Concepts and Models of Parallel and Data-centric Programming

Apache Spark – Job Scheduling and Fault Tolerance

Lecture, Summer 2020

Simon Schwitanski
Dr. Christian Terboven

i12 High Performance Computing
RWTH AACHEN UNIVERSITY

Outline

0. Organization
1. Foundations
2. Shared Memory
3. GPU Programming
4. Bulk-Synchronous Parallelism
5. Message Passing
6. Distributed Shared Memory
7. Parallel Algorithms
8. Parallel I/O
9. MapReduce
10. **Apache Spark**

a. Spark Programming Model
b. Resilient Distributed Datasets (RDDs)
c. **Job Scheduling and Fault Tolerance**
d. Streaming and Applications
e. Concluding Remarks

Job Scheduling (1)

- Job scheduler gets active when driver program runs an RDD action
- Scheduler examines RDD's lineage graph → Build DAG of execution stages
  - Try to pipeline as many transformations as possible (for narrow dependencies)
  - Shuffle operations are boundaries of the stages
  - Again: Try to use already computed partitions to speedup computation
- Scheduler launches tasks to compute missing partitions from each stage
- Respect data locality for task assignment
  - First priority: Node with the required partition stored in memory
  - Second priority: Node with preferred location for a partition in the file system (HDFS)

Job Scheduling – Example

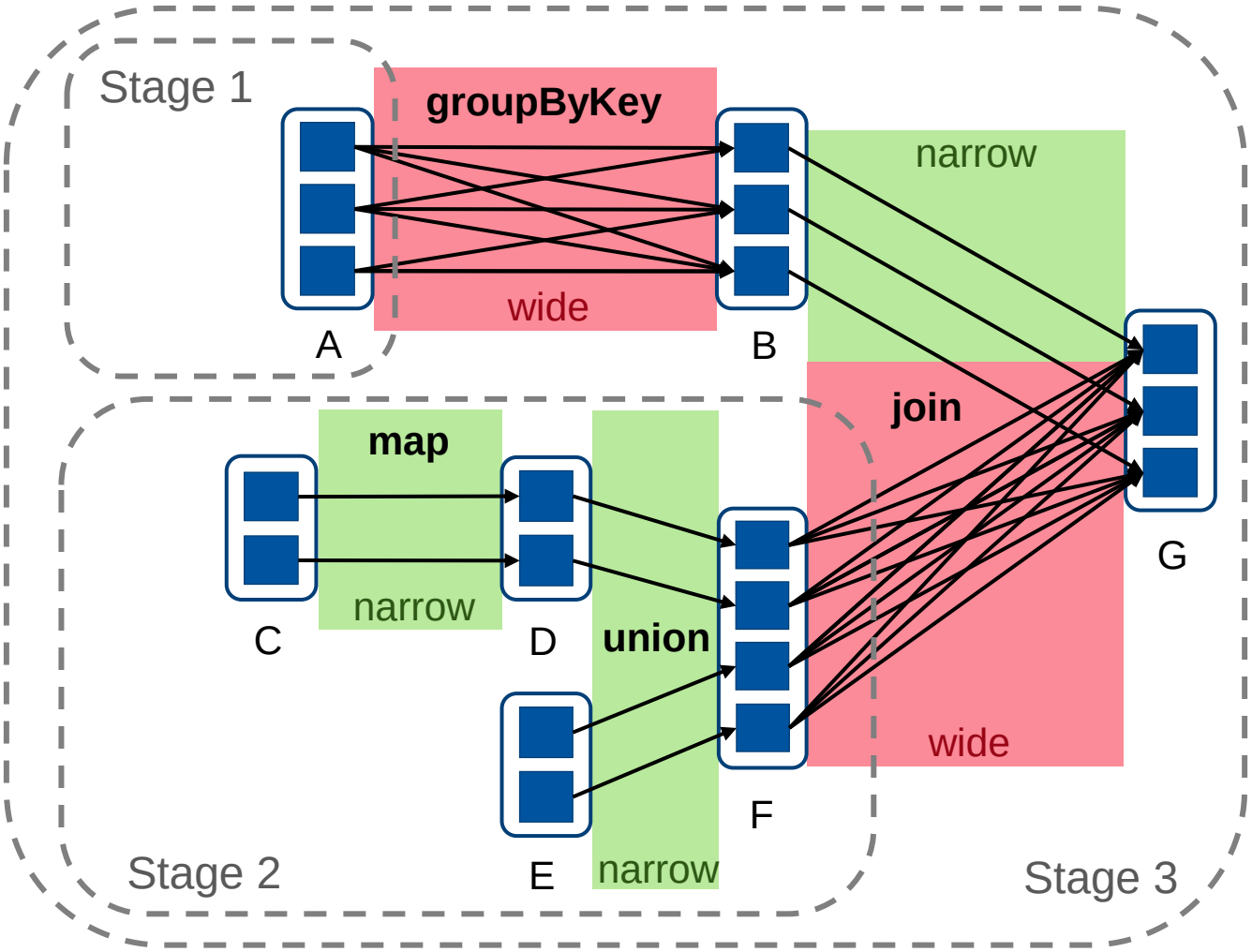


Image Source: Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauly, Michael J. Franklin, Scott Shenker, Ion Stoica. "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing." NSDI2012: 15-28

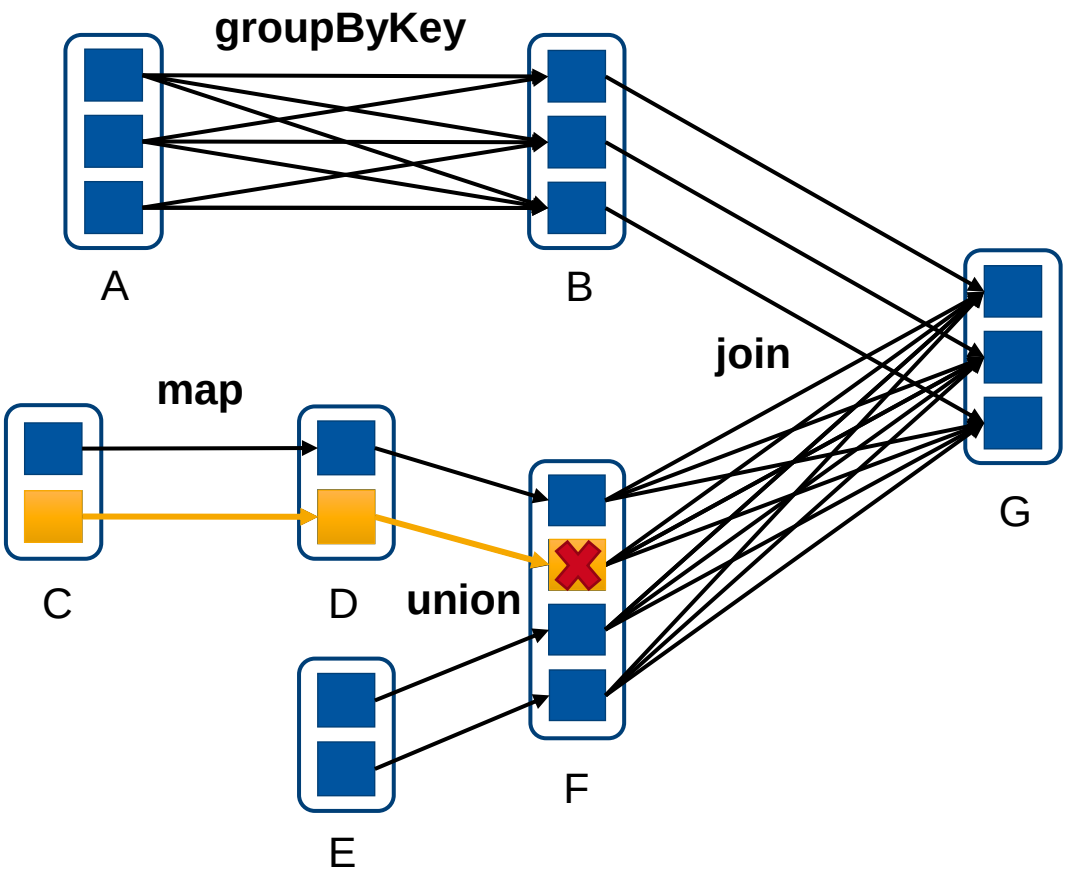
Job Scheduling (2)

