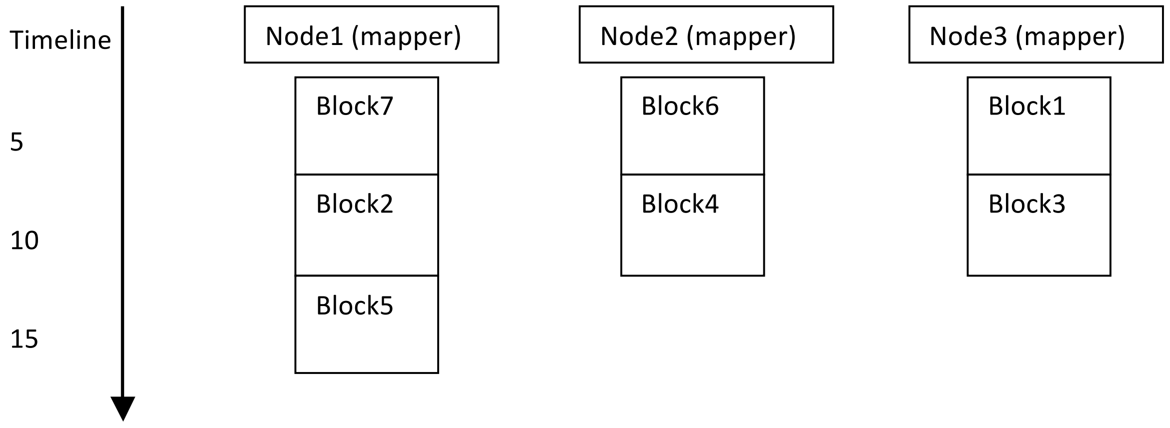
Estimating the Distributed Execution Cost in MapReduce

MapReduce execution consists of the following steps. Given an input file consisting of n blocks:

1. Individual mappers have to process all of the n blocks across available nodes, generating key-value pairs (the output is stored in local storage on each node, not in HDFS)
2. Next, Hadoop framework delivers keys and values to reducers according to the partitioning rule (over the network). The keys are given to each individual reducer in a sorted order.
3. Each reducer then processes the keys in the received order, outputting the final result back to HDFS. By default a single file per reducer is written.

For the purposes of assignment 2, we are only considering the cost associated with step #1. The rest will be revisited in later homework assignments (e.g., the cost of network transfer or reducer processing).
Consider the following example, similar to the homework assignment:
Suppose you are given an input file with a size of 7 blocks. The cost to process each block is 5 minutes. How long would it take for 1-node and 3-node cluster?

1-node example is trivial. Each block takes 5 minutes to process; 7 blocks on the single node would then take exactly 7 * 5 = 35 minutes.
Three node example is more interesting (please resist the temptation to divide 35 minutes by 3). The processing is done in terms of **whole** blocks. Below is one possible example execution, with 3 blocks evaluated at Node1 and 2 blocks evaluated at Node2 and Node3.



Note that blocks are indivisible and cannot be partially evaluated in two places. In this case, processing using a 3-node cluster takes 15 minutes, based on the slowest node (Node1). Node2 and Node3 are idle (or running another job) in the last 5 minutes, but they cannot expedite the processing of Block5 above. The execution could instead process 3 blocks / 3 blocks / 1 block (i.e., process 3 blocks at Node2 and 1 at Node3) which will also result in 15 minute runtime.

If the data is not distributed evenly and one of the nodes processes more than 3 blocks, the total execution could take even longer – which is why we discussed the need for balancing the data as evenly as possible across the cluster.