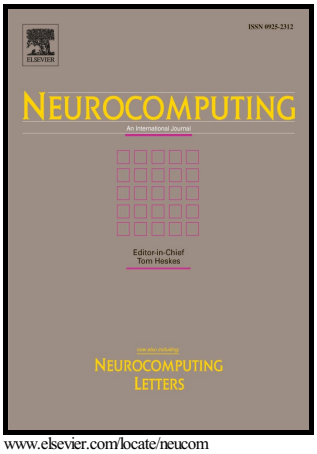

www.elsevier.com/locate/neucom

A New Big Data Storage Architecture with Intrinsic Search Engines

Jianjun Luo, Lingyan Fan, Zhenhua Li, Chris Tsu

PII: S0925-2312(15)01907-4
DOI: http://dx.doi.org/10.1016/j.neucom.2015.06.103
Reference: NEUCOM16498

To appear in: *Neurocomputing*

Received date: 14 March 2015
Revised date: 10 June 2015
Accepted date: 20 June 2015

Cite this article as: Jianjun Luo, Lingyan Fan, Zhenhua Li and Chris Tsu, A New Big Data Storage Architecture with Intrinsic Search Engines, *Neurocomputing* http://dx.doi.org/10.1016/j.neucom.2015.06.103

A New Big Data Storage Architecture with Intrinsic Search Engines

Jianjun Luo[a], Lingyan Fan[a,], Zhenhua Li[b], Chris Tsu[c]

[a] *Micro-Electronics Research Institute, Hangzhou Dianzi University, Zhejiang, 310018, China*
[b] *Zhejiang Vocational College of Commerce, Zhejiang, 310018, China*
[c] *Sage Microelectronics Corp., 910 Campisi Way, Campbell, CA95008, USA*

Abstract

The data storage system is central in determining the performance and cost in data mining or ITS. As the computing power of servers has increasedso have the problems caused by the bottlenecks from slower storage protocol interfaces, which restrict data throughput and the accessing raw data from the physical storage systems. This paper presented new big data storage architecture to optimize the efficiency of data mining or mass surveillance by integrating a distributed and embedded searching engine inside each storage drive. By integrating the intrinsic search engine (iSearch) into the core controller chip some of the work of searching for patterns and keywords takes place inside the drive freeing up resources of a higher level host and ultimately the server. Only those drives, in which the expected pattern or keywords were detected, are analyzed by the higher level host. Not only does iSearch free up the server for other high level computing tasks it also helps preserve as the bandwidth of the big data storage interface.

Keywords: Big Data, Data Storage, SSD, iSearch

1. Introduction

A disk array is used for computer servers or data centers. Increasingly cloud based applications and websites are placing ever greater demands on big storage systems. Intelligent-Transportation System (ITS) is one of the best applications for surveillance and is widely deployed in this fast growing vertical market [1][2]. Today's data bases are growing at exponential rates and tomorrows data storage systems will need to grow exponential to accommodate them. At the same time, cloud computing is seeking higher data processing abilities. High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning [3][4]. ITS or mass surveillance is trying to process huge files composed of video based information rather than that

Email address: fanlingyan@hdu.edu.cn (Lingyan Fan)

of text, file or document. For example, facer recognition technology is trying to detect and recognize a criminal wanted by the law enforcement among the thousands of people who are walking through a railway station or airport[5-9].

All these systems rely on disk arrays to store the raw data, and support the servers to search and analyze these raw data at higher and higher levels of performance. Fig.1 is a simplified architecture revealing a typical big data storage system. Redundant array of inexpensive disks (RAID) is the fundamental technology to provide data reliability and recover data from any disk errors [10]. Here RAID-5 is shown as an example. Disk 5 is a redundant disk for the parity. Each basic disk array (BDA) consists of five drives. The disk arrays are used to construct larger storage space by high level RAID and finally connected to servers by data switcher [11]. Each server has its own buffer to access the whole storage system. The serviers job is to load and the process this buffered information.

