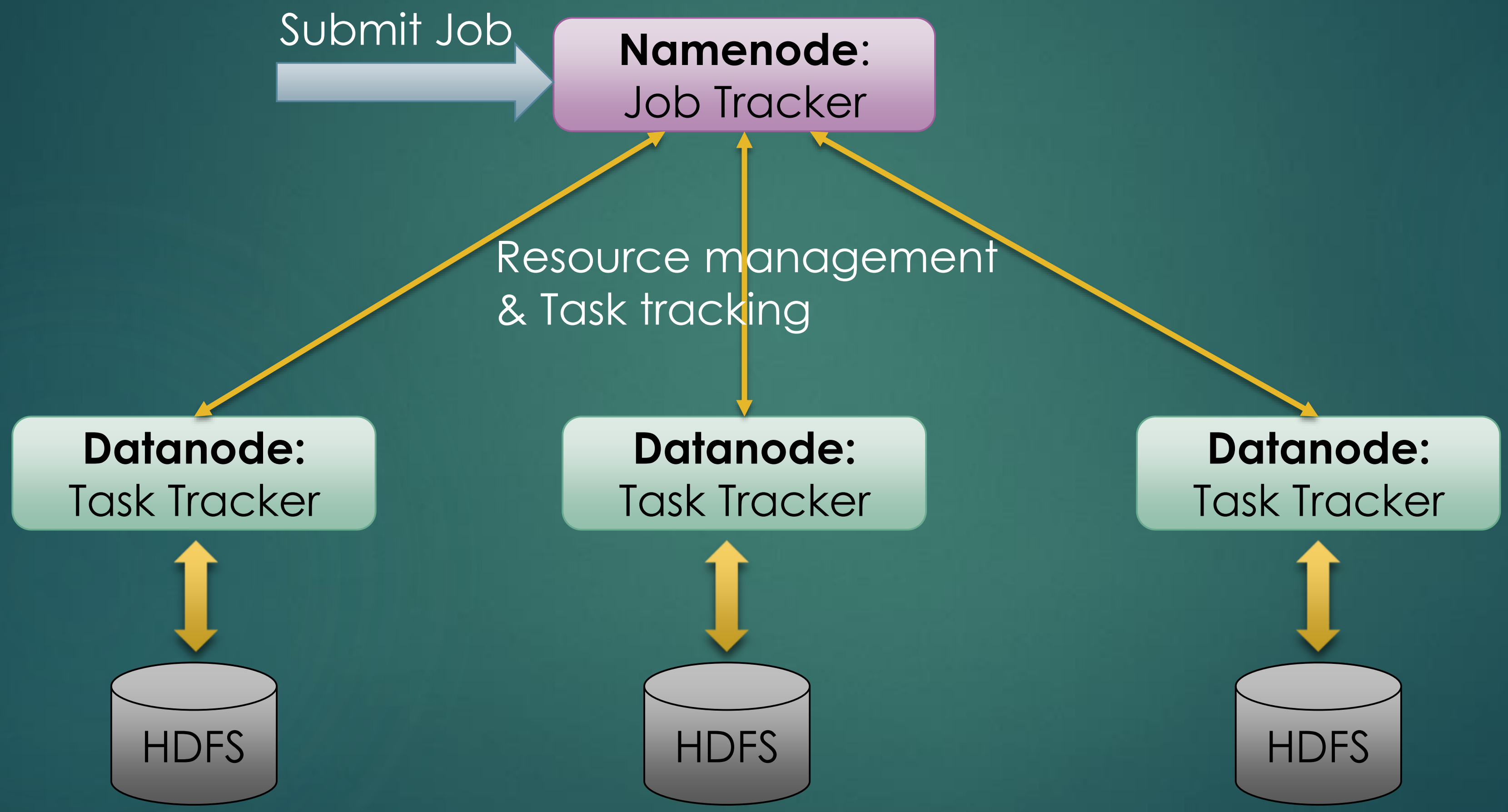
Recap

- HDFS filesystem
  - Block
  - Read and Write operations
  - CLI commands
  - Network topology and rack awareness
- MapReduce

