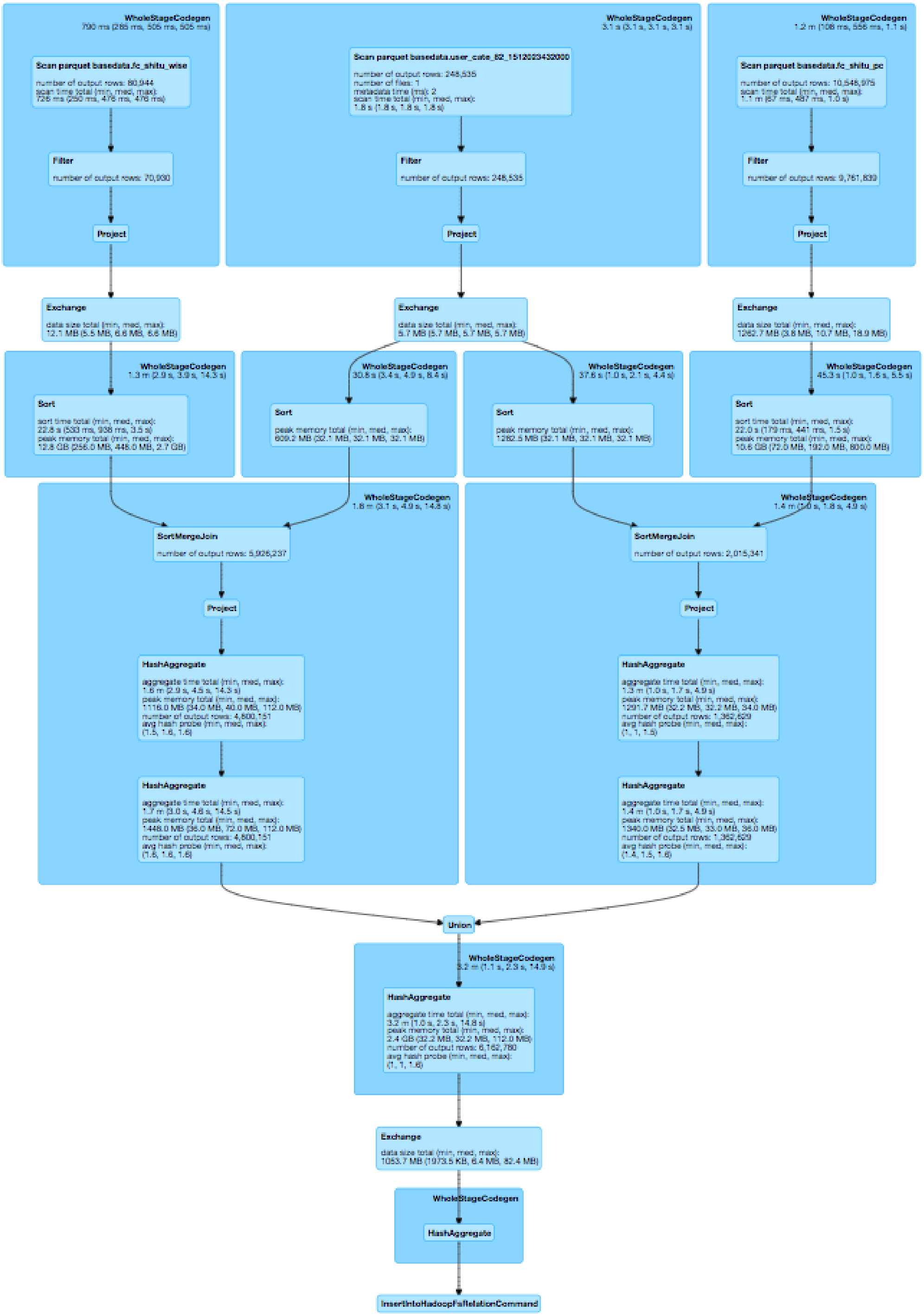
Adaptive Execution in Baidu

This document shares some typical user cases about Spark Adaptive Execution in Baidu. Here we divide them mainly into 3 scenario:

