An Empirical study on real-world OOM cases in MapReduce jobs

Lijie Xu

Institute of Software, Chinese Academy of Sciences

Abstract: This empirical study aims to understand and summarize the causes of OOM errors in real-world MapReduce jobs. These cases come from StackOverflow, Hadoop mailing list, developer's blogs, and two popular MapReduce books. The two MapReduce books are Data-Intensive Text Processing with MapReduce and MapReduce Design Patterns. The causes and cause distribution are illustrated in the following table.

We totally found 82 cases. The causes of 43 cases have been identified by experts (Hadoop committers or experienced developers). The causes of the left 39 cases are unknown from the job's error description and expert's answers.

Causes / Sources	Improper configuration		Hotspot key	Memory-consuming user code		
	Large framework buffer	Improper data partition	Dynamic hotspot key	Large external data	Large intermediate results	Large accumulated results
StackOverflow	3	1	9 (7)*	2	5	11
Hadoop mailing list			1 (1)			4
Developer's blogs		2	1 (1)			2
MapReduce books			4 (4)	2		9
Total	3	3	15 (13)	4	5	26

9 (7) * means the causes of 9 cases are hotspot key and 7 cases lead to the large accumulated results.

The concrete cases are list below:

1. OOM cases in StackOverflow.com

Large framework buffer

1

Q: CDH 4.1: Error running child : java.lang.OutOfMemoryError: Java heap space

User: I've trying to overcome sudden problem. Befor that problem I've used old VM. I've downloaded the new one VM and still can't make my job run. I get Java heap space error. I've alredy read this one post: out of Memory Error in Hadoop

Expert: lower the buffer size. a compbination of 256 JVM to 128 sort could be your problem. Try an io.sort.mb size of 64mb – One could also try setting the mapred.job.shuffle.input.buffer.percent to 20%. By default, this is set to 70%, which could be a lot if you are working on a very large set of data.

Causes: Large framework buffer

Source code: Stack Trace

asked dec 2 '12 by Sergey

2

Q: Out of memory error in Mapreduce shuffle phase

User: I am getting strange errors while running a **wordcount-like** mapreduce program. I have a hadoop cluster with 20 slaves, each having 4 GB RAM. I configured my map tasks to have a heap of 300MB and my reduce task slots get 1GB. I have 2 map slots and 1 reduce slot per node. Everything goes well until the first round of map tasks finishes. Then there progress remains at 100%. I suppose then the **copy phase** is taking place. Each map task generates something like:

Expert: I think the clue is that the heapsize of my reduce task was required almost completely for the reduce phase. But the **shuffle phase is competing for the same heapspace**, the conflict which arose caused my jobs to crash. I think this explains why the job no longer crashes if I lower the shuffle.input.buffer.percent.

Causes: Large framework buffer

asked oct 10 '13 by DDW

3

A: pig join gets OutOfMemoryError in reducer when mapred.job.shuffle.input.buffer.percent=0.70

User: We're doing a simple pig join between a small table and a big skewed table. We cannot use "using skewed" due to another bug (pig skewed join with a big table causes "Split metadata size exceeded 10000000") :(

If we use the default mapred.job.shuffle.input.buffer.percent=0.70 some of our reducers fail in the shuffle stage:

Expert: As you mentioned, when mapred.job.shuffle.input.buffer.percent=0.30, only 30% heap is used for storing shuffled data, heap is hard to be full.

Causes: Large framework buffer

answered aug 14 '13 by Lijie Xu

Improper data partition

1

Q: Reducer's Heap out of memory

User: So I have a few Pig scripts that keep dying in there reduce phase of the job with the errors that the Java heap keeps running out of space. To this date **my only solution has been to increase Reducer counts**, but that doesn't seem to be getting me anywhere reliable. Now part of this may be just the massive growth in data we are getting, but can't be sure.

Expert: Obviously you are running out of memory somewhere. **Increasing the number of reducers is actually quite reasonable.** Take a look at the stats on the JobTracker Web GUI and see how many bytes are going out of the mapper. Divide that by the number of reduce tasks, and that is a pretty rough estimate of what each reducer is getting. Unfortunately, this only works in the long run if your keys are evenly distributed.

Causes: Improper data partition (small reduce number)

Source code: StackTrace

asked jan 2 '12 by NerdyNick

Hotspot key

1

Q: Reducer's Heap out of memory

Expert: In some cases, JOIN (especially the replicated kind) will cause this type of issue. This happens when you have a "hot spot" of a particular key. For example, say you are doing some sort of join and one of those keys shows up 50% of the time. Whatever reducer gets lucky to handle that key is going to get clobbered. You may want to investigate which keys are causing hot spots and handle them accordingly. In my data, usually these hot spots are useless anyways. To find out what's hot, just do a GROUP BY and COUNT and figure out what's showing up a lot. Then, if it's not useful, just FILTER it out.
Causes: Hotspot key
Source code: StackTrace
asked jan 2 '12 by NerdyNick

2

Q: Hadoop Error: Java heap space

User: I am literally running an empty map and reduce job. However, the job does take in an input that is, roughly, about 100 gigs. For whatever reason, I run out of heap space. Although the job does nothing.
Expert: Note Got it figured out. Turns out I was setting the configuration to have a different terminating token/string. The format of the data had changed, so that token/string no longer existed. **So it was trying to send all 100gigs into ram for one key.**
Causes: Hotspot key.
asked may 7 by Xibz

Large external data

1

Q: OutOfMemory Error when running the wikipedia bayes example on mahout

User: i ran mahout wikipedia example with the 7 gig wiki backup.. , but when testing the classifier, i am getting the a OutOfMemory Error
i have pasted the output below, i set the mahout heap size and java heap size to 2500m
Expert: You need to increase the memory available to mappers. Set mapred.map.java.child.opts to something big enough to hold the model.
It may be that you are trying to load something unrealistically large into memory. he's running out of memory where the mapper side-loads a model. It is not affected by how much overall data goes into the mapper.
Causes: Large external data (training model)
asked apr 9 '12 by Nauman Bashir

2

Q: How to run large Mahout fuzzy kmeans clustering without running out of memory?