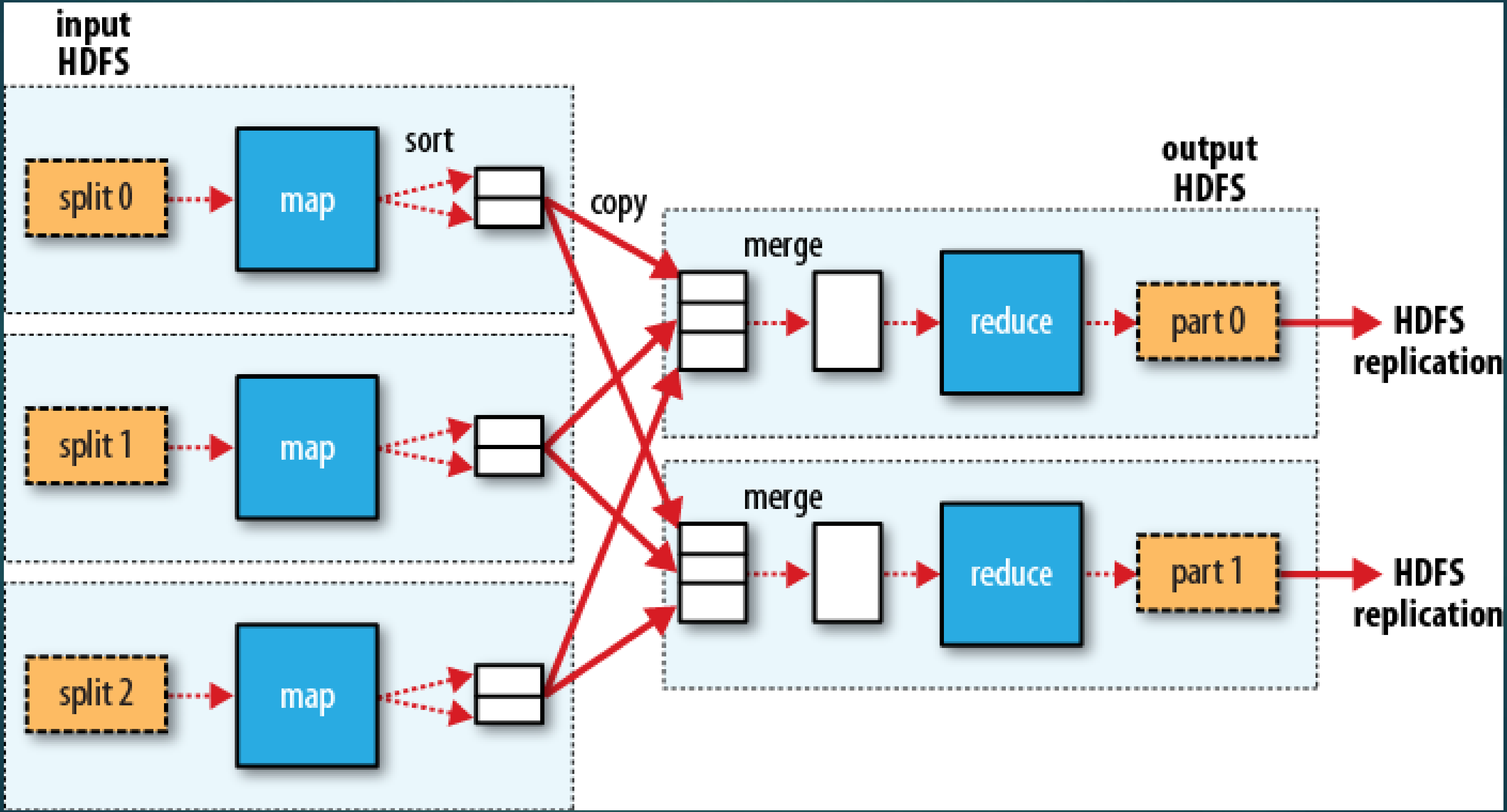
Agenda for today

- MapReduce detailed discussion
- Running your first MapReduce program
- Hadoop 1 vs 2: YARN

