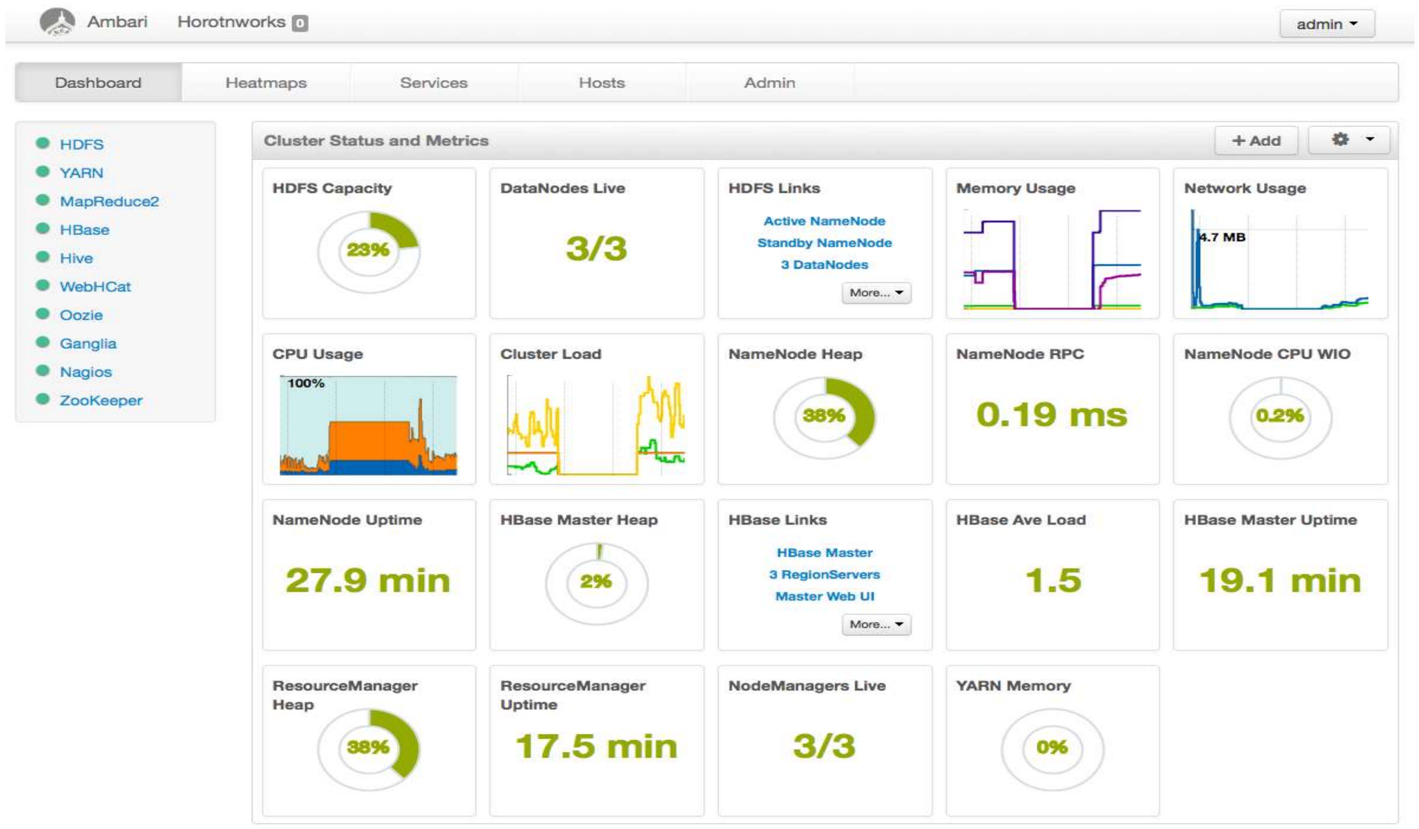
Optimizing MapReduce Jobs

Monitoring Performance



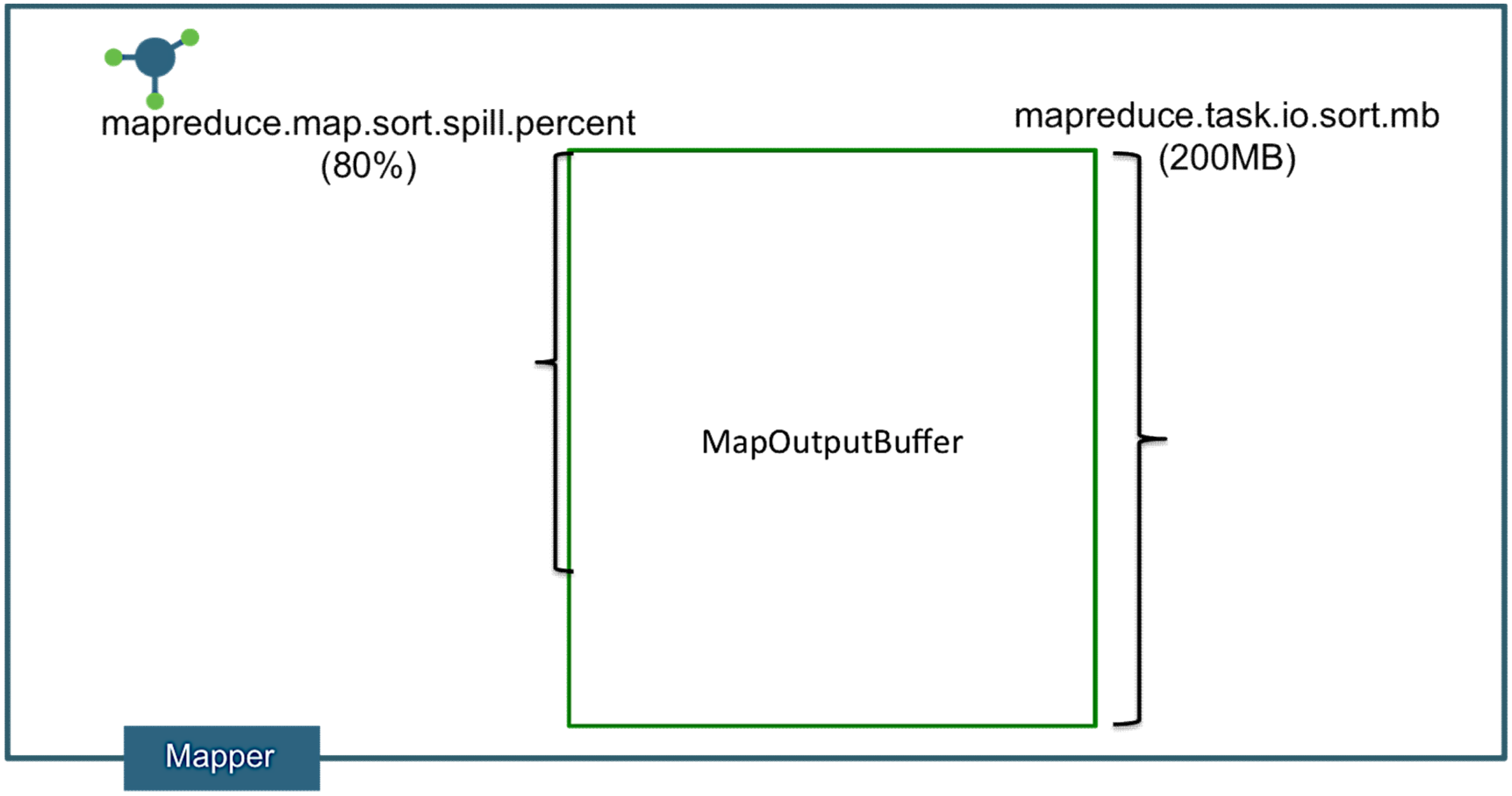
Cluster Status and Metrics

HDFS Capacity 23%	DataNodes Live 3/3	HDFS Links: Active NameNode, Standby NameNode, 3 DataNodes	Memory Usage	Network Usage
CPU Usage	Cluster Load	NameNode Heap 38%	NameNode RPC 0.19 ms	NameNode CPU WIO 0.2%
NameNode Uptime 27.9 min	HBase Master Heap 2%	HBase Links: HBase Master, 3 RegionServers, Master Web UI	HBase Ave Load 1.5	HBase Master Uptime 19.1 min
ResourceManager Heap 38%	ResourceManager Uptime 17.5 min	NodeManagers Live 3/3	YARN Memory 0%	

