Wed July 2, 2008

Apache Hadoop Wins Terabyte Sort Benchmark

by Ajay Anand

22 Comments Bookmark Share Tweet Like

One of Yahoo's Hadoop clusters sorted 1 terabyte of data in **209 seconds**, which beat the previous record of 297 seconds in the annual general purpose (daytona) terabyte sort benchmark. The sort benchmark, which was created in 1998 by Jim Gray, specifies the input data (10 billion 100 byte records), which must be completely sorted and written to disk. This is the first time that either a Java or an open source program has won. Yahoo is both the largest user of Hadoop with 13,000+ nodes running hundreds of thousands of jobs a month and the largest contributor, although non-Yahoo usage and contributions are increasing rapidly.

The cluster statistics were:

- 910 nodes
- 2 quad core Xeons @ 2.0ghz per a node
- 4 SATA disks per a node
- 8G RAM per a node
- 1 gigabit ethernet on each node
- 40 nodes per a rack
- 8 gigabit ethernet uplinks from each rack to the core
- Red Hat Enterprise Linux Server Release 5.1 (kernel 2.6.18)
- Sun Java JDK 1.6.0_05-b13

The benchmark was run with Hadoop trunk (pre-0.18) with a couple of optimization patches to remove intermediate writes to disk. The sort used 1800 maps and 1800 reduces and allocated enough memory to buffers to hold the intermediate data in memory. All of the code for the benchmark has been checked in as a Hadoop example.

Owen O'Malley Yahoo! Grid Computing Team

Ajay Anand

22 Comments Bookmark Share Tweet Like

Categories Uncategorized

Previous Post « » Next Post

MORE FROM AJAY ANAND

Hadoop 0.20.S Virtual Machine Appliance
Hadoop Summit 2010 – Agenda is available!
Hadoop computes the 10^15+1st bit of π
Hadoop Sorts a Petabyte in 16.25 Hours and a Terabyte in 62 Seconds
Hadoop Summit 2009 – Open for registration

22 COMMENTS

Stephane Grenier 148 weeks ago | Report Abuse

Who says Java is slow… :)

Permalink

The Apache Hadoop project develops open-source software for reliable, scalable, distributed computing.

HADOOP ON TWITTER

ALL summit yhadoop ydn

Wisdom of @ZURB on the evolution of their Verify testing product. Sketch first, then write code faster. http://bit.ly/jkr4Tn #hack
3 hours ago

RT @caridy proud of what we built using #yui: http://bit.ly/kiyDnc yahoo #searchdirect for screenreader #accessibility cc @vick08
7 hours ago

Hands-on Node.js book http://bit.ly/lhQ44l (via @jcleblanc)
8 hours ago

Eager to hear from the @ZURB guys about building awesome products. Yahoo #hack lunch talk today in SNV. http://bit.ly/jCLB10
8 hours ago

YouTube founders acquire Tap11 real-time biz intelligence platform proclaim 'strong synergy' w/Delicious http://bit.ly/lV5e3M
22 hours ago

Follow @hadoopsummit on Twitter

RECENT COMMENTS

r_angani on HCatalog, tables and metadata for Hadoop

Alan Gates on HCatalog, tables and metadata for Hadoop

Mark Tsimelzon on HCatalog, tables and metadata for Hadoop

Thomas Koch on HCatalog, tables and metadata for Hadoop

gerritjw@googlemail.com on HCatalog, tables and metadata for Hadoop

CATEGORIES

Announcements Developer Notes

Web2PDF
converted by Web2PDFConvert.com

Anonymous 148 weeks ago | Report Abuse

Given that this implementation had 3640 cores and the previous winner only had 800 cores, I would say that Java is slow.

Permalink

Mihai 148 weeks ago | Report Abuse

It's not just the numbers of cores, it's a whole distributed system. They are spread on different machines on different racks, it's how you use them together to get a fast result.

Permalink

Venkat 148 weeks ago | Report Abuse

Anonymous

2005 winner used only 80 cores and achieved it in 435 seconds. So with 800 cores what 2007 winner achieved is 297 seconds ?

Its not only number of cores its how the logic to use parallel nodes properly to do a particular task is important.