The searching process is executed as a server loading data from every disk and comparing data in the buffer, shown in Fig. 2. Assume each drive has the capacity of C_{Disk} in Fig. 1, and the number of BDA is N. The total capacity of this disk array C_{DA} is

$$C_{DA} = 4 \times C_{Disk} \times N \tag{1}$$

A single drive has a high speed interface such as ATA (Advanced Technology Attachment), Serial ATA (SATA) [12] , PCI (Peripheral Component Interconnect) or PCI Express (PCIE), etc. Assume SATA-III interface with 6 Giga bit per second (Gb/s) bus speed is applied here, a single drive's throughput is 600 Mega Byte per second (MB/s), which is the ideal speed without the discount of bus overhead, spindle speed of a hard disk drive (HDD) or programming wait period of a solid state drive (SSD). It takes time T_{Disk} to transfer the data in the whole disk to the RAID controller.

A basic disk array of RAID-5 can be PCIE interface. Assume PCIE-III interface applied with four lanes, BDA has the bus speed as fast as 3.2GB/s, which is faster than the striping speed of the five drives. It takes time T_{DA} to transfer all the data in a disk array. The top interface connected to a server should be fast enough to match the whole system performance, such as multiport 10Gb/s Ethernet or 40Gb/s Fiber Channel. It takes time T_{SW} to transfer all the data to the server buffer.

A big data storage system may have a scale of thousands of drives. For example, if a data base's size C_{Total} is 100 petabyte, it may be built up with 100,000 drives and each drive has 1 terabyte (TB) density. It is not simple to calculate the data load time by a formula of T_{Disk}, T_{DA} and T_{SW} because of parallel accessing.

If the server access the database by the speed of S_{Load}, it will take time T_{Load} to load raw data into buffer for searching:

$$T_{Load} = \frac{C_{Total}}{S_{Load}} \tag{2}$$

The serve executes the search operation, and it takes time T_{Com} which is the function of processor's speed S_{Com}

$$T_{Com} = \frac{C_{Total}}{S_{Com}} \tag{3}$$

Finally, it task time Tsys to complete the target search:

$$T_{Load} = C_{Total}(\frac{1}{S_{Load}} + \frac{1}{S_{Com}}) \tag{4}$$

Here, if the bottleneck is happened at the interface of BDA, S_{Load} is 3.2GB/s, then T_{Load} is calculated as 31,250 seconds. This is not a practical parameter in any systems. This is the original methodology to do big data mining. Many advanced technologies have been developed to improve it [8]. The basic idea is to reduce the data loaded into buffer. For example, file systems and indexes were applied to re-allocate the searching space. There were many algorithms presented and and proven efficient [14]. Applied with these technologies, the final result was so successful that the data mining or online website searching could be carried out in several seconds or minutes in most cases.

However, these methods were mostly developed as software algorithms and the server's computing resource was regarded as endless or free. This is not the real world. In the real world, neither the computing capability of a server's processors nor the buffer (cache) size is infinite. It is better to consider that both software and hardware methods are executed at the same time to reach the optimized result.

This paper presented a hardware method to reduce the raw data which a server has to directly deal with by distributed and embedded hardware searching engine in all the drives. For the sake of analysis please assume there are no file systmes, indeses or other software methods involved. So the new structure, shown in Fig. 2, can directly be deduced from the diagram in Fig. 1.

2. STORAGE SYSTEM WITH HARDWARE SEARCH ENGINES

The storage system architecture in Fig. 3 has the same blocks as that shown in Fig. 1. The difference is that each disk is no longer only a pure storage drive but also is embedded with the intelligent function of "searching". The first hard disk drive invented was really a pure data storage device. Although there was micro-processor unit (MPU) embedded inside a drive to take care of ATA command sets as well as the workload of management, this MPU or the device controller had never been assigned to any intelligent tasks other than storage.