User: I am running Mahout 0.7 fuzzy k-means clustering on Amazon's EMR (AMI 2.3.1) and I am running out of memory.
Expert: So 4 * (100000 entries) * (20 bytes/entry) * (128 clusters) = 1.024G. This algorithm is a memory hog.
Causes: Large external data (training data)
asked apr 19 '13 by dfrankow

Large intermediate results

1

Q: Heap error when using custom RecordReader with large file

User: I've written a custom file reader to not split my input files as they are large gzipped files and I want my first mapper job to simply gunzip them. I followed the example in 'Hadoop The Definitive Guide' but I get a heap error when trying to read in to the BytesWritable. I believe this is because the byte array is of size 85713669, but I'm not sure how to overcome this issue.
Expert: In general you can not load whole file into memory of Java VM. You should find some streaming solution to process large files - read data chunk by chunk and save the results w/o fixing in memory whole data set.
Cause: Large intermediate results
Large bye[] in the RecordRead. In other words, large intermediate results are generated for each input record.
Source code: Yes
asked feb 11 '13 by Shane

2

Q: Hadoop Streaming Memory Usage

User: Reading the file from the HDFS and constructing a Text-Object should not amount to more than 700MB Heap - assuming that Text does also use 16-Bit per Character - I'm not sure about that but I could imagine that Text only uses 8-Bit.
So there is these (worst-case) 700MB Line. The Line should fit at least 2x in the Heap but I'm getting always out of memory errors.
Expert: Input File is a 350MByte file containg a single line full of a's.
I'm assuming you file has a single line of all a's with a single endline delimiter.
If that is taken up as a value in the map(key, value) function, I think, you might have memory issues, since, y**ou task have can use only 200MB and you have a record in memory which is of 350MB**.
Causes: Large intermediate results (Large input record)
Source code: None
asked jul 31 '13 by mt_

3

Q: java.lang.OutOfMemoryError on running Hadoop job

User: I have an input file (~31GB in size) containing consumer reviews about some products which I'm trying to lemmatize and find the corresponding lemma counts of. The approach is somewhat similar to the WordCount example provided with Hadoop.
I've 4 classes in all to carry out the processing: StanfordLemmatizer [contains goodies for lemmatizing from Stanford's coreNLP package v3.3.0], WordCount [the driver], WordCountMapper [the mapper], and WordCountReducer [the reducer].
I've tested the program on a subset (in MB's) of the original dataset and it runs fine. Unfortunately, when I run the job on the complete dataset of size ~31GB, the job fails out.
Expert: You need to make the size of the individual units that you are processing (i.e., each Map job in the map-reduce) reasonable. The first unit is the size of document that you are providing to the StanfordCoreNLP's annotate() call. The whole of the piece of text that you provide here will be tokenized and processed in memory. In tokenized and processed form, it is over an order of magnitude larger than its size on disk. So, the document size needs to be reasonable. E.g., you might pass in one consumer review at a time (and not a 31GB file of text!)
Secondly, one level down, the POS tagger (which precedes the lemmatization) is annotating a sentence at a time, and it uses large temporary dynamic programming data structures for tagging a sentence, which might be 3 orders of magnitude larger in size than the sentence. So, the length of individual sentences also needs to be reasonable. If there are long stretches of text or junk which doesn't divide into sentences, then you may also have problems at this level. One simple way to fix that is to use the pos.maxlen property to avoid POS tagging super long sentences.
Causes: Large intermediate results, it uses large temporary dynamic programming data structures for tagging a sentence, which might be 3 orders of magnitude larger in size than the sentence.
Source code: Yes
asked nov 27 '13 by Aditya

4

A: Hadoop Pipes: how to pass large data records to map/reduce tasks

User: I'm trying to use map/reduce to process large amounts of binary data. The application is characterized by the following: the number of records is potentially large, such that I don't really want to store each record as a separate file in HDFS (I was planning to concatenate them all into a single binary sequence file), and each record is a large coherent (i.e. non-splittable) blob, between one and several hundred MB in size. The records will be consumed and processed by a C++ executable. If it weren't for the size of the records, the Hadoop Pipes API would be fine: but this seems to be based around passing the input to map/reduce tasks as a contiguous block of bytes, which is impractical in this case.
Expert: Hadoop is not designed for records about 100MB in size. You will get OutOfMemoryError and uneven splits because some records are 1MB and some are 100MB. By Ahmdal's Law your parallelism will suffer greatly, reducing throughput.
Causes: Large intermediate results (large input record)
answered oct 26 '10 by Spike Gronim

5

Q: Writing a Hadoop Reducer which writes to a Stream

User: I have a Hadoop reducer which throws heap space errors while **trying to produce very long output records**.
Is there a way to write a Reducer to use Streams for output, so that I can run through the data for **the record without marshalling the whole record into memory**?
Expert:
Causes: large intermediate results (the whole record in memory)
Source code: None
asked sep 10 by mcintyre321

Large accumulated results

1

Q: Out of memory due to hash maps used in map-side aggregation

User: MY Hive Query is throwing this exception.
I tried this on 8 EMR core instances with 1 large master instance on 8Gb of data. First i tried with external table (location of data is path of s3). After that i downloaded data from S3 to EMR and used native hive tables. But in both of them i got the same error
Expert: if mapper is not sorting: consider bumping the hash aggregate memory % as indicated in the log to a higher number. if mapper is sorting - just bump up task memory to a larger number. yes, my mapper is sorting.....i tried to debug the problem and it comes out that my mapper was using 5.5 gb of RAM. So, i increased the RAM and it worked. Before increasing the RAM i also tried set hive.map.aggr.hash.percentmemory = 0.25. But it didn't work.
Causes: Large accumulated results (in-memory sort)
asked may 22 '13 by Naresh

2

Q: Getting java heap space error while running a mapreduce code for large dataset

