Lecture 26: Pig: Making Hadoop Easy
(Some Slides provided by: Alan Gates, Yahoo!Research)

Friday, Dec 3, 2010
What is Pig?

- An engine for executing programs on top of Hadoop
- It provides a language, Pig Latin, to specify these programs
- An Apache open source project
Map-Reduce

• Computation is moved to the data
• A simple yet powerful programming model
  – Map: every record handled individually
  – Shuffle: records collected by key
  – Reduce: key and iterator of all associated values
• User provides:
  – input and output (usually files)
  – map Java function
  – key to aggregate on
  – reduce Java function
• Opportunities for more control: partitioning, sorting, partial aggregations, etc.
Romeo, Romeo, wherefore art thou Romeo?

What, art thou hurt?

Romeo, 1
Romeo, 1
wherefore, 1
art, 1
thou, 1
Romeo, 1
art, (1, 1)
hurt (1),
thou (1, 1)
art, 2
hurt, 1
thou, 2
Romeo, (1, 1, 1)
wherefore, (1)
what, (1)
Romeo, 3
wherefore, 1
what, 1
Making Parallelism Simple

• Sequential reads = good read speeds
• In large cluster failures are guaranteed; Map Reduce handles retries
• Good fit for batch processing applications that need to touch all your data:
  – data mining
  – model tuning
• Bad fit for applications that need to find one particular record
• Bad fit for applications that need to communicate between processes; oriented around independent units of work
Why use Pig?

Suppose you have user data in one file, website data in another, and you need to find the top 5 most visited sites by users aged 18 - 25.
In Map-Reduce

```java
import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
...
        jc.addJob(joinJob);
        jc.addJob(groupJob);
        jc.addJob(limit);
        jc.run();
    }
}
```
In Pig Latin

Users = load ‘users’ as (name, age);
Fltrd = filter Users by
    age >= 18 and age <= 25;
Pages = load ‘pages’ as (user, url);
Jnd = join Fltrd by name, Pages by user;
Grpd = group Jnd by url;
Smmd = foreach Grpd generate group,
    COUNT(Jnd) as clicks;
Srtd = order Smmd by clicks desc;
Top5 = limit Srtd 5;
store Top5 into ‘top5sites’;

9 lines of code, 15 minutes to write
But can it fly?

Pig Performance vs Map-Reduce

- Sep 11 08: 7.6
- Nov 11 08: 2.5
- Jan 20 09: 1.8
- Feb 23 09: 1.6
- Apr 20 09: 1.5
- Jun 28 09: 1.4
- Aug 28 09: 1.2
- Oct 18 09: 1.0
Essence of Pig

- Map-Reduce is too low a level to program, SQL too high
- Pig Latin, a language intended to sit between the two:
  - Imperative
  - Provides standard relational transforms (join, sort, etc.)
  - Schemas are optional, used when available, can be defined at runtime
  - User Defined Functions are first class citizens
  - Opportunities for advanced optimizer but optimizations by programmer also possible
How It Works

Script
A = load
B = filter
C = group
D = foreach

Logical Plan ≈ relational algebra

Parser

Semantic Checks

Logical Optimizer

Logical Plan

Logical to Physical Translator

Physical Plan

MapReduce Launcher

Map-Reduce Plan

Physical To MR Translator

Physical Plan

Map-Reduce Plan = physical operators broken into Map, Combine, and Reduce stages

Physical Plan = physical operators to be executed

Plan standard optimizations

Jar to hadoop
Cool Things We’ve Added In the Last Year

• Multiquery – Ability to combine multiple group bys into a single MR job (0.3)
• Merge join – If data is already sorted on join key, do join via merge in map phase (0.4)
• Skew join – Hash join for data with skew in join key. Allows splitting of key across multiple reducers to handle skew. (0.4)
• Zebra – Contrib project that provides columnar storage of data (0.4)
• Rework of Load and Store functions to make them much easier to write (0.7, branched but not released)
• Owl, a metadata service for the grid (committed, will be released in 0.8).
Users = \text{load} \ 'users' \ as \ (name, \ age); \\
Pages = \text{load} \ 'pages' \ as \ (user, \ url); \\
\text{Jnd} = \text{join} \ Pages \ by \ user, \ Users \ by \ name \ using \ "replicated";
Hash Join

Users = load 'users' as (name, age);
Pages = load 'pages' as (user, url);
Jnd = join Users by name, Pages by user;

Map 1
User
block n

Map 2
Page
block m

Reducer 1
(1, user)
(1, fred)
(2, fred)
(2, fred)

Reducer 2
(2, name)
(1, jane)
(2, jane)
(2, jane)
Skew Join

Users = load 'users' as (name, age);
Pages = load 'pages' as (user, url);
Jnd = join Pages by user, Users by name using "skewed";

Map 1
Pages block n
Map 2
Users block m
Reducer 1
(1, user)
(1, fred, p1)
(1, fred, p2)
(2, fred)
Reducer 2
(2, name)
(1, fred, p3)
(1, fred, p4)
(2, fred)
Merge Join

Users = load ‘users’ as (name, age);
Pages = load ‘pages’ as (user, url);
Jnd = join Pages by user, Users by name using “merge”;

Map 1

Map 2

Pages Users Pages Users

aaron... aaron... amr... amr...

... ... ...

... ... ...

zach zach... zach...
Multi-store script

A = load 'users' as (name, age, gender, city, state);
B = filter A by name is not null;
C1 = group B by age, gender;
D1 = foreach C1 generate group, COUNT(B);
    store D into 'bydemo';
C2 = group B by state;
D2 = foreach C2 generate group, COUNT(B);
    store D2 into 'bystate';
Multi-Store Map-Reduce Plan

map

filter

split

local rearrange  local rearrange

reduce

demux

package  package

foreach  foreach
What are people doing with Pig

- At Yahoo ~70% of Hadoop jobs are Pig jobs
- Being used at Twitter, LinkedIn, and other companies
- Available as part of Amazon EMR web service and Cloudera Hadoop distribution
- What users use Pig for:
  - Search infrastructure
  - Ad relevance
  - Model training
  - User intent analysis
  - Web log processing
  - Image processing
  - Incremental processing of large data sets
What We’re Working on this Year

• Optimizer rewrite
• Integrating Pig with metadata
• Usability – our current error messages might as well be written in actual Latin
• Automated usage info collection
• UDFs in python
Research Opportunities

• Cost based optimization – how does current RDBMS technology carry over to MR world?
• Memory Usage – given that data processing is very memory intensive and Java offers poor control of memory usage, how can Pig be written to use memory well?
• Automated Hadoop Tuning – Can Pig figure out how to configure Hadoop to best run a particular script?
• Indices, materialized views, etc. – How do these traditional RDBMS tools fit into the MR world?
• Human time queries – Analysts want access to the petabytes of data available via Hadoop, but they don’t want to wait hours for their jobs to finish; can Pig find a way to answer analysts question in under 60 seconds?
• Map-Reduce-Reduce – Can MR be made more efficient for multiple MR jobs?
• How should Pig integrate with workflow systems?
• See more:  http://wiki.apache.org/pig/PigJournal
Learn More

- On line tutorials
  - From Cloudera, [http://www.cloudera.com/hadoop-training](http://www.cloudera.com/hadoop-training)
- A couple of Hadoop books are available that include chapters on Pig, search at your favorite bookstore
- Join the mailing lists:
  - [pig-user@hadoop.apache.org](mailto:pig-user@hadoop.apache.org) for user questions
  - [pig-dev@hadoop.apache.com](mailto:pig-dev@hadoop.apache.com) for developer issues
- Contribute your work, over 50 people have so far