The searching process was regarded as two parts shown in Fig. 4. The primary searching was running inside each drive independently, and no data was necessary to be load or read into server buffer. The primary searching reduced the searching range by hardware. Then the secondary searching running by the server itself focused on the locations reported by the primary searching with higher efficiency

It is possible that a SSD controller is embedded with a hardware engine which can monitor or detect indexes, keywords or patterns on the main data bus without disturbance to the data stream. This type of search engine can match the speed of the data throughput so no performance discount will be caused.

A SSD unit with intrinsic search (iSearch) engine can load (read) data into its own buffer while detecting the indexes, keywords or patterns. It is not necessary to transfer the data for a single disk to its higher level host. A host's first step is to issue the target contents, i.e. indexes, keywords or patterns, to all the lower level storage drives, and check the result reported from these drives.

Not all drives will detect and find the target contents in its stored raw data. For example, in Fig.2, Disk 4 in Basic Disk Array 2 reported no target contents detected. Such kind of disk is called irrelevant drive. Those drives, which were reported that iSearch engine found the contents, are called relevant drives. All irrelevant drives were primary searched by iSearch engine and judged no content potentially the server was looking for, thus, it was not necessary for a server to read and compare the stored information in these drives.

Y_{Drive} is defined as the ratio of relevant drives in a BDA. If R_{Drive} is the number of relevant drives in a BDA, Y_{Drive} is calculated as

$$Y_{Drive} = \frac{R_{Drive}}{\text{Total Drive Number}} \tag{5}$$

For RAID-5, there are 5 drives in a BDA, so

$$Y_{Drive} = \frac{R_{Drive}}{5} \tag{6}$$

There is one additional drive for parity, so the maximum number of relevant drive is 4, and. $Y_{Drive} \leq 0.8$. When $Y_{Drive} = 0.8$, it means that all drives in the BDA have no target contents detected, and such kind of BDA is called irrelevant BDA, shown as BDA N in Fig. 2. A server can ignore any read or compare activity to an irrelevant BDA and all disks inside this BDA.

As mentioned in Section I, some software algorithms were also developed with the similar functions, such as minimizing Y_{Drive} and predicting the target locations. For example, managing an index table for the database, or developing the database with dedicated structure, was proven practical, and actually widely applied. These methods all took a server's computing or buffer resource as well as the communication or peripheral bandwidth. Here, iSearch can carry out the primary searching without server resource involved. This will released more server capability in system level from being exhausted in data search, data collection, storage and communications.

Furthermore, it is not necessary for a server to dig the whole relative drive, only a certain range located at the address of every detected pattern need to be reloaded and compared. Assume this data size for secondary search step is D_{size} and the detected points in a disk is N_{Detect}, the total secondary data

volume should be search in total N basic disk arrays:

$$D_{server} = \sum_{i=1}^{R_{Drive}} (\sum_{j=1}^{5} Dij_{size} \times Nij_{Detect}) \tag{7}$$

So the total searching time T^*_{Load} is

$$T^*_{Load} = D_{Server}(\frac{1}{S_{Load}} + \frac{1}{S_{Com}}) \tag{8}$$

Therefore, the new method's search time compared with the original methods is

$$\frac{T^*_{Load}}{T_{Load}} = \frac{D_{Server}}{C_{Total}} < 1 \tag{9}$$

The less the target contents are distributed in drives, the shorter the search time will be. If the target contents are sparse, iSearch engine can provide highly efficient primary search results.

3. DESIGN A REAL EVALUATION DISK ARRAY

Fig. 5 is a SSD structure in which there is only a SSD controller and a group of flash memories. Most of the flash memory chips by the major global vendors are applicable. The interface is assumed SATA. Actually it can be any one of common storage interfaces, such as SCSI, IDE, SATA, PCI/PCIE, etc.

