RHadoop Advanced Usage

RevolutionAnalytics

December 3, 2013

Outline

1. Review
2. Converting Models to MapReduce
3. Regression with MapReduce
4. K-Means with MapReduce
5. Performance Considerations
6. How `rmr` Works

MapReduce Basics: Map

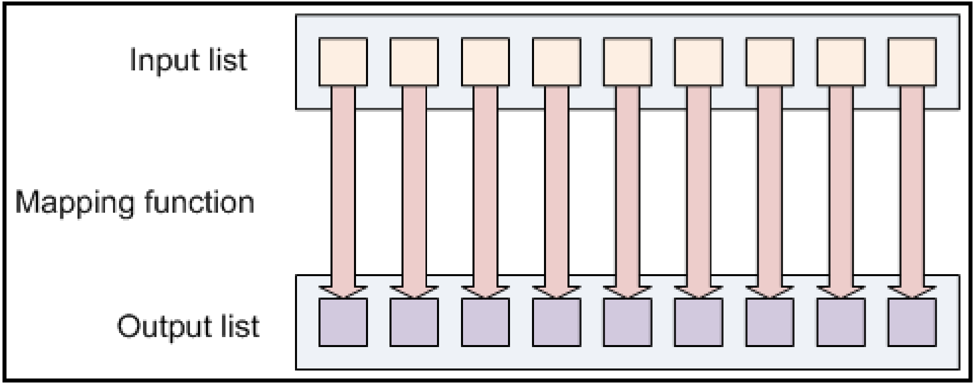


Figure : Map

MapReduce Basics: Reduce

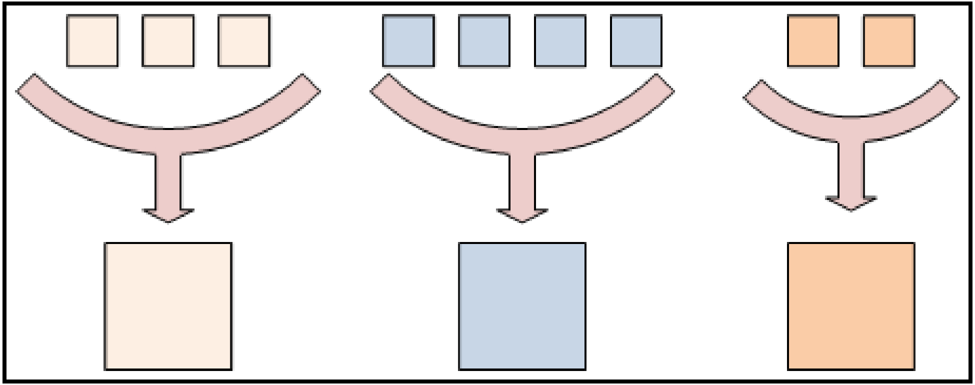
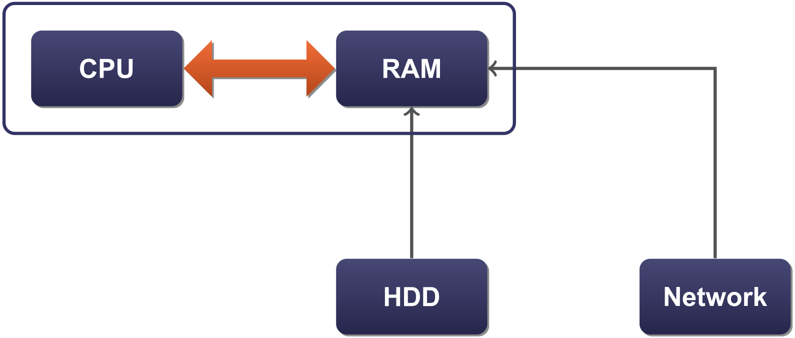
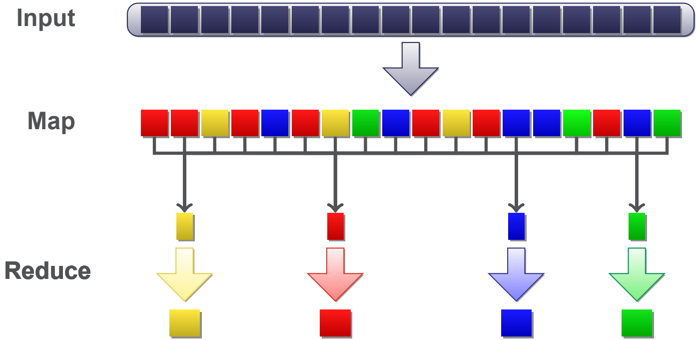


