Paper Trail

Computer Systems, Distributed Algorithms and Databases

The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google

Note: this is a personal blog post, and doesn't reflect the views of my employers at Cloudera

Map-Reduce is on its way out. But we shouldn't measure its importance in the number of bytes it crunches, but the fundamental shift in data processing architectures it helped popularise.

This morning, at their I/O Conference, Google revealed that they're not using Map-Reduce to process data internally at all any more.

We shouldn't be surprised. The writing has been on the wall for Map-Reduce for some time. The truth is that Map-Reduce as a processing paradigm continues to be severely restrictive, and is no more than a subset of richer processing systems.

It was known for decades that generalised dataflow engines adequately capture the map-reduce model as a fairly trivial special case. However, there was real doubt over whether such engines could be efficiently implemented on large-scale cluster computers. But ever since Dryad, in 2007 (at least), it was clear to me that Map-Reduce's days were numbered. Indeed, it's a bit of a surprise to me that it lasted this long.

Map-Reduce has served a great purpose, though: many, many companies, research labs and individuals are successfully bringing Map-Reduce to bear on problems to which it is suited: brute-force processing with an optional aggregation. But more important in the longer term, to my mind, is the way that Map-Reduce provided the justification for re-evaluating the ways in which large-scale data processing platforms are built (and purchased!).

If we are in a data revolution right now, the computational advance that made it possible was **not** the 'discovery' of Map-Reduce, but instead the realisation that these computing systems can and should be built from relatively cheap, shared-nothing machines (and the real contribution from Google in this area was arguably GFS, not Map-Reduce).

The advantages of this new architecture are enormous and well understood: storage and compute become incrementally scalable, heterogeneous workloads are better supported and the faults that more commonly arise when you have 'cheaper' commodity components are at the same time much easier to absorb (paradoxically, it's easier to build robust, fault-tolerant systems from unreliable components). Of course lower cost is a huge advantage as well, and is the one that has established vendors stuck between the rock of having to cannibalise their own hardware margins, and the hard place of being outmaneuvered by the new technology.

In the public domain, Hadoop would not have had any success without Map-Reduce to sell it. Until the open-source community developed the maturity to build successful replacements, commodity distributed computing needed an app – not a 'killer' app, necessarily, but some new approach that made some of the theoretical promise real. Buying into Map-Reduce meant buying into the platform.

Now we are much closer to delivering much more fully on the software promise. MPP database concepts, far from being completely incompatible with large shared-nothing deployments, are becoming more and more applicable as we develop a better understanding of the way to integrate distributed and local execution models. I remember sitting in a reading group at Microsoft Research in Cambridge as we discussed whether joins could ever be efficient in cluster computing. The answer has turned out to be yes, and the techniques were already known at the time. Transactions are similarly thought to be in the 'can never work' camp, but Spanner has shown that there's progress to be made in that area. Perhaps OLTP

will never move wholesale to cluster computing, but data in those clusters need not be read-only.

As these more general frameworks improve, they subsume Map-Reduce and make its shortcomings more evident. Map-Reduce has never been an easy paradigm to write new programs for, if only because the mapping between your problem and the rigid two-phase topology is rarely obvious. Languages can only mask that impedance mismatch to a certain extent. Map-Reduce, as implemented, typically has substantial overhead attributable both to its inherent 'batchness', and the need to have a barrier between the map and reduce phases. It's a relief to offer end-users a better alternative

So it's no surprise to hear that Google have retired Map-Reduce. It will also be no surprise to me when, eventually, Hadoop does the same, and the elephant is finally given its dotage.

Published: June 25, 2014
Filed Under: Uncategorized

30 Responses to "The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google"

1. *The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google | Blog* says:
June 25, 2014 at 7:54 pm

   […] The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google […]

2.  *Jeff Smith* says:
June 25, 2014 at 9:27 pm

   All the more impressive to me that companies like LexisNexis built their own proprietary languages to do this type of general purpose distributed data processing 15 years before anyone else even thought about it. Ironically, MR and Hadoop forced them to open source pieces of their systems to combat the recent drop in customers.

3. *Bookmarks for June 25th from 19:33 to 22:50 : Extenuating Circumstances* says:
June 26, 2014 at 12:59 am

   […] The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google : Paper Trail – Map-Reduce is on its way out. But we shouldn't measure its importance in the number of bytes it crunches, but the fundamental shift in data processing… […]

4.  *Nicolas* says:
June 26, 2014 at 2:36 am

   It should be obvious to anyone who has enough brain to actually understand what map reduce is.

