Real-world OOM Errors in Hadoop MapReduce Applications

Lijie Xu

Institute of Software, Chinese Academy of Sciences

Abstract: This study aims to summarize root causes and fix patterns of OOM errors in real-world Hadoop MapReduce applications. These cases come from StackOverflow.com, 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 summarized causes and cause patterns are illustrated in the following table.

We totally found 151 cases. The causes of 56 cases have been identified by experts (Hadoop committers, experienced developers, or the book authors), users themselves or us in the reproduced errors. The causes of the left 95 cases are unknown from the users' error descriptions or the expert's answers.

TABLE I: DISTRIBUTION OF THE OOM ERRORS

Framework	**Sources**	**Row code**	**Pig**	**Hive**	**Mahout**	**Cloud9**	**Total**
Hadoop	StackOverflow.com	20	4	2	4	0	30
	Hadoop mailing list	5	5	1	0	1	12
	Developers' blogs	2	1	0	0	0	3
	MapReduce books	8	3	0	0	0	11
Total	**All sources**	**35**	**13**	**3**	**4**	**1**	**56**

TALBE II: OOM CAUSE PATTERNS

Category	**Pattern**	**Pattern description**	**Hadoop**	**Ratio**
Improper job configurations	Large framework buffers	Large intermediate data stored in buffers	6	10%
	Improper data partition	Some data partitions are extremely large	3	5%
	Subtotal		**9**	**15%**
Data skew	Hotspot key	Large (k, list(v))	15	28%
	Large single record	Large (k, v)	6	10%
	subtotal		**21**	**38%**
Memory-consuming user code	Large external data	User code loaded large external data	8	15%
	Large intermediate results	Large computing results are generated during processing a record	4(3)	7%
	Large accumulated results	Large computing results are accumulated in user code	30[13]	53%
	Subtotal		**42**	**75%**
Total			**56+16**	**128%**

Notation: 4(3) means 3 out of the 4 errors also have the *large single record* cause pattern. 30[13] means 13 out of the 30 errors also have the *Hotspot key* cause pattern.

Contents

1. Large framework buffers (6)

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

User: I've trying to overcome sudden problem. Before 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 already read this one post: out of Memory Error in Hadoop
Expert: lower the buffer size. a combination 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.

Job type: User-defined (StackOverflow)
Causes: Large framework buffer (identified by expert, Reproduced)
Fix suggestions: lower buffer size (accepted)
Fix details: Set io.sort.mb from 128MB to 64MB (fixed)
Hadoop version: Cloudera Hadoop 4.1

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: I think this explains why **the job no longer crashes if I lower the shuffle.input.buffer.percent**.
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.

Job type: User-defined (StackOverflow)
Causes: Large framework buffer (identified by user, Reproduced)
Fix suggestions: lower buffer size (user)
Fix details: Lower shuffle.input.buffer.percent from 0.7 to 0.2 (fixed)
Hadoop version: 1.2.1

3. Q: 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.

Job type: Pig (StackOverflow)
Causes: Large framework buffer (identified by user, Reproduced)
Fix suggestions: lower buffer size (user), use "skewed"
Fix details: Change shuffle.input.buffer.percent from 0.7 to 0.3 (fixed), although in 2 hours.
Hadoop version: unknown

4. 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.]+'

Job type: User-defined (StackOverflow)
Causes: Large framework buffer (reproduced)
Fix suggestions: lower buffer size (us)
Fix details: Change io.sort.mb from 400MB to 200MB (fixed by us)
Hadoop version: 1.0.0

5. 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

Job type: User-defined (StackOverflow)
Causes: Large framework buffer (reproduced)
Fix suggestions: lower buffer size (us)
Fix details: Change io.sort.mb from 500MB to 200MB (fixed by us)
Hadoop version: unknown

6. Out of heap space errors on TTs

User: I am running hive and I am trying to join two tables (2.2GB and 136MB) on a cluster of 9 nodes (replication = 3)
Expert: by default it will be 200mb. But your io.sort.mb(300) is more than that. So, configure more heap space for child tasks.