Figure : Reduce

Hadoop Basics



MapReduce Data Flow



Hadoop: The Metaphor

You've decided to hire the monkeys, what could possibly go wrong?

- Certain parts of the pile are heavier than others
- They see something shiny
- They get hungry

Hadoop: Potential Problems

What problems can happen when trying to run a job?

- Certain parts of the data take longer to process
- Hardware problems: machine dies, network traffic
- Software problems: run out of memory

REVOLUTION ANALYTICS

Hadoop: Solutions

Hadoop can handle all of these things and more, using:

- Data redundancy
- Load-balancing
- Job monitoring and resubmitting

Hadoop: Architecture

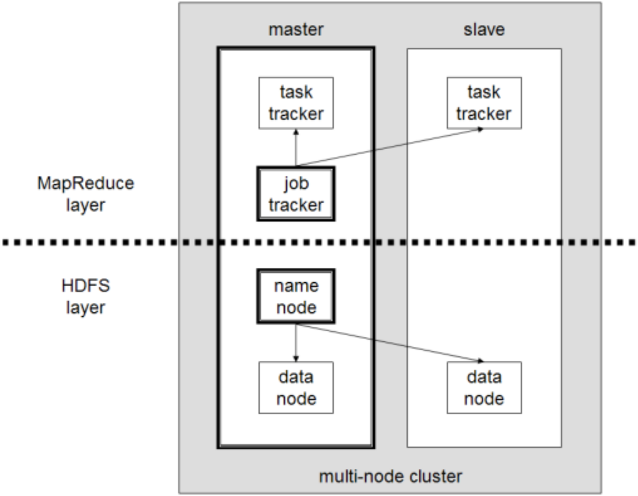


Figure : MapReduce and HDFS on Hadoop

Hadoop: Terminology

- Job: High-level MapReduce “program” you submit
- Task: Lower-level subtask assigned to nodes

Centralized Jobs are composed of Tasks, which are distributed and redundant, so that Task failures don’t result in Job failures.