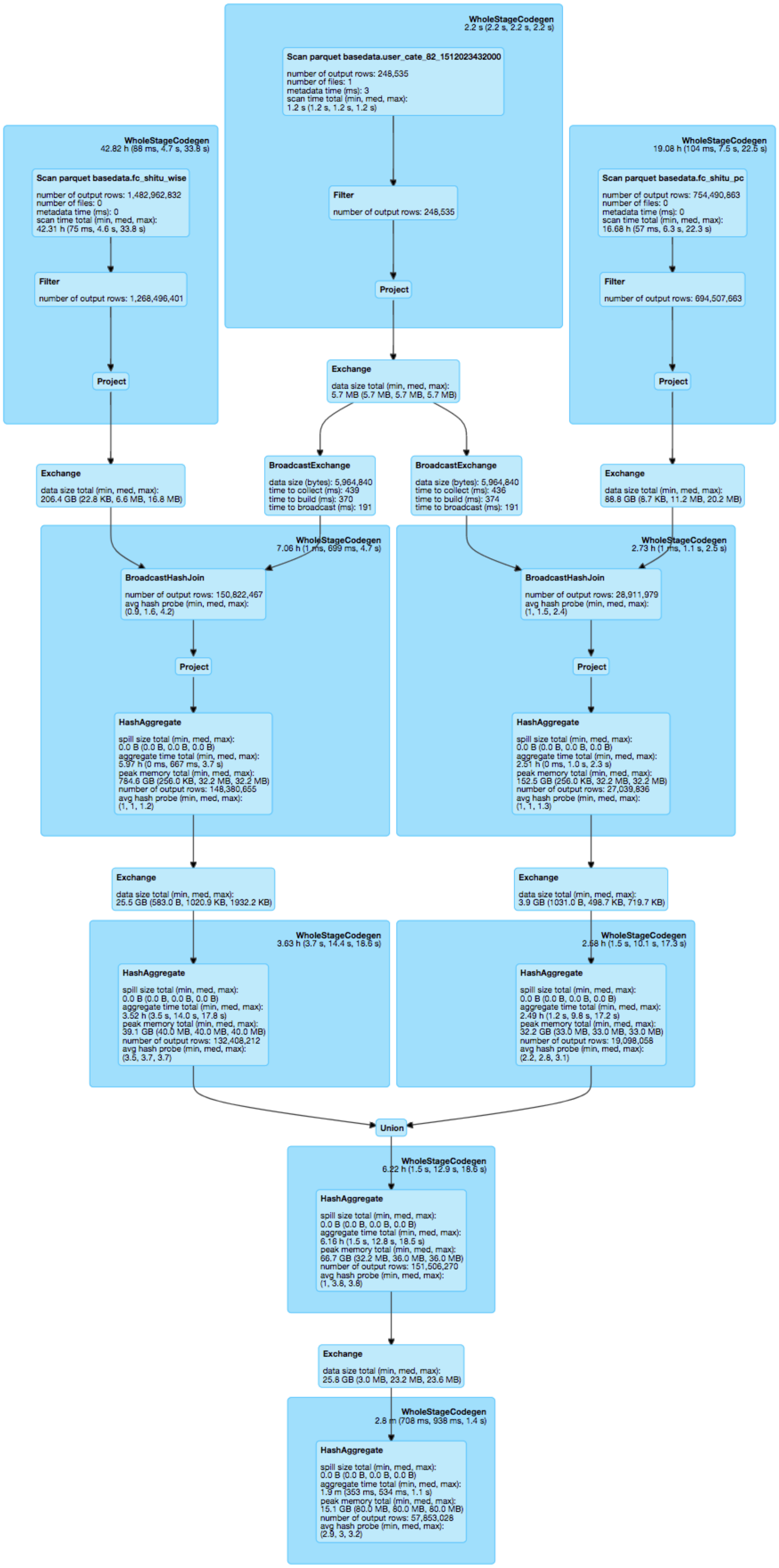
1. SortMergeJoin to BroadcastJoin

The SortMergeJoin transform to BroadcastJoin over deeply tree node can bring us 50% to 200% boosting on query performance, and this strategy alway hit the BI scenario like join several tables with filter strategy in subquery, see the example below:

a) with AE closed


WholeStageCodegen
790 ms (285 ms, 505 ms, 505 ms)
Scan parquet basedata.fc_shitu_wise
number of output rows: 80,944
scan time total (min, med, max):
726 ms (250 ms, 476 ms, 476 ms)
Filter
number of output rows: 70,930
Project
WholeStageCodegen
3.1 s (3.1 s, 3.1 s, 3.1 s)
Scan parquet basedata.user_cate_82_1512023432000
number of output rows: 248,535
number of files: 1
metadata time (ms): 2
scan time total (min, med, max):
1.8 s (1.8 s, 1.8 s, 1.8 s)
Filter
number of output rows: 248,535
Project
WholeStageCodegen
1.2 m (108 ms, 556 ms, 1.1 s)
Scan parquet basedata.fc_shitu_pc
number of output rows: 10,548,975
scan time total (min, med, max):
1.1 m (67 ms, 487 ms, 1.0 s)
Filter
number of output rows: 9,761,839
Project
Exchange
data size total (min, med, max):
12.1 MB (5.5 MB, 6.6 MB, 6.6 MB)
Exchange
data size total (min, med, max):
5.7 MB (5.7 MB, 5.7 MB, 5.7 MB)
Exchange
data size total (min, med, max):
1262.7 MB (3.8 MB, 10.7 MB, 18.9 MB)
WholeStageCodegen
1.3 m (2.9 s, 3.9 s, 14.3 s)
Sort
sort time total (min, med, max):
22.8 s (533 ms, 938 ms, 3.5 s)
peak memory total (min, med, max):
12.8 GB (256.0 MB, 448.0 MB, 2.7 GB)
WholeStageCodegen
30.8 s (3.4 s, 4.9 s, 8.4 s)
Sort
peak memory total (min, med, max):
809.2 MB (32.1 MB, 32.1 MB, 32.1 MB)
WholeStageCodegen
37.6 s (1.0 s, 2.1 s, 4.4 s)
Sort
peak memory total (min, med, max):
1282.5 MB (32.1 MB, 32.1 MB, 32.1 MB)
WholeStageCodegen
45.3 s (1.0 s, 1.6 s, 5.5 s)
Sort
sort time total (min, med, max):
22.0 s (179 ms, 441 ms, 1.5 s)
peak memory total (min, med, max):
10.6 GB (72.0 MB, 192.0 MB, 800.0 MB)
WholeStageCodegen
1.8 m (3.1 s, 4.9 s, 14.8 s)
SortMergeJoin
number of output rows: 5,926,237
Project
HashAggregate
aggregate time total (min, med, max):
1.6 m (2.9 s, 4.5 s, 14.3 s)
peak memory total (min, med, max):
1116.0 MB (34.0 MB, 40.0 MB, 112.0 MB)
number of output rows: 4,800,151
avg hash probe (min, med, max):
(1.5, 1.6, 1.6)
HashAggregate
aggregate time total (min, med, max):
1.7 m (3.0 s, 4.6 s, 14.5 s)
peak memory total (min, med, max):
1448.0 MB (36.0 MB, 72.0 MB, 112.0 MB)
number of output rows: 4,800,151
avg hash probe (min, med, max):
(1.6, 1.6, 1.6)
WholeStageCodegen
1.4 m (1.0 s, 1.8 s, 4.9 s)
SortMergeJoin
number of output rows: 2,015,341
Project
HashAggregate
aggregate time total (min, med, max):
1.3 m (1.0 s, 1.7 s, 4.9 s)
peak memory total (min, med, max):
1291.7 MB (32.2 MB, 32.2 MB, 34.0 MB)
number of output rows: 1,362,629
avg hash probe (min, med, max):
(1, 1, 1.5)
HashAggregate
aggregate time total (min, med, max):
1.4 m (1.0 s, 1.7 s, 4.9 s)
peak memory total (min, med, max):
1340.0 MB (32.5 MB, 33.0 MB, 36.0 MB)
number of output rows: 1,362,629
avg hash probe (min, med, max):
(1.4, 1.5, 1.6)
Union
WholeStageCodegen
3.2 m (1.1 s, 2.3 s, 14.9 s)
HashAggregate
aggregate time total (min, med, max):
3.2 m (1.0 s, 2.3 s, 14.8 s)
peak memory total (min, med, max):
2.4 GB (32.2 MB, 32.2 MB, 112.0 MB)
number of output rows: 6,162,780
avg hash probe (min, med, max):
(1, 1, 1.6)
Exchange
data size total (min, med, max):
1053.7 MB (1973.5 KB, 6.4 MB, 82.4 MB)
WholeStageCodegen
HashAggregate
InsertIntoHadoopFsRelationCommand