Job type: Apache Hive (Mailing list)
Causes: Large framework buffer (Expert)
Fix suggestions: lower buffer size (Expert)
Fix details: Lower io.sort.mb (from 300MB, suggestion)
Hadoop version: 0.20.2

3. Improper data partition (3)

1. 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.

Job type: Pig (Mailing list)
Causes: Improper data partition (Expert, Reproduced)
Fix suggestions: no
Hadoop version: MapR M7

2. 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.

Job type: Pig script (StackOverflow)
Causes: Improper data partition (user)
Fix suggestions: add reduce number and lower framework buffer (User)
Fix details: increase the reduce number (partially fixed)
Pig version: 0.8.1

3. 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.

Job type: User-defined (Mailing list)
Causes: Improper data partition (us, Reproduced)
Fix suggestions: add reduce number (us)
Fix details: increase the reduce number (fixed by us)
Hadoop version: 0.20.2

4. Hotspot key (2 errors here + 13 errors in Large accumulated results)

1. Q: Reducer's Heap out of memory

Expert: In some cases, JOIN (especially the replicated kind) will cause this type of issue. T**his 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.

Job type: User-defined (StackOverflow)
Causes: Hotspot key (Expert)
Fix suggestions: Filter the useless the hot key (Expert)
Fix details: Skip the hotspot records (fixed)
Pig version: 0.8.1

2. 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.

Job type: Pig (Book, MapReduce Design Patterns)
Causes: Hotspot key (Expert)
Fix suggestions: no
Pig version: unknown

5. Large single record (3 errors here + 3 errors in Large intermediate results)

1. 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, **you task have can use only 200MB and you have a record in memory which is of 350MB**.

Job type: User-defined (StackOverflow)
Causes: Large single record (Expert)
Fix suggestions: no
Hadoop version: unknown

2. 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.

Job type: User-defined (StackOverflow)
Causes: Large single record (Expert)
Fix suggestions: break up the large record into smaller records (Expert)
Fix details: Your first map task must break up the data into smaller records for further processing. Further tasks then operate on the smaller records. If you really can't break it up, make your map reduce job operate on file names. The first mapper gets some file names, runs them thorough your mapper C++ executable, stores them in more files. The reducer is given all the names of the output files, repeat with a reducer C++ executable. This will not run out of memory but it will be slow. (suggestion)
Hadoop version: unknown

3. OutOfMemory Error

User: The key is of the form "ID :DenseVector Representation in mahout with dimensionality size = 160k"
For example: C1:[,0.00111111, 3.002, ... 1.001,....]
So, typical size of the key of the mapper output can be 160K*6 (assuming double in string is represented in 5 bytes)+ 5 (bytes for C1:[]) + size required to store that the object is of type Text
Yeah. That was the problem. And Hama can be surely useful for large scale matrix operations

Expert: I guess vector size seems too large so it'll need a distributed vector architecture (or 2d partitioning strategies) for large scale matrix operations.

Job type: User-defined (Mailing list)
Causes: Large single record (Expert)
Fix suggestions: no
Hadoop version: 0.17.1

6. Large external data (8)

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)

Job type: Apache Mahout (StackOverflow)
Causes: Large external data (large training model) (Expert, Reproduced)
Fix suggestions: Add memory space (Expert)
Hadoop version: unknown

2. A: Hive: Whenever it fires a map reduce it gives me this error "Can not create a Path from an e...

User: I have solved the problem.
I looked up the log file and in my case **the table is an external table referring to a directory located on hdfs.** This directory contains more than 300000 files. So **while reading the files it was throwing an out of memory exception** and may be for this reason it was getting an empty string and throwing 'Can not create a Path from an empty string' exception.
I tried with a smaller subset of files and it worked.

Job type: Apache Hive (StackOverflow)
Causes: Large external data (large external table) (User)
Fix suggestions: decrease the dataset (User)
Fix details: I tried with a smaller subset of files and it worked
Hive version: 0.10

3. 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.

Job type: Pig join (Book, MapReduce Design Pattern)
Causes: Large external data (large table for join) (Expert, reproduced)
Fix suggestions: use reduce join (Expert)
Interesting: Yes
Hadoop version: 1.0.3

4. 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, t**he 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.

Job type: Pig join (Book, MapReduce Design Pattern)
Causes: Large external data (user data) (Expert)
Fix suggestions: no
Hadoop version: 1.0.3

5. 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

Job type: Apache Mahout (StackOverflow)
Causes: Large external data (large model) (us, reproduced)
Fix suggestions: no
Hadoop version: 1.2.1, Mahout version: 0.9

6. 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.

Job type: User-defined (StackOverflow)
Causes: Large external data (external table) (us, reproduced)
Fix suggestions: no
Hadoop version: unknown

7. 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:
Job type: Apache Mahout (StackOverflow)
Causes: Large external data (external table) (us, reproduced)
Fix suggestions: no
Mahout version: 0.6

8. OutOfMemoryError of PIG job (UDF loads big file)

User: I am running a hadoop job written in PIG. It fails from out of memory because a UDF function consumes a lot of memory, it loads a big file. What are the settings to avoid the following OutOfMemoryError?
Expert:

Job type: Apache Pig (Mailing list)
Causes: Large external data (User)
Fix suggestions: no
Hadoop version: unknown

7. Large intermediate results (4)

1. 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. T**he 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.

Job type: User-defined with third library (StackOverflow)
Causes: Large intermediate results + large single record (Expert & Reproduced)
Fix suggestions: split long records to multiple small records (Expert, accepted)
Fix details: the document size needs to be reasonable, avoid POS tagging super long sentences, If there are long stretches of text or junk which doesn't divide into sentences (suggestions)
Hadoop version: 0.18.0

2. 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**?

Job type: User-defined (StackOverflow)
Causes: Large intermediate results (User)
Fix suggestions: no
Hadoop version: unknown

3. 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.
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.**

Job type: User-defined (StackOverflow)
Causes: Large intermediate results + Large single record (User)
Fix suggestions: Split the large single record
Fix details: 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 (fixed)
Hadoop version: 2.2

4. 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.

Job type: User-defined (StackOverflow)
Causes: Large intermediate results + Large single record (Expert)
Fix suggestions: read data chunk by chunk, avoid load the whole file (Expert)
Fix details: 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 This specific task - unzip is probably not suited for the MR since there is no logical division of data into records. (suggestion)

8. Large accumulated results (30)

1. 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.

Job type: User-defined (StackOverflow)
Causes: Large accumulated results + Hotspot key (Expert, Reproduced)
Fix suggestions: avoid accumulation (Expert)
Fix details: You have to avoid keeping all the values correponding to a key in an object in java heap.
System requirement: memory+disk data structures
Hadoop version: unknown

2. 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.

Job type: User-defined (StackOverflow)
Causes: Large accumulated results + Hotspot key (User and Expert, reproduced)
Fix suggestions: Redesign the key, use "skew join" (Expert) create artificial keys and spread the key distribution
Fix details: Redesign the key: suffix "1" to 50% of your key "K1" and "2" the other 50%. (create artificial keys and spread the key distribution), could some kind of sampling algorithm. (suggestion)
Hadoop version: unknown

3. Q: Detailed dataflow in hadoop's mapreduce?

User: I 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**.

Job type: User-defined (StackOverflow)
Causes: Large accumulated results (User)
Fix suggestions: no
Hadoop version: unknown

4. 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).

Job type: User-defined (StackOverflow)
Causes: Large accumulated results + Hotspot key (Expert, Reproduced)
Fix suggestions: Job split, do the operation in several passes (Expert)
Fix details: do the operation in several passes (suggestion)
Hadoop version: unknown

5. 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
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.
Expert: Out of memory due to hash maps used in map-side aggregation. 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.

Job type: Apache Hive (StackOverflow)
Causes: Large accumulated results (Expert, Reproduced)
Fix suggestions: lower hive.map.aggr.hash.percentmemory (Expert)
Hadoop version: unknown

6. 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.

Job type: Pig UDF (StackOverflow)
Causes: Large accumulated results (Expert)
Fix suggestions: no
Pig version: 0.8.1

7. 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. For your example, your mappers will end up writing a lot of times the same values, while you only care about

how many uniques you have for a given key. Add a combiner in your job that will run just after the mapper but before the reducer, and will only keep uniques before sending to the reducer. Modify your mapper to keep a mapping of what you already sent, and not send if you've already sent this mapping before.

Job type: User-defined (StackOverflow)
Causes: Large accumulated results + Hotspot key (Expert)
Fix suggestions: add combiner to reduce the values for a key, in-memory combing (Expert)
Fix details: Add a combiner in your job that will run just after the mapper but before the reducer.
Hadoop version: maybe 0.21.0

8. 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.

Job type: Apache Mahout (StackOverflow)
Causes: Large accumulated results (User)
Fix suggestions: no (Enlarge memory limit cannot work)
Mahout version: 0.7

9. 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:

Job type: User-defined (StackOverflow)
Causes: Large accumulated results (User)
Fix suggestions: no
System requirement: disk-based data structure
Hadoop version: unknown

10. 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.

Job type: User-defined (StackOverflow)
Causes: Large accumulated results + Hotspot key (Expert, Reproduced)
Fix suggestions: multiple outputs (Expert) (Enlarge memory limit cannot work)
Hadoop version: unknown

11. 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?)

Job type: Pig script (Mailing list)
Causes: Large accumulated results (Expert)
Fix suggestions: Filter the unrelated columns (TOP)
Fix details: have fewer columns. ie only do "TOP" of the things you really care about
Hadoop version: unknown

12. 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;

Job type: Pig script (Mailing list)
Causes: Large accumulated results (Expert, Reproduced)
Fix suggestions: Try setting the property pig.cachedbag.memusage to 0.1 or lower (-Dpig.cachedbag.memusage=0.1 on java command line). It controls the memory used by pig internal bags, including those used by distinct. If that does not work, you can try computing count-distinct for each type of user separately and then combining the result. (suggestion)
Pig version: 0.8

13. 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.

Job type: Cloud9 (Mailing list)
Causes: Large accumulated results (Expert, Reproduced)
Fix suggestions: Change sort by using Hadoop secondary sort (Expert)
Fix details: redesigned our algorithm to not require holding lots of values in memory by taking advantage of Hadoop's sorting capability and secondary sorting capability (fixed)
Hadoop version: unknown

14. Subject: ORDER ... LIMIT failing on large data

User: I have a small pig script that outputs the top 500 of a simple computed relation. It works fine on a small data set but fails on a larger (45 GB) data set. I don't see errors in the hadoop logs (but I may be looking in the wrong places). On the large data set the pig log shows. 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!
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

Job type: Pig script (Mailing list)
Causes: Large accumulated results (Expert, Reproduced)
Fix suggestions: two groups (Expert)
Fix details: use two groups (fixed)
Pig version: 0.8.1

15. 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.

Job type: User-defined (developer's blog)
Causes: Large accumulated results (User, Reproduced)
Fix suggestions: Map aggregation => Reduce aggregation, avoid in-memory aggregation (us)
Fix details: spill the accumulated results periodically or achieve a certain threshold. Emit partial results after processing every n key-value pairs
Hadoop version: unknown

16. 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.

Job type: User-defined (developer's blog)
Causes: Large accumulated results + Hotspot key (User, Reproduced)
Fix suggestions: output several records for the same key (us)
Fix details: spill the accumulated results periodically. Output several records (e.g., 1000) for the same key
Hadoop version: unknown

17. 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.**

Job type: User-defined (book, MapReduce Design Patterns)
Causes: Large accumulated results (Expert, Reproduced)
Fix suggestions: no
Interesting: Yes
Hadoop version: 1.0.3

18. 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.

Job type: User-defined (book, MapReduce Design Patterns)
Causes: Large accumulated results (Expert)
Fix suggestions: no
Hadoop version: 1.0.3

19. 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).

Job type: User-defined (book, Data-Intensive Text Processing with MapReduce)
Causes: Large accumulated results (Expert)
Fix suggestions: control the sort order (Expert) (tricky method)
Cloud9 version: 2.0

20. 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.

There is a simple solution to this problem. Since the execution framework guarantees that keys arrive at each reducer in sorted order, one way to overcome the scalability bottleneck is to let the MapReduce runtime do the sorting for us. Instead of emitting key-value pairs of the following type:

Job type: User-defined (book, Data-Intensive Text Processing with MapReduce)
Causes: Large accumulated results (Expert)
Fix suggestions: Use framework sort mechanism (Expert) (fixed)
Interesting: Yes
Cloud9 version: 2.0

21. 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.

Job type: User-defined (book, Data-Intensive Text Processing with MapReduce)
Causes: Large accumulated results (Expert)
Fix suggestions: no
Cloud9 version: 2.0

22. 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.**

The insight lies in properly sequencing data presented to the reducer. If it were possible to somehow compute (or otherwise obtain access to) the marginal in the reducer before processing the joint counts, the reducer could simply divide the joint counts by the marginal to compute the relative frequencies.

Job type: User-defined (book, Data-Intensive Text Processing with MapReduce)
Causes: Large accumulated results (Expert)
Fix suggestions: no
Cloud9 version: 2.0

23. 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.

Job type: User-defined (book, Data-Intensive Text Processing with MapReduce)
Causes: Large accumulated results + Hotspot key(Expert)
Fix suggestions: no
Cloud9 version: 2.0

24. Case: In-memory combining [page 54]

User/Expert: For both algorithms, the in-mapper combining optimization discussed in the previous 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 approach 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.

Job type: User-defined (book, Data-Intensive Text Processing with MapReduce)
Causes: Large accumulated results + Hotspot key(Expert)
Fix suggestions: no
Cloud9 version: 2.0

25. Q: Fail to join large groups

I have two data structures with the same structure:

{(id, (record1, record2, record3))}
I want to join them, by on the value of record1. In order to do that I wrote this script:

```
data_1_group = group data_1 by $1.record1;
data_2_group = group data_2 by $1.record1;
jj = join data_1_group by group, data_2_group by group;
```

But, since the both data_1 and data_2 contains millions of records while record1 can assume only 20 different values, the groups are very large and the script runs out of memory and fails.

Job type: Pig (StackOverflow)
Causes: Large accumulated results + Hotspot key (us, reproduced)
Fix suggestions: no

Hadoop version: unknown

26. Hashing two relations

User: I know that map-side join does this (on pre-partitioned data), but I want to do it on reduce side. Using job-chaining, I can output (hash(key), value) by two map tasks on the two input files, but when it comes to the reduce stage, i have to take the same partition from both the hash tables. I am not sure how can I accomplish this. get a partition from each
Expert: If you want to do the join in reduce side, MapReduce framework enable this by grouping all the matching tuples together. Why bother to build hash table to buffer the entire partition in memory? This probably brings you a out-of-memory error. The default reduce join should be your choice in this case

Job type: User-defined (Mailing list)
Causes: Large accumulated results (Expert)
Fix suggestions: no
Hadoop version: unknown

27. OutOfMemory during Plain Java MapReduce

User: during the Reduce phase or afterwards (i don't really know how to debug it) I get a heap out of Memory Exception. I guess this is because the value of the reduce task (a Custom Writable) holds a List with a lot of user ids.
I had a look to the stacktrace and it says the problem is at the reducer:
userSet.add(iterator.next().toString());

Expert: When you implement code that starts memory-storing value copies for every record (even if of just a single key), things are going to break in big-data-land. Practically, post-partitioning What if we are working with a high frequency sensor or sensor readings over a long period of time? if you really really want to do this, or use an alternative form of key-value storage where updates can be made incrementally (Apache HBase is such a store, as one example).