A single drive (disk) in Fig. 5 can execute search function inside the disk by the embedded iSearch engine. Its diagram is shown in Fig. 5. A host side controller is actually a device interface to be connected with a host. As discussed in Section I, it is SATA device controller as a typical case. The main controller has a core embedded micro-processor unit (MPU) with its peripheral registers or state-machines dedicated to configure the whole disk, set parameters for physical layer circuits and some storage oriented features such as S.M.A.R.T., i.e. Self-Monitoring, Analysis and Reporting Technology. It also takes care of iSearch engine by loading the target items, such as keywords and patterns, and reporting the result detected. The data storage controller transfer data to and from the media by driving spindles in HDD. The data storage controller changed the function in SSD to manage NAND flash memory chips by wear-leveling algorithms and error correction coding (ECC) [15].

The iSearch engine's only function is to search the targets contents (here keywords or patterns are applied for the evaluation purpose) in the data stream transferring though the high speed bus inside a storage controller. There are group of content detectors inside iSearch engine. Therefore, a group of vectors can be searched independently in parallel. Fig. 6 shows multiple patterns detectors searching patterns on the bus between storage controller and a drive's buffer.

A pattern or keyword detectorshown in Fig. 7, latches data from the bus and compares them with the latched contents. The result is reported to MPU.

In order to verify the proposed architecture, a SSD unit with SATA interface was designed based on Fig. 5. The storage media is an array of flash memory chips as shown Fig. 8. The data stream is stripped to multiple flash channels which are accessed in parallel to achieve the high speed matching that of SATA interface, up to 300MB/s for SATA-II protocol.

4. RESULT

A real storage system was built to evaluate the iSearch function and efficiency. A server was applied by running data search function, i.e. keyword search, to simulate the simple data mining operation.

Because iSearch engine is hardware based, FPGA (Field Programmable Gate Array) was firstly applied to setup the SSD disk with embedded iSearch engines but its timing parameter caused the bottleneck of performance for high speed throughput. Finally, a SSD controller was designed and turned into a real silicon chip with 110nm semiconductor process. Fig. 9(a) is an effect picture of single SSD with iSearch and Fig. 9(b) is a photo of a 2.5" SSD board mounted with the real silicon chip. A disk array with 24 SSD slots in a 2U standard rack is shown in Fig. 9(c).

The lab test was done by applying 24 iSearch embedded SSD units, and the basic performance, shown in Fig.10, was close to 1GB/s matching the maximum speed of the iSCSI interface connected by 10Gb/s Ethernet with a server.

The disk array was built up with 24 SSD units, and each unit had 1TB density. A 24TB data base was loaded to run the evaluation program. A complete search without iSearch engine involved is about 24 seconds according to formula (4). The curve of Original Method in Fig. 11 is the result when there were no iSearch engines enabled. The curve of test1 and test2 in Fig. 11 show the different curves when the target contents distributed in different density. All the target contents were distributed evenly in all the drives for the curve of test1, the search time was 5.5ms in average. When all the target contents distributed in the first RAID5 array, other array's Y_{Drive} was 0. The serve only need to search the first RAID5 array. So the search time is reduced. From these curves, it reveals that

(1) The original methods took the longest time, around 24 seconds. iSearch engines reduced the searching time to 5.5ms.

(2) With iSearch engines, the more target contents were distributed in database, the more times it took to found all these locations. The curve was almost linear.

(3) If the target contents were very popular distributed in the database, the iSearch efficiency became worse, and finally reduced to the original method.

5. Conclusion

A real silicon SSD controller chip was designed with embedded iSearch engines and applied to build a disk array for big data storage system. The search engines were distributed in each SSD unit. A server could issue primary search tasks to these engines and put them into parallel searching inside each drive without transferring data to a server. The preliminary search actions, finished by iSearch engines, helped a server to do the secondary searching more accurately and much less data accessing to the database. Therefore, iSearch engines built in each storage drive can make the data mining or data analysis faster in hardware method. This research work is still in its early stages. The next step is to combine with a real database and its optimized indexes or file systems, which is proven to work well as a software method. The ultimate objective is to find the optimal combination of software and hardware searching methods for the mining of big data.

This work was supported in part by Zhejiang Provincial Natural Science

Foundation of China under Grant No. LQ12F01001, and Zhejiang Province Science and Technology Innovation Focused Team Foundation under Grant No.2013TD03the National Innovation Fund for Small and Medium Enterprises Chinese under Grant No.13C26213302221.

References

[1] Ashok Srivastava and Mehran Sahami., 2009, Text Mining: Classification, Clustering, and Applications. Boca Raton, FL: CRC Press, pp. 20-35.

[2] Hemlata Sahu, Shalini Shrma and Seema Gondhalakar, 2011, A Brief Overview on Data Mining Survey, International Journal of Computer Technology and Electronics Engineering, 1(3), pp. 114-121.

[3] S.Hameetha Begum, 2013, Data Mining Tools and Trends -An Overview, International Journal of Emerging Research in Management & Technology, pp.7.

[4] Mohamed Medhat Gaber, Arkady Zaslavsky and Shonali Krishnaswamy, 2005, Mining Data Streams: A Review, SIGMOD Record, 34(2), pp. 18-26.

[5] Luming Zhang, Yue Gao, Yingjie Xia, Qionghai Dai and Xuelong Li, 2015, A Fine Crained Image Categorization System by Cellet Encoded Spatial Pyramid Modeling. IEEE Transactions on Industrial Electronics, 62(1), pp. 564-571.

[6] Luming Zhang, Yingjie Xia, Rongrong Ji and Xuelong Li, 2015, Spatial Aware Object Level Saliency Prediction by Learning Graphlet Hierarchies, IEEE Transactions on Industrial Electronics 62(2), pp. 1301-1308.

[7] Luming Zhang, Yingjie Xia, Kuang Mao, He Ma and Zhenyu Shan, 2015, An Effective Video Summarization FrameworkToward Handheld Devices. IEEE Transactions on Industrial Electronics 62(2), pp. 1309-1316.

[8] Luming Zhang, Yue Gao, Rongrong Ji, Yinjie Xia and Qionghai Dai, Xuelong Li. Actively Learning Human Gaze Shifting Paths for SemanticsAware Photo Cropping, IEEE Transactions on Image Processing, 23(5), pp. 2235-2245.

[9] Luming Zhang, Yue Gao, Yingjie Xia, Ke Lu, Jialie Shen and Rongrong Ji, 2014, Representative Discovery of Structure Cues for Weakly Supervised Image Segmentation. IEEE Transactions on Multimedia, 16(2), pp. 470-479.

[10] David A. Patterson, Garth Gibson, and Randy H. Katz, 1998, A Case for Redundant Arrays of Inexpensive Disks (RAID), University of California Berkeley, pp. 1-12

[11] Meeta Gupta, 2002, Storage Area Network Fundamentals, Cisco Press, pp. 20-55.

[12] Serial ATA International Organization: Serial ATA Revision 3.0, 2009.

[13] Goebel, Michael and Gruenwald, Le, 1999, A Survey of Data Mining and Knowledge Discovery Software Tools, SIGKDD Explorations, 1(1), pp. 20-33.

[14] Mohammed J. Zaki and Wagner Meira JR., 2014, Data Mining and Analysis-Fundamental Concepts and Algorithms, Cambridge University Press, pp. 40-66.

[15] Li-Pin Chang, Tei-Wei Kuo, 2002, An adaptive striping architecture for flash memory storage systems of embedded systems, IEEE Real-Time and Applications Symposium, pp. 187-196.

