International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume: 3 | Issue: 4 | May-Jun 2019 Available Online: www.ijtsrd.com e-ISSN: 2456 - 6470

An Efficient Virtual Memory using Graceful Code

Divya YA

Assistant Professor, Department of Information Science and Engineering,
GSSS Institute of Engineering and Technology for Women, Mysuru, Karnataka, India

How to cite this paper: Divya YA "An Efficient Virtual Memory using Graceful Code" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-3 | Issue-4, June 2019, pp.623-626, URL: https://www.ijtsrd.com/papers/ijtsrd23878.pdf

IJTSRD23878

ABSTRACT

Memory is hardware that is used by computer to load the operating system and run programs. It is buildup of RAM chip that has different memory modules. The amount of main memory in a computer is limited to the amount of RAM that has installed. Generally memory sizes are 256 MB, 512 MB, & 1 GB, because of computer has limited amount of RAM. When too many programs are simultaneously it is possible to run a program out of memory. This is the concept where virtual memory comes. Virtual memory enhance the available memory of a computer has by enlarging the address space or place in memory where data can be stored. Hard disk is used for additional memory allocation .However, since secondary storage is much slower than the RAM, program which is in Virtual Memory must be mapped back to virtual memory in order to be used. The process of mapping data and forth between the hard disc and RAM takes longer than accessing it directly from the memory. It means virtual memory is increased, the more it will slow your computer down. While virtual memory enables your computer to run more than one program it could, otherwise it is the best way to having as main memory as possible.

Keywords: *Addressing, Mapping, Swapping, Graceful code, Segmentation*

I. PROBLEM DEFINITION

We are having 64 MB Main Memory and 1 GB Virtual Memory. To run a program we need 512 MB RAM.

Here 32 MB of Main memory is reserved for operating system and application files, so free space is 32 MB. Program an active portion is only loaded in main memory that is 33 to 64 MB .Hence remaining program (passive portion) i.e. 512-32 MB loaded on Virtual memory. To keep Track of this file active and passive portion of a program GCC is coming into this picture. Save 128 MB Active location is (128*1024) is addressed onto the virtual memory by GCC through its constant factor.

II. INTRODUCTION

Virtual Memory is a main factor of operating system that provides all the facilities of a process to use RAM (memory address space) that is completely independent of other process running simultaneously and uses space that is larger than the actual amount of RAM. Virtual memory combines active RAM and inactive hard disk to form a larger range of contiguous address. Virtual memory is a technique used to develop for multi-tasking. Virtual memory allowing designing a program that behaves like directly addressable memory RAM. Virtual memory specifies each application program easier by hiding fragmentation of main memory or to access memory with relative addressing. Virtual memory is generally a concept of generalization of memory virtualization. Virtual memory used not to extend RAM, but to make an extension as easy as possible for users to use.

Virtual memory is specialized to design to automate the movement of data and code between secondary memory and RAM to give the appearance of a single large store. When program code exceeds the main memory size, this technique is used to simplify the programmer‟s job. Virtual memory eliminates external fragmentation and minimizes internal fragmentation.

III. LITERATURE REVIEW

Liang Shi et al. [1] presented the main use of flash memory is being widely used in mobile devices and embedded system .Flash memory is small and light weight from factor, low power consumption and shock resistance ideal candidate in replacing traditional hard disk as the storage device flash memory emphasis asymmetric speed of read and write operations .Write operation on flash memory is related with erase operation. In Flash memory based system virtual memory management is use to reduce the number of write activities and improve input /output performance traditional management strategies and virtual memory are designed based on hard disk as storage system.

