Declarative MapReduce
MapReduce Examples

- Filter
- Aggregate
- Grouped aggregated
- Equi-join
- Non-equi-join

Map
Reduce
Map
Reduce
Map
Reduce
Map
Reduce
Declarative Languages

- Describe what you want to do not how to do it
- The most popular example is SQL
- Can we compile SQL queries into MapReduce program(s)?
Pig

- A system built on-top of Hadoop (Now supports Spark as well)
- Provides a SQL-ETL-like query language termed Pig Latin
- Compiles Pig Latin programs into MapReduce programs
Examples

Filter: Return all the lines that have a user-specified response code, e.g., 200.

log = LOAD 'logs.csv' USING PigStorage()
   AS (host, time, method, url, response, bytes);
ok_lines = FILTER log BY response = '200';
STORE ok_lines into 'filtered_output' ;
Examples

- Grouped aggregate
- Find the total number of bytes per response code

```
log = LOAD 'logs.csv' USING PigStorage()
   AS (host, time, method, url, response, bytes: int);
grouped = GROUP log BY response;
grouped_aggregate =
   FOREACH grouped GENERATE group, SUM(bytes);
STORE grouped_aggregate into 'grouped_output';
```
Examples

- Grouped aggregate
- Find the **average** number of bytes per response code

```pig
log = LOAD 'logs.csv' USING PigStorage()
   AS (host, time, method, url, response, bytes: int);
grouped = GROUP log BY response;
grouped_aggregate =
   FOREACH grouped GENERATE group, AVG(bytes);
STORE grouped_aggregate into 'grouped_output';
```
Examples

- Join: Find pairs of requests that ask for the **same URL**, coming from the **same source**

log1 = LOAD 'logs.csv' USING PigStorage()
  AS (host, time, method, url, response, bytes: int);

log2 = LOAD 'logs.csv' USING PigStorage()
  AS (host, time, method, url, response, bytes: int);

joined = JOIN log1 BY (url, host),
         log2 BY (url, host);
Examples

Join: Find pairs of requests that ask for the same URL, coming from the same source and happened within an hour of each other

```
log1 = LOAD 'logs.csv' USING PigStorage()
   AS (host, time, method, url, response, bytes: int);
log2 = LOAD 'logs.csv' USING PigStorage()
   AS (host, time, method, url, response, bytes: int);
joined = JOIN log1 BY (url, host),
         log2 BY (url, host);
fILTERed = FILTER joined BY
  ABS(log1::time - log2::time) < 3600000;
```
How it works

- **LOAD operation**
  - Determines the input path and InputFormat

- **STORE operation**
  - Determines the output path and OutputFormat

- **FILTER and FOREACH**
  - Translated into map-only jobs

- **AGGREGATE and JOIN**
  - Translated into map-reduce jobs

- All are compiled into one or more MapReduce jobs
Additional Features

- Lazy execution
  - Nothing gets actually executed until the STORE command is reached

- Consolidation of map-only jobs
  - Map-only jobs (FILTER and FOREACH) can be consolidated into a next job’s map function or a previous job’s reduce function
A Complex Example

log1 = LOAD 'logs.csv' USING PigStorage() AS (...);
log2 = LOAD 'logs.csv' USING PigStorage() AS (...);
joined = JOIN log1 BY (url, host),
         log2 BY (url, host);
filtered = FILTER joined BY
           ABS(log1::time - log2::time) < 3600000;
grouped = GROUP filtered BY log1::host;
agg_groups = FOREACH grouped GENERATE group, COUNT(*);
STORE agg_groups INTO 'final_result';
Further Readings

- Pig home page: https://pig.apache.org
- Detailed documentation: http://pig.apache.org/docs/r0.17.0/
- The original Pig Latin paper: