Parallel and Distributed Joins in H2O

Data by the Bay, San Francisco

16 May 2016

Matt Dowle

True radix sorting

Terdiman, 2000

http://codercorner.com/RadixSortRevisited.htm

Herf, 2001

http://stereopsis.com/radix.html

Dowle & Srinivasan, 2015

http://user2015.math.aau.dk/presentations/234.pdf

Now included in R itself thanks to Michael Lawrence.

But

Single threaded

Single node

Limited to 2 billion rows (2^31)

=> H2O

Every step has now been parallelized and distributed

data.table join

- Find order of the left join columns
- Find order of the right join columns
- Binary merge the two sorted indexes

- No hash table at all
- Fast ordered joins; e.g. rolling forwards, backwards, nearest and limited staleness

https://github.com/Rdatatable/data.table/wiki

Cardinality

Not just the number of rows, but what's in the rows

A 10 billion row file could contain :

- 500 stock tickers (low cardinality)
- Millions of people (medium cardinality)
- Billions of devices (high cardinality)

Create some test data

```
#!/usr/bin/awk -f
nrow = ARGV[2]
while(i<nrow)
{
    printf "%d,%d\n",
        rand()*nrow,
        rand()*nrow*2 - nrow;
    i++;
}
```

1e6 rows; 14MB

```
$ head X
KEY,X2
82967,-9233
550105,-819078
963516,-663146
706905,-128965
766103,774695
```

```
$ head Y
KEY,Y2
610198,322685
872395,-887505
340972,535361
23067,346231
295498,918692
```

1e7 rows; 156MB

```
$ head X
KEY,X2
829673,-92335
5501052,-8190789
9635168,-6631465
7069052,-1289657
7661030,7746956
```

```
$ head Y
KEY,Y2
6101982,3226855
8723957,-8875053
3409724,5353612
230673,3462315
2954985,9186925
```

1e8 rows; 1.8GB

```
$ head X
KEY,X2
8296733,-923350
55010523,-81907897
96351686,-66314650
70690522,-12896576
76610309,77469562
```

```
$ head Y
KEY,Y2
61019825,32268551
87239579,-88750532
34097244,53536122
2306734,34623153
29549857,91869251
```

1e9 rows; 19GB

```
$ head X
KEY,X2
82967333,-9233501
550105235,-819078974
963516860,-663146506
706905226,-128965762
766103099,774695629
```

```
$ head Y
KEY,Y2
610198251,322685514
872395790,-887505326
340972449,535361227
23067343,346231535
295498572,918692512
```

1e10 rows; 200GB

```
$ head X
KEY,X2
829673335,-92335012
5501052357,-8190789743
9635168607,-6631465068
7069052265,-1289657629
7661030994,7746956291
```

```
$ head Y
KEY,Y2
6101982517,3226855142
8723957901,-8875053263
3409724497,5353612273
230673439,3462315357
2954985724,9186925123
```

H2O commands

```
library(h2o)

h2o.init(ip="mr-0xd6", port=55666)

X = h2o.importFile("hdfs://mr-0xd6/datasets/mattd/X1e9_2c.csv")

Y = h2o.importFile("hdfs://mr-0xd6/datasets/mattd/Y1e9_2c.csv")

ans = h2o.merge(X, Y, method="radix")

system.time(print(head(ans)))
```

Scaling

4 node	
800GB/128cpu	
1e6	6s
1e7	7s
1e8	13s
1e9	49s

10 node		
2TB/320cpu		
1e6	11s, 6s	
1e7	6s	
1e8	9s	
1e9	30s	
1e10	10m	<= demo