Hadoop MapReduce Performance on SSDs: The Case of Complex Network Analysis Tasks

Marios Bakratsas[1], Pavlos Basaras[1], Dimitrios Katsaros[1,2(✉)], and Leandros Tassiulas[2]

[1] Department of Electrical and Computer Engineering, University of Thessaly, Volos, Greece
mmpakrat@gmail.com, pbasaras@gmail.com, dkatsar@inf.uth.gr

[2] Department of Electrical Engineering and Yale Institute for Network Science, Yale University, New Haven, USA
leandros.tassiulas@yale.edu

Abstract. This article investigates the relative performance of SSDs versus hard disk drives (HDDs) when they are used as underlying storage for Hadoop's MapReduce. We examine MapReduce tasks and data suitable for performing analysis of complex networks which present different execution patterns. The obtained results confirmed in part earlier studies which showed that SSDs are beneficial to Hadoop; we also provide solid evidence that the processing pattern of the running application plays a significant role.

1 Introduction

Processing of modern Online Social Networks on a single machine (centralized) is doomed to fail due to lack of resources. The Hadoop instead was designed to solve problems where the "same, repeated processing" had to be applied to peta-scale volumes of data. Hadoop's initial design was based on magnetic disk characteristics. With the advent of Solid State Drives (SSDs) research is emerging to test/exploit the potential of the new technologically advanced drive [4, 8]. The lack of seeking overhead gives them a significant advantage with respect to Hard Disk Drives (HDDs) for workloads whose processing requires random access instead of sequential access. Providing a clear answer to the question of whether SSDs significantly outperform or offer increased performance in same cases compared to HDDs in the Hadoop environment is not straightforward, because the results of a system-analysis-based investigation are affected by the network speed and topology, by the cluster (size, architecture) and by the nature of the benchmarks used (MapReduce algorithms, input data). This article starts the investigation from a new basis and attempts to provide a clear answer to the following question [7]: *Ignoring any network biases and storage media cost considerations, do SSDs provide improved performance over HDDs for real workloads that are not dominated by either reads or writes?*

P. Angelov et al. (eds.), *Advances in Big Data*, Advances in Intelligent Systems and Computing 529, DOI 10.1007/978-3-319-47898-2_12

2 Related Work

Investigating the usage of SSDs in Hadoop clusters has been a hot issue of discussion very recently. The most relevant work to ours is included in the following articles [4, 5, 8, 9, 11]. The first effort [5] to study the impact of SSDs on Hadoop was on a virtualized cluster (multiple Hadoop nodes on a single physical machine) and showed up to three times improved performance for SSDs versus HDDs. However, it remains unclear whether the conclusions still hold in non-virtualized environments. The work in [8] compared Hadoop's performance on SSDs and HDDs on hardware with non-uniform bandwidth and cost using the Terasort benchmark. The major finding is that SSDs can accelerate the shuffle phase of MapReduce. However, this work is confined by the very limited type of application/workload used to make the investigation and the intervention of data transfers across the network. Cloudera's employees in [4], using a set of same-rack-mounted machines (not reporting how many of them), focus on measuring the relative performance of SSDs and HDDs for equal-bandwidth storage media. The MapReduce jobs they used are either read-heavy (Teravalidate, Teraread, WordCount) or network-heavy (Teragen, HDFS data write), and the Terasort which is read/write/shuffle "neutral". Thus, neither the processing pattern is mixed nor the network effects are neutral. Their findings showed that SSD has higher performance compared to HDD, but the benefits vary depending on the MapReduce job involved, which is exactly where the present study aims at [7].