User: am a beginner of MapReduce programming and have coded the following Java program for running in a Hadoop cluster comprising 1 NameNode and 3 DatanNodes :
Each row has an ID followed by 3 comma-separated attributes. My problem is to find the frequency of the value of each attribute(along the column not across the row if the dataset is seen as a 2-D array) of each ID and then sum up the frequencies of each attribute for an ID and find the average.Thus for the above the dataset:
Expert: In your first job you are keeping all the values corresponding to a specific key in a list. As you have 5cr rows and each row have 9 attributes the size of all the values corresponding to a specific key will be too large for a normal List in java to keep in heap memory.That is the reason for java.lang.OutOfMemoryError: Java heap space exception.You have to avoid keeping all the values correponding to a key in an object in java heap.
Causes: Large accumulated results, hotspot key.
asked apr 13 by Monami Sen

3

Q: Memory problems with Java in the context of Hadoop

User: These errors occur when I hash the input records and only then for the moment. During the hashing I have quite many for loops which, produce a lot of temporary objects. For this reason I get the 1) problem. So, I solved the 1) problem by setting K = StringBuilder which is a final class. In other words I reduced the amount of temporary objects by having only few objects which their value, content changes but not themselves.
Expert: Use an ArrayList instead of a LinkedList and it will use a lot less memory.Also I suggest using a HashMap instead of Hastable as the later is a legacy class.
Causes: Large accumulated results, Hotspot key
Source code: None
asked sep 22 '13 by vpap

4

Q: Detailed dataflow in hadoop's mapreduce?

User: am struggling a bit to understand the dataflow in mapreduce. Recently a very demanding job crashed when my disks ran out of memory in the reduce phase. I find it difficult to estimate how much disk my job will need. I will describe the dataflow in detail. My map tasks are very similar to wordcount, so they need little memory. **My reduce tasks work with permutation groups of words. Since identical words need to be joined the reduce function requires a temporary hash map which is always <= 3GB**.
Expert:
Causes: Large accumulated results (in hash map)
asked oct 21 '13 by DDW

5

Q: Mahout Canopy Clustering, K-means Clustering : Java Heap Space - out of memory

User: More then a certain number of canopy centers, **the jobs keep failing with a "Error: Java heap space" message at 67% of reduce phase.**
K-means clustering also has same heap space problems, if the value of K is increasing.

I've heard canopy-center vectors and k-center vectors are held in memory on every mapper and reducer. That would be **num of canopy center(or k) x sparsevector(300000 size) = enough for 4g memory** things, which doesn't seem too bad.
Expert:
Causes: Large accumulated results, hotspot key
asked jun 12 '13 by Nebulach

6

Q: Hadoop UniqValueCount Map and Aggregate Reducer for Large Dataset (1 billion records)

User: a set that has approximately 1 billion data points. There are about 46 million unique data points I want to extract from this. I want to use Hadoop to extract the unique values, but keep getting "Out of Memory" and Java heap size errors on Hadoop - at the same time, I am able to run this fairly easily on a single box using a Python Set (hashtable, if you will.)
and then running the "aggregate" reducer to get the results, which should look like this for the above data set:
Expert: You're probably getting the memory error in the shuffle, remember that Hadoop sorts the keys before starting the reducers. Sort itself is not necessary for most apps, but Hadoop uses this as a way to aggregate all value belonging to a key.
Causes: Large accumulated results (values are cached and sorted in memory)
asked jan 18 '13 by Suman

7

Q: Reducer's Heap out of memory

Expert: Another source of this problem is a Java UDF that is aggregating way too much data. For example, if you have a UDF that goes through a data bag and collects the records into some sort of list data structure, you may be blowing your memory with a hot spot value.
Causes: Large accumulated results, Hotspot key
Source code: None
asked jan 2 '12 by NerdyNick

8

Q: A join operation using Hadoop MapReduce

User: This solution is very good and works for majority of the cases but in my case my issue is rather different. **I am dealing with a data which has got billions of records and taking a cross product of two sets is impossible because in many cases the hashmap will end up having few million objects.** So I encounter a Heap Space Error.
Expert: You should look into how Pig does skew joins. The idea is that if your data contains too many values with the same key (even if there is no data skew), you can create artificial keys and spread the key distribution. This would make sure that each reducer gets less number of records than otherwise. For e.g. if you were to suffix "1" to 50% of your key "K1" and "2" the other 50% you will end with half the records on the reducer one (1K1) and the other half goes to 2K2.
Causes: Large accumulated results, hotspot key
asked may 19 '13 by Eastern Monk

9

Q: Building Inverted Index exceed the Java Heap Size

User: I am building an inverted index for large data set (One day worth of data from large system). The building of inverted index get executed as a map reduce job on Hadoop. Inverted index is build with the help of scala. Structure of the inverted index as follows: {key:"New", ProductID:[1,2,3,4,5,...]} these get written in to avro files.
During this process I run into Java Heap Size issue. I think the reason is that terms like "New" as I showed above contain large number of productId(s). I have a, rough idea where the issue could take place in my Scala code:
When I execute the map reduce job for small data set I don't get the error. Which means that as the data increase number of items/product_ids that I index for words like New or old etc get bigger which cause the Heap Size to overflow.
So, the question is how can avoid java heap size overflow and accomplish this task.
I don't have a exact number it varies because some key words only return 1 or 2 items but some may return 100000

Expert: you might have to do the operation **in several passes so you don't accumulate too much data in memory** (it looks to me like you have all product IDs for all keys in memory at the same time).
Causes: Large accumulated results, hotspot key
asked jul 31 '13 by Null-Hypothesis

10

Q: HBase: How to handle large query results

User: In my current approach, my function scans for the records and gets them just fine. I then try to convert the ResultScanner contents into ArrayList form so I can pass them to the calling function. I could just pass the Scanner, but I want to close it.
My problem is when the **ArrayList gets filled to about 2 million records**, I get an OutOfMemoryError:
Expert:
Causes: Large accumulated results
asked jul 21 by Bryany

11

Q: OOM exception in Hadoop Reduce child

User: I am getting OOM exception (Java heap space) for reduce child. In the reducer, I am appending all the values to a StringBuilder which would be the output of the reducer process. The number of values aren't that many.
I had a set accumulating all the values - so that there are no duplicate values. I removed it later on.
Some sample sizes of iterator are as follows: 238695, 1, 13, 673, 1, 1 etc. These are not very large values. Why do I keep getting the OOM exception? Any help would be valuable to me.
Expert: 238695 seems pretty excessive - what's the typical length of the values being accumulated? If you really want to do this, and not get the OOM error then you'll need to look into a custom output format so you don't accumulate the values in memory
So for your example, you want to output the values for a particular key as a space separated list of the values (as the output key), and an empty text as the output value.
Causes: Large accumulated results, Hotspot key
asked oct 11 '12 by Raghava

