CS744: Big Data Systems Notes

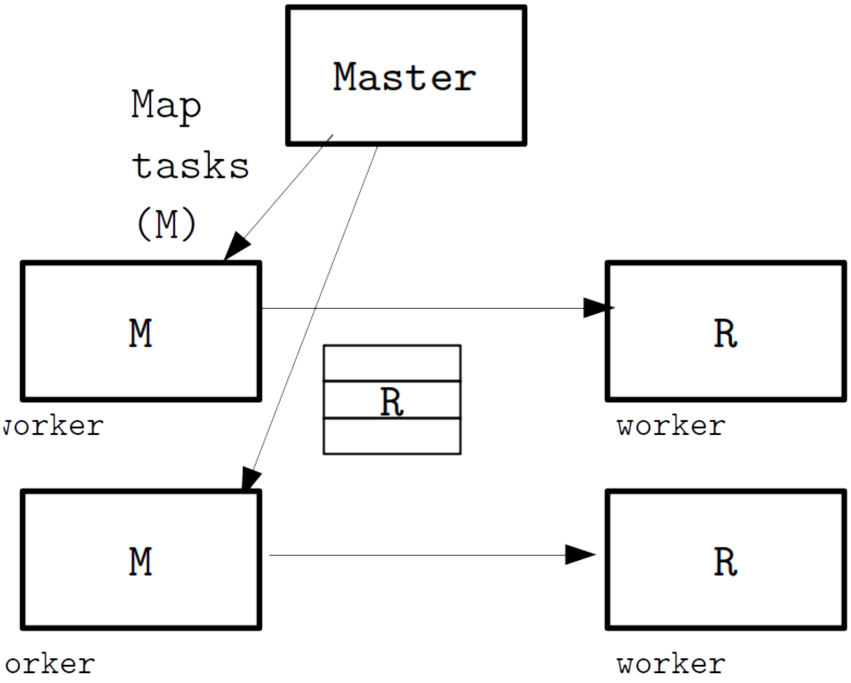
Jack Truskowski

February 6, 2019

Contents

1 2.4.19 MapReduce

- Programming model
- Execution
- Runtime issues
- M-R library handles execution and run-time issues
  - Transparent to programmers



1.1 Operators

1. Map
   - Input = (key,value) –> (key, <v>)
2. Reduce
   - Operates share a key
   - (key,value) is sorted and values passed to reducer

1.2 Failures and Slowdowns

- Handled by the master

1.2.1 Possible failures

1. Map / Reduce
   - Worker fails, some maps and some reduces completed
   - Reduce data is already written to HDFS, doesn't need to be recomputed
   - Maps must be re-executed to recover intermediate data, since it hasn't been written to HDFS

2 2.6.19 Spark

- Programming model

2.1 RDDs

- Partitioned collection of records
- SQL, D-Streams, Graphx
- Intermediate data stored in memory
- Low overhead fault tolerance achieved through lineage

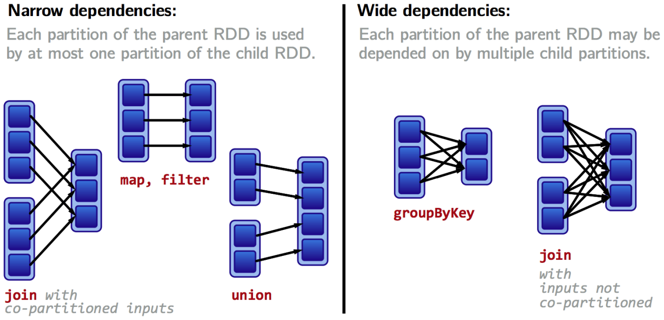
2.2 Benefits

1. Speed up iterative computations
2. Load datasets into memory
   - can't be done in MapReduce
3. Higher level programs

```
RDD -> transformations -> action
```

- `Persist` (deserialized, serialized, on-disk)
  - RDDs only exist logically unless `persist` is called

- Only then materialized (unless wide dependencies)


Narrow dependencies:
Each partition of the parent RDD is used by at most one partition of the child RDD.
map, filter
join with co-partitioned inputs
union
Wide dependencies:
Each partition of the parent RDD may be depended on by multiple child partitions.
groupByKey
join with inputs not co-partitioned

- `REL` (reliable flag): checkpoint to disk or other memory locations

- Partitioning
- Lazy computation

2.3 Example: PageRank

1. Gather
2. Applies
3. Scatter