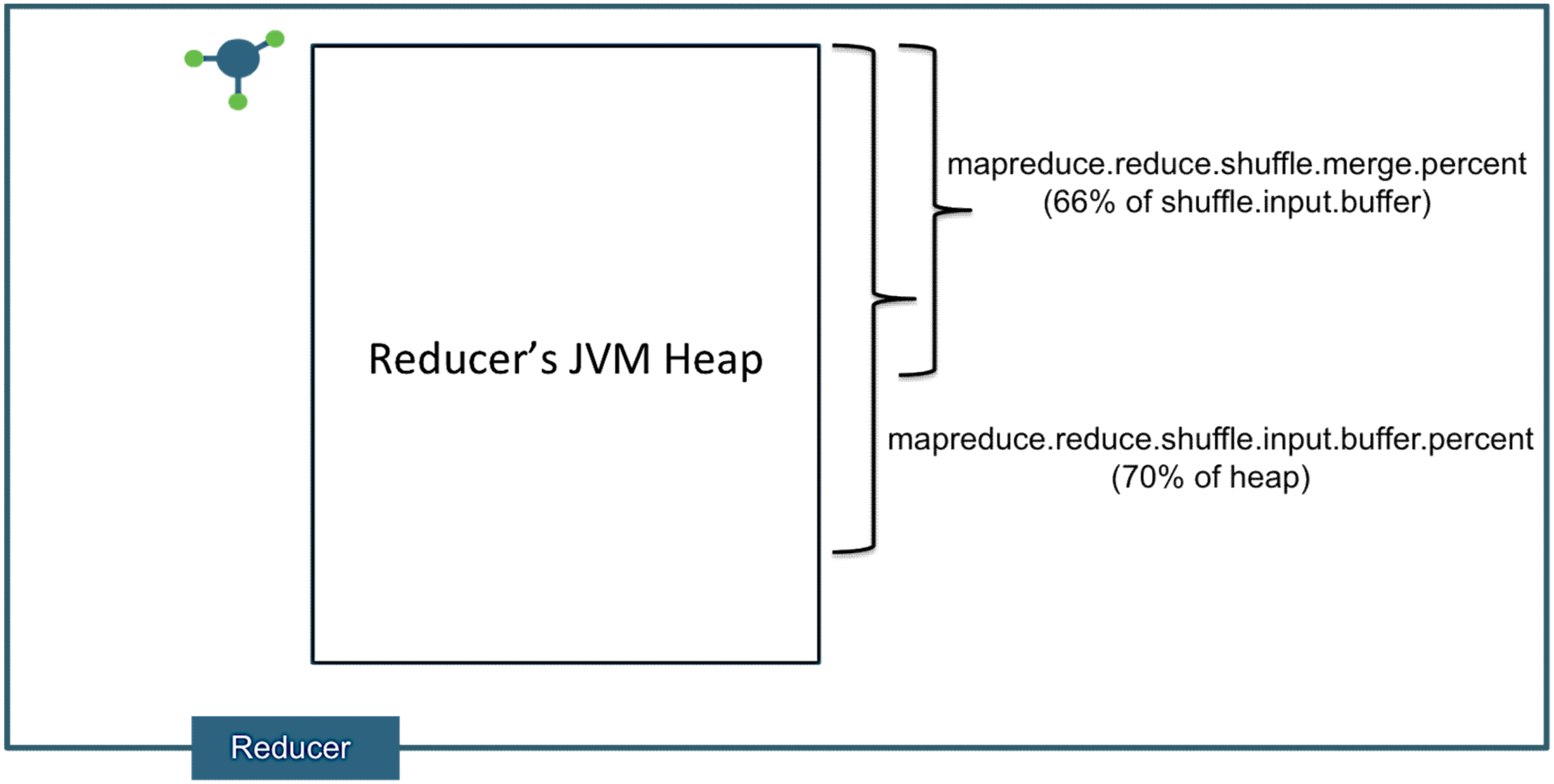
Optimization Best Practices

- Distribute workload evenly across NodeManagers
- Use a combiner
- Avoid instantiating new objects, including string literals
- Prefer StringBuilder over string concatenation
- Use data compression
- Avoid converting numeric types to text
- Define and configure a RawComparator
- Prefer StringUtils.split over String.split

Optimizing the Map Phase



Optimizing the Reduce Phase


Reducer's JVM Heap
mapreduce.reduce.shuffle.merge.percent
(66% of shuffle.input.buffer)
mapreduce.reduce.shuffle.input.buffer.percent
(70% of heap)
Reducer

Data Compression

- **Snappy**: org.apache.hadoop.io.compress.SnappyCodec
- **gzip**: org.apache.hadoop.io.compress.GzipCodec
- **bzip2**: org.apache.hadoop.io.compress.BZip2Codec
- **LZO**: com.hadoop.compression.lzo.LzopCodec
- **DEFLATE**: org.apache.hadoop.io.compress.DefaultCodec

Limitations of Compression

- **Space vs. time**: is the gain in filesystem space or smaller network traffic worth the additional time it takes to compress and decompress the data?
- **Splittable vs. Non-splittable**: a major concern in MapReduce, since large files are split up into chunks across the cluster.

Configuring Data Compression

```
Configuration conf = job.getConfiguration();
conf.setBoolean(MRJobConfig.MAP_OUTPUT_COMPRESS,
true);
conf.setClass(MRJobConfig.MAP_OUTPUT_COMPRESS_CODEC,
                              SnappyCodec.class,
                              CompressionCodec.class);

conf.setBoolean(FileOutputFormat.COMPRESS, true);
conf.setClass(FileOutputFormat.COMPRESS_CODEC,
                              SnappyCodec.class,
                              CompressionCodec.class);
```