This has been discussed before IIRC, and if the goal were to store the outputs onto a file then its better to just **directly serialize them with a file opened instead of keeping it in a data structure and serializing it at the end**. The caveats that'd apply if you were to open your own file from a task are described at

Looking at your reducer code, it appears that you are trying to compute the distinct set of user IDs for a given reduce key. Rather than computing this by holding the set in memory, **use a secondary sort of the reduce values**, then while iterating over the reduce values, look for changes of user id. Whenever it changes, write out the key and the newly found value.

Job type: User-defined (Mailing list)
Causes: Large accumulated results + Hotspot Key (Expert, reproduced)
Fix suggestions: Write the computing results continuously, user a secondary sort (Expert)
Fix details: You'd probably need to write out something continuously if you really really want to do this, or **use an alternative form of key-value storage where updates can be made incrementally** (Apache HBase is such a store, as one example).
Hadoop version: unknown

28. how to solve reducer memory problem?

User: I have a map reduce program that do some matrix operations. in the reducer, it will average many large matrix(each matrix takes up 400+MB(said by Map output bytes). so if there 50 matrix to a reducer, then the total memory usage is 20GB. so the reduce task got exception
one method I can come up with is use Combiner to save sums of some matrixs and their count but it still can solve the problem because the combiner is not fully controled by me.
Expert: In your implementation, you Could OOM as you **store more and more data into "TrainingWeights result"**. So the question is for each "Reducer group", or "Key", how many data it could be?
If a key could contain big values, then all these values will be saved in the memory of "result" instance. That will require big memory. If so, either you have to have that much memory, or redesign your key, make it more lower level, so requires less memory
Job type: User-defined (Mailing list)
Causes: Large accumulated results + Hotspot Key (Expert)
Fix suggestions: Use combiner, redesign the key (User) (suggestions)
Hadoop version: unknown

29. memoryjava.lang.OutOfMemoryError related with number of reducer?

User: I can fix this by changing heap size. But what confuse me is that when i change the reducer number from 24 to 84, there's no this error. Seems i found the data that causes the error, but i still don't know the exactly reason.
I just do a group with pig latin:
The group key (custid, domain, level, device) is significantly skewed, about 42% (58,621,533 / 138,455,355) of the records are the same key, and only the reducer which handle this key failed.

Expert: When you increase the number of reducers they each have less to work with provided the data is distributed evenly between them - in this case about one third of the original work. It is essentially the same thing as increasing the heap size - it's
just distributed between more reducers.

It is because of the nested distinct operation relies on the RAM to calculate unique values.

Job type: Apache Pig (Mailing list)
Causes: Large accumulated results - Count(distinct) + Hotspot Key (Expert)
Fix suggestions: Add reduce number (User)
Fix details: increase reducer number
Hadoop version: unknown

30. Q: Why does the last reducer stop with java heap error during merge step

User: I keep increasing the number of reducers and I see that while all except one reducers run quickly and finish their job, one last reducer just hangs at the merge step with this message in its tasktracker log:
Yes, you are right. So I am wondering what is the solution for this.
Expert: My guess is that you have a single key with a huge number of values and the following line in your reducer is causing you problems: You can probably confirm this by putting in some debug and inspecting the logs for the reducer that never ends:

Job type: User-defined (StackOverflow)
Causes: Large accumulated results + Hotspot Key (Expert)
Fix suggestions: no
Hadoop version: unknown

11. Errors that their root causes are unknown (95)

1. 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:

Job type: User-defined (StackOverflow)
Causes: Unknown
Hadoop version: 1.0.3

2. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

3. 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

Job type: User-defined (StackOverflow)
Causes: Unknown (maybe large external data)

4. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

5. 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:
Job type: Apache Hive (StackOverflow)
Causes: Unknown

6. 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
Job type: User-defined (StackOverflow)
Causes: Unknown

7. 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

8. 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:

Job type: Apache Pig (StackOverflow)
Causes: Unknown

9. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

10. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

11. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

12. 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.
Job type: User-defined (StackOverflow)
Causes: Unknown

13. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

14. 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:
Job type: Apache Mahout (StackOverflow)
Causes: Unknown

15. 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:
Job type: Apache Mahout (StackOverflow)
Causes: Unknown

16. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

17. 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:
Job type: Apache Hive (StackOverflow)
Causes: Unknown

18. 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.
Job type: Apache Hive (StackOverflow)
Causes: Unknown

19. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

20. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

21. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

22. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

23. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

24. 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:

Job type: User-defined (StackOverflow)
Causes: Unknown

25. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

25. 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 parallelize the algorithm and exploit hadoop.
Job type: User-defined (StackOverflow)
Causes: Unknown (Maybe large external data)

26. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

27. 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
Job type: User-defined (StackOverflow)
Causes: unknown

28. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

29. 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:
Job type: Apache Pig (StackOverflow)
Causes: Unknown

30. 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.

Job type: Apache Pig (Mailing list)
Causes: Unknown

31. 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.
Job type: Apache Mahout (StackOverflow)
Causes: Unknown

32. 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.
Job type: User-defined (StackOverflow)
Causes: Unknown

33. 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:
Job type: Apache Pig (Mailing list)
Causes: Unknown

34. 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:
Job type: User-defined (StackOverflow)
Causes: Unknown

35. OOM Error Map output copy.

User: I am encountering the following out-of-memory error during the reduce phase of a large job. I am currently using 12 reducers and I can't increase this count by much to ensure availability of reduce slots for other users. mapred.job.shuffle.input.buffer.percent --> 0.70
Fortunately, our requirements for this job changed, allowing me to use a combiner that ended up reducing the data that was being pumped to the reducers and the problem went away.
Expert:
Job type: User-defined (Mailing list)
Causes: Unknown

36. java.lang.OutOfMemoryError: Direct buffer memory

Job type: User-defined (Mailing list)
Causes: Unknown

36. OOME only with large datasets

Job type: User-defined (Mailing list)
Causes: Unknown

37. reducer outofmemoryerror

Job type: User-defined (Mailing list)
Causes: Unknown

38. Nor "OOM Java Heap Space" neither "GC OverHead Limit Exeeceded"

Job type: User-defined (Mailing list)
Causes: Unknown

39. Yarn container out of memory when using large memory mapped file

Job type: User-defined (Mailing list)
Causes: Unknown

40. ReducerTask OOM failure

Job type: User-defined (Mailing list)
Causes: Unknown

41. Reducer Out of Memory
Job type: User-defined (Mailing list)
Causes: Unknown

42. out of memory error

Job type: User-defined (Mailing list)
Causes: Unknown

43. RE: out of memory running examples

Job type: User-defined (Mailing list)
Causes: Unknown

44. Reduce tasks running out of memory on small hadoop cluster

Job type: User-defined (Mailing list)

45. Map Task is failing with out of memory issue

Job type: User-defined (Mailing list)

46. Out of Memory error in reduce shuffling phase when compression is turned on

Job type: User-defined (Mailing list)

47. java.lang.OutOfMemoryError: Direct buffer memory

Job type: User-defined (Mailing list)

48. OOM error with large # of map tasks

Job type: User-defined (Mailing list)

49. OutOfMemory error processing large amounts of gz files

Job type: User-defined (Mailing list)

50. OutofMemory Error, inspite of large amounts provided

Job type: User-defined (Mailing list)

51. OutOfMemoryError with map jobs

Job type: User-defined (Mailing list)

52. Caused by: java.lang.OutOfMemoryError: Java heap space - Copy Phase

Job type: User-defined (Mailing list)

53. OOME only with large datasets

Job type: User-defined (Mailing list)

54. Mapper OutOfMemoryError Revisited !!

Job type: User-defined (Mailing list)

55. OutOfMemory Error

Job type: User-defined (Mailing list)

56. reducer outofmemoryerror

Job type: User-defined (Mailing list)

57. reducer out of memory?