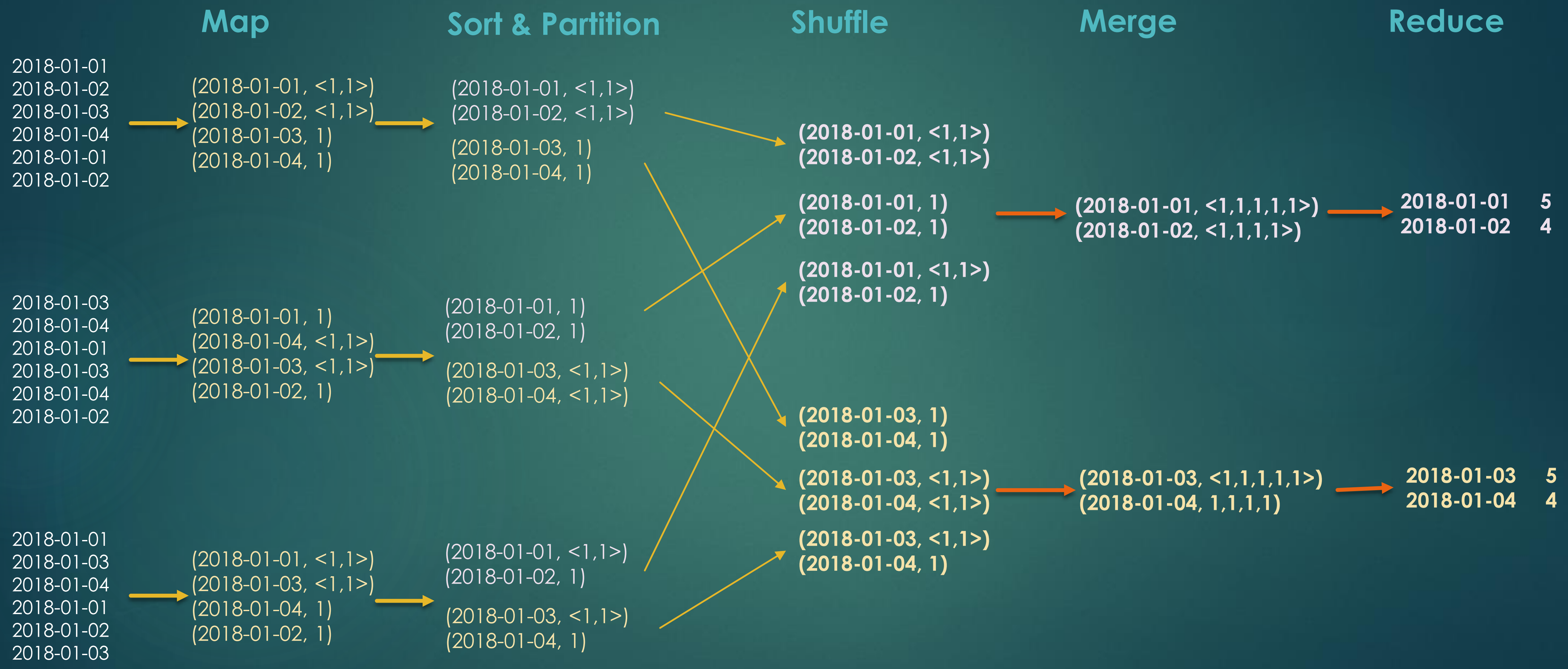
Job management



MapReduce Stages

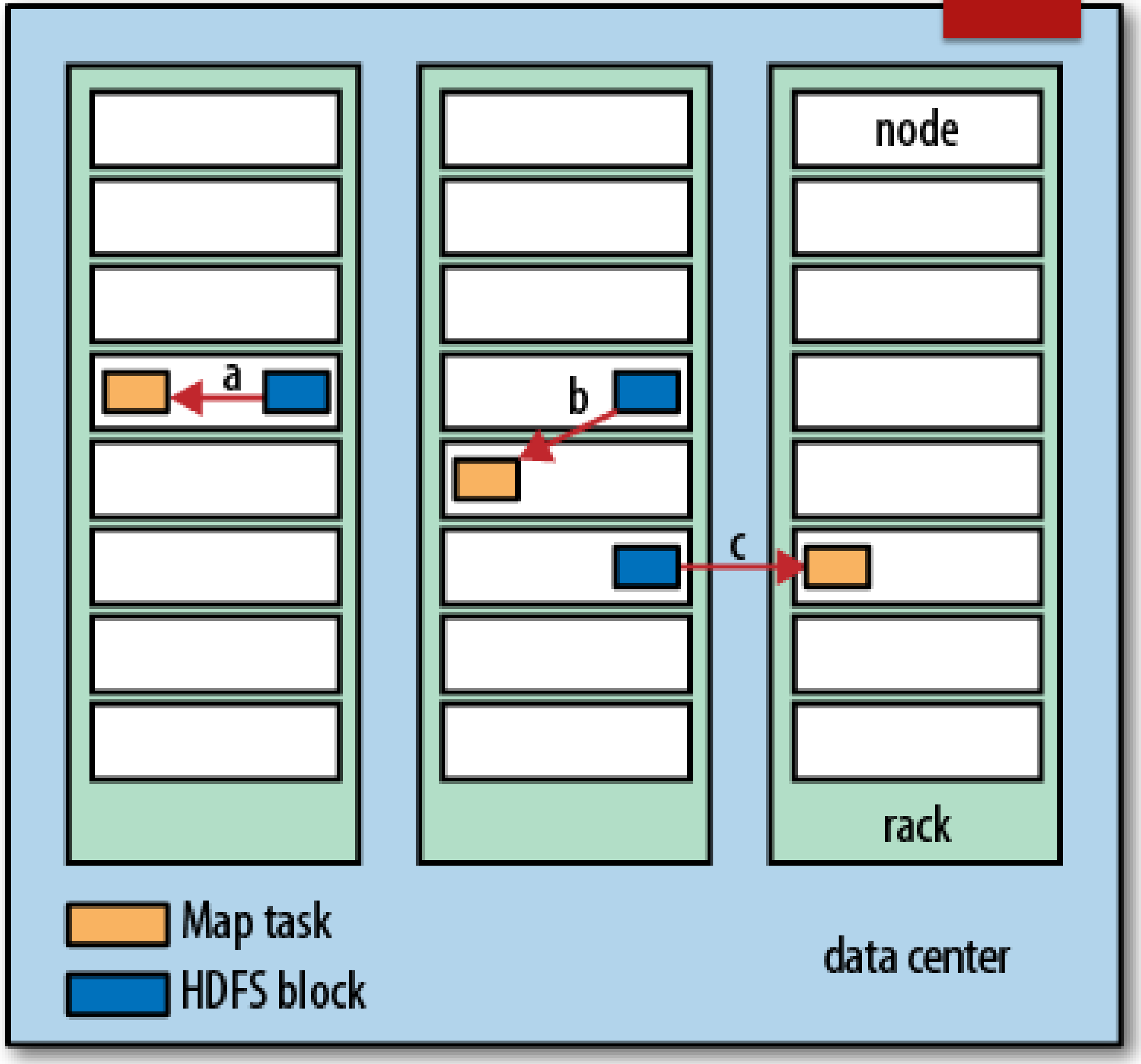


Mapreduce: working example



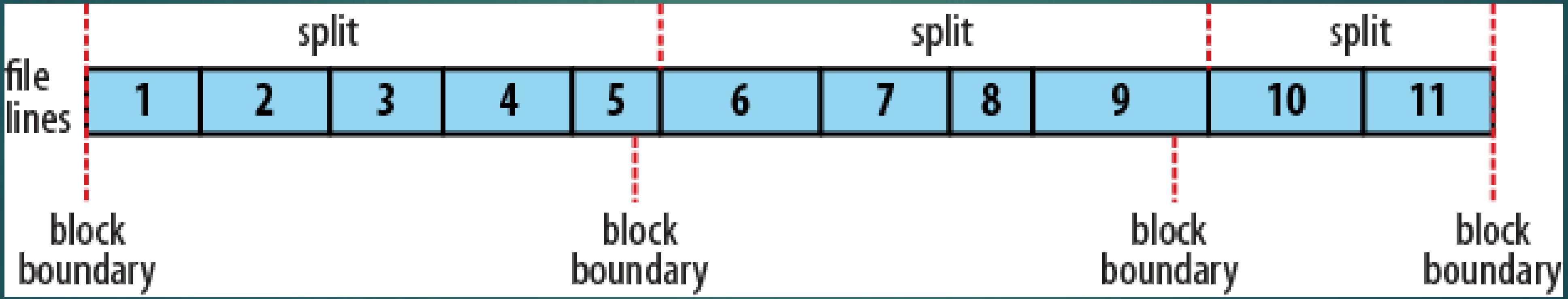
Task to node mapping

- Notion of data locality



Input Splits

- Blocks are of fixed size
- Good chances of records being split between two block



MapReduce: Mapper code

```
public class WebHitCounterMapper extends
Mapper<Input Key, Input Value, Output Key, Output Value>
{

    public void map(Input Key, Input Value, Context context)
throws IOException, InterruptedException {

     <MAP Logic goes here>

     context.write(Output Key, Output Value)
    }
}
```

MapReduce: Reducer code

```
public class WebHitCounterReducer extends