2. OOM cases from Hadoop mailing list

1

Subject: java.lang.OutOfMemoryError while running Pig Job

User: I am running following Pig script in Pig 0.8 version
filter_pe = FILTER page_events BY is_iab_robot == 0 AND is_pattern_match_robot == 0 AND day == '2011-05-10'; select_pe_col = FOREACH filter_pe GENERATE day, is_cookied_user, anon_cookie, reg_cookie, referrer_id, has_web_search_phrase, business_unit_id; select_ref_col = FOREACH referrer_group_map GENERATE referrer_id, has_web_search_phrase, referral_type_id; jn = JOIN select_ref_col BY (referrer_id, has_web_search_phrase), select_pe_col BY (referrer_id, has_web_search_phrase);

Expert: The stack trace shows that the OOM error is happening when the distinct is being applied. It looks like in some record(s) of the relation group_it, one more of the following bags is very large - logic.c_users, logic.nc_users or logic.registered_users;
Causes: Large accumulated results, hotspot key

2

Subject: java.lang.OutOfMemoryError when using TOP udf

User: The immediate cause of the problem I had (failing without erroring out) was slightly different formatting between the small and large input sets. Duh. When I fixed that, I did indeed get OOM due to the nested distinct. I tried the workaround you suggested Jonathan using two groups, and it worked great! In a separate run I also tried SET pig.exec.nocombiner true; and found that worked also, and the runtime was the same as using the two group circumlocution.
Expert: I took a look and it is being reasonable. I do that that that is the issue: the way that **it works is by holding a priority queue of however many items you care about, adding one, then popping the bottom one. If it has to hold almost 3M objects in memory, memory issues is a real likely thing.** A couple things you can do: - have fewer columns. ie only do "TOP" of the things you really care about - more memory (don't you love that?)
Causes: Large accumulated results

3

Subject: 0.9.1 out of memory problem

User: I'm having an out of memory problem that seems rather weird to me. Perhaps you can help me.
bySite = GROUP wset BY site; report = FOREACH bySite { duids = DISTINCT wset.duid; GENERATE group, COUNT(duids), SUM(wset.replicas), SUM(wset.nbfiles), SUM(wset.rnbfiles), SUM(wset.length), SUM(wset.rlength); }; STORE report INTO 'testfile.out';

If I omit the COUNT(DISTINCT), it works brilliantly and fast. With the COUNT(DISTINCT) it dies like this. Now, I don't know where to go from here. I'm running Hadoop and Pig with default settings, except I've increased child.opts to -Xmx1024M (24GB machines) so it would be great if you could tell me what to do, because I'm stuck.
Expert: Nested distincts are dangerous. They are not done in a distributed fashion, they have to be loaded into memory. So that is what is killing it, not the order/limit. The alternative is to do two groups, first group by (citeddocid,CitingDocids) to get the distinct and then by citeddocid. to get the count
Causes: Large accumulated results

4

Subject: Set number Reducer per machines.
User: I was using the following file for mapreduce job.
Cloud9/src/dist/edu/umd/cloud9/example/cooccur/ComputeCooccurrenceMatrixStripes.java
I am trying to run a job on my hadoop cluster, where I get consistently get heap space error. I increased the heap-space to 4 GB in hadoop-env.sh and reboot the cluster. However, I still get the heap space error.
Expert: It looks like both the mapper and reducer are using a map structure which will be created on the heap. **All the values from the reducer are being inserted into the map structure. If you have lots of values for a single key then you're going to run out of heap memory really fast.** Do you have a rough estimate for the number of values per key? We had this problem when we first started using map-reduce (we'd create large arrays in the reducer to hold data to sort). Turns out this is generally a very bad idea (it's particularly bad when the number of values per key is not bounded since sometimes you're algorithm will work and other times you'll get out of memory errors). In our case we redesigned our algorithm to not require holding lots of values in memory by taking advantage of Hadoop's sorting capability and secondary sorting capability.
Causes: Large accumulated results

3. OOM cases in developer's blogs

1

Subject: MapReduce Algorithm - in Map Combining
User/Expert: One common solution to limiting memory usage when using the in-mapper combining technique is to "block" input key-value pairs and "flush" in-memory data structures periodically. The idea is simple: instead of emitting intermediate data only after every key-value pair has been processed, emit partial results after processing every n key-value pairs. This is straightforwardly implemented with a counter variable that keeps track of the number of input key-value pairs that have been processed. As an alternative, the mapper could keep track of its own memory footprint and flush intermediate key-value pairs once memory usage has crossed a certain threshold. In both approaches, either the block size or the memory usage threshold needs to be determined empirically: with too large a value, the mapper may run out of memory, but with too small a value, opportunities for local aggregation may be lost.
Causes: Large accumulated results

2

Subject: Efficient Sharded Positional Indexer
User/Expert: For frequent terms such as "the", the reducer output record may exceed the memory limit of the JVM, resulting in out of memory error. This is because Hadoop keeps the whole record (in this case the whole postings list for "the") in memory before sending it to disk. Partitioning the collection into more shards helps, but it's a suboptimal hack. One way to avoid such error is to partition these large postings into manageable sized chunks, and output several records for the same key (the word "the"). E.g. record1: <key="the", value=<<"doc1", tf, positions>, <"doc2", tf, pos> ...<"doc1000", tf, pos>>>, record2: <key="the", value=<<"doc1001", tf, pos>, <"doc1002", tf, pos> ...<"doc2000", tf, pos>>>, ..., recordN.
Causes: Large accumulated results, hotspot key

3

Subject: Reducers fail with OutOfMemoryError while copying Map outputs
User: Hi , I am using M7,Reducers fail while copying Map outputs with following exception:
View Diagnostics:
Error: java.lang.OutOfMemoryError: Java heap space at
org.apache.hadoop.mapred.ReduceTask$ReduceCopier$MapOutputCopier.shuffleInMemory(ReduceTask.java:1774) at
org.apache.hadoop.mapred.ReduceTask$ReduceCopier$MapOutputCopier.getMapOutputFromFile(ReduceTask.java:1487) at
org.apache.hadoop.mapred.ReduceTask$ReduceCopier$MapOutputCopier.copyOutput(ReduceTask.java:1361)
Expert: What I see is that in the second map-reduce, you have 8 reducers, of which 7 completed successfully. 1 reducer failed, presumably this is the out-of-memory task.

Since this sounds like a problem that depends on the volume of input, I suspect that something about your query has trouble with a very large number of items being sent to a single reducer. This can happen a number of different ways, but the problem of skew in data volumes to reducers is a very common one. There are often clever tricks to avoid the OOM error that this can cause.
Causes: Improper data partition

4

Subject: OutOfMemoryError in ReduceTask shuffleInMemory

User/Expert: We were able to capture a heap dump of one reduce task. The heap contained 8 byte arrays **that were 127 MB each.** These byte arrays were all referenced by their own DataInputBuffer. Six of the buffers were referenced by the linked lists in ReduceTask$ReduceCopier.mapOutputsFilesInMemory. **These six byte arrays consume 127 MB * 6 = 762 MB of the heap**. Curiously, this 762 MB exceeds the 717 MB limit. The ShuffleRamManager.fullSize = 797966777 = 761MB, so something is a bit off in my original value of 717... But this is not the major source of trouble.
Causes: Improper data partition (small reduce number)

4. OOM cases in two MapReduce books

4.1 OOM cases in MapReduce Design Patterns

1

Case: Median and standard deviation [page 25]

User/Expert: The easiest way to perform these operations involves copying the list of values into a temporary list in order to find the median or iterating over the set again to determine the standard deviation. With large data sets, this implementation may result in Java heap space issues, because each value is copied into memory for every input group.
Any sort of memory growth in the reducer has the possibility of blowing through the Java virtual machine's memory. For example, if you are collecting all of the values into an ArrayList to perform a median, that ArrayList can get very big. This will not be a particular problem if you're really looking for the top ten items, but if you want to extract a very large number you may run into memory limits.
Causes: Large accumulated results, hotspot key

2

Case: Replicated Join [page 119]

User/Expert: A replicated join is an extremely useful, but has a strict size limit on all but one of the data sets to be joined. All the data sets except the very large one are essentially read into memory during the setup phase of each map task, which is limited by the JVM heap.
Causes: Large external data

3

Case: Cogroup in Pig [page 75]

User/Expert: The next major concern is the possibility of **hot spots in the data that could result in an obscenely large record.** With large data sets, it is conceivable that a particular output record is going to have a lot of data associated with it. Imagine that for some reason a post on StackOverflow has a million comments associated with it. That would be extremely rare and unlikely, but not in the realm of the impossible.
Causes: Large accumulated results and hotspot key

4

Case: Sort XML Object [page 75]

User/Expert: If you are building some sort of XML object, all of those comments at one point might be stored in memory before writing the object out. This can cause you to blow out the heap of the Java Virtual Machine, which obviously should be avoided.
Causes: Large accumulated results

5

Case: ParserUserData [page 122]

User/Expert: During the setup phase of the mapper, the user data is read from the DistributedCache and stored in memory. Each record is parsed and the user ID is pulled out of the record. Then, the user ID and record are added to a HashMap for retrieval in the map method. This is where an out of memory error could occur, as the entire contents of the file is stored, with additional overhead of the data structure itself. If it does, you will either have to increase the JVM size or use a plain reduce side join.
Causes: Large external data

4.2 OOM in Data-Intensive Text Processing with MapReduce

1

Case: Reduce-side Join [page 66]

User/Expert: All the tuples from S with the same join key will be encountered first, which the reducer can buffer in memory. As the reducer processes each tuple from T, it is crossed with all the tuples from S. Of course, we are assuming that the tuples from S (with the same join key) will fit into memory, which is a limitation of this algorithm (and why we want to control the sort order so that the smaller dataset comes first).

Causes: Large accumulated results

2

Case: BuildInvertedIndex [page 77]

User/Expert: The inverted indexing algorithm presented in the previous section serves as a reasonable baseline. However, there is a significant scalability bottleneck: the algorithm assumes that there is sufficient memory to hold all postings associated with the same term. Since the basic MapReduce execution framework makes no guarantees about the ordering of values associated with the same key, the reducer first buffers all postings (line 5 of the reducer pseudo-code in Figure 4.2) and then performs an in-memory sort before writing the postings to disk.7 For efficient retrieval, postings need to be sorted by document id. However, as collections become larger, postings lists grow longer, and at some point in time, reducers will run out of memory.

inverted indexing is nothing but a very large distributed sort and group by operation! We began with a baseline implementation of an inverted indexing algorithm, but quickly noticed a scalability bottleneck that stemmed from having to buffer postings in memory. Application of the value-to-key conversion design pattern (Section 3.4) addressed the issue by offloading the task of sorting postings by document id to the MapReduce execution framework.

A two-pass solution that involves first buffering the postings (in memory) would suffer from the memory bottleneck we've been trying to avoid in the first place.

Causes: Large accumulated results

3

Case: One-to-man join [page 65]

User/Expert: one-to-many join. Assume that tuples in S have unique join keys (i.e., k is the primary key in S), so that S is the "one" and T is the "many". The above algorithm will still work, but when processing each key in the reducer, we have no idea when the value corresponding to the tuple from S will be encountered, since values are arbitrarily ordered. The easiest solution is to buffer all values in memory, pick out the tuple from S, and then cross it with every tuple from T to perform the join. However, as we have seen several times already, this creates a scalability bottleneck since we may not have sufficient memory to hold all the tuples with the same join key.

Causes: Large accumulated results, hotspot key

4

Case: word co-occurrence [page 59]

User/Expert: The computation of the word co-occurrence matrix is quite simple if the entire matrix fits into memory—however, in the case where the matrix is too big to fit in memory, a naïve implementation on a single machine can be very slow as memory is paged to disk. Although compression techniques can increase the size of corpora for which word co-occurrence matrices can be constructed on a single machine, it is clear that there are inherent scalability limitations. We describe two MapReduce algorithms for this task that can scale to large corpora.

This algorithm will indeed work, but it suffers from the same drawback as the stripes approach: as the size of the corpus grows, so does that vocabulary size, and at some point there will not be sufficient memory to store all co-occurring words and their counts for the word we are conditioning on.

Causes: Large accumulated results

5

Case: SortByKey [page 61]

User/Expert: However, since MapReduce makes no guarantees about the ordering of values associated with the same key, the sensor readings will not likely be in temporal order. The most obvious solution is to buffer all the readings in memory and then sort by timestamp before additional processing. However, it should be apparent by now that any in-memory buffering of data introduces a potential scalability bottleneck. What if we are working with a high frequency sensor or sensor readings over a long period of time? What if the sensor readings themselves are large complex objects? This approach may not scale in these cases—the reducer would run out of memory trying to buffer all values associated with the same key.

Causes: Large accumulated results

6

Case: In-memory combining [page 54]

User/Expert: For both algorithms, the in-mapper combining optimization discussed in the pre- vious section can also be applied; the modification is sufficiently straightforward that we leave the implementation as an exercise for the reader. However, the above caveats remain: there will be far fewer opportunities for partial aggregation in the pairs ap- proach due to the sparsity of the intermediate key space. The sparsity of the key space also limits the effectiveness of in-memory combining, since the mapper may run out of memory to store partial counts before all documents are processed, necessitating some mechanism to periodically emit key-value pairs (which further limits opportunities to perform partial aggregation). Similarly, for the stripes approach, memory management will also be more complex than in the simple word count example. **For common terms, the associative array may grow to be quite large**, necessitating some mechanism to periodically flush in-memory structures.
Causes: Large accumulated results, hotspot key

5. Cases that their causes are unknown

1

Q: Mahout - Exception: Java Heap space

User: I'm trying to convert some texts to mahout sequence files using:
mahout seqdirectory -i Lastfm-ArtistTags2007 -o seqdirectory
But all I get is a OutOfMemoryError, as here:
Expert: add heap size
Causes: Probably Large external data (training data)
asked apr 7 by user3422072

2

Q: OutofMemoryError when reading a local file via DistributedCache

User: However, when I execute it in the Hadoop cluster (fully distributed mode), I get an "OutOfMemoryError: Java heap space"
Expert:
Causes: Probably Large external data (read into HashMap)
asked nov 19 '12 by iporto

3

A: out of Memory Error in Hadoop

User: I tried installing Hadoop following thishttp://hadoop.apache.org/common/docs/stable/single_node_setup.html document. When I tried executing this
bin/hadoop jar hadoop-examples-*.jar grep input output 'dfs[a-z.]+'
Expert: add heap size
Causes: Probably Large framework buffer
answered nov 16 '12 by etlolap

4

Q: Hadoop searching words from one file in another file

User: I want to build a hadoop application which can read words from one file and search in another file.
If the word exists - it has to write to one output file If the word doesn't exist - it has to write to another output file
I tried a few examples in hadoop. I have two questions
Two files are approximately 200MB each. Checking every word in another file might cause out of memory. Is there an alternative way of doing this?
Expert:
Causes: Probably Large external data (cached file)
asked jan 24 '10 by Boolean

5

A: Hadoop example job fails in Standalone mode with: “Unable to load native-hadoop library”

User: I'm trying to get the simplest Hadoop "hello world" setup to work, but when I run the following command:
hadoop jar /usr/share/hadoop/hadoop-examples-1.0.4.jar grep input output 'dfs[a-z.]+'
Expert:
Causes: Probably Large framework buffer (map buffer)
answered dec 1 '12 by Lorand Bendig

6

Q: Java Heap space error in Hadoop

User: I am currently reading file of 50 MB in reducer setup step successfully, But when file is larger than that approx(500MB) it gives me "out of memory error".
Expert: Increase the heap size
Causes: Unknown
Source code: None
asked 14 hours ago by Sanjay Bhosale

7

Q: Java heap space error while running hadoop

User: I am trying to run hadoop-examples.jar-1.2.1 from hadoop examples. I am using 64 -bit Linux system. No i have not tried that. The memory i set up for the virtual machine is IGB. Each time I run the job it shows Java heap space exception and the job fails.
Expert:
Causes: Unknown
Source code: None
asked sep 6 by NeethuPL

8

Q: Out of heap error when creating Index in Apache Hive

User: when we execute the “alter” statement to actually build it this fails with "java.lang.OutOfMemoryError: Java heap space". For a partitioned table the index for each partition seems to be built individually so in this case about 1500 separate jobs are launched (we have tried both to let them run one at a time and several in parallel with the same result) and a large number of jobs actually seem to run as they should but after a while we start to see jobs failing and after this point most of the remaining jobs actually fail.
Expert:
Causes: Unknown
Source code: None
asked aug 26 by Magnus Eriksson

9

Q: Error: Java heap space

User: running the hadoop example :
$bin/hadoop jar hadoop-examples-1.0.4.jar grep input output 'dfs[a-z.]+'
In log, I am getting the error.
Expert: add the heap size
Causes: Unknown
Source code: Yes
asked mar 25 '13 by Senthil Porunan

10

Q: hadoop mapper over consumption of memory(heap)

User: I wrote a simple hash join program in hadoop map reduce. The idea is the following:
A small table is distributed to every mapper using DistributedCache provided by hadoop framework. The large table is distributed over the mappers with the split size being 64M. **The setup code of the mapper creates a hashmap reading every line from this small table. In the mapper code, every key is searched(get) on the hashmap, and if the key exists in the hash map it is written out.** There is no need of a reducer at this point of time. This is the code which we use:

While testing this code, **our small table was 32M, and large table was 128M**, one master and 2 slave nodes.
This code fails with the above inputs when I have a 256M of heap. I use -Xmx256m in the mapred.child.java.opts in mapred-site.xml file. When I increase it to 300m it proceeds very slowly and with 512m it reaches its max throughput.
Expert:
Causes: Unknown (probably large external data)
Source code: Yes
asked mar 9 '13 by mc_87

11

Q: increase jvm heap space while runnig from hadoop unix

User: I am running a java class test.java from hadoop command :
$ hadoop test
I am using a stringBuilder, and its size is going out of memory :
Expert:
Causes: Unknown
Source code: StackTrace
asked aug 31 '13 by naveen kumar

12

Q: PIG using HCatLoader, Java heap space error

User: When I try to load a table created in hive into pig using HCatloader, it is giving the Java heap space error.The detailes are as follow
Expert:
Causes: Unknown
Source code: StackTrace
asked apr 12 by Dorababu G

13

Q: Error running child : java.lang.OutOfMemoryError: Java heap space

User: I have read a lot on the internet, but found no solution for my problem. I use Hadoop 2.6.0.
The main goal for the *MapReduce* is to run through a *SequenceFile* and do some analysis on the key/value pairs.
Expert:
Causes: Unknown
Source code: StackTrace
asked 14 hours ago by Christian D.