Job type: User-defined (Mailing list)

58. out of memory running examples

Job type: User-defined (Mailing list)

59. Out of memory in identity mapper?

Job type: User-defined (Mailing list)

60. Canopy generation out of memory troubleshooting

Job type: User-defined (Mailing list)

61. out of memory for Reducer possible?

Job type: Pig (Mailing list)

62. Java heap size increase caused MORE out of memory exceptions.

Job type: User-defined (Mailing list)

63. Yarn container out of memory when using large memory mapped file

Job type: User-defined (Mailing list)

64. Has anyone else seen out of memory errors at the start of combiner tasks?

Job type: User-defined (Mailing list)

65. Hadoop 1.2.1 corrupt after restart from out of heap memory exception

Job type: User-defined (Mailing list)

66. Out of Java heap space

Job type: User-defined (Mailing list)

67. Out of heap space errors on TTs

Job type: User-defined (Mailing list)

68. OutOfMemoryError of PIG job (UDF loads big file)

Job type: Pig (Mailing list)

69. OutOfMemoryError: Cannot create GC thread. Out of system resources

Job type: User-defined (Mailing list)

70. Shuffle In Memory OutOfMemoryError

Job type: User-defined (Mailing list)

71. Hashing two relations

Job type: User-defined (Mailing list)

72. OOM Error Map output copy.

Job type: User-defined (Mailing list)

73. OutOfMemoryError during reduce shuffle

Job type: User-defined (Mailing list)

74. Caused by: java.lang.OutOfMemoryError: Java heap space - Copy Phase

Job type: User-defined (Mailing list)

75. java.lang.OutOfMemoryError: Java heap space

Job type: Pig (Mailing list)

76. OutOfMemory during Plain Java MapReduce

Job type: User-defined (Mailing list)

77. OutOfMemoryError: unable to create new native thread

Job type: User-defined (Mailing list)

78. Reduce java.lang.OutOfMemoryError

Job type: User-defined (Mailing list)

79. OutOfMemory in ReduceTaskReduceCopierMapOutputCopier.shuffleInMemory

Job type: User-defined (Mailing list)

80. ReducerTask OOM failure

Job type: User-defined (Mailing list)

81. OOM error and then system hangs

Job type: User-defined (Mailing list)

82. how to prevent JAVA HEAP OOM happen in shuffle process in a MR job?

Job type: User-defined (Mailing list)

83. java.lang.OutOfMemoryError: Java heap space

Job type: User-defined (Mailing list)

84. java.lang.OutOfMemoryError: GC overhead limit exceeded

Job type: User-defined (Mailing list)

85. memoryjava.lang.OutOfMemoryError related with number of reducer?

Job type: User-defined (Mailing list)

86. Possible memory "leak" in MapTask$MapOutputBuffer

Job type: User-defined (Mailing list)

87. about the exception in mapreduce program?

Job type: User-defined (Mailing list)

88. MapReduce failure

Job type: User-defined (Mailing list)

89. High memory usage in Reducer

Job type: User-defined (Mailing list)

90. Child JVM memory allocation / Usage

Job type: User-defined (Mailing list)

91. Memory exception in the mapper

Job type: User-defined (Mailing list)

92. Killed : GC overhead limit exceeded

Job type: User-defined (Mailing list)

93. Solving "heap size error"

Job type: Apache Mahout (Mailing list)

94. heap size problem durning mapreduce

Job type: User-defined (Mailing list)

95. Error with Heap Space.

Job type: User-defined (Mailing list)