|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Experiment | Shared Memory (m1.medium 1GB) | Linux Sort (m1.medium 1GB) | Shared Memory (m1.medium 10GB) | Linux Sort (m1.medium 10GB) | Shared Memory (m1.xlarge 40GB) | Linux Sort (m1.xlarge 40GB) |
| Data Read (GB) | 1GB + 256 MB\*4 + 500MB\* 2(calculation based on 4 thread on a 2 core machine) |  | Did not finish properly |  | Did not have m1.xlarge machine | Did not have m1.xlarge machine |
| Data Write (GB) | 256MB\*4+500MB\*2 |  | Did not finish properly |  | Did not have m1.xlarge machine | Did not have m1.xlarge machine |
| Sort Time (sec) | 148 seconds | 11 seconds | Did not finish properly | 5 minutes 23 seconds | Did not have m1.xlarge machine | Did not have m1.xlarge machine |
| Overall I/O Throughput (MB/sec) | 5GB/148 = 33MB /second |  | Did not finish properly |  | Did not have m1.xlarge machine | Did not have m1.xlarge machine |

Linux sort execution:

================

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ time sort input\_1GB > sorted\_1GB

real 0m11.774s

user 0m11.376s

sys 0m1.880s

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ time sort input\_10GB > sorted\_10^C

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ time sort input\_10GB > sorted\_10GB

real 5m23.301s

user 2m17.692s

sys 0m30.584s

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ valsort sorted\_1GB

Records: 10000000

Checksum: 4c48a881c779d5

Duplicate keys: 0

SUCCESS - all records are in order

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ valsort sorted\_10GB

Records: 100000000

Checksum: 2faf0ab746e89a8

Duplicate keys: 0

SUCCESS - all records are in order

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

TeraSort.java

===========

**With 1GB input**

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ java TeraSort

Picked up \_JAVA\_OPTIONS: -Xms512m -Xmx1024m

Exception in thread "Thread-1" java.lang.OutOfMemoryError: Java heap space

at java.util.Arrays.copyOfRange(Arrays.java:3664)

at java.lang.String.<init>(String.java:207)

at java.io.BufferedReader.readLine(BufferedReader.java:356)

at java.io.BufferedReader.readLine(BufferedReader.java:389)

at java.io.BufferedReader$1.hasNext(BufferedReader.java:571)

at java.util.Spliterators$IteratorSpliterator.tryAdvance(Spliterators.java:1811)

at java.util.stream.ReferencePipeline.forEachWithCancel(ReferencePipeline.java:126)

at java.util.stream.AbstractPipeline.copyIntoWithCancel(AbstractPipeline.java:498)

at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:485)

at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)

at java.util.stream.ForEachOps$ForEachOp.evaluateSequential(ForEachOps.java:151)

at java.util.stream.ForEachOps$ForEachOp$OfRef.evaluateSequential(ForEachOps.java:174)

at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)

at java.util.stream.ReferencePipeline.forEach(ReferencePipeline.java:418)

at FileSplit.run(TeraSort.java:66)

Exception in thread "Thread-3" Exception in thread "Thread-2" java.lang.OutOfMemoryError: GC overhead limit exceeded

java.lang.OutOfMemoryError: GC overhead limit exceeded

at java.util.Arrays.copyOfRange(Arrays.java:3664)

at java.lang.String.<init>(String.java:207)

at java.io.BufferedReader.readLine(BufferedReader.java:356)

at java.io.BufferedReader.readLine(BufferedReader.java:389)

at java.io.BufferedReader$1.hasNext(BufferedReader.java:571)

at java.util.Spliterators$IteratorSpliterator.tryAdvance(Spliterators.java:1811)

at java.util.stream.ReferencePipeline.forEachWithCancel(ReferencePipeline.java:126)

at java.util.stream.AbstractPipeline.copyIntoWithCancel(AbstractPipeline.java:498)

at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:485)

at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)

at java.util.stream.ForEachOps$ForEachOp.evaluateSequential(ForEachOps.java:151)

at java.util.stream.ForEachOps$ForEachOp$OfRef.evaluateSequential(ForEachOps.java:174)

at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)

at java.util.stream.ReferencePipeline.forEach(ReferencePipeline.java:418)

at FileSplit.run(TeraSort.java:66)

Done Writing Thread-0

op1 Started Merging op2

op3 Started Merging op4

op3 Done Merging op4

op1 Done Merging op2

op101 Started Merging op102

op101 Done Merging op102

148.760203577

**cc@hw6-abose2**:**~/gensort/64**$

The above sort is able to complete ¼ of the data successfully before it hits exception. Valsort also works on the intermediate file which is generated from the above execution. Intermediate file name op201, which has close to 250 MB of size.

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ valsort op201

Records: 2500000

Checksum: 13146a2b24bfce

Duplicate keys: 0

SUCCESS - all records are in order

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$ du -h op201

239M op201

**cc@hw6-abose2**:**~/gensort/64**$

**cc@hw6-abose2**:**~/gensort/64**$