Flash memory can reduce the page swapping cost significantly, Flash memory is a good device for use as swap space in virtual memory .Flash memory supports read, write and erase commands. A flash memory page if it is already been written It cannot be overwritten, and the corresponding block should be erased before data is written to the page. These constraint are known as „erase- before-write‟ constraint proposed in Seunggu Ji et al. [2] In order to reduce the execution cost of a program, a data segmented program and program‟s code is to be rearranged in virtual

address space, this technique is called as program restructuring. Virtual memory is used to increase the capabilities and, makes potential computer system, and is divided into two parts hierarchical organization built up of small, fast primary memory and large slow secondary memory. In Paged virtual memory system, a program‟s or user application‟s virtual address space divided into matching size page and main memory is divided into equal size chunks called page frames. The main object of program restructuring increases page memory utilization decrease the number of page faults and space time execution cost of executing program. A static program restructuring is accomplished at before loading time, information collected by interpreter described in Stephen J. Hartley et al. [3].

Steven P.Smith & John Kuban et al. [4] describe the performance of page based virtual memory system is influenced by page fault frequency. Now inestimable energy has gone to decrease the hit ratio of virtual to physical address mapping. Imaginary memory is typically divides into pages of equal size, and transfers from disk into physical memory has made in units of pages. When a virtual address referenced which is not currently in physical memory, a page replacement algorithm is used to determine which page currently in main memory is replaced by referenced page. Simulation references behavior is quantified through analysis of address traces taken from APOLLO then address traces were collected using I.M.S (integrated measurement system) .The address traces were used in virtual memory system using the LRU replacement algorithm. Through software modeling approach page size and physical memory capacity can be easily varied to access the effect on overall performance.

Rafal Kolanski et al. [5] Presented Virtual memory or Imaginary memory is a concept of secondary memory, it is treated as primary memory. Virtual memory is used to complete the process when RAM is not enough to run the process that time we requires secondary memory that becomes primary memory until the process is not completed, called virtual memory.

Bensoussan, R. C. Daley et al. [6] presented Multics emphasis direct to hardware addressing by user and application and system programs of all information, that is independent of its physical storage. A no. of techniques is being used by 2.6 Linux kernel that improves the large amount of memory .This article emphasis on some important changes includes reverse mapping that includes for page reclaim, large memory pages, storage of page table entries in memory. Memory which is used by kernel increases efficiency, flexibility and stability of memory manager. In the earlier system it has been a need for more memory than exist physically in a system. To overcome this drawback, a successful aspect we get that is virtual memory. Kernel is used to write the contents of currently unused block of memory to the hard disk .so the memory can be used for some purpose presented in Hartley S.J. [7].

The early days of computing, Accounting to the size of program, the limited amount of physical memory posed the evaluation of Virtual Memory. The management of physical memory got hidden inside the operating system, and an address space backed by physical memory and secondary storage in which the programs were placed. Paging a concept of memory can be used to free by the application, if it is involved in physical memory. Most of the applications use algorithms that are use a as elastic in terms of their ability the usage of other resources. Database and web browsers maintain the memory cache of disk can increase performance by enlarging the cache. A useful extension model of this preferences uses page replacement algorithm, so that real time applications only when only when memory pressure is secured, and daemons becomes alternative users for page eviction presented in Sita

ram Iyer [8]. Yousef A.Khalidi et al. [9] presents an Implementation of Virtual Memory in current system such as SUNOS, VMS, NT, MACH and CHORUS share two concept regarding Main Memory Management. There is one page size, each size may be multiple of Main Memory Unit page size, each page size range can be 512 – 8k bytes. Physical Memory is not very large. Somewhere like the range o0f 4M-256 M bytes. Main Memory Replacement algorithm are normally turned for the common size. TLB (Translation look aside) buffer is a cache of virtual to physical address translations. It is typically used to reduce the average address translation time. When required Translation is not in TLB a software and a hardware miss handler is executed to enter the translation in TLB.

In embedded system virtual memory has been known as elegant mechanism for transparent hardware resources sharing and utilization. The address space accessed by the program is referred to as virtual address space and that is divided into equally size virtual page, which are converted into actual physical page .Each and every virtual page is identified by a set of significant virtual address bits also known as virtual page number i. similarly virtual pages are identified by their physical corresponding physical page number .Many contemporary high ended processor such ARM9 & X scale ,offers a hardware that supports for virtual memory in the form of main memory unit which captures the most frequent address translation. System software usually maintain a data structure which provides the mapping between virtual pages in the application address space and frames in the physical memory. This page table is treated as data structure, is always used in tabular form which occupies a signify memory and traverse hardware .Main Memory Unit cannot Provide the physical address when it is missed presented in X.Zhou and P Petrov [10].

Hung-Wei Tseng et al. [11] presented the effects of the subpaging method and storage cache management. A full page is written back to the secondary storage on a page fault in the traditional virtual memory system. M. Huang et al. [12] describes an energy-management framework that challenges both energy efficiency and temperature control in a unified manner. This approach is called Dynamic Energy Efficiency and Temperature Management (DEETM). Framework combines several energy-management techniques and can activate them groups or individually in a fine-grained manner according to a given rule.

R.M. Jones [14] compares results that have been attained for several virtual memory swapping algorithms. Algorithms were tested as software components of a multiple computer on-line system. C. Park et al. [15] present energy-aware demand paging technique to decrease the energy consumption of embedded systems seeing the characteristics of interactive embedded applications with large memory footprints. And also describes a page replacement policy which can decrease the number of write and erase operations in NAND flash memory.

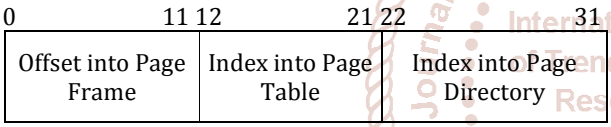
Hiroshi Tezuka et al. [16] illustrate pin-down cache technique for zero-copy message communication. It reclaims the pinned-down area to decline the amount of calls to pin-down and free primary one. This technique applied in the low-level communication library on the RWC PC Cluster II.

Giuseppe Psaila [17] presents a Virtual DOM which is a Java package provides an effective representation technique for large XML documents. It accepts a specifically designed virtual memory technique- memory blocks allotted to denote the document are exchanged by skipping the operating system swapping mechanism. Actual main memory needs are under control, the thrashing phenomenon is avoided even for large documents.

IV. ADDRESS MAPPING AND TRANSLATION

Virtual Memory mapping using graceful code is the diversity of communication model .It provides concurrent access the active portion of a program which is in main memory, and provides a way to run the program successfully with the help of virtual memory.

Another feature of address space is mapping and translations, often consist of number of layers. It means that higher level must be translated to lower level ones in some way. For example: on a logical disc file system operates linear sector numbers, which have to be translated. Address Mapping maps logical address to physical address presented in Dr. Vivek Chaplot [18]. CPU scrutinizes any virtual address which classify the address into three fields



1. Page Frame- This field occupies 12 bits. It provides offset to one of 4096 bytes in Page frame
2. Page Table -This field occupies 10 bits. It selects one of 1024 array entries in page table
3. Page Directory This field occupies 10 bits. It selects one of 1024 array entries in page directory

Address in virtual memory consists of logical_page, offset and page_size. Logical_page denotes page number within the logical address space, offset defines the offset into that page and page_size means the size of the page (which is a multiple of 2).

In Figure 1. P defines Page Number, O denotes Page offset and F means Page Frame. There are two types of Virtual address translation

1. Virtual Address Translation Using TLBs

When a program makes a memory reference, the virtual address directed to the TLB to determine if it contains a translation for the address presented in Yaman Cakmakci et al. [19]. For TLB hit, returns the physical address of the data, and the memory reference continues. For TLB miss, system searches the page table for the translation.

2. Virtual address translation using page registers

Register in each frame comprising three bits. Residence bit shows whether or not the frame is occupied, Occupier defines page number of the page occupying and Protection bits described in Hanna Alam et al. [20]

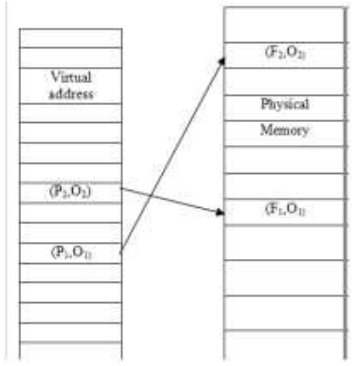


Figure1. Maps Virtual address to Physical address

V. SOLUTION METHODOLOGY

Virtual address space is memory mapping mechanism available in operating system, which provides a relationship between physical memory and virtual memory presented in Andon coleman et al. [22].

It provides security through process isolation. An address generated by process is called logical address (virtual address) and is mapped virtual address space.

Address space defines a range of discrete addresses, each of which may correspond a network host, peripheral device, disk sector or logical and physical entity presented in Pinchas Weisberg et al. [21].

In the below figure we are having 64 MB Main Memory and 1 GB main memory. Here our program length is 512 length. To run a program we need 512 RAM. But here available RAM is only 64 MB. Where 32 MB is reserved for operating system and application files. And remaining space is lifted of 32 MB. So current program"s active portion is only loaded in main memory that is 33 to 64 MB. Hence remaining current program„s passive portion is (512-32) MB loaded on virtual memory. Active and passive portion of current program is completely divided in maim memory and virtual memory respectively. To keep track of active and passive portion of a program, we use graceful code.

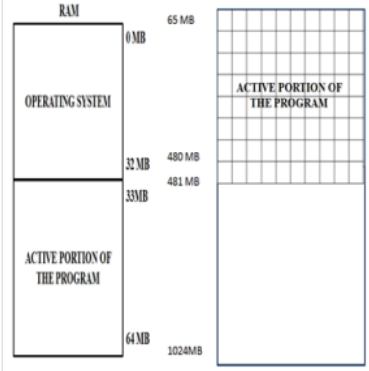


Figure2. Virtual memory mapping using graceful code

To find the actual address of an instruction using formula: Actual address of an instruction = Length of instruction /number of blocks

VI. ALGORITHM FOR GRACEFUL CODE

STEP -1. Ask to user the length of a program.
STEP -2. Check size of RAM.
STEP -3. Check size of Virtual memory, (RAM <VIRTUAL MEMORY).
STEP -4. Enter the user instruction address.
STEP -5. Check whether it is in RAM or in Virtual memory.
STEP -6. Fetch the address through the constant factor with Factorials.

STEP -7. (a*0!) + (b*1!) + (c*2!) + (d*3!)+……..
STEP -8. Find out the constant values of a, b, c, d….
STEP -9. Identify the location.
STEP -10. End.

VII. ADVANTAGES OF GARCEFUL CODE

Virtual Memory using graceful code provides several advantages over the traditional, Kernel dispatch- based message passing. Main advantages of virtual memory mapping communication is to perform low overhead communication since data can move between one process to another without message dispatching and context dispatching. Another advantages of virtual memory mapping is that it moves the memory buffer management to user level. Graceful code provides such active portion of running program with the help of virtual memory.

VIII. RESULT

Here Main memory is showing active part of a running program and the remaining portion of program is in virtual memory that performing a role to complete the program.

Input		Output	
M.M	V.M	M.M	V.M
64 KB	1024 KB	0A 0B 2C 2D 0E 0F 0G 5H 1I	0A 0B 2C 2D 0E 2F 0G 0H 8I 2J
16 KB	256 KB	0A 0B 2C 2D 2E 4F 1G 3H	0A 0B 2C 2D 2E 4H 6I
8 KB	128 KB	0A 0B 1C 1D 1E 2F 4G 1H	0A 0B 1C 1D 1E 2H 3I

CONCLUSION

This paper describes efficiency and implementation of virtual memory using graceful code. The virtual memory is a mapping that provides direct data transfer between receiver‟s virtual address space and sender‟s virtual address space. This code eliminates operating system involvement in communication, supports user buffer management, zero copy protocol, provides protection and minimizes the software overheads associated with implementation communication. With this work we provide a first implementation of graceful code on commercially available hardware platform.

With this work we prove that Virtual memory graceful code is a fairly portable graceful code, not tried to one particular hardware platform. A better approach of virtual memory graceful code implementation at the cost of a specialized network interface and more operating system modification.

REFERENCES

[1] Rafal Kolanski, “A logic for virtual memory“, Electronic Notes in theoretical computer science 217, Pages 61-77, 2008.