14

Q: Hadoop: Can you silently discard a failed map task?

User: I am processing large amounts of data using hadoop MapReduce. The problem is that, ocassionaly, a corrupt file causes Map task to throw a java heap space error or something similar.
Expert:
Causes: Unknown
Source code: None
asked jan 9 '14 by miljanm

15

Q: Querstion regarding hadoop-env.sh

User: I am Facing Error: Java heap space and Error: GC overhead limit exceeded
So i started looking into hadoop-env.sh.
so thats what i understand so far, Please correct me if i am wrong.
Expert:
Causes: Unknown
Source code: None
asked jun 27 by user2950086

16

Q: how to change mapper memory requirement in hadoop?

User: In a map-reduce job, i got the error "java.lang.OutOfMemoryError: Java heap space". Since I get this error in a mapper function; I thought that when I lower the input size to the mapper I will have no more error, so I changed the mapred.max.split.size to a much more lower value.
Then, I started the job again and i saw that "number of mapper tasks to be executed" has increased, so i thought that lowering mapred.max.split.size was a good idea: more mappers with lower memory requirements.
BUT, I got the "java.lang.OutOfMemoryError: Java heap space" error again, again and again.
Causes: Unknown
Source code: None
asked sep 11 '13 by ndemir

17

Q: Hadoop conf to determine num map tasks

User: I have a job, like all my Hadoop jobs, it seems to have a total of 2 map tasks when running from what I can see in the Hadoop interface. However, this means it is loading so much data that I get a Java Heap Space error.
I've tried setting many different conf properties in my Hadoop cluster to make the job split into more tasks but nothing seems to have any effect.
I have tried setting mapreduce.input.fileinputformat.split.maxsize, mapred.max.split.size, dfs.block.size but none seem to have any effect.
Expert:
Causes: Unknown
Source code: None
asked jul 23 '13 by Katie

18

Q: Mahout on Elastic MapReduce: Java Heap Space

User: I'm running Mahout 0.6 from the command line on an Amazon Elastic MapReduce cluster trying to canopy-cluster ~1500 short documents, and the jobs keep failing with a "Error: Java heap space" message.
Expert:
Causes: Unknown (Probably large external data)
Source code: Yes, and StackTrace
asked apr 29 '12 by David M.

19

A: Java Heap Space Error in running mahout item similarity job on Amazon EMR

User: I am trying to run a mahout item similarity job on a input consists of ~250 Million Pairs(row) in a Amazon EMR Cluster(m3.2xLarge,10 core nodes).I am facing Java Heap Size error while running the similarity job.
Expert:
Causes: Unknown
Source code: Yes
answered jan 6 by ChristopherB

20

Q: Mahout on EMR Error: Java heap space

User: I've ran a clustering job on EMR. The dataset is huge. Everything worked well until:
Expert:
Causes: Unknown
asked sep 29 '12 by denys

21

Q: How to handle unsplittable 500 MB+ input files in hadoop?

User: However, I noticed that hadoop works very poorly with output values that can sometimes be really big (700 MB is the biggest I've seen). In various places in the MapReduce framework, **entire files are stored in memory, sometimes twice or even three times.** I frequently encounter out of memory errors, even with a java heap size of 6 GB.

Expert:

Causes: Unknown (Probably Large external data or Large intermediate results)

asked may 25 by Michael

22

Q: Hive Issue - java.lang.OutOfMemoryError: Java heap space

User: I have started hive server and i am trying to login to Hive and run a basic command. Below is snapshot of the issue.

Expert:

Causes: Unknown

asked apr 22 by user3528338

23

A: hive collect_set crashes query

User: I've got the following table:

where I remove the collect_set command. So my question: Has anybody an idea why collect_set might fail in this case?

Expert: This is probably the memory problem, since collect_set aggregates data in the memory.

Causes: Unknown

answered jan 14 '14 by Nigel Tufnel

24

Q: MapReduce jobs in hive-0.8.1-cdh4.0.1 Failed.

User: Queries in **hive-0.8.1-cdh4.0.1** that invoke the Reducer results in Task Failed. The queries having MAPJOIn is working fine but JOIN gives error.

Expert:

Causes: Unknown

asked oct 15 '12 by Jickson T George

25

Q: how to determine the size of “mapred.child.java.opts” and HADOOP_CLIENT_OPTS in mahout canopy

User: I’ve got a "dictionary.file-0" file and its size is about 50M.

Expert:

Causes: Unknown

asked dec 6 by user3156087

26

Q: setting hadoop job configuration programmatically

User: I am getting OOM exception (Java heap space) for reduce child. I read in the documentation that increasing the value of mapred.reduce.child.java.opts to -Xmx512M or more would help. Since I am not the admin, I cannot change that value in mapred-site.xml. I would like to set that value only for my job through the java program. I tried setting it using Configuration class as follows, but that didn't work.

Expert:

Causes: Unknown

asked oct 10 '12 by Raghava

27

Q: Is it possible to intervene if a task fails?

User: I have a mapreduce job running on many urls and parsing them. I need way to handle a scenario in which one parsing task crashes on a fatal error like OOM error. In the normal hadoop behaivour a task is retried for a defined number of time and than the job fails. the problem is with urls that are corrupted in some way causing this error. These urls will fail in all the retries.

