Memory Management System Based on Inverted Page Table with Hash Technology

Haoyu Yang[1, a], Linying Gong[2, b] and Yuping Yuan[2, c]

[1]College of software, Jilin University, Changchun 136000, China;

[2] College of computer science and technology, Jilin University, Changchun 136000, China.

[a]yanghaoyujlu@163.com, [b]1425380508@qq.com, [c]1393604763@qq.com

Keywords: Memory management, paging, inverted page table, hash technology.

Abstract. Faced with the development of information technology and the arrival of the era of big data, dynamic memory management is one of the most important operations in many C/C++ applications. Traditional paging memory management apply page table which aims at process. When the address of the process is large, the page table of the process will waste lots of memory space. This paper proposes a new memory management method based on the inverted table to wasting space .Besides, we apply a hash technology to the inverted table in order to improve the speed of searching.

1. Introduction

Memory management includes inner and External memory management .There are many ways in operating system for inner memory management, such as single contiguous region memory management, paging memory management ,segmentation memory management and so on. Single contiguous region method request one process possesses only one contiguous region, so it will produce fragment. On the other hand, paging memory management permit one process to possess many contiguous regions in memory space, and the lengths of the space are the same. As a result, a memory management system applying paging memory management will not produce fragment. Moreover, the inverted table can solve the problem of wasting space. The hash technology can improve the speed of searching [2].

2. Space Division

2.1 Memory Space Division.

The memory space can by divided by some regions of which the length is the same. The region is called page frame. Every page frame has 2i (i is integer) units. They are addressed from low to high. Assuming the capacity of the memory is 2n B, we get 2n-i page frames. The address of them are called page space. The address of memory unit is called physical space.

Physical address =first address of page frame +the address in the page
=the page frame number* 2i + the address in the page

Actually, regarding page frame number as high bit and the address in the page as the low bit, we can get physical address.

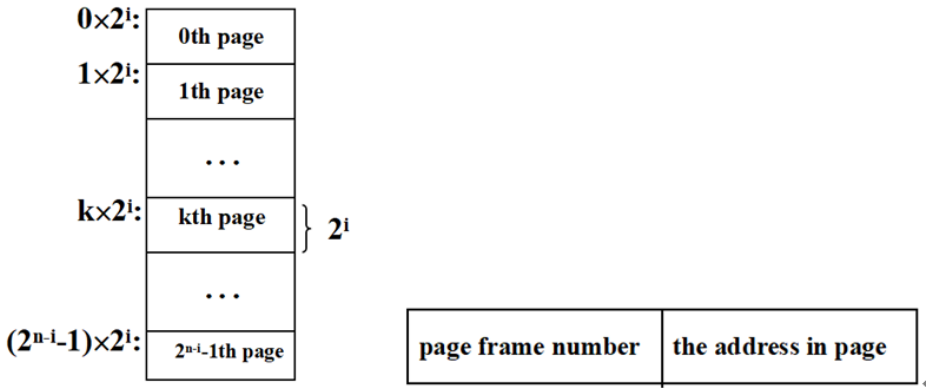

Fig. 1 Memory space division and physical address

2.2 Process Space division.

The process space can by divided by some regions of which the length is the same. The region is called logical page. The length of logical page is the same as page frame, which is to say it also has 2i (i is integer) units. They are addressed from 0. Assuming the the process has l logical pages, they are numbered from 0 and addresses of them are called logical page number.

Logical address =first address of logical page +the address in the page
=the logical page number* 2i + the address in the page

Actually, regarding logical page number as high bit and the address in the page as the low bit, we can get logical address.

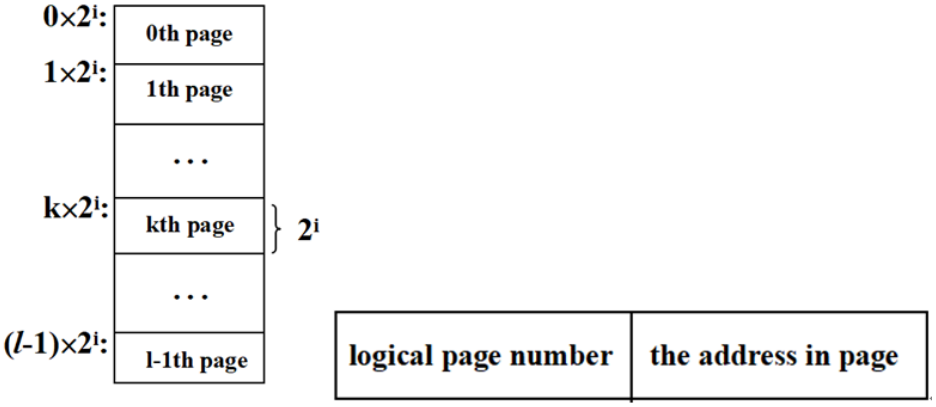

Fig. 2 Process space division and logical address

2.3 Mapping from Logical Address to Physical Address.

In order to transform the logical address in the user address space into the physical address in the memory space, an address translation mechanism must be set in the system. The basic task of this organization is to convert logical address to physical address. As the page address and the physical address is one to one, the address translation's task is actually just that the logical address of the page number is converted to physical address in the memory. [3, 5]The role of page mapping table is used to achieve the transformation from the page number to the physical block number, therefore, the address transformation task is done by means of the page table.