MapReduce on Hadoop

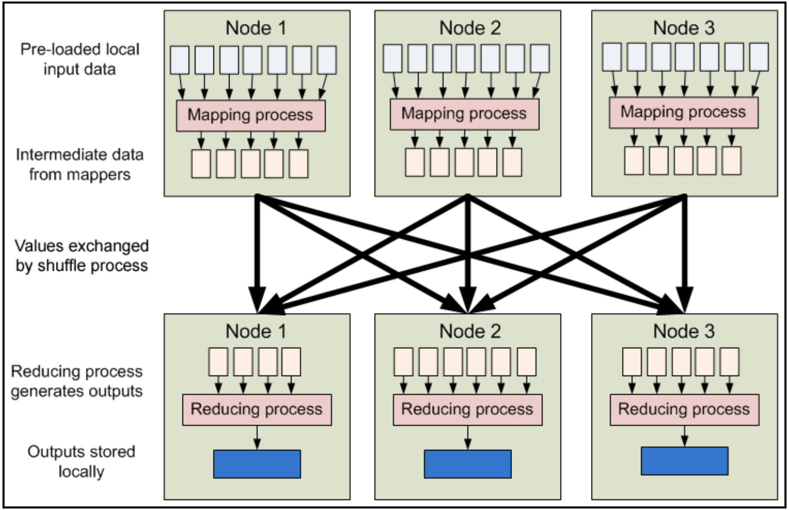
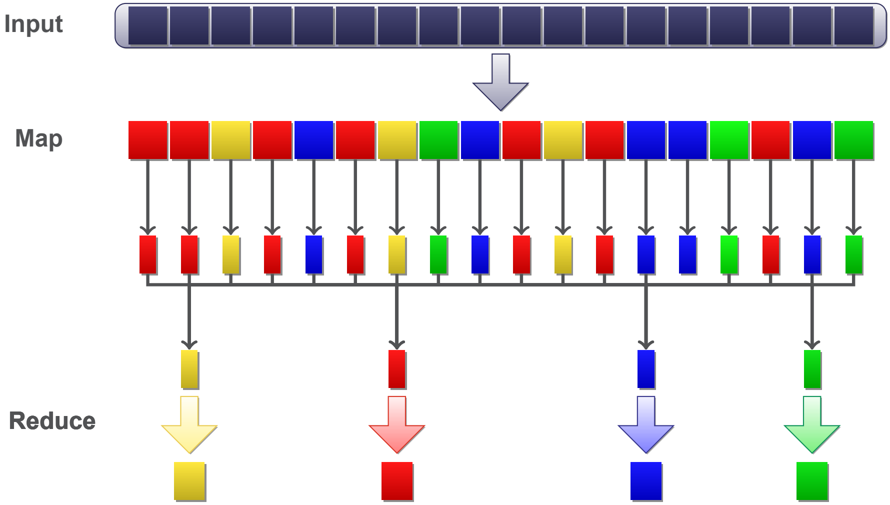


Figure : MapReduce and HDFS on Hadoop

MapReduce Data Flow



Outline

Core Concepts

- Identify computational bottlenecks
- Track the data “size” through your model

Data vs Computation

- Different perspectives, similar results
- Largely a matter of your intuition

Outline

Least-Squares Regression

```r
x <- 1:100
y <-  x + 10*rnorm(100)
mod <- lm(y~x)
plot(mod)
```

Least-Squares Regression: How?

- $y = \beta X + \epsilon$

Least-Squares Regression: How?

- $y = \beta X + \epsilon$
- How do we solve for β

Least-Squares Regression: How?

- $y = \beta X + \epsilon$
- How do we solve for β
- $\beta = (X^T X)^{-1} X^T y$

Least-Squares Regression: How?

- $y = \beta X + \epsilon$
- How do we solve for β
- $\beta = (X^T X)^{-1} X^T y$
- How do we MapReduce this?

How Big is the Data?

For least-squares regression:

$$\beta = (X^T X)^{-1} X^T y \tag{1}$$

Discuss the following:

- What **can** you do without Hadoop?
- What **can't** you do without Hadoop?
- How would you implement this on Hadoop?

How Big is the Data?

- For large number of "rows": n, and much smaller number of "columns": k
- X is n by k
- $(X^T X)$ is only k by k
- $X^T y$ is only k by ka

REVOLUTION ANALYTICS

OLS via MapReduce

```
XtX =
  values(
    from.dfs(
      mapreduce(
        input = X,
        map =
          function(., Xi)
            keyval(1, list(t(Xi) %*% Xi)),
        reduce = Sum,
        combine = TRUE))[[1]]
```

OLS via MapReduce

```
Xty =
  values(
    from.dfs(
      mapreduce(
        input = X,
        map = function(., Xi)
          keyval(1, list(t(Xi) %*% y)),
        reduce = Sum,
        combine = TRUE)))[[1]]
```

OLS via MapReduce

```r
solve(XtX, Xty)
```

Outline

K-Means Review

- Unsupervised learning
- Cluster analysis
- Finds $\mathbb{K}$ clusters that minimize distances from cluster centers

K-Means Example

```r
require(graphics)

# a 2-dimensional example
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 1, sd = 0.3), ncol = 2))
colnames(x) <- c("x", "y")
(cl <- kmeans(x, 2))
plot(x, col = cl$cluster)
points(cl$centers, col = 1:2, pch = 8, cex=2)
```

K-Means Algorithm

Iteratively assign points to clusters and update cluster centers around their assigned points

1. Initialize cluster centers
2. Assign each point to the closest cluster
3. Update each cluster center to be the average of the points assigned to it
4. Repeat 2-3 until no cluster center changes

Where is the Bottleneck?