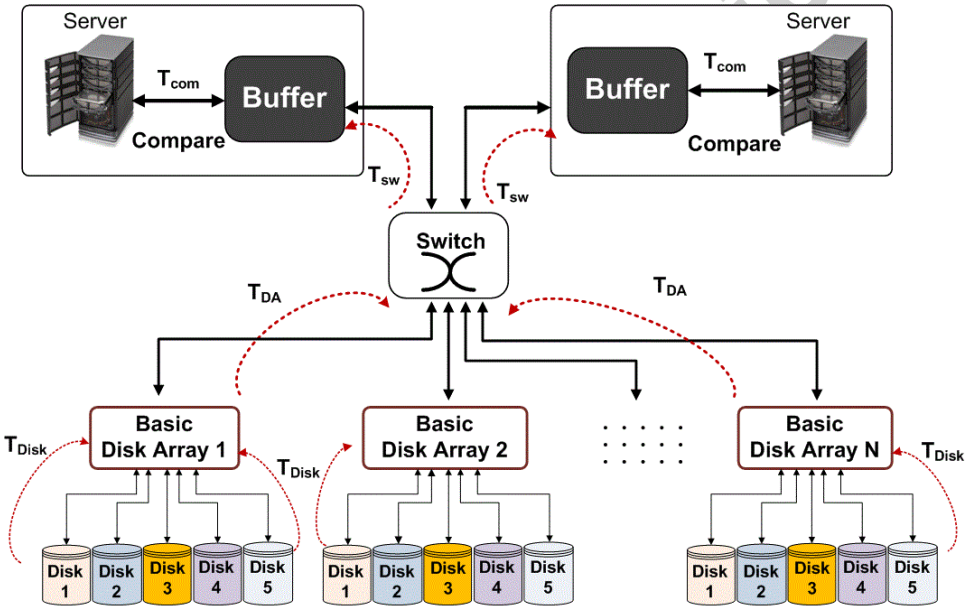

Figure 1: A simplified big data storage architecture.

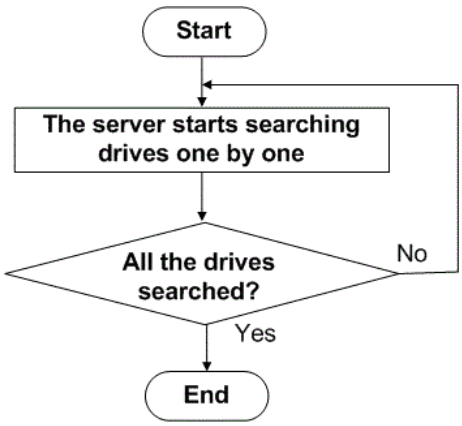

Figure 2: The flowchart of searching.

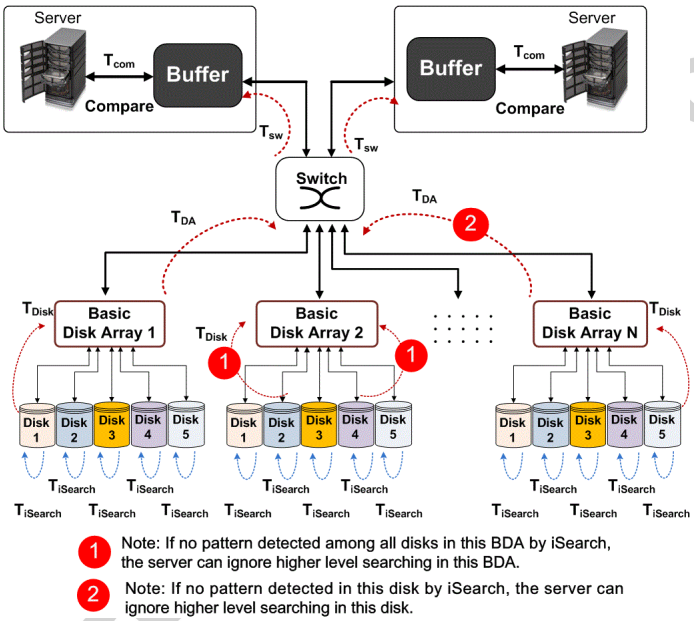

Figure 3: A Storage System with Distributed iSearch Engines.