Permalink

Anonymous 148 weeks ago | Report Abuse

Hadoop core is written in ANSI C, and all the Java is layered on top of that. Saying Java is fast after reading this is like saying Matlab is fast because it can invert a matrix really, really fast (Matlab calls super-optimized open-source Fortran codes to invert matrices).

Permalink

hadoop user 148 weeks ago | Report Abuse

Actually, hadoop is written in pure java.

Permalink

bernz 148 weeks ago | Report Abuse

See, this is what's wrong with the Internet. One poster claims hadoop core is C, and the next poster counter-claims that it's pure (read: completely, core included) Java. One of you is wrong.

It's each and every poster's responsibility to CHECK THE FACTS, OR DON'T POST — otherwise, every other user will either have to waste his time finding the truth, or be somewhat convinced of something that's not even true (which leads to stupidity).

Luckily, in this case, the fact under dispute is not very important, but that does not in any way invalidate my point — ONLY POST FACTS. If you are not willing to stake your OWN LIFE on the veracity of your post, then, FFS, just DON'T POST.

BTW, nice job, hadoop community!

Permalink

Robert Gutierrez 148 weeks ago | Report Abuse

I am curious of the networking aspect of this setup. I will assume the "8 uplinks from each rack to the core" is a standard Port-Channel. Now I am just beginning to be familiar with Hadoop, but I don't know about it's traffic patterns. Did you optimize on traffic in keeping a majority within a group of racks (a TOR switch in each rack) connected to each distribution point (lets say a Cisco WS-6748-GE-TX line card) without having to cross a backplane (again assuming a Cisco 6500 series)?

Or is this moot if you used a Cisco 7609 series and it's 720gbps backplane? Or something more exotic (and non-Cisco)?

Very interesting stuff here!

Permalink

Steve 148 weeks ago | Report Abuse

I'll be that this is an I/O bound benchmark – by design. The hdfs (hadoops *distributed* filesystem) is an abstraction layered atop native (OS-embedded) filesystems. These

Hadoop User Group MapReduce
Miscellaneous Pig
Presentations Testing

ARCHIVES Select Month

Web2PDF
converted by Web2PDFConvert.com

native filesystems are implemented as optimized, native code. These native filesystems are always (originally) written/tuned in native C (and optimized by C compilers).

There may be significant performance overhead/latencies in the TCP/IP stack (processing) too. TCP/IP drivers are implemented as optimized, native code (even if off-loaded to a fancy TCP/IP Off-load engine – TOE). All of this is originally written/tuned in C as well.

In addition, Hadoop's designers choose to employ native compression libraries – for 'performance reasons' (and due to non-availability of Java alternatives). See http://hadoop.apache.org/core/docs/current/native_libraries.html

So yes, this bencmark manages to use a dollup of Java code to spread the heck out of some relatively simple 'record-sort' benchmark logic. A lot of the run-time observed is system time – not user time. The JVM hardly matters. In this benchmark, it just runs a dollup of logic that spreads around the sort work – to lots of commodity class servers (running Linux). On each server, that logic drives lots of native/C code - in parallel. By ganging thousands of such servers (versus mere hundreds of such servers in the previous record-setting attempt), this new bechmark effort manages to set a world speed record. Sure. So what?

This same result could have been achieved with Python … or Perl … or Ruby … or whatever. This is really just a demonstratation the power of distributed, parallel 'sorting' - irrespective of the implementation languages involved. It is hard to imagine that the modicum of top-level logic is anywhere close to being the rate limiting step. Therefore, this benchmark says little about the performance of Java – or any other scripting language that might have been employed to this same end.

This isn't about Java (performance). This is about the performance advantages of the distributed, map-reduce algorithm. It hardly matters whether this distributed, map-reduced sorting benchmark was implemented in JAVA, C/C++, Python, VB, Ruby, Perl or whatever! Lets not miss the whole point.

Cheers, Steffen.

Permalink

Jeffrey W. Baker 148 weeks ago | Report Abuse

Those of you crowing about inefficiency would be wise to note that only 1456 CPU clock cycles per row were used. That's not at all bad!