For k-means clustering, discuss the following:

- Where is the computational bottleneck?
- How could you rewrite that bottleneck as a Map?
- How could you collect the results of that Map with Reduce?

K-Means with MapReduce

- What is the computational bottleneck?

K-Means with MapReduce

- What is the computational bottleneck?
- Distance calculations: ie finding the closest cluster

K-Means via MapReduce

```
dist.fun =
      function(C, P) {
        apply(
          C,
          1,
          function(x)
            colSums((t(P) - x)^2))}
```

K-Means via MapReduce

```
ns.map =
      function(., P) {
        nearest = {
          if(is.null(C))
            sample(
              1:num.clusters,
              nrow(P),
              replace = T)
          else {
            D = dist.fun(C, P)
            nearest = max.col(-D)}}
        if(!(combine || in.memory.combine))
          keyval(nearest, P)
        else
          keyval(nearest, cbind(1, P))}
```

K-Means via MapReduce

```
kmeans.reduce = {
      if (!(combine || in.memory.combine) )
        function(., P)
          t(as.matrix(apply(P, 2, mean)))
      else
        function(k, P)
          keyval(
            k,
            t(as.matrix(apply(P, 2, sum))))}
```

K-Means + MapReduce: Good Idea

- K: How many clusters?

K-Means + MapReduce: Good Idea

- K: How many clusters?
- That was one iteration

Outline

Tasks Execute on One Node

Each application of a `map` or `reduce` function occurs on a single machine in the cluster.

- What if the data can't fit on a node?

Tasks Execute on One Node

Each application of a `map` or `reduce` function occurs on a single machine in the cluster.

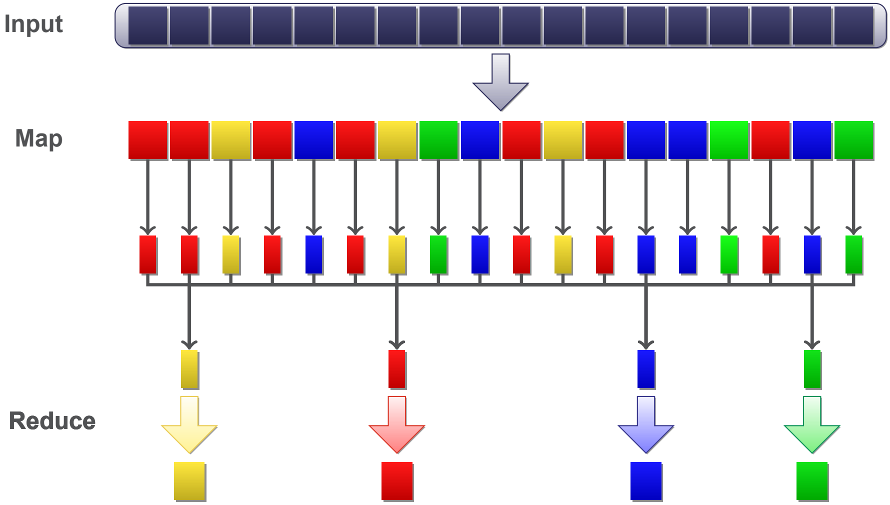
- What if the data can't fit on a node?
- What if the task is taking "too long"

Tasks Execute on One Node

Each application of a `map` or `reduce` function occurs on a single machine in the cluster.

- What if the data can't fit on a node?
- What if the task is taking "too long"
- Your jobs may fail

The Combine



The Combine

- Whatâ□□s a combiner?
- A function used to consolidate mapper output locally before the big shuffle/sort happens across nodes

- When to use?

The Combine

- Whatâ□□s a combiner?
- A function used to consolidate mapper output locally before the big shuffle/sort happens across nodes

- When to use?
- Whenever you can

The Combine

- Whatâ□□s a combiner?
- A function used to consolidate mapper output locally before the big shuffle/sort happens across nodes

- When to use?
- Whenever you can
- Why?

The Combine

- Whatâ□□s a combiner?
- A function used to consolidate mapper output locally before the big shuffle/sort happens across nodes

- When to use?
- Whenever you can
- Why?
- Combine allows you to reduce the size of the `map` output

The Combine

- Whatâs a combiner?
- A function used to consolidate mapper output locally before the big shuffle/sort happens across nodes

- When to use?
- Whenever you can
- Why?
- Combine allows you to reduce the size of the `map` output
- So what?

REVOLUTION ANALYTICS

The Combine

- Whatâ□□s a combiner?
- A function used to consolidate mapper output locally before the big shuffle/sort happens across nodes

- When to use?
- Whenever you can
- Why?
- Combine allows you to reduce the size of the `map` output
- So what?
- Less data to send over the network, shuffle-sort and reduce, with less data being cached to HD

REVOLUTION ANALYTICS

Combine vs Reduce: Standard deviation

How could you rewrite this as a combine function?

```
reduce = function(k,v) {
    keyval(k,sd(v))
}
```

- Remeber this? $Var[X] = E[X - \mu_X]^2 = E[X^2] - E[X]^2$

Combine vs Reduce: Standard deviation

```
combine = function(k,v) {
    keyval(k,list(v=sum(v),vsquared=sum(v^2)))
}
```

When can't you combine?

- median
- other quantiles
- any other time you need relative position information

wordcount: Combine

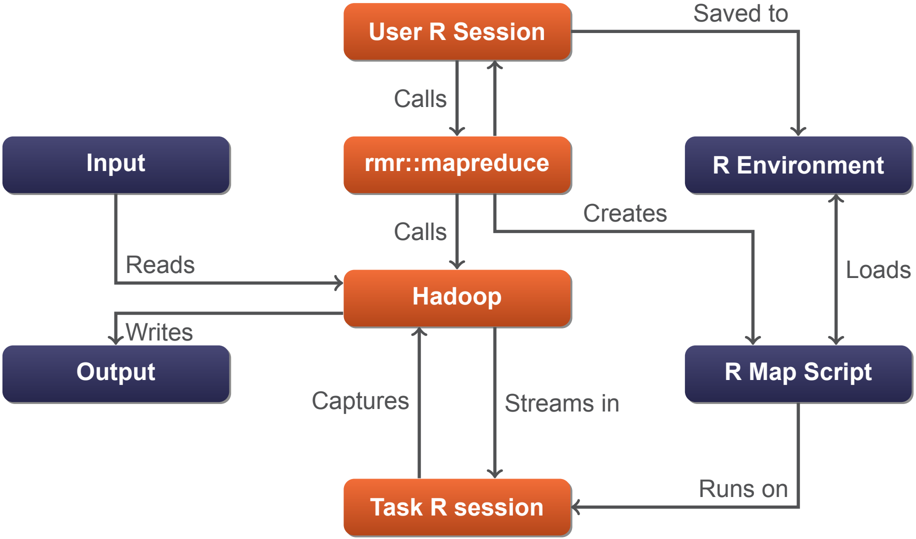
…and thatâ□□s why the reducer code uses sum() and not length():

```
reduce = function(word, counts) {
  keyval(word, sum(counts))
}
```

Outline

REVOLUTION ANALYTICS

The `rmr` Data Flow



`rmr` Is Hadoop Streaming

Ultimately, `rmr2::mapreduce` simply initializes a Hadoop Streaming job and submits it to the cluster. Most job configuration is simply configuration of the Hadoop Streaming job.

Here are some options you may want to know:

- `mapred.child.java.opts` - Command-line options for the jvm, useful to reduce java memory consumption
- `mapred.map.tasks` - Explicitly set the number of map tasks
- `mapred.reduce.tasks` - Explicitly set the number of reduce tasks
- `mapred.job.name` - Give your job a name, you can more easily monitor it

For more information, see the documentation.