Managing Large Datasets and

Computation Workflows with Disco

and Blaze

- Benjamin Zaitlen

- Continuum Analytics

MapReduce With Disco

- Developed at Nokia

- MapReduce Implementation written in Python and Erlang

- Scalable Distributed Computation

- Useful for Processing Big Data

- http://discoproject.org/

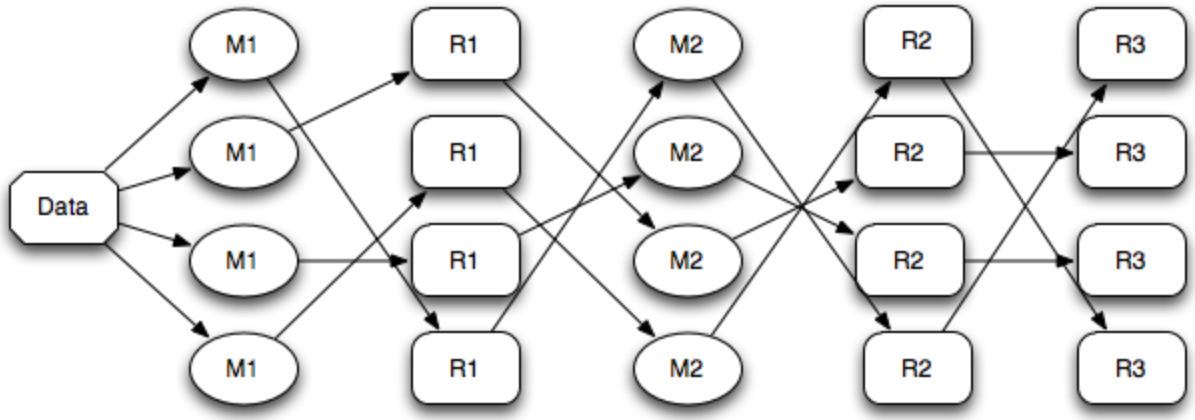
Map Reduce Background

- **Map** Data: (key,value) pairs to buckets.

- **Partitioning** define how keys move to buckets

- **Reduce** process data in buckets with the same key

- *Repeat if necessary*


Data
M1
M1
M1
M1
R1
R1
R1
R1
M2
M2
M2
M2
R2
R2
R2
R2
R3
R3
R3
R3

Disco Distributed Filesystem (DDFS)

- Tag Based System

- Data, URL, S3, etc.

- Chunked or Whole File

- Horizontally Scalable

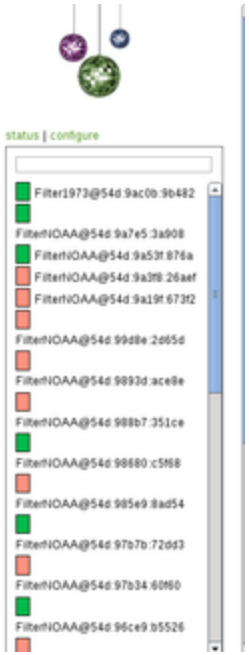
Disco Web Interface (DDFS)

disco status

master		
13030	2	25

node001		
12995	16	16

node002		
13755	16	18



Example: Hello Weather

- Daily Recordings of a Surface Data

- 10000+ Weather Stations

- Data Store: FTP

- Data Contents:

- Average: Temperature, Dew Pt,Pressue, Wind Speed, etc

- Max/Min: Temperature

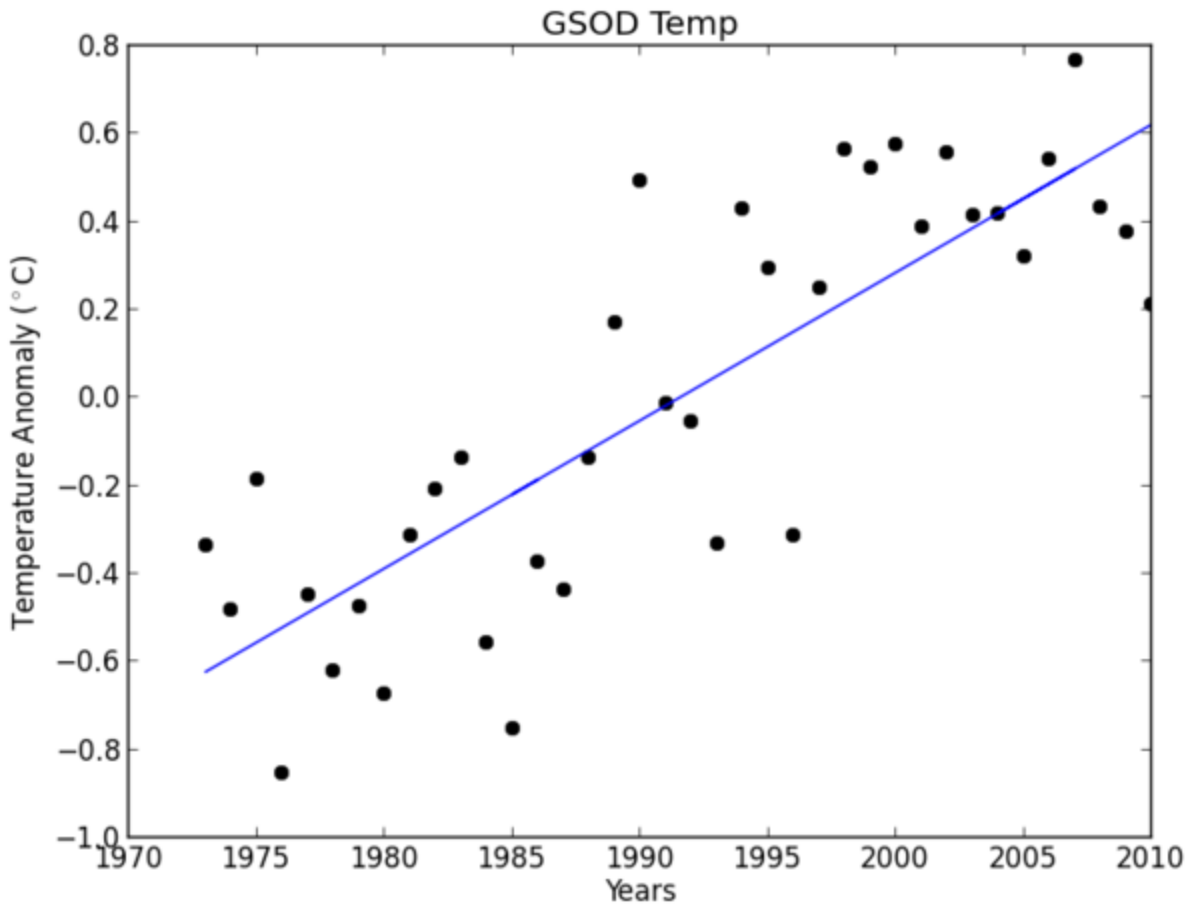
- Totals: Precipitation, Snow Depth

- Data Hierarchy:

- Year->File

- 1981/227100-99999-1981.op.gz (WMO-WBAN-Year.op.gz)

Global Temperature Anomaly


GSOD Temp
Temperature Anomaly (°C)
Years
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
1970
1975
1980
1985
1990
1995
2000
2005
2010

Map Step

- Goal: Generate average temperatures per year across many stations

- Map dates and station IDs

```
1 @staticmethod
```

```
2 def map(WeatherDateStat, params):
```

```
3     date = WeatherDateStat.split('/')[....#get date
```

```
    yield (date, WeatherDateStat)
```

Reduce Function Part 1

- Modules are imported in each function. No Globals.

```
@staticmethod
```

```
2 def reduce(StatIDs, out, params):
```

```
3     from disco.util import kvgroup
```

```
import numpy as np
```

```
import ftplib
```

```
import iopro
```

```
7
```

```
    for date, WeatherDateStat in kvgroup(sorted(StatIDs)):
```

```
9         ftp = ftplib.FTP('ftp.ncdc.noaa.gov')
```

```
10          ftp.login()
```

Reduce Function Part 2

- Compute average of several stations across a given year

```
avg_temp = np.empty(0)
```

2

```
3     for file in WeatherDateStat:
```

```
4          cache = open(file.split('/')[-1],'wb')
```