- On task failure
  - Re-run task on another node if stage's parents still available
  - If some stages got unavailable, resubmit tasks to compute missing partitions in parallel
- Scheduler is single point of failure
- However: Replication of the RDD lineage graph straightforward

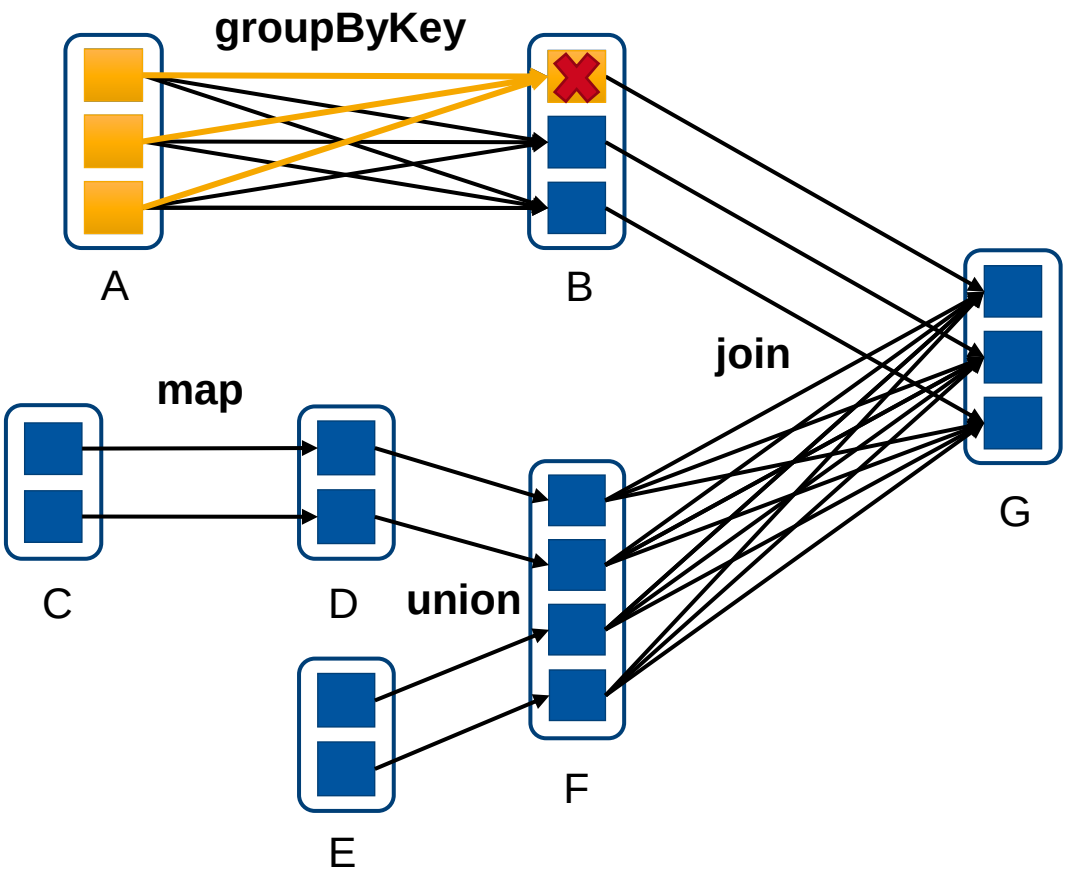
Fault Tolerance with Lineage Graphs

- RDDs can be rebuilt on node failure using the lineage graph
- If some partition of an RDD is lost (e.g., due to node failure)
  - Check the dependencies of this partition in the lineage graph
  - Recompute the lost partition by reapplying the RDD transformations
  - Try to use already computed or cached parent RDD partitions for reconstruction
- Partitions with only narrow dependencies fast to recover
- Partitions with wide dependencies slow to recover
- Fault tolerance without writing every transformed RDD to disk

Fault Tolerance with Lineage Graphs – Example (1)



Fault Tolerance with Lineage Graphs – Example (2)



Fault Tolerance with Lineage Graphs – Example (3)