The analysis performed in [9] using Intel's HiBench benchmark [2] concluded that "… the performance of SSD and HDD is nearly the same", which contradicts all previously mentioned works. A study of both pure (only with HDDs or only with SSDs) and hybrid systems (combined SSDs and HDDs) is reported in [11] using a five node cluster and the HiBench benchmark. In contrast to the current work, the authors in [11] investigated the impact of HDFS's block size, memory buffers, and input data volume on execution time. The results illustrated that when the input data set size and/or the block size increases, the performance gap between a pure SSD system and a pure HDD system widens in favor of the SSD. Moreover, for hybrid systems, the work showed that more SSDs result in better performance. These conclusions are again expected since voluminous data imply increased network usage among nodes. Earlier work [3, 10] studied the impact of interconnection on Hadoop performance in SSDs identifying bandwidth as a potential bottleneck. Finally, some works propose extensions to Hadoop with SSDs. For instance, VENU [6] is a proposal for an extension to Hadoop that will use SSDs as a cache (of the HDDs) not for all data, but only for those that are expected to benefit from the use of SSDs. This work still leaves open the question about how to tell which applications are going to benefit from the performance characteristics of SSDs.

3 Investigated Algorithms

Complex network analysis comprises a large set of diverse tasks (algorithms for finding communities, centralities, epidemics, etc.) that cannot be enumerated here. Among all these problems and their associated MapReduce solutions, we had to select some of

them based on (a) their usefulness in complex network analysis tasks, (b) in their suitability to the MapReduce programming paradigm, (c) the availability of their implementations (free/open code) for purposes of reproducibility of measurements, and (d) complexity in terms of multiple rounds of map-reduce operations. Based on these criteria, we selected three problems/algorithms for running our experimentations[1]. The first algorithm deals with a very simple problem which is at the same time a fundamental operation in Facebook, that of *finding mutual friends*. The second algorithm deals with a network-wide path-based analysis for *finding connected components* which finds applications in reachability queries, techniques for testing network robustness and resilience to attacks, epidemics, etc. The third algorithm is about *counting triangles* which is a fundamental operation for higher level tasks such as calculating the clustering coefficient, or executing community finding algorithms based on clique percolation concepts. Table 1 summarizes the "identity" of the tasks.

Table 1. Characterization of problems/algorithms examined.

Primitive task	Type of analysis	Type of analysis
Mutual friends	Neighbor-based	Local network (neighborhood) properties Recommendation queries
Connected Components	Path-based	Large-scale network properties Reachability queries Resilience queries
Triangle counting	Mixed (extended neighborhood & paths)	Large-scale network properties Clustering/communities finding queries

We deferred a more advanced method for measuring the performance for multi-job workload such as the one described in [1], because the standalone, one-job-at-the-time method allows for the examination of interaction between MapReduce and storage media without the interventions of job scheduling and task placement algorithms. We aim at showing that the conclusions about the relative performance of SSDs versus HDDs are strongly depended on the features of the algorithms examined, which has largely been neglected in earlier relative studies [4, 5, 8], and based on these features we draw some conclusions on the relative benefits of SSDs.

4 System Setup

A commodity computer (Table 2) was used for the experiments. Three storage media were used (Table 2) with capacities similar to that used in [8]. On each of the three drives (one HDD and two SSDs) a separate and identical installation of the latest

[1] The MapReduce codes (along with many experiments) can be found in the technical report at http://www.inf.uth.gr/~dkatsar/Hadoop-SSD-HD-for-SNA.pdf.

version of required software was used. We emphasize at this point that since we need to factor out the network effects, we used single machine installations. Three different incremental setting setups were used: (a) with default settings, allowing 6 parallel maps, (b) with modified containers, allowing 3 parallel maps, and (c) with custom settings (Table 3). In all these setups, speculative execution was disabled and no early shuffling was permitted.

Table 2. Computer specifications.

CPU	Intel i5 4670 3.4 GHz (non HT)
RAM	8 GB 1600 MHz DDR3 (1333 MHz with disabled XMP)
Disk 1 (HDD)	Western Digital Blue WD10EZEX 1TB
Disk 2 (SSD1)	Samsung 840 EVO 120 GB
Disk 3 (SSD2)	Crucial MX100 512 GB

Table 3. Custom settings.

mapreduce.reduce.shuffle.parallel.copies	5 -> 50
mapreduce.task.io.sort.factor	10 -> 100
mapreduce.map.sort.spill.percent	0.80 -> 0.90
io.file.buffer.size	4 kb -> 64 kb

5 Input Data and Performance Measures

For the evaluation of the two disk types, we used *ten real social network data* (Table 4). They were retrieved from https://snap.stanford.edu/ and http://konect.uni-koblenz.de/.