   15 years ago it might have been new to some (including me) but then folks were using altavista…

5.  *Howard Chu* says:
June 26, 2014 at 10:25 am

   re: (paradoxically, it's easier to build robust, fault-tolerant systems from unreliable components) not a paradox at all, and also not universally true.

As usual, you can find supporting evidence in nature – simpler life forms reproduce by producing millions of eggs that, if fertilized, may yield millions of offspring that eventually survive as only thousands or hundreds of adults.

More complex life forms have small numbers of offspring, but also smaller mortality rates.

Yes, large numbers of small cheap nodes guarantees that at least some will have a long enough uptime to get some useful work done. But those nodes will never have the ability to get the most complex jobs done – for that you need to put all your eggs in one basket, and use small numbers of large, high complexity, high investment nodes.

And yes, when one of those big nodes goes down, you lose a lot. That's the inevitable cost of higher capability.

6.  *Raj N* says:
June 26, 2014 at 1:43 pm

   Yes and we are all Google – we should stop using Map Reduce before we even know what it is. The key is "anymore" and that is taken out of context. Google wants to do millisecond time response queries against petabytes of data, so they moved on to dremel. We use Map Reduce to run scrubbing and validations that took hours .We use Pentaho to do it so that the tool can pick the right processing paradigm – Map Reduce today, something else tomorrow for the right workload

7.  *ESV* says:
June 26, 2014 at 3:22 pm

   @Howard Chu, can you give an example of a "complex job" for which you would need a "large, high complexity, high investment node"?

8.  *Giotti* says:
June 26, 2014 at 5:58 pm

   I'd be interested in your thoughts on the stratosphere project (http://stratosphere.eu), which enables more complex operations (e.g. joins, unions, iterations, etc.) for data flows. And is, of course, open source (i.e. this is not a commercial plug, but a genuine question).

9. *The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google - HackerExchange.com* says:
June 26, 2014 at 6:29 pm

   […] Continue Reading : The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google […]

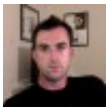
10.  *Doug Meil* says:
June 26, 2014 at 7:01 pm

    re: "and the real contribution from Google in this area was arguably GFS, not Map-Reduce). "

I disagree. It's not enough just to store data – you need to be able to do something with it. MapReduce was a breakthrough in widescale adoption of distributed processing. The fact that MapReduce was surpassed doesn't mean that it sucked in total – far from it in my opinion.

11.  *Mark T* says:
June 26, 2014 at 7:39 pm

    What get's me is how much I found in principle the activities were to problems that have existed in DB management for literally decades. What is really novel – it seems to me – is not so much that this or that language is in play, but the low-cost barrier – for a couple of thousand bucks you can setup a simple cluster, and push some fairly respectable numbers around your equipment, that to me is the real opportunity ahead.

12.  *Henry* says:
June 27, 2014 at 11:29 am

    Doug – the GFS paper much more clearly lays out the benefits and challenges of designing for large shared-nothing clusters (e.g. "First, component failures are the norm rather than the exception").

    Yes, you need a processing framework, and yes Map-Reduce was a product of its time, but I don't personally think it set the technical direction as clearly as GFS.

13. *Moving past Map-Reduce | Die wunderbare Welt von Isotopp* says:
June 28, 2014 at 8:11 am

    […] http://the-paper-trail.org/blog/the-elephant-was-a-trojan-horse-on-the-death-of-map-reduce-at-google… » This morning, at their I/O Conference, Google revealed that they're not using Map-Reduce to process data internally at all any more. […]

14.  *Alex x* says:
June 28, 2014 at 8:54 am

    @ESV banks and other mainframe buyers

15. *The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google | Business Intelligence, Data Analytics, Infographics, and Life* says:
June 28, 2014 at 7:22 pm

    […] Reference: The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google […]

16. *Hadoop Happenings: Google Abandons MapReduce, Splice Machine Goes Beta | Qubole* says:
June 30, 2014 at 11:37 am

    […] The-Paper-Trail.org- With Google's announcement that it is no longer using MapReduce, this post covers the legacy of MapReduce, and its role in the shift in data processing architectures. Read More […]

17. *Five short links « Pete Warden's blog* says:
July 1, 2014 at 4:04 pm

[…] The elephant was a Trojan horse – I almost always use Hadoop as a simple distributed job system, and rarely need MapReduce. I think this eulogy for the original approach captures a lot of why MapReduce was so important as an agent of change, even if it ended up not being used as much as you'd expect. […]

18.  *Petar Maymounkov* says:
July 2, 2014 at 3:37 am

Isn't every good idea a Trojan Horse though, as it dissipates through the slower understanding wider circles? It is natural by then that the creators would have moved on at their faster pace. No?