b) with AE opened

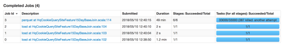
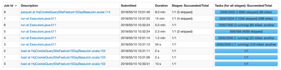


```
WholeStageCodegen
2.2 s (2.2 s, 2.2 s, 2.2 s)

Scan parquet basedata.user_cate_82_1512023432000
number of output rows: 248,535
number of files: 1
metadata time (ms): 3
scan time total (min, med, max):
1.2 s (1.2 s, 1.2 s, 1.2 s)

Filter
number of output rows: 248,535

Project

WholeStageCodegen
42.82 h (88 ms, 4.7 s, 33.8 s)

Scan parquet basedata.fc_shitu_wise
number of output rows: 1,482,962,832
number of files: 0
metadata time (ms): 0
scan time total (min, med, max):
42.31 h (75 ms, 4.6 s, 33.8 s)

Filter
number of output rows: 1,268,496,401

Project

WholeStageCodegen
19.08 h (104 ms, 7.5 s, 22.5 s)

Scan parquet basedata.fc_shitu_pc
number of output rows: 754,490,863
number of files: 0
metadata time (ms): 0
scan time total (min, med, max):
16.68 h (57 ms, 6.3 s, 22.3 s)

Filter
number of output rows: 694,507,663

Project

Exchange
data size total (min, med, max):
5.7 MB (5.7 MB, 5.7 MB, 5.7 MB)

Exchange
data size total (min, med, max):
206.4 GB (22.8 KB, 6.6 MB, 16.8 MB)

BroadcastExchange
data size (bytes): 5,964,840
time to collect (ms): 439
time to build (ms): 370
time to broadcast (ms): 191

BroadcastExchange
data size (bytes): 5,964,840
time to collect (ms): 436
time to build (ms): 374
time to broadcast (ms): 191

Exchange
data size total (min, med, max):
88.8 GB (8.7 KB, 11.2 MB, 20.2 MB)

WholeStageCodegen
7.06 h (1 ms, 699 ms, 4.7 s)

BroadcastHashJoin
number of output rows: 150,822,467
avg hash probe (min, med, max):
(0.9, 1.6, 4.2)

Project

HashAggregate
spill size total (min, med, max):
0.0 B (0.0 B, 0.0 B, 0.0 B)
aggregate time total (min, med, max):
5.97 h (0 ms, 667 ms, 3.7 s)
peak memory total (min, med, max):
784.6 GB (256.0 KB, 32.2 MB, 32.2 MB)
number of output rows: 148,380,655
avg hash probe (min, med, max):
(1, 1, 1.2)

WholeStageCodegen
2.73 h (1 ms, 1.1 s, 2.5 s)

BroadcastHashJoin
number of output rows: 28,911,979
avg hash probe (min, med, max):
(1, 1.5, 2.4)

Project

HashAggregate
spill size total (min, med, max):
0.0 B (0.0 B, 0.0 B, 0.0 B)
aggregate time total (min, med, max):
2.51 h (0 ms, 1.0 s, 2.3 s)
peak memory total (min, med, max):
152.5 GB (256.0 KB, 32.2 MB, 32.2 MB)
number of output rows: 27,039,836
avg hash probe (min, med, max):
(1, 1, 1.3)

Exchange
data size total (min, med, max):
25.5 GB (583.0 B, 1020.9 KB, 1932.2 KB)

Exchange
data size total (min, med, max):
3.9 GB (1031.0 B, 498.7 KB, 719.7 KB)

WholeStageCodegen
3.63 h (3.7 s, 14.4 s, 18.6 s)

HashAggregate
spill size total (min, med, max):
0.0 B (0.0 B, 0.0 B, 0.0 B)
aggregate time total (min, med, max):
3.52 h (3.5 s, 14.0 s, 17.8 s)
peak memory total (min, med, max):
39.1 GB (40.0 MB, 40.0 MB, 40.0 MB)
number of output rows: 132,408,212
avg hash probe (min, med, max):
(3.5, 3.7, 3.7)

WholeStageCodegen
2.58 h (1.5 s, 10.1 s, 17.3 s)

HashAggregate
spill size total (min, med, max):
0.0 B (0.0 B, 0.0 B, 0.0 B)
aggregate time total (min, med, max):
2.49 h (1.2 s, 9.8 s, 17.2 s)
peak memory total (min, med, max):
32.2 GB (33.0 MB, 33.0 MB, 33.0 MB)
number of output rows: 19,098,058
avg hash probe (min, med, max):
(2.2, 2.8, 3.1)

Union

WholeStageCodegen
6.22 h (1.5 s, 12.9 s, 18.6 s)

HashAggregate
spill size total (min, med, max):
0.0 B (0.0 B, 0.0 B, 0.0 B)
aggregate time total (min, med, max):
6.16 h (1.5 s, 12.8 s, 18.5 s)
peak memory total (min, med, max):
66.7 GB (32.2 MB, 36.0 MB, 36.0 MB)
number of output rows: 151,506,270
avg hash probe (min, med, max):
(1, 3.8, 3.8)

Exchange
data size total (min, med, max):
25.8 GB (3.0 MB, 23.2 MB, 23.6 MB)

WholeStageCodegen
2.8 m (708 ms, 938 ms, 1.4 s)

HashAggregate
spill size total (min, med, max):
0.0 B (0.0 B, 0.0 B, 0.0 B)
aggregate time total (min, med, max):
1.9 m (353 ms, 534 ms, 1.1 s)
peak memory total (min, med, max):
15.1 GB (80.0 MB, 80.0 MB, 80.0 MB)
number of output rows: 57,853,028
avg hash probe (min, med, max):
(2.9, 3, 3.2)
```