The two SSDs were of different size disallowing the execution of some datasets. The most important measures we captured were the Map and Reduce execution times,

Table 4. Social networks used for evaluation.

#	Social network name	#Nodes	#Edges
1	Brightkite location based online social network	58,228	214,078
2	Gowalla location based online social network	196,591	950,327
3	Amazon product co-purchasing network	334,863	925,872
4	DBLP collaboration network	317,080	1,049,866
5	YouTube online social network	1,134,890	2,987,624
6	YouTube (ver. 2) online social network	3,223,589	9,375,374
7	Flickr	1,715,255	15,550,782
8	LiveJournal online social network	3,997,962	34,681,189
9	LiveJournal (ver. 2) online social network	5,204,176	49,174,620
10	Orkut online social network	3,072,441	117,185,083

as also Sort (merge) and Shuffle phase. One common side effect is "cache hits" from previous executions, that was also experienced in [8]. In order to give each experiment an equal environment, Hadoop was halted and page cache was flushed, after each experiment. Before each test, HDFS was re-formatted.

6 The Results

1. Mutual Friends

The complexity of this algorithm is exponential due to the mapper of the 2nd MapReduce job. Thus, the 2nd MapReduce job is the most resource-intensive of the three jobs, rendering it a good inspection point for our measures (see Table 5), whereas the 1st and 3rd MapReduce jobs were fast-executed and almost identical for all disks. For Amazon, Brightkite and DBLP, the three disks performed almost equally. For the bigger datasets, the magnetic disk gives competitive (with respect to both SSD drives) execution times for the reduce phase, but the HDD performs worse for the map phase. The SSD2 displays superior performance at shuffling.

Table 5. Average times for each phase for 2nd job (creating triples) of "mutual friends" algorithm

Defaults: Triples (2nd Job)												
	Avg Map			Avg Shuffle			Avg Merge			Avg Reduce		
	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2
Brightkite	52	52	52	1	1	1	0	0	0	11	10	10
Amazon	36	35	35	2	1	1	0	0	0	8	7	8
Gowalla	1780	1752	1593	120	103	42	0	0	0	178	195	194
DBLP	90	89	89	5	2	3	0	0	0	16	17	17
YouTube	11197	-	9708	812	-	258	0	-	0	916	-	984

2. Counting Triangles

Here, the SSDs outperform the HDD for all evaluated datasets. At "forming the triads" job, HDD illustrated competitive behavior at reduce phase (Table 7). The "counting the triangles" job demonstrated greater variations in execution times. Our evaluation shows that with small datasets the performance differentiations between the two disk types are small (Table 6), whereas with larger ones (like YouTube dataset), SSDs capabilities become evident for shuffle and merge (sort) phases.

For the 1st MR job (creating triads), map, shuffle and merge phases finished quite fast and with almost zero differentiations among disks. Reduce phase lasted significantly longer with both disks performing equally (Table 7). With containers settings, the biggest dataset of Flickr gets significant improvement for both disk types (Table 8).

To optimize performance, increasing the following settings provided best results for the magnetic disk, compared to "containers" settings:

Table 6. Average times for each phase for 2nd job (calculate triangles) of "counting triangles"

A) Defaults: Triangles (2nd) job	Avg Map			Avg Shuffle			Avg Merge			Avg Reduce		
	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2
Brightkite	18	18	18	1	1	1	0	0	0	4	4	3
Amazon	9	9	9	1	1	1	0	0	0	2	2	2
Gowalla	38	39	38	52	62	21	79	86	70	106	106	110
DBLP	14	14	14	1	1	1	0	0	0	7	5	5
YouTube	42	-	41	655	-	141	820	-	668	689	-	551

Table 7. Average times for each phase for 1st job (create triads) of "counting triangles" algorithm