19. *Daily Links – 2nd July 2014 | Ewan Silver* says:
July 2, 2014 at 2:25 pm

[…] The Elephant was a Trojan Horse: On the Death of Map-Reduce at Google […]

20.  *Z* says:
July 3, 2014 at 7:04 am

@ ESV :

A good example would be many of the numerical solvers that are used in simulation/modeling software. Of course efforts are being made to better distribute these tasks, but even then, each node still requires some serious computational power.

If 3D printing virtual prototyping or IoT system simulations become mainstream (as in used by the hobbyist or consumer markets), there may be a push for less dependency on this type of horsepower. For now, though, the innovation has been focused more pushing the boundaries of the tool features and algorithms themselves rather than scaling the tools through commodity hardware.

21. *Noticias 07-07-2014 - La Web de Programación* says:
July 7, 2014 at 12:58 am

[…] Google abandona Map-Reduce: Hadoop es cosa del pasado para ellos ya, ahora usan Dataflow. […]

22. *Tasting first CouchBase | milansimonovic.com* says:
July 8, 2014 at 12:46 am

[…] In this post I'll tell you about installing and using CouchBase in a test-first manner. Wiping out all the data between tests turned out to be very slow (4s), and for anything but key lookups indexes have to be used, so the elephant is still in the room. […]

23.  *Pere* says:
July 12, 2014 at 2:52 am

Something is not quite clear to me, when Google guys present a month ago a machine learning system that does use MapReduce: https://www.youtube.com/watch?v=QoUVwGZb9tA

Since many years ago, "killing MapReduce" is been the easy marketing motif… But I don't think

someone actually killed it – not even Google

24. *Quora* says:
September 2, 2014 at 4:07 am

Why do people on Quora seem to hate Tata Consultancy Services?

The respect you deserves depends on how you respect others. And after working 4+ years in TCS I can say that TCS doesn't respects its employees. One can imagine the respect where people are "resources". You mentioned Labs and Innovation center. I am…

25. *The Problem with Hadoop - Off- and Online* says:
September 22, 2014 at 3:33 am

[…] many cases, you don't want to force MapReduce into the equation. Google confessed to more or less having stopped using MapReduce as far back as 2010. And even Apache Mahout, the project named for the elephant driver steering […]

26. *Aimee* says:
September 24, 2014 at 10:17 am

After getting some tutorials on MapReduce, I couldn't help but think this alogithm is NOT new at all. In college CS classes, we were taught Parallel algorithms. The Shuffle/Sort is exactly something a 2nd year CS major would learn. Why didn't anyone thought to commercialize such a simple basic CS theory earlier than Google? Exception is LexisNexis which had its own proprietary algorithm but didn't learn to commercialize as big as Google did.

27. *Hadoop: 5 Undeniable Truths | The News Gazette* says:
October 23, 2014 at 9:52 am

[…] Death of MapReduce," proclaimed a blog post on the Paper Trail earlier this year, as soon as Google said it had adopted DataFlow and stopped […]

28. *IWM | Hadoop: cinco verdades innegables* says:
October 29, 2014 at 2:08 pm

[…] de la fatalidad que le esperaba a la base de datos relacional. Luego, a principios de año, una publicación en el blog en Paper Trail proclamó "la muerte de MapReduce", cuando Google dijo que había dejado […]

29. *Six Business Intelligence Predictions For 2015 | Beyond Intelligence Blog* says:
December 20, 2014 at 4:05 am

[…] major Hadoop developments I foresee in 2015 include MapReduce being reduced to the ash heaps of history while Hadoop Spark having a banner year and becoming a mainstream […]

30. *Apache Spark: Defining the Future of Reactive Data? | Voxxed* says:
January 28, 2015 at 8:48 am

[…] Scala based Apache Spark may have only graduated from incubator status last February, but in a short space of time, it's become the most active open source big data project around – a rise that's segued with the decline of old faithful analytics tool Hadoop MapReduce. […]

Leave a Comment

Name: *Required* Email: *Required, not published*
Homepage:

Comment:

Post Comment

« Previous Post
Next Post »

Elsewhere

Search