[2] Bensoussan, R.C.Daley, ”The multics virtual memory: concept and design“, volume 15, no.5, PP 3078,318,1 May 1972.

[3] X. Zhou, P Petrov, “Towards virtual memory supports in realtime and memory constraint embedded applications the interval page system”, Received on 11th march 2009, Revised on 19 th march 2009.

[4] Archana s. sumant, Pramila M. Chawan, “Virtual memory techniques in 2.6 kernel and challenges“, vol 2, no., ISSN : 1793-823; 2, april 2010.

[5] S. P, Smith,, ”Modelling and Enhancing Virtual Memory Performance in Logic Simulation”, on page 264-267,Product type-conference publication, 7-10 Nov 1988.

[6] Y. A. Khalidi,, ”Virtual Memory Supports for Multiple page Sizes”, on pages (104-109), 14-15 oct 1993.

[7] S. JHartley, “Compile Time Progaram Restructuring in Multiprogrammed Virtual Memory System”, volume 14, issues:11, on page(s)-1640-1644, November 1988.

[8] Sitaram lyer, ”Application - Assisted Physical Memory Management”, Rice University, 6100 Main Street, Ms-132 ,Houston, TX 77005, USA, FEBRUARY 1994.

[9] V.Delaluz et al., ”Schduler-based dram energy power managemant”, In designing Automation conference 39,2002.

[10] D.culler, ”Logp,performance assessment of fast network interfaces”, IEEE MICRO.1996.

[11] Black, et. al., “Translation Lookaside BufferConsistencyA: software Approach”, December1988, CMU-CS-88-201.

[12] Hung-Wei Tseng, Han-Lin Li, Chia-Lin Yang, “An Energy-Efficient Virtual Memory System with Flash Memory as the Secondary Storage”, Low Power Electronics and Design, ISLPED'06, 2006.

[13] M. Huang, J. Renau, S.-M. Yoo, J. Torrellas, "The design of DEETM: a framework for dynamic energy efficiency and temperature management", Journal of Instruction-Level Parallelism, vol. 3, 2002.

[14] R.M. Jones, “Factors Affecting the Efficiency of a Virtual Memory IEEE Transactions on Computers”, Volume: C-18, Issue: 11, Nov. 1969.

[15] C. Park, J.-U. Kang, S.-Y. Park, J.-S. Kim, "Energy-aware demand paging on NAND flash-based embedded storages", Proceedings of the IEEE/ACM International Symposium on Low Power Electronics and Design, August 2004.

[16] Hiroshi Tezuka, Francis O'Carroll, Atsushi Hori, and Yutaka Ishikawa, “Pin-down Cache: A Virtual Memory Management Technique for Zero-copy Communication”, IPPS/SPDP 1998. Proceedings of the First Merged International and Symposium on Parallel and Distributed Processing 1998.

[17] Giuseppe Psaila, ”Virtual DOM-An Efficient Virtual Memory Representation for Large XML Documents”, Database and Expert Systems Application, 2008. DEXA '08.

[18] Dr. Vivek Chaplot, “Virtual Memory Benefits and Uses”, International Journal of Advance Research in Computer Science and Management Studies, Volume 4, Issue 9, September 2016.

[19] Yaman Cakmakci, Oguz Ergin, “Exploiting Virtual Addressing for Increasing Reliability”, IEEE Computer Architecture Letters, Volume: 13, Issue: 1, Jan.-June 28 2014.

[20] Hanna Alam, Tianhao Zhang, Mattan Erez, Yoav Etsion, ”Do-It-Yourself Virtual Memory Translation”, ISCA “17, June 24-28, 2017.

[21] Pinchas Weisberg and Yair Wiseman, “Virtual Memory Systems Should Use Larger Pages rather than the Traditional 4KB Pages”, International Journal of Hybrid Information Technology, Vol.8, No.8 (2015), pp.57-68.

[22] Andon coleman, Janusz zalewski, “A study of Real-time memory management: Evaluating operating system‟s performance”, Automatkya/Automatics, Vol. 17, No. 1, 2013.