Reducer<Input Key, Input Value, Output Key, Output Value>
{
    public void reduce(Input Key, Iterable<Value Data type>
values,Context context) throws IOException,
InterruptedException {

     <REDUCE logic goes here>

        context.write(Output Key, Output Value);
    }
}
```

MapReduce: Driver Code

```
public class WebHitCounterMain {
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Daily Web Hit Counter");


    }
}
```

MapReduce: Driver Code

```
public class WebHitCounterMain {
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Daily Web Hit Counter");

        job.setJarByClass(main.WebHitCounterMain.class);
        job.setMapperClass(mapper.WebHitCounterMapper.class);
        job.setReducerClass(reducer.WebHitCounterReducer.class);

    }
}
```

MapReduce: Driver Code

```
public class WebHitCounterMain {
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Daily Web Hit Counter");

        job.setJarByClass(main.WebHitCounterMain.class);
        job.setMapperClass(mapper.WebHitCounterMapper.class);
        job.setReducerClass(reducer.WebHitCounterReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

    }
}
```

MapReduce: Driver Code

```
public class WebHitCounterMain {
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Daily Web Hit Counter");

        job.setJarByClass(main.WebHitCounterMain.class);
        job.setMapperClass(mapper.WebHitCounterMapper.class);
        job.setReducerClass(reducer.WebHitCounterReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

    }
}
```