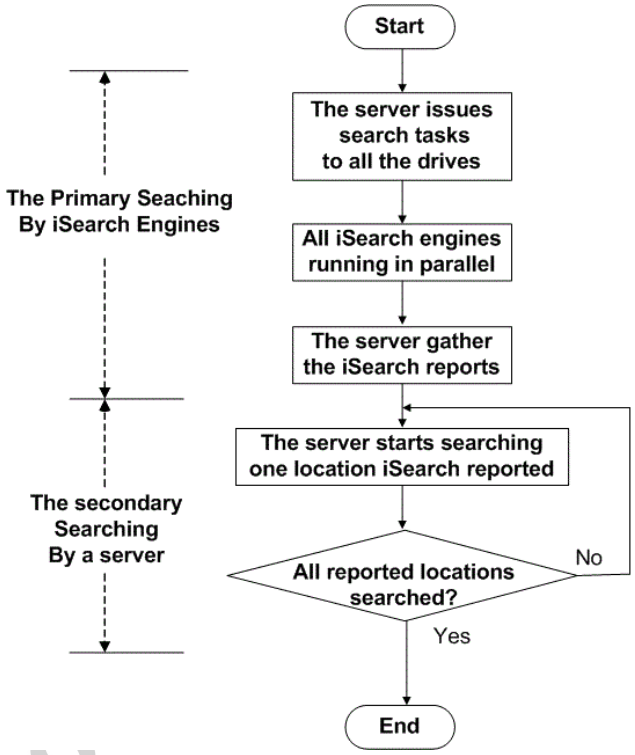

Figure 4: The flowchart of data searching with iSearch engines.

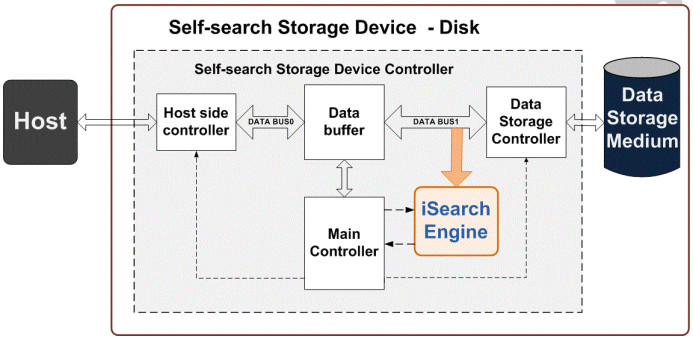

Figure 5: A single drive with iSearch engine embedded.

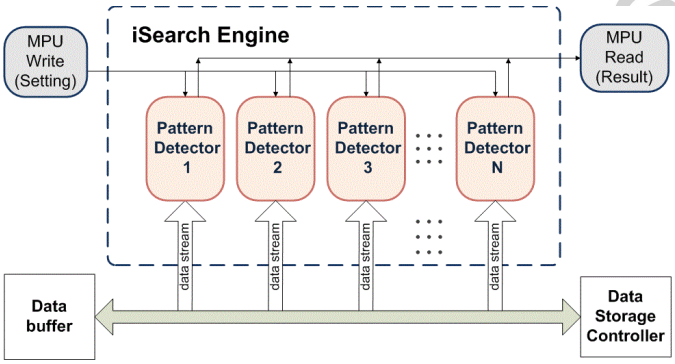

Figure 6: The iSearch engine is a group of pattern detectors.

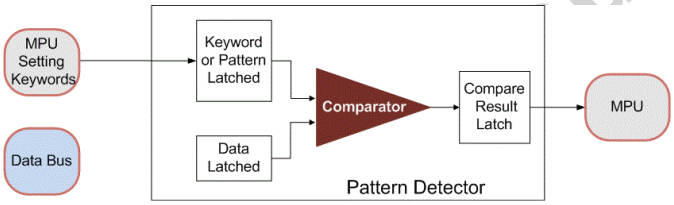

Figure 7: The structure of a pattern detector.

