Neurocomputing 181 (2016) 147–152

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

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

ARTICLE INFO

Article history:
Received 14 March 2015
Received in revised form
10 June 2015
Accepted 20 June 2015
Available online 23 December 2015

Keywords:
Big data
Data storage
SSD
iSearch

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 increased so 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.

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 tomorrow's 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 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 servers 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 that 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

$$C_{DA} = 4 \times C_{Disk} \times N \quad (1)$$

A single drive has a high speed interface such as ATA (Advanced Technology Attachment), Serial ATA (SATA) [12] , and PCI (Peripheral Component Interconnect) or PCI Express (PCIE). Assume that 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 that PCIE-III interface applied with four lanes, BDA has the bus speed as fast as 3.2 GB/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 multi-port 10 Gb/s Ethernet or 40 Gb/s Fiber Channel. It takes time T_{SW} to transfer all the data to the server buffer.

* Corresponding author.
E-mail address: fanlingyan@hdu.edu.cn (L. Fan).

http://dx.doi.org/10.1016/j.neucom.2015.06.103
0925-2312/

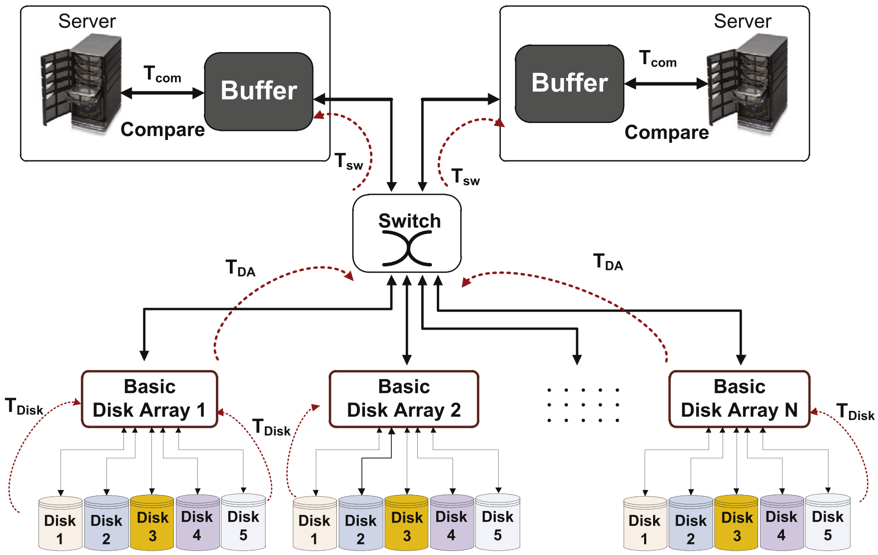

Fig. 1. A simplified big data storage architecture.

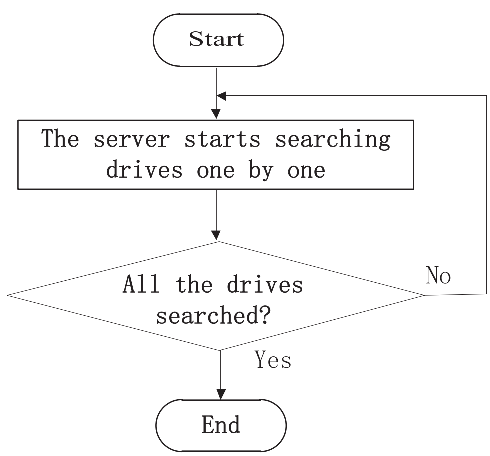

Fig. 2. The flowchart of searching.

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:

$$T_{Load} = \frac{C_{Total}}{S_{Load}} \quad (2)$$

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

$$T_{Com} = \frac{C_{Total}}{S_{Com}} \quad (3)$$

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

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

Here, if the bottleneck is happened at the interface of BDA, S_{Load} is 3.2 GB/s, then T_{Load} is calculated as 31,250 s. 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 [13]. 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 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 that there are no file systems, indexes 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