Defaults: Triads (1st) job	Avg Map		Avg Shuffle		Avg Merge		Avg Reduce	
	HDD	SSD2	HDD	SSD2	HDD	SSD2	HDD	SSD2
Gowalla	2	2	1	1	0	0	142	140
YouTube	6	6	1	1	0	0	706	694
Flickr	13	13	1	1	0	0	5053	5125

Table 8. Average times for each phase for 1st job (create triads) of "counting triangles" algorithm, with changed container's settings

Containers: Triads (1st) job	Avg Map		Avg Shuffle		Avg Merge		Avg Reduce	
	HDD	SSD2	HDD	SSD2	HDD	SSD2	HDD	SSD2
Gowalla	2	2	1	1	0	0	141	138
YouTube	6	6	1	1	1	1	697	707
Flickr	13	13	1	1	6	6	4163	4140

Table 9. Performance difference for YouTube dataset at "Counting Triangles", increasing sort factor, for HDD

just containers and io.sort.factor 10->100				
Elapsed	Avg Map	Avg Shuffle	Avg Merge	Avg Reduce
40mins, 26sec (-12mins, 17sec)	25	471(-94)	14(-582)	667(-53)

Table 10. Performance difference for YouTube dataset at "Counting Triangles", increasing sort factor, for SSD2

just containers and io.sort.factor 10->100				
35mins, 15sec (-5mins, 53sec)	25	371(+12)	16(-323)	497 (-41)

(a) *The number of streams to merge at once while sorting files. Minimizes merge time for both disk types. Improves HDD shuffling time as well* (Tables 9 and 10).
(b) *The buffer size for I/O (read/write) operations* (Table 11).

On the other hand, increasing the buffer size for I/O operations had minimal effect on SSD2 performance (Tables 12, 13 and 14).

Table 11. Performance difference for YouTube dataset at "Counting Triangles", increasing file buffer size, for HDD

just containers and io.file.buffer.size 4kb->128kb				
46mins, 44sec (-5mins, 59sec)	25	445(-120)	470(-126)	619(-101)

Table 12. Performance difference for YouTube dataset at "Counting Triangles", increasing file buffer size, for SSD2

just containers and io.file.buffer.size 4kb->128kb				
41mins, 9sec (+1 sec)	24 (-1)	361 (+2)	331 (-8)	554 (+16)

Table 13. Percentage difference between "customs" and "containers settings for YouTube dataset, at "Counting Triangles" algorithm

"Customs" Difference to "Containers"				
	map	shuffle	merge	reduce
HDD	4.00%	-28.85%	-97.65%	-11.39%
SSD2	0.00%	-2.23%	-95.28%	-10.41%

Table 14. Percentage difference between "customs" and "containers settings for YouTube dataset, at "Mutual Friends" algorithm

"Customs" difference to "Containers"				
	map	shuffle	merge	reduce
HDD	-26.14%	-16.59%	-	-9.72%
SSD2	-18.83%	0.78%	-	4.36%

3. Connected Components

Comparing SSD1 to the HDD, the Connected Components algorithm seems to slightly favor the SSD1 for small datasets, at reduce phase. Map, shuffle and phase times are close for both disk types (Table 15). For the datasets of Flickr and LiveJournal the magnetic disk takes the lead at reduce phase which is mostly characterized as "write"

procedure for the Hadoop framework. Surprisingly, SSD1 performs quite slowly at shuffle phase for the LiveJournal dataset. The SSD2 generally delivers great performance especially at map and shuffle phase, noticeably as the datasets' size increase. For the reduce phase HDD falls behind SSD2, but not with a great margin.