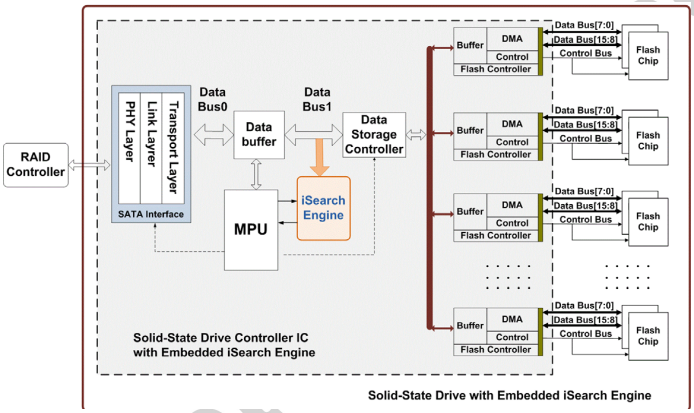

Figure 8: A single drive with iSearch engine embedded.

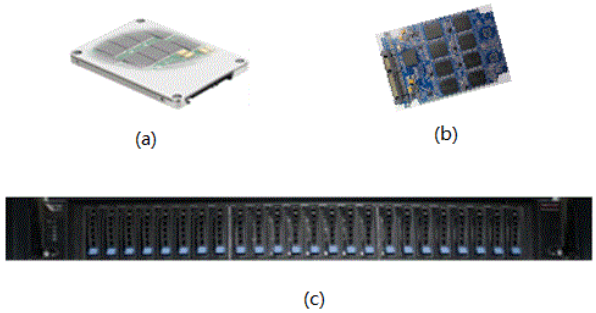

Figure 9: The real Silicon chip and SSD board, and disk array.

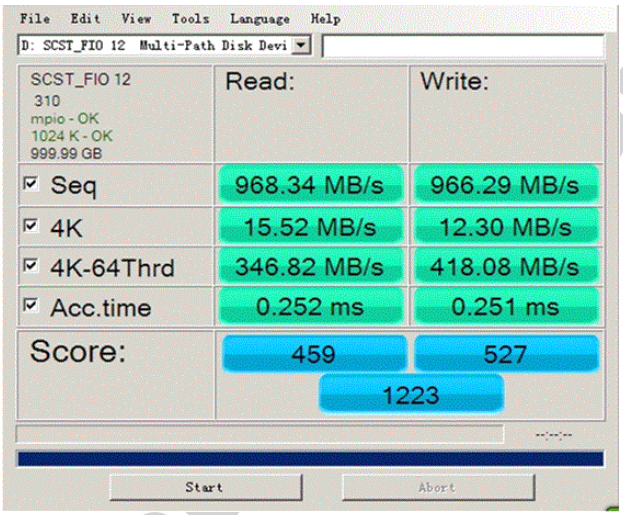

Figure 10: The performance test result of the disk array.

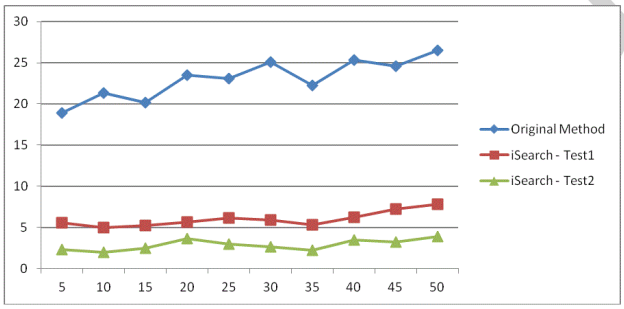

Figure 11: Search time results.