3. Inverted Page Table

Traditional page table aims to process space. There is an entry for each process logical page. When the process space is large, the page table is large. As a result, it can cause wasting space. However, inverted page table aims at physical memory space. One entry is for each page frame and it has fixed size. The number of the entry is the number of page frame. The content of the entry is the ordered pair marked by process mark pid and logical page number p. The system only needs to set one page table which is used for all processes.

When mapping from logical address to physical address, we search the inverted page table according to (pid,p). Once we find matched entry, the offset is the page frame number. We achieve the inverted page table by JAVA.

```
Public void toTable_2() {
int n = dtm2.getRowCount();
For (int i = 1; i < _vc_2.size(); i++) {
_vc_2.remove(i);}
System.out.println("vector num: " + _vc_n2.length);
For (int i = n; i > 0; i--) {
_vc_n2[i].removeAllElements();
}
System.out.println("rows count: " + (n + 1));
ptable = new Ptable(); // produce
boolean bo;
For (int i = 0; i < Source.quantity; i++) {
For (int j = 0; j < pro[i].getPnum(); j++) {
bo = true;
int num = ptable.hash(i, j);
If (num > ptable.getPnumber()) {
num = 0; }
While (bo) {
If (ptable.pta[num].getBusy() < 0) {ptable.pta[num].setNum(num);ptable.pta[num].setBusy(1);
ptable.pta[num].setPid(i);
ptable.pta[num].setP(j); bo = false;
} else {
int cr1 = ptable.pta[num].getCra();
ptable.pta[num].setCra(cr1 + 1);
num = num + ptable.pta[num].getCra();if (num > ptable.getPnumber()) {num = 0;
}
}
```

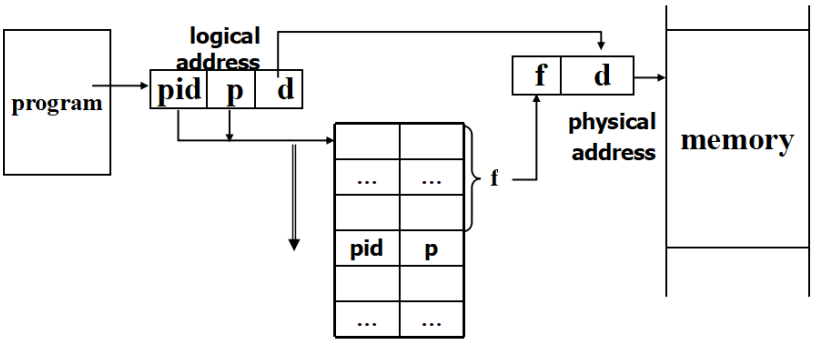

Fig. 3 Inverted page table

4. Hash Technology

The problem of inverted page table is speed. The sequence searching of the inverted page table needs access the memory many times. In order to improve the speed of the speed, we adopt the hash technology, and add conflict counting in the inverted page table. When mapping, calculate the entrance address of the inverted page table, search the target entry from this entrance address. Offset f is the page frame number we want.

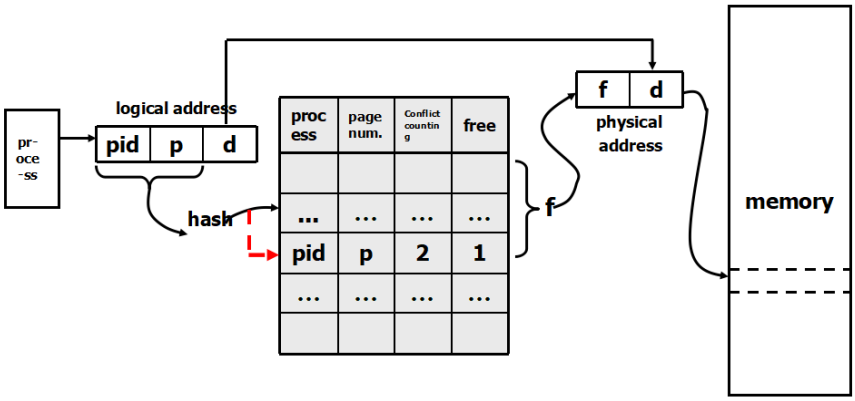

Fig. 4 Inverted page table with hash technology

5. Conclusion

An operating system is system software that manages computer hardware and software resources and provides common services for computer programs. For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware, although the application code is usually executed directly by the hardware and frequently makes system calls to an OS function or is interrupted by it. [3]

Memory management is becoming overwhelmingly significant and indispensible in modern operating system. Paging memory management solves the problem of fragment. Inverted page table can make use of the space in order not to cause space wasting. The inverted page table can improve the speed of searching. [4, 6]The system based on inverted table with hash develpoed by JAVA is shown in the Fig.5.

