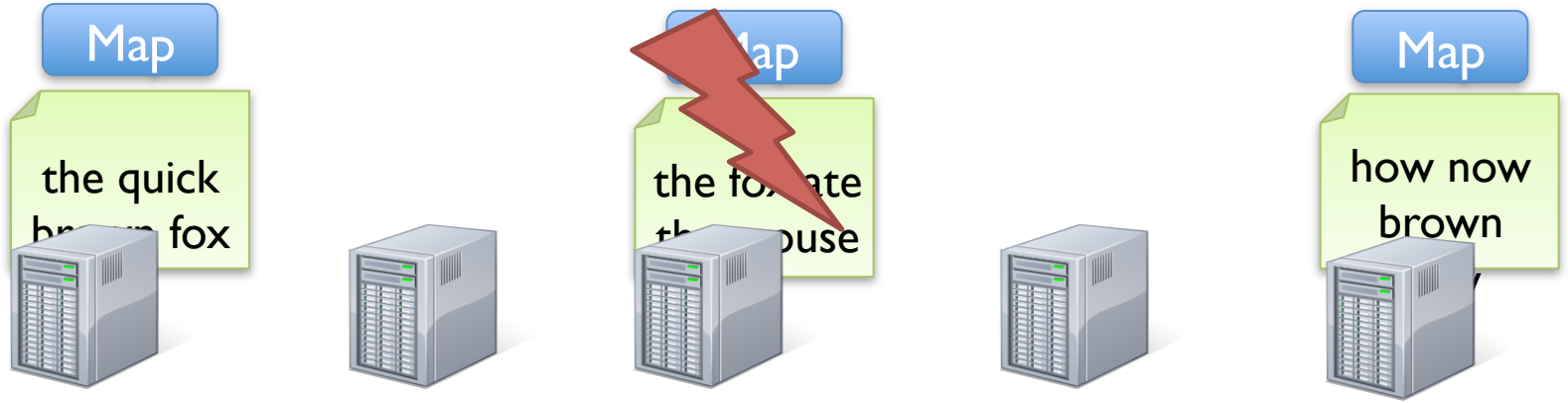
- Retry on another node
- If the same task repeatedly fails, end the job



FAULT RECOVERY

If a task crashes:

- Retry on another node
- If the same task repeatedly fails, end the job



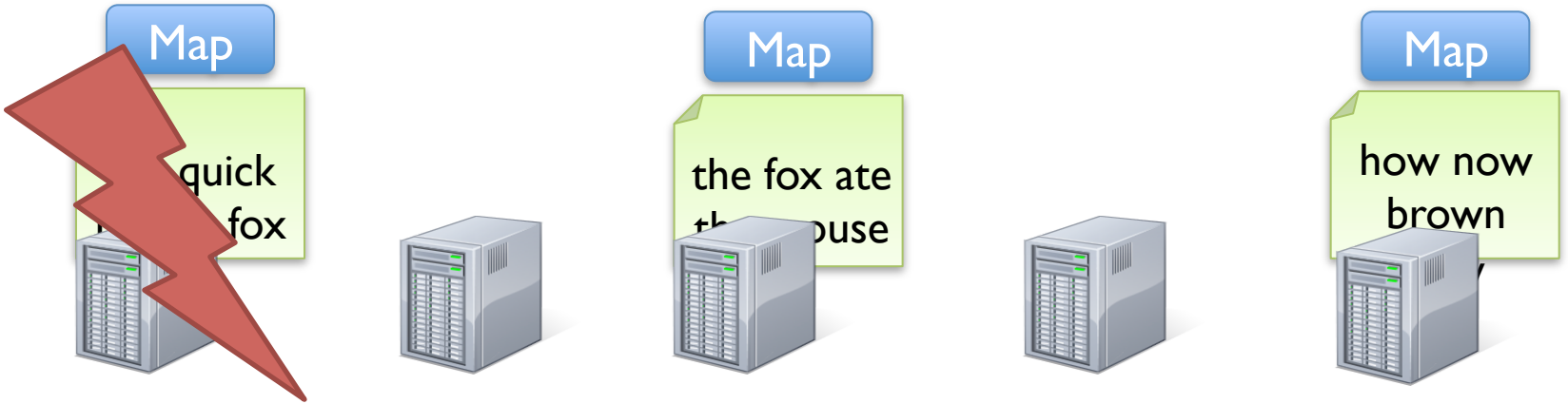
Requires user code to be **deterministic**