We can see the both side of SortMergeJoin transform to BoradcastHashJoin， there's no complex config on this scenario, just reused the broad cast threshold. In this case, AE brings us 200%(9min−>3min) boosting on performance improvement.

2. Long running application or use Spark as a service

In this case, long running application refers to the duration of application near 1 hour. Using Spark as a service refers to use spark−shell and keep submit sql or use the service of Spark like Zeppelin, Livy or our internal sql service Baidu BigSQL. In such scenario, all spark jobs share same partition number, so enable AE and add configs about expected task info including data size, row number, min\max partition number and etc, will bring us 50%−100%boosting on performance improvement.

AE Enable	Application Detail (AE will generate extra job to do some metrics collection)	Duration	AE config	Resource Config
False	 Completed Jobs (4) Job Id ▾ · Description · Submitted · Duration · Stages: Succeeded/Total · Tasks (for all stages): Succeeded/Total 3 · parquet at HqCookieQuerySiteFeature15DayBaseJoin.scala:114 · 2018/05/10 12:40:15 · 49 min · 6/6 · 33000/33000 (387 killed: another attempt 2 · load at HqCookieQuerySiteFeature15DayBaseJoin.scala:104 · 2018/05/10 12:40:10 · 2 s · 1/1 · 1/1 1 · load at HqCookieQuerySiteFeature15DayBaseJoin.scala:103 · 2018/05/10 12:40:04 · 2 s · 1/1 · 1/1 0 · load at HqCookieQuerySiteFeature15DayBaseJoin.scala:102 · 2018/05/10 12:38:50 · 1.2 min · 1/1 · 1/1	52 min	null	spark.executor.instances 100 spark.executor.memory 10G spark.executor.cores 4 spark.task.cpus 1
True	 Job Id ▾ · Description · Submitted · Duration · Stages: Succeeded/Total · Tasks (for all stages): Succeeded/Total 8 · parquet at HqCookieQuerySiteFeature15DayBaseJoin.scala:114 · 2018/05/10 10:51:40 · 8.0 min · 1/1 (5 skipped) · 2000/2000 (14890 skipped) (89 killed: 7 · run at Executors.java:511 · 2018/05/10 10:37:25 · 14 min · 1/1 (4 skipped) · 3334/3334 (11556 skipped) (288 killed: 6 · run at Executors.java:511 · 2018/05/10 10:32:24 · 5.0 min · 1/1 · 7000/7000 (3 running) (86 killed: another 5 · run at Executors.java:511 · 2018/05/10 10:32:24 · 2.3 min · 1/1 (2 skipped) · 556/556 (4000 skipped) 4 · run at Executors.java:511 · 2018/05/10 10:31:14 · 1.2 min · 1/1 · 2000/2000 (1 running) (22 killed: another 3 · run at Executors.java:511 · 2018/05/10 10:31:13 · 34 s · 1/1 · 2000/2000 (11 running) (123 killed: another 2 · load at HqCookieQuerySiteFeature15DayBaseJoin.scala:104 · 2018/05/10 10:31:11 · 2 s · 1/1 · 1/1 1 · load at HqCookieQuerySiteFeature15DayBaseJoin.scala:103 · 2018/05/10 10:31:06 · 2 s · 1/1 · 1/1 0 · load at HqCookieQuerySiteFeature15DayBaseJoin.scala:102 · 2018/05/10 10:30:51 · 10 s · 1/1 · 1/1	30 min	spark.sql.adaptive.enabled true spark.sql.adaptive.minNumPostShufflePartitions 400 spark.sql.adaptive.maxNumPostShufflePartitions 10000 spark.sql.adaptive.shuffle.targetPostShuffleInputSize 512M	spark.executor.instances 100 spark.executor.memory 10G spark.executor.cores 4 spark.task.cpus 1

3. GraphFrame jobs

The last scenario is the application use GraphFrame, in this case, user has a 2−dimension graph with 1 billion edges, use the connected componentsalgorithm in GraphFrame. With enabling AE, the duration of app reduce from 58min to 32min, almost 100% boosting on performance improvement.