5

```
6         ftp.retrbinary("RETR " + file, cache.write, 8*1024)
```

```
7
```

```
8         cache.close()
```

```
9         adapter = iopro.text_adapter(cache.name,...)
```

```
        avg_temp = np.concatenate((avg_temp,adapter[:]['TEMP']))
```

```
11
```

```
    out.add(date, (avg_temp.mean(),avg_temp.std() ) )
```

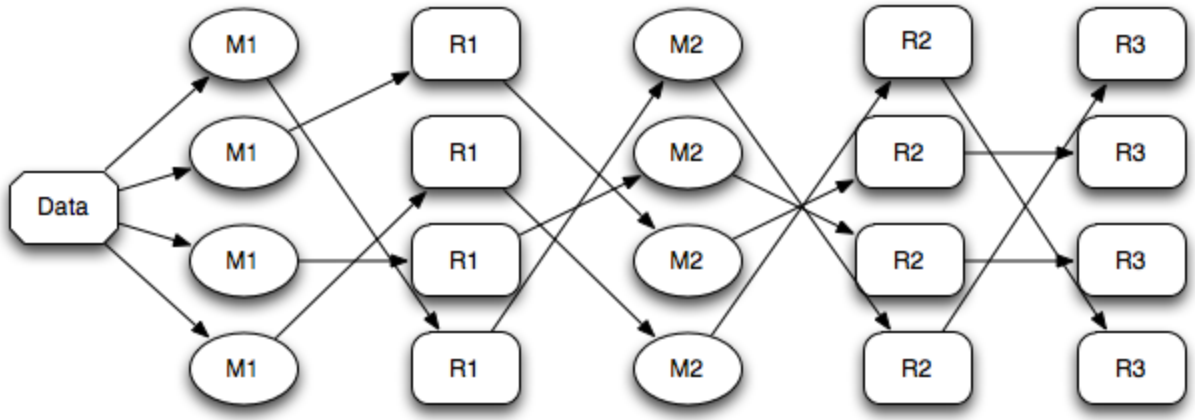
NOAA Data is Messy

- Incomplete Dates

- NA/9999.9

- Stations Don't Persist

Chaining Jobs


Data
M1
M1
M1
M1
R1
R1
R1
R1
M2
M2
M2
M2
R2
R2
R2
R2
R3
R3
R3
R3

Chaining Jobs (cont.)

- Filter list of stations in 1973

- Filter list of stations which persist

- Pass list to original job

MapReduce Final Thoughts

- Data cleansing - no one wants to talk about it but it's everyone's pain point

- Framework which invites thinking about how tasks can be broken up

- good for code management

- hides -- in a good-way -- data management

- Can be inefficient

- Be aware of overheard

- Job Organization

Blaze

- Next generation of NumPy

- Handles out-of-core computations on large datasets.

- *Will* handle data from multiple sources and filesystems

- http://blaze.pydata.org/

Blazing NOAA

```
from blaze import Table, mean, std, params, select, open
```

```
2 from blaze.algo.select import select2
```

3

```
4 adapter = iopro.text_adapter('noaa_gsod_example.op',...)
```

Blazing NOAA (cont.)

```
if not os.path.exists('./noaa_data'):
```

```
p = params(clevel=5, storage='./noaa_data')
```

3

```
4     t = Table([], dshape='{f0: int, f1:int, f2:int, f3:float}', \
```

```
                    params=p)
```

6

```
    t.append(adapter[:])
```

```
    t.commit()
```

```
else:
```

```
10     t = open('ctable://noaa_data')
```

11

```
mean(t, 'TEMP')
```

```
13 std(t, 'TEMP')
```

14

```
qs1 = select(t, lambda x:  20120131 > x > 20110101, 'YEARMODA')
```

```
16
```

```
17 mean(t[qs2],'TEMP')
```

Blaze Future

```
t = open('ctable://noaa_data')
```

```
2 t = open('http://...')
```

```
3 t = open('ftp://...')
```

```
4 t = open('hdfs://...')
```

```
5 t = open('ddfs://...')
```

```
6 ...
```

Closing Thoughts

- MapReduce (Disco)

- Easy To Break up work

- Distribute Computation

- Distribute Data

- Blaze

- Empower Domain Expert

- Empower Algorithm Designers

- Empower Parallelism Researchers

Thank You!

- Blaze

- http://blaze.pydata.org/

- Disco

- http://discoproject.org/

- Continuum

- http://continuum.io/