Expert:

Causes: Unknown

asked may 8 '12 by user1251654

28

Q: Limit CPU / Stack for Java method call?

User: I am using an NLP library (Stanford NER) that throws OOM errors for rare input documents.
I plan to eventually isolate these documents and figure out what about them causes the errors, but this is hard to do (I'm running in Hadoop, so I just know the error occurs 17% through split 379/500 or something like that). As an interim solution, I'd like to be able to apply a CPU and memory limit to this particular call.

Expert:

Causes: Unknown

asked jul 4 '09 by Kevin Peterson

29

Q: Hadoop YARN Map Task running out of physical and virtual memory

User: have the following method that I run from my map task in a multithreaded execution , however this works fine in a standalone mod e, but when I runt this in Hadoop YARN it runs out of the physical memory of 1GB and the virtual memory also shoots up.

Expert:

Causes: Unknown

asked jan 7 '14 by user1965449

30

Q: Why the identity mapper can get out of memory?

User: After some hours of researching and trial and error I realized that the machines I provisioned for the TASK group were small instances with not much memory and, more interestingly, that the point in which I was running out of memory was during shuffling instead of mapping.

Expert:

Causes: Unknown (Probably improper data partition)

asked sep 6 '12 by sortega

31

Q: Java Heap space error in Hadoop

User: My problem is with hadoop heap memory. I am currently reading file of 50 MB in reducer setup step successfully, But when file is larger than that approx(500MB) it gives me "out of memory error". When i browse through http:/ip_address:50070/dfshealth.html#tab-overview it gives me below information.

Expert: The reason why you use something like hadoop is because you cant fit the entire data set into memory. Either you don't change the logic and try to find a computer that's big enough or you parallel-ize the algorithm and exploit hadoop.

Causes: Unknown (Probably large external data)

asked 15h ago by Sanjay Bhosale

32

Q: Will reducer out of java heap space

User: My question is how to deal with java out of space problem, I added some property configuration into xml file, but it didn't work. Increasing number of reducers doesn't work for me either. Because in my program every reducer needs large sparse whole matrix, and I am not allowed to change this logic. Yet every reducer will receive an entry with column id as key, and column vector as value.Is there any way I can get out of this dilemma?

Expert:

Causes: Unknown

asked apr 20 '13 by shirley

33

A: running an elementary mapreduce job with java on hadoop

User: The assignment is to run:

bin/hadoop jar hadoop-cookbook-chapter1.jar chapter1.WordCount input output
And this is the response that I get:
Expert: add heap size
Causes: Unknown (Probably large framework buffer)
answered may 1 '13 by greedybuddha

34

Q: Apparent memory-leak in hadoop

User: I'm running on what should be very small data sets in an initial trial, so I shouldn't be hitting any memory limit. More to the point I don't want to change the hadoop configuration; if the program can't run with the current configuration the program needs rewritten.
Can anyone help me figure out how to diagnose this issue? ise there a command line argument to get a stack trace of memory usage? any other way of tracking this issue?
Expert:
Causes: Unknown
asked nov 30 '12 by dsollen

35

Q: Hadoop: Heap space and gc problems

User: My algorithm works fine for small datasets, but for a medium dataset has heap space problems. My algorithm reaches a certain tree level and then it goes out of heap space, or has gc overhead problems. At that point, i made some calculations and i saw that every task doesnt need more than 100MB memory. So for 8 tasks, i am using about 800MB of memory. I don't know what is going on. I even updated my hadoop-env.sh file with these lines:
Expert: add heap size
Causes: unknown
asked mar 14 '12 by jojoba

36

A: Hadoop JobClient: Error Reading task output

User: I had a similar problem and was able to find a solution. The problem lies on how hadoop deals with smaller files. In my case, I had about 150 text files that added up to 10MB. Because of how the files are "divided" into blocks the system runs out of memory pretty quickly.
Expert:
Causes: Unknown
answered mar 15 by Enrique

37

A: Pig: Hadoop jobs Fail

User: I have a pig script that queries data from a csv file.
The script has been tested locally with small and large .csv files.
In Small Cluster: It starts with processing the scripts, and fails after completing 40% of the call
The error is just, Failed to read data from "path to file"
What I infer is that, The script could read the file, but there is some connection drop, a message lose
But I get the above mentioned error only.
Expert:
Causes: Unknown
answered dec 17 by Paulo Fidalgo

38

Q: Mahout on Elastic MapReduce: Java Heap Space

User: I'm running Mahout 0.6 from the command line on an Amazon Elastic MapReduce cluster trying to canopy-cluster ~1500 short documents, and the jobs keep failing with a "Error: Java heap space" message.
Expert:
Causes: Unknown

39

Subject: New type of JOIN specialized for filtering

User: In many cases I am close to be able to use a replicated join (100-150 MB of data) but it still blows up despite upping the Java heap to a few GB. e.g.
A = LOAD 'bigdata'
B = LOAD 'smalldata'
C = FOREACH B GENERETE key1, key2 -- < 150 MB
D = JOIN A BY (key1, key2), C BY (key1, key2) USING 'replicated'
....
java.lang.OutOfMemoryError: Java heap space

I looked at the code of the POFRJoin and its primary goal is to JOIN some extra columns. Would it make sense to have a new type of join for efficiently doing some filtering on the map side? (something similar to LookupInFiles but working transparently with Pig relations) Do better techniques exist already?
Expert: Seems like it would be more memory-efficient to pull the group keys out of the hashmap value:
e.g.:
x: key, a, b
y: key, c, d
join x by key, y by key using 'replicated';

Current FRJoin:
construct a hashmap of { key --> (key, a, b) }
Proposed optimization:
construct a hashmap of { key --> (a, b) }, keep track of where key should
be inserted to reconstruct the tuple.

That *almost* gets us where you suggest -- a pure filtering join would let us drop the whole hashmap and replace with a hashset -- but it also lets us avoid some extra complexity so that may be a good thing. I think giving users too many types of joins may be a bad thing.

If we implemented the IN operator, we could do what you suggest without the complexity overhead.. IN could take a relation or bag of single-valued tuples, or a list of scalars, and for all 3 cases it would build up an in-memory hashset.
Causes: Unknown