International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

General Article

Design of Swap Space in Virtual Memory Management System

Amandeep Singh[Á*], Gulshan Goyal[Á] and Jasneet Kaur[Á]

[Á]Department of Computer Science and Engineering, Chandigarh University, Punjab, India

Accepted 30 May 2014, Available online 01 June 2014, **Vol.4, No.3 (June 2014)**

Abstract

Swap space is storage space on any storage media (like a disk or flash storage) which is designed to reduce the scope of large disk optimization. The pages are swapped out to swap space from RAM periodically with different page replacement policies (LRU, FIFO, Workset etc.). In present systems, disk or flash drives are used to design swap space. Present paper reviews the design of swap space on disk and flash media. Based on the review, MFURVSS (Most Frequent Used Page Replacement to Volatile Swap Space) approach is proposed to design a volatile swap space using main memory. This approach will be beneficial to reduce the number of page faults as compared to traditional approaches.

Keywords: *The Swap Space, MFURVSS, Page Replacement, Regular Segment, Swap Segment.*

1. Introduction

Memory is an important part of computer system and it should be carefully managed. Management of memory is the sole responsibility of an operating system. Swap space has an important role in memory management system, Swap space is storage space designed on any storage media (disk or flash drive) which is designed to reduce large disk optimization. The pages are swapped out to the swap space from RAM periodically. The pages are replaced to swap space because of its less usage. Replacement is done using different page replacement policies (like LRU, FIFO, Clock, Second chance, Workset etc.). In common computer system architectures, disk storage is used to design swap space. According to the memory hierarchy disk storage is slow in data access operations. Disk takes much time to perform read and write operations because of its high read write latency. Transfer rate of disk storage is very slow as compared RAM and Flash storage. So if a page fault occurs then it takes much time to swap in the page from swap space to RAM. Large number of page faults can degrade the performance of the system. In hot researches, flash storage is very popular in the memory hierarchy as storage media. Flash storage has low read and write latency as compared to mechanical disk storage. The transfer rate of the flash drive is better as compared to disk storage. Flash drives are cheaper than RAM. In recent research flash memory is used to design swap space in Virtual Memory Management System but still flash memory faces the problem of a limited number of writes to each block of flash.

The present paper proposed a MFURVSS (Most Frequent Used Page Replacement to Volatile Swap Space) approach to design a volatile swap space using main memory. To design volatile swap space, the main memory is divided into two fixed segments, the first segment is known as *Swap Segment*, and the second segment is known as *Regular Segment*. One fourth portion of main memory is assigned to Swap Segment and remaining portion of RAM is assigned to Regular Segment. A Regular Segment portion is used for transactions on pages as traditional methods. Swap Segment is backed up to nonvolatile storage buffer periodically as the data of Swap Segment should be safe after switch off the power supply. Swap Segment data is prefetched during the booting process of the system. To improve the performance of the system MFURVSS approach is used to swap the frequently used pages to the Swap Segment, so that only most frequent pages are prefetched during system booting. This approach will be beneficial to reduce the page faults.

Further sections of the paper describe the related work, the basic concept of design of swap space using disk and Flash and proposed work.

2. Related Work

The first virtual memory machine was developed in 1959. By the late 1970's the virtual memory system was most popular in commercial computer. In 1985 Intel offered virtual memory and cache in the 386 microprocessor and Microsoft offered multiprogramming in Windows 3.1. And after that virtual memory found its place in our everyday lives

Jacob B. *et al.* described the concept of virtual memory. they described the basic architecture of a virtual memory management system and compared the memory

*Corresponding author **Amandeep Singh** is a Research Scholar; **Gulshan Goyal** and **Jasneet Kaur** are working as Associate Professor and Assistant Professor respectively.

management design in three commercial architectures (Power PC, MIPS, x86). They presented a study on address translation without any specific hardware support. Address translation though specific hardware consumed more power supply during translation. Elimination of TLB reduced the power supply in significance manner.

Li H.L. *et al*. described different features of flash memory. This paper focused on energy efficient aspect. They proposed three energy efficient techniques for flash memory management in the virtual memory management system. These techniques are: Sub paging, Hot cache scheme and DA-GC. An average Energy saving of these three techniques was 42.2%.

Saxena M *et al*. described some issues related to the use of flash as virtual memory. They introduced a new system design that was FlashVM to efficient use of flash as virtual memory. They used FlashVM for virtual memory management in Flash. FlashVM focused on three main aspects of flash: Performance, Reliability, Garbage collection. FlashVM facilitated 94% reduction in execution time, efficient reliability and 10 times faster garbage collection than previous approaches.

Ji S. *et al*. proposed a technique which exploits data redundancy between the main memory and flash memory in the virtual memory management system. This approach was used to minimize the garbage collection overhead. The proposed technique improved the performance by 37% on average compared to the previous techniques.

Sudan K. *et al*. in their paper NAND-Flash: Fast Storage or Slow Memory? described two state of art systems: Hybrid storage system using NAND Flash and Hybrid main memory using NAND Flash.

Pham, B. *et al*. proposed Coalesced Large-Reach TLBs (CoLT), which coalesce multiple virtual to physical page translations into single TLB entries. They proved that CoLT implementations eliminate 40% to 58% of TLB misses on average, improved performance by 14%.

Shi L. *et al*. proposed cooperative management schemes for virtual memory and write buffer. I/O performance and reduction of the number of erase and write operations can be achieved compared to the other approaches.

Basu A. *et al*. described the mapping part of the process's linear virtual address with direct segment rather than pages. Direct segment used minimal hardware base, limit and offset register per core to map the contiguous virtual memory regions directly to the contiguous physical memory. The proposed approach reduced the execution time wasted on TLB misses to less than 0.5%.

3. Swap Space Using Disk and Flash

Initially, the only disk drive was used to design swap space. It means the virtual address space is designed with the help of disk storage. In recent research, flash storage is also used to design swap space in a virtual memory system.

3.1 Swap Space using Disk Storage

In traditional computer system architectures, a portion of the disk storage is used to design swap space. The page size of virtual address space or swap space is equal to the physical address space or actual memory address space. Mapping of virtual addresses to physical address is the responsibility of a Memory Management Unit (MMU). Each process has its own page table which is used to identify the page frame in main memory corresponding to a particular page. Large number of pages of a particular process can cause of a large number of page table entries which are difficult to locate in main memory. Searching of particular page table entry is a very time consuming process. To overcome these problems high speed buffer used which contains a small amount of page entries. This buffer is known as TLB (Translation Lookaside Buffer). Initially TLB was a small hardware device which consumed more power supply during address translation. Software managed TLBs are used to reduce the power consumption.

Some of the common page replacement policies are used as a FIFO, LRU, NRU, Clock Page Replacement, Second Chance Page Replacement, and Working Set Page Replacement etc.

3.2 Swap Space using Flash Storage

Flash storage is faster in read and writes operations as compared to disk storage. In Window Vista, new feature was introduced known as ReadyBoost. It gave the facility to attach a thumb drive into a PC and use its capacity as cache, to make things faster.

The architecture of flash memory storage system consists of

1) Flash Translation Layer (FTL) which is responsible for data allocation on flash memory and garbage collection.
2) Memory Technology Device (MTD) driver which handles routines to read, write and erase operations between FTL and flash memory.
3) Flash chips as storage media.

The block size of the typical NAND flash is 16KB and page size is 512B. It means if system follows 4KB page of RAM, and 512 B page of flash, then each page fault required eight read/write operations to flash memory. The main problem occurs using flash is limited write cycles per block of flash memory. Typical flash devices can limited up to 100,000 to 1 million overwrites. To manage the limited write operations the write buffer used. To reduce the write operations in flash based system Han-lin-li *et al*. proposed a HotCache technique, in which SRAM was used as cache to keep frequent writes. Every write is cached in HotCache and page replacement was based on the multiplicity of the timestamp and frequency factors of the page. Lower weight pages after multiplication are considered as evicted pages. Different write buffer techniques are used to improve the write performance of the flash memory. Recently, Write buffer aware virtual memory management approach is introduced by Liang Shi *et al*. They used Write Buffer aware LRU (WBLRU) policy to improve the write performance of flash based virtual memory system.

4. Proposed Work

In this paper, MFURVSS (Most Frequent Used Page Replacement to Volatile Swap Space) approach to design swap space using main memory to minimize the page faults is proposed. The Microsoft Windows operating system is hardware independent and it also has easily accessible and user friendly graphical user interface. Due to these two features of Microsoft Windows, it becomes very popular in personal use computers as well as in industrial environment. So here, the memory usage and CPU utilization are identified. Present work, describes the memory usage and CPU utilization as running the following applications on Microsoft windows 7 (32 bit operating system) is identified:

1) Google Chrome (10 different Web Pages).
2) Microsoft Visual Studio 10
3) VM-Ware Workstation.
4) MS Office Word 2007.
5) MS Office Excel 2007.
6) Paint.
7) Adobe Reader 10.0.

Table 1. Memory usage and CPU utilization

RAM Capacity	No. of Processes	Memory Usage	CPU Utilization
2 GB	75-78	65-70%	1-10%

The resulting performance of the system including memory usage and CPU utilization is shown as Table I. this performance is identified with the help performance manager feature of the Microsoft Windows Operating System.

4.1 Overview

To design volatile swap space, the main memory is divided into two fixed segments, the first segment is known as *Swap Segment* and the second segment is known as *Regular Segment.* The block diagram of volatile swap space shown in figure 1.

The memory is segmented on the basis of memory usage. Here, one fourth portion of the memory is allocated to Swap Segment and remaining portion of the memory is allocated to Regular Segment. Backup storage contains the backup of the pages of the Swap Segment.

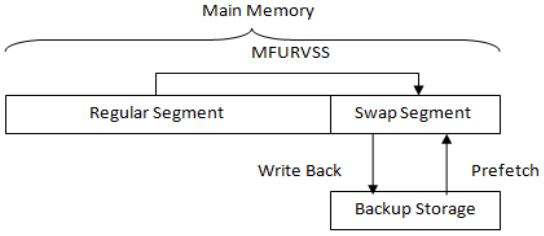

Figure 1 Block diagram for volatile swap space.

These pages are reviewed and changed periodically as per changing the content of Swap Segment. Frequently used pages are swapped to the Swap Segment periodically. These pages are replaced by comparing with minimum used pages in Swap Segment. If minimum used pages in the Swap Segment are less frequent than maximum frequent used pages in Regular Segment then pages of the Regular Segment are replaced with the less frequent page of the Swap Segment. It means Swap Segment contains only most frequent pages. Swap Segment prefetched during the booting process of the system.

4.2 MFURVSS

MFURVSS means Most Frequent Used Page Replacement to Volatile Swap Space. This policy used to swap most frequently used pages to Swap Segment. Using this policy most frequently used pages of Regular Segment is compared with less frequent used pages of Swap Segment. If less frequent used pages in the Swap Segment is less frequent than most frequent used pages in Regular Segment then most frequent pages of the Regular Segment is replaced with the less frequent page of the Swap Segment. Frequently used pages are identified by the time and frequency factors of the pages. The time-frequency policy was used by Li H.L. *et al*. in their research to identify less frequent used blocks in HotCache for replacement purpose. Here, time factor is based on the timestamps. Suppose a first page came at timestamp T1, after that second page is coming at timestamp T2, then T1 will be lesser than T2 (T1<T2). Frequency is measured by hit counts of a page in a particular time interval. Replacement is based on the calculative weight by multiplying the timestamp value and frequency count. Pages with maximum weight value in Regular segment are replaced with the Pages with minimum weight value in Swap Segment. The only condition is that the minimum weight value of the pages in Swap Segment is also lesser than the maximum weight value of the pages in Regular Segment.

4.3 Prefetching

Backup Storage contains the backup of Swap Segment pages. It is periodically changed as per the changes in the content of Swap Segment. Pages from the Backup storage are prefetched during the booting process of the computer system. Backup storage can be reserved area on disk. To speed up the booting process, other fast storage media like flash memory or Phase Change Memory to backup the Swap Segment pages can be used.

Conclusion

With existing technologies, disk storage and flash storage are used to design swap space. There are different page replacement policies are used to swap out and swap in operations. If a page fault occurs then it takes much time to read a page from disk and flash storage. To reduce the numbers of page faults proposes a MFURVSS approach to design swap space using main memory. It will be significantly reducing the number of page faults and improve the performance of system and application startup.

References

Basu A., Gandhi J. Chang J. Hill M.D. Swift M.M. (2013), Efficient Virtual Memory for Big Memory Server. ACM SIGARCH Computer Architecture News – ICSA'13, Vol. 41, issue 3, pp. 237-248.

Dan's Data. et http://dansdata.com /flashswap.htm.

Jacob B. and Mudge T. (1998) Virtual memory: Issues of implementation, IEEE Computer, vol. 31, no. 6, pp. 33-43.

Jacob B. and Mudge T. (2001), Uniprocessor virtual memory without TLBs, IEEE Transactions on Computers, vol. 50, no. 5, pp. 482-499.

Ji S., Shin D (2010). An Efficient Garbage Collection for Flash Memory-Based Virtual Memory Systems, Consumer Electronics, IEEE Transactions, vol. 56, issue 4, pp. 2355-2363.

Li H.L., Yang C.L., Tseng H.W. (2008), Energy-Aware Flash Memory Management in Virtual Memory System, IEEE transactions on very large scale integration (VLSI) systems, vol. 16, no. 8, pp. 952-964.

Lim G., Min C., Eom Y.I. (2013), Virtual Memory Partitioning for Enhancing Application Performance in Mobile Platforms, IEEE Transactions on Consumer Electronics, Vol. 59, No. 4, pp. 786-794.

Pham B. *et al*. (2012). CoLT: Coalesced Large Reach TLBs, Proceedings of 45th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 258-269.

S. Tanenbaum A , Memory Management, in Modern Operating Systems, 3rd ed., USA, Pearson Education, 2009, pp. 173.

Saxena M and Swift. M.M. (2009), FlashVM: Revisiting the Virtual Memory Hierarchy, Usenix Workshop on Hot Topics in Operating Systems. vol. 34, no. 4.

Saxena M and Swift. M.M. (2010), FlashVM: virtual memory management on flash, USENIX annual technical conference, pp.187-200.

Shi L., Li J. *et al*. (2013), Cooperating Virtual Memory and Write Buffer Management for Flash-Based Storage Systems, IEEE transactions on very large scale integration (VLSI) systems, vol. 21, no. 4, pp. 706-719.

Sudan K, Badam A, Nellans D (2012), NAND-Flash: Fast Storage or Slow Memory? Non-Volatile Memory Workshop (NVMW-2012) , San Diego.

History of Virtual Memory, The Core of Information Technology, George Mason University. http://www.cs.gmu.edu/cne/itcore/virtualmemory/vmhistory.html