FAULT RECOVERY

If a node crashes:

- Relaunch its current tasks on other nodes

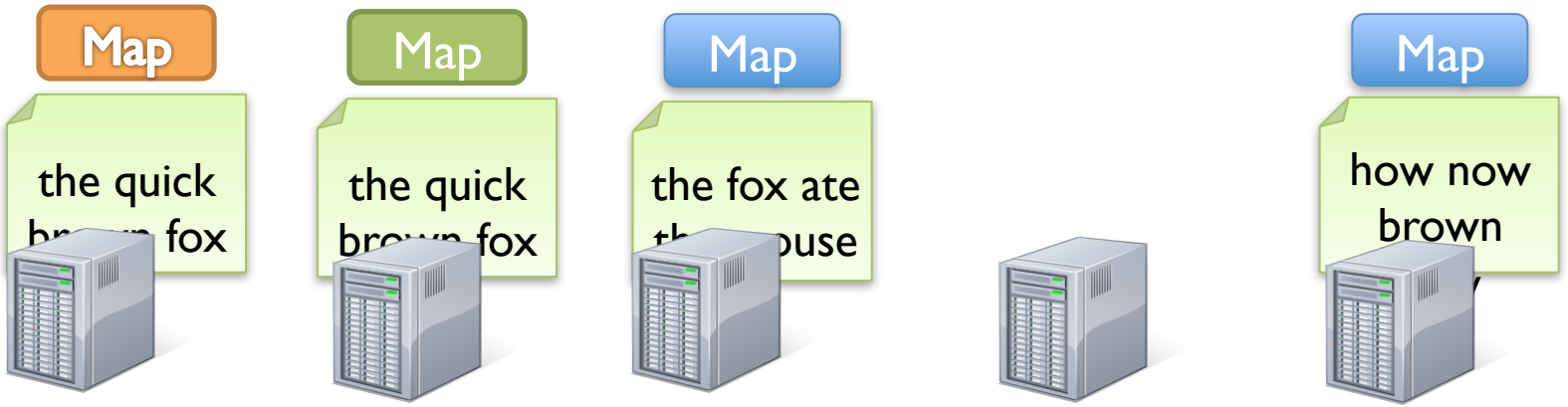
  What about task inputs ? File system replication



FAULT RECOVERY

If a task is going slowly (straggler):

- Launch second copy of task on another node
- Take the output of whichever finishes first



REFINEMENTS

- Combiner functions
- Counters
- Side effects

MapReduce Usage Statistics Over Time

	Aug, ‘04	Mar, ‘06	Sep, '07	Sep, ’09
Number of jobs	29K	171K	2,217K	3,467K
Average completion time (secs)	634	874	395	475
Machine years used	217	2,002	11,081	25,562
Input data read (TB)	3,288	52,254	403,152	544,130
Intermediate data (TB)	758	6,743	34,774	90,120
Output data written (TB)	193	2,970	14,018	57,520
Average worker machines	157	268	394	488

Jeff Dean, LADIS 2009

DISCUSSION: PROGRAMMABILITY

Most real applications require multiple MR steps

- Google indexing pipeline: 21 steps
- Analytics queries (e.g. sessions, top K): 2-5 steps
- Iterative algorithms (e.g. PageRank): 10's of steps

Multi-step jobs create spaghetti code

- 21 MR steps → 21 mapper and reducer classes

DISCUSSION: PERFORMANCE

MR only provides one pass of computation

- Must write out data to file system in-between

Expensive for apps that need to *reuse* data

- Multi-step algorithms (e.g. PageRank)
- Interactive data mining

QUESTIONS / DISCUSSION ?