Hadoop Parallel Execution

Hadoop information:

m4.xlarge
8 vCore, 16 GiB memory, EBS only storage
EBS Storage:32 GiB
1 Master node
2/4/8 Core nodes

Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
CPU(s): 4
On-line CPU(s) list: 0-3
Thread(s) per core: 2
Core(s) per socket: 2
Socket(s): 1
NUMA node(s): 1
Vendor ID: GenuineIntel
CPU family: 6
Model: 79
Model name: Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
Stepping: 1
CPU MHz: 2300.022
BogoMIPS: 4600.04
Hypervisor vendor: Xen
Virtualization type: full
L1d cache: 32K
L1i cache: 32K
L2 cache: 256K
L3 cache: 46080K
NUMA node0 CPU(s): 0-3

Amazon Linux AMI release 2017.03
4.4.35-33.55.amzn1.x86_64
Python 2.7.12

'--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--target=x86_64-amazon-linux-gnu' '--program-prefix=' '--prefix=/usr' '--exec-prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/etc' '--datadir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/lib64' '--libexecdir=/usr/libexec' '--localstatedir=/var' '--sharedstatedir=/var/lib' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--enable-ipv6' '--enable-shared' '--enable-unicode=ucs4' '--with-dbmliborder=gdbm:ndbm:bdb' '--with-system-expat' '--with-system-ffi' '--with-dtrace' '--with-tapset-install-dir=/usr/share/systemtap/tapset' '--with-valgrind' 'build_alias=x86_64-redhat-linux-gnu' 'host_alias=x86_64-redhat-linux-gnu' 'target_alias=x86_64-amazon-linux-gnu' 'CC=gcc' 'CFLAGS=-O2 -g -pipe -Wall -Wp,-D_FORTIFY_SOURCE=2 -fexceptions -fstack-protector --param=ssp-buffer-size=4 -m64 -mtune=generic -D_GNU_SOURCE -fPIC -fwrapv ' 'LDFLAGS= ' 'CPPFLAGS= ' 'PKG_CONFIG_PATH=%{_PKG_CONFIG_PATH}:/usr/lib64/pkgconfig:/usr/share/pkgconfig'

Description of the experiment:

I used the Distributed Grep MapReduce code and the large version of the movielens data set (file ratings.csv) to show the ratings with 5.0 stars.

First I executed the MapReduce code on a cluster with 2, 4 and 8 m4.xlarge instances.

Hadoop cluster:

hadoop jar /usr/lib/hadoop/hadoop-streaming-2.7.3-amzn-3.jar -file /home/hadoop/P11_mapper.py -mapper P11_mapper.py -file /home/hadoop/P11_reducer.py -reducer P11_reducer.py -input ex/ratings.csv -output ex/output

2 instances:

Total time spent by all maps in occupied slots (ms)=25215600
Total time spent by all reduces in occupied slots (ms)=7858176
Total time spent by all map tasks (ms)=525325
Total time spent by all reduce tasks (ms)=818560

4 instances:

Total time spent by all maps in occupied slots (ms)=40564560
Total time spent by all reduces in occupied slots (ms)=68500128
Total time spent by all map tasks (ms)=845095
Total time spent by all reduce tasks (ms)=713543

8 instances:

Total time spent by all maps in occupied slots (ms)=17650800
Total time spent by all reduces in occupied slots (ms)=19722528
Total time spent by all map tasks (ms)=367725
Total time spent by all reduce tasks (ms)=205443

Later I tried tuning two different parameters of the Hadoop configuration on the larger cluster with 8 nodes.

Tuning for cluster with 8 nodes:

hadoop jar /usr/lib/hadoop/hadoop-streaming-2.7.3-amzn-3.jar -D mapreduce.job.maps=5 -file /home/hadoop/P11_mapper.py -mapper P11_mapper.py -file /home/hadoop/P11_reducer.py -reducer P11_reducer.py -input ex/ratings.csv -output ex/output

Total time spent by all maps in occupied slots (ms)=21441072
Total time spent by all reduces in occupied slots (ms)=15565440
Total time spent by all map tasks (ms)=446689
Total time spent by all reduce tasks (ms)=162140

hadoop jar /usr/lib/hadoop/hadoop-streaming-2.7.3-amzn-3.jar -D mapreduce.job.reduces=5 -file /home/hadoop/P11_mapper.py -mapper P11_mapper.py -file /home/hadoop/P11_reducer.py -reducer P11_reducer.py -input ex/ratings.csv -output ex/output

Total time spent by all maps in occupied slots (ms)=46250496
Total time spent by all reduces in occupied slots (ms)=89137728
Total time spent by all map tasks (ms)=963552
Total time spent by all reduce tasks (ms)=928518

Discussion about performance, speed-up and tuning:

In the cluster with 8 nodes time spent by all map tasks and all reduce tasks is significantly shorter than for clusters with lower number of instances. Time spent by all reduce tasks decreases as the number of instances increases.

Tuning different configuration parameters did not lead to performance improvements. Increasing number of map and reduce tasks should improve the performance but it did not occur in this experiment.