MapReduce: Driver Code

```
public class WebHitCounterMain {
    public static void main(String[] args) throws Exception {

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Daily Web Hit Counter");

        job.setJarByClass(main.WebHitCounterMain.class);
        job.setMapperClass(mapper.WebHitCounterMapper.class);
        job.setReducerClass(reducer.WebHitCounterReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Programming Exercise

Challenges with Hadoop 1

- Applications were limited to MapReduce implementations only
- Namenode machine crash or maintenance activity
- Namespace scaling
- Backup and Recovery
- Batch oriented architecture
- Support for various file formats
- Dual responsibilities of Job tracker

Image Ref: https://www.greycampus.com/blog/big-data/top-differences-between-hadoop-1-0-and-hadoop-2-2

Hadoop 2

- Support for other data processing engines
- High Availability
- HDFS Federation
- HDFS Snapshot
- Introduced Streaming and Interactive analysis
- Support for various file formats
- Yarn

YARN

- Yet Another Resource Negotiator

MapReduce 1	**YARN**
Job Tracker	Resource Manager, Application Master and Timeline server
Task Tracker	Node Manager
Slot	Containers

YARN model

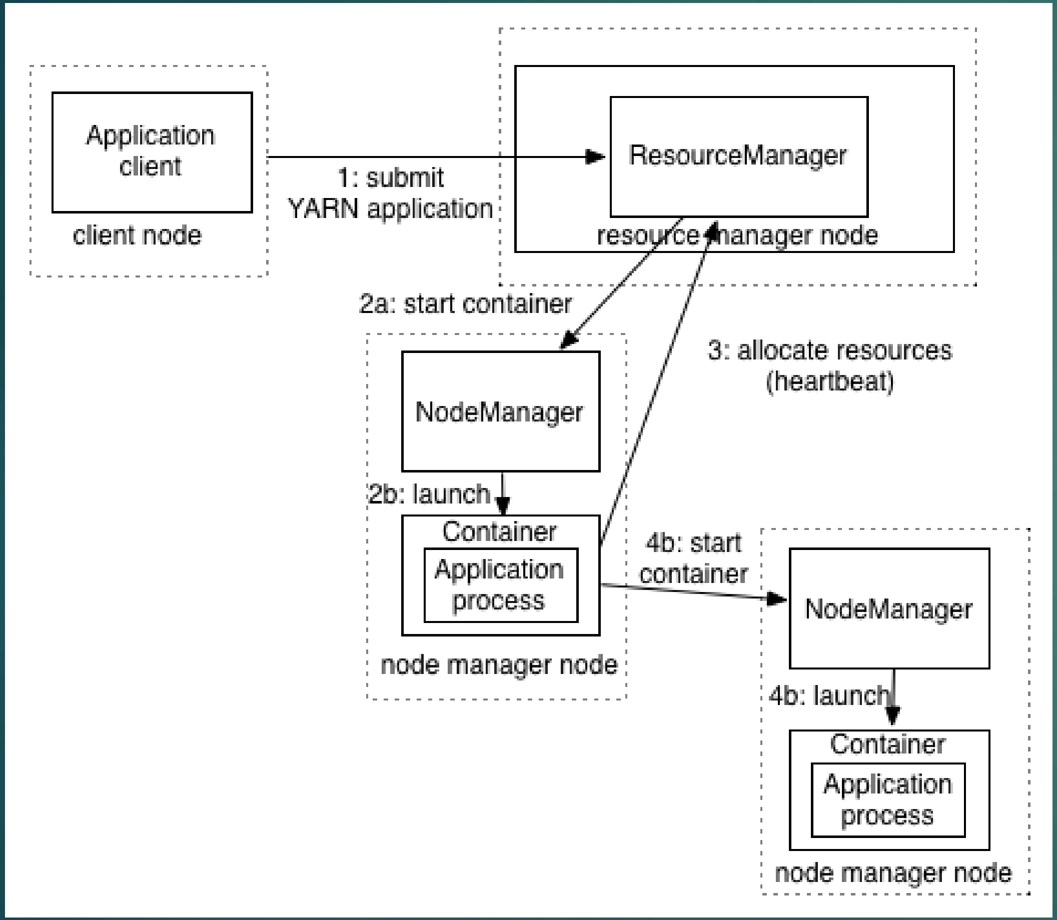


Image Ref: Hadoop definitive guide 4th edition

Pros of YARN

- Scalability
- Availability
- Utilization
- Multitenancy

Hadoop Installation

Standalone	Everything in one JVM. No HDFS installation
Pseudo-distributed	Mimic a distributed cluster on single physical machine
Distributed	Fully distributed cluster with multiple physical machines

Reference

- Hadoop standalone vs pseudo-distributed

https://stackoverflow.com/questions/23435333/what-is-the-difference-between-single-node-pseudo-distributed-mode-in-hadoop

- Hadoop installation differences

https://medium.com/@nidhinmahesh/getting-started-hadoop-mapreduce-hdfs-and-yarn-configuration-and-sample-program-febb1415f945

- Download Ubuntu

https://www.ubuntu.com/download/desktop

- Hadoop installation step by step guide

http://www.bogotobogo.com/Hadoop/BigData_hadoop_Install_on_ubuntu_16_04_single_node_cluster.php