Table 15. Sum of average times for each phase for the iterative Jobs of "Connected Components"

	Avg Map			Avg Shuffle			Avg Merge			Avg Reduce		
	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2	HDD	SSD1	SSD2
Brightkite	14	14	14	11	11	11	0	0	0	0	0	0
Amazon	104	106	103	34	34	34	0	0	0	74	61	62
Gowalla	27	26	26	10	10	10	0	0	0	14	14	16
DBLP	54	54	54	15	15	15	0	0	0	35	34	33
YouTube	126	124	123	14	14	14	0	0	0	101	96	98
YouTube 2	247	*243*	244	28	*24*	24	0	*0*	0	428	*424*	408
Flickr	170	168	167	30	19	20	0	0	0	309	314	304
LiveJournal	353	380	322	104	143	45	1	0	0	666	682	651
LiveJournal 2	417	-	347	137	-	57	0	-	0	930	-	912
Orkut	456	-	324	552	-	154	295	-	231	1448	-	1204

7 Conclusions

We compared the performance of solid state drives and hard disk drives for social network analysis. SSDs didn't come out as the undisputed winner. The second SSD performed significantly better. In many cases SSD1 and the magnetic disk came into a draw. Although SSD1 was slightly faster in many tests, in some cases the magnetic disk outperformed the SSD1. Even comparing to the faster SSD2, the magnetic disk provided competitive times for reduce phase, especially with the "mutual friends" algorithm, where it performed marginally better. Magnetic disk's shuffle times can be reduced. SSD's performance doesn't present further improvement. Nevertheless, HDD can't catch up with SSD's superior performance at shuffling. With tweaking merge-sort can be performed in less steps minimizing merge's phase times for both disk types, slightly favoring magnetic disk that would perform slower otherwise. For map phase both disk types can get similar performance improvement.

Acknowledgement. This work was supported by the Project "REDUCTION: Reducing Environmental Footprint based on Multi-Modal Fleet management System for Eco-Routing and Driver Behaviour Adaptation," funded by the EU.ICT program, Challenge ICT-2011.7.

References

1. Chen, Y., Ganapathi, A., Griffith, R., Katz, R.: The case for evaluating MapReduce performance using workload suites. In: Proceedings of IEEE MASCOTS, pp. 390–399 (2011)
2. Huang, S., Huang, J., Dai, J., Xie, T., Huang, B.: The HiBench benchmark suite: Characterization of the MapReduce-based data analysis. In: Proceedings of ICDE Workshops (2010)

3. Islam, N., Rahman, M., Jose, J., Rajachandrasekar, R., Wang, H., Subramoni, H., Murthy, C., Panda, D.: High performance RDMA-design of HDFS over InfiniBand. In: Proceedings of SC (2012)
4. Kambatla, K., Chen, Y.: The truth about MapReduce performance on SSDs. In: Proceedings of LISA, pp. 109–117 (2014)
5. Kang, S.-H., Koo, D.-H., Kang, W.-H., Lee, S.-W.: A case for flash memory SSD in Hadoop applications. Int. J. Control Autom. **6**, 201–210 (2013)
6. Krish, K.R., Iqbal, M.S., Butt, A.R.: VENU: orchestrating SSDs in Hadoop storage. In: Proceedings of IEEE BigData, pp. 207–212 (2014)
7. Min, C., Kim, K., Cho, H., Lee, S.-W., Eom, Y.I.: SFS: random write considered harmful in solid state drives. In: Proceedings of USENIX FAST (2012)
8. Moon, S., Lee, J., Kee, Y.S.: Introducing SSDs to the Hadoop MapReduce framework. In: Proceeding of IEEE CLOUD, pp. 272–279 (2014)
9. Saxena, P., Chou, J.: How much solid state drive can improve the performance of Hadoop cluster? Performance evaluation of Hadoop on SSD and HDD. Int. J. Mod. Commun. Technol. Res. **2**(5), 1–7 (2014)
10. Sur, S., Wang, H., Huang, J., Ouyang, X., Panda, D.: Can high-performance interconnects benefit Hadoop distributed file system. In: Proceedings of the Workshop MASVDC (2010)
11. Wu, D., Xie, W., Ji, X., Luo, W., He, J., Wu, D.: Understanding the impacts of solid-state storage on the Hadoop performance. In: Proceedings of Advanced Cloud and Big Data, pp. 125–130 (2013)