Permalink

Owen O'Malley 148 weeks ago | Report Abuse

The framework schedules maps tasks close to their input, where close usually means the same node or at least the same rack. The traffic pattern between the maps and reduces is all to all. So all 900 nodes had roughly even data for every other node. In terms of the 8 gb uplinks from each rack, I believe that there are 4 core switches and each rack has 2 1gb links to each of the core switches. I don't know the details of which switches we have where…

Permalink

Owen O'Malley 148 weeks ago | Report Abuse

Robert, I didn't use the compression codecs, because it wasn't clear whether or not it was "legal" to do so. It is clearly against the rules to use compression on the input or output, but the transient data should be an implementation detail. Hopefully, that will be clarified in the rules for next year. So I was running 100% pure Java.

I will agree with you that Java hasn't been the critical limiting factor on performance yet. Much more important is good architecture and design, identifying and removing choke points, reducing memory copies, using appropriate data structures, etc.

Permalink

Allen Wittenauer 148 weeks ago | Report Abuse

There is a (fairly simplified) diagram of the network layout on page 18 of the presentation found at http://tinyurl.com/5foamm . One thing worth mentioning is that this is all layer 3.

Permalink

Web2PDF
converted by Web2PDFConvert.com

dojo 147 weeks ago | Report Abuse

I certainly agree that this is impressive but it isn't clear to me what "completely sorted and writen to disk" means for a 910-node cluster. The benchmark ground rules state that the output must be a sequential file (with open/seek/read access) and if you want to distribute the data file on different disks, you need use the OS RAID software. This seems to preclude distributing the output file but even assuming that the input, map, distribute and reduce tasks take zero seconds, just writing 1TB in 209 seconds roughly 5GB/sec which is at least 50x faster than any node with four SATA disks could write. If, on the other hand, the input and output files are spread over 910 nodes, how is it partitioned? Did each node simply sort its (roughly) 1.1GB of data? Or does node[0] end up with the lowest 1/910th values and node[909] the highest? Sorting 1.1GB in 209 seconds is only about 5.25MB/sec which is a lot less impressive.

Permalink

deepesh 146 weeks ago | Report Abuse

Did google participate?

Permalink

Dmitry 144 weeks ago | Report Abuse

This is like saying that C is nothing because all of the real work is done by assembler. Apart from JVM, everything else is written in Java. The only bottle necks are disks and network. Maybe some of you can replicate this – build xml delta utility that takes 2 xml files and creates a delta xml comparing one to the other and producing xml delta – my java implementation, single instance, around 100 meg xml (mRSS) files in around 5 seconds.

Permalink

christian coryell 144 weeks ago | Report Abuse

Congrats to the Hadoop team. Another example of the value of opensource projects.

Permalink

Richard Corey 143 weeks ago | Report Abuse

Stake my life ??? A little perspective, perhaps? This isn't an internal classified memo being used to regulate catapult speeds on a carrier — and if anyone was writing software for that and was happy just using whatever they saw in a few posts sans testing… I'd ask to transfer to the Air Force.

Permalink

Aster 141 weeks ago | Report Abuse

Folks – just wanted to make everyone aware that Aster has just announced the world's first integrated MapReduce database:

http://www.asterdata.com/product/mapreduce.html

Not all database vendors think MapReduce is a "major step backwards"…Aster thinks MapReduce is definitely the wave of the future…

Permalink

igc 128 weeks ago | Report Abuse

Google just posted their results. Similar number of computers, same size of data, sorted in 68 seconds. They also sorted 1PB of data. And they seem to keep 3 replicas of the data at least in the 1PB test.

http://googleblog.blogspot.com/2008/11/sorting-1pb-with-mapreduce.html

Permalink

Sam 128 weeks ago | Report Abuse

Aster data is another failure of commercial use of MapReduce

Permalink

kavya 54 weeks ago | Report Abuse

Web2PDF
converted by Web2PDFConvert.com

hello anyone plz tell me how to run our own programs on hadoop? and one more doubt i have, can we run c programs on hadoop? if so plz tell me steps because i have one client server program written in c…but im not able to run it on hadoop…plz plz plz help me..

Permalink

Post a Comment

You must be logged in to Yahoo! to comment. Log In.

Follow Yahoo! Developer Network on

Blogs > Yahoo! Hadoop Blog > 2008 > Apache Hadoop Wins Terabyte Sort Benchmark

Copyright © 2011 Yahoo! Inc. All rights reserved.

Copyright | Privacy Policy | Terms of Use

Contact Us | Community | Suggestions

Web2PDF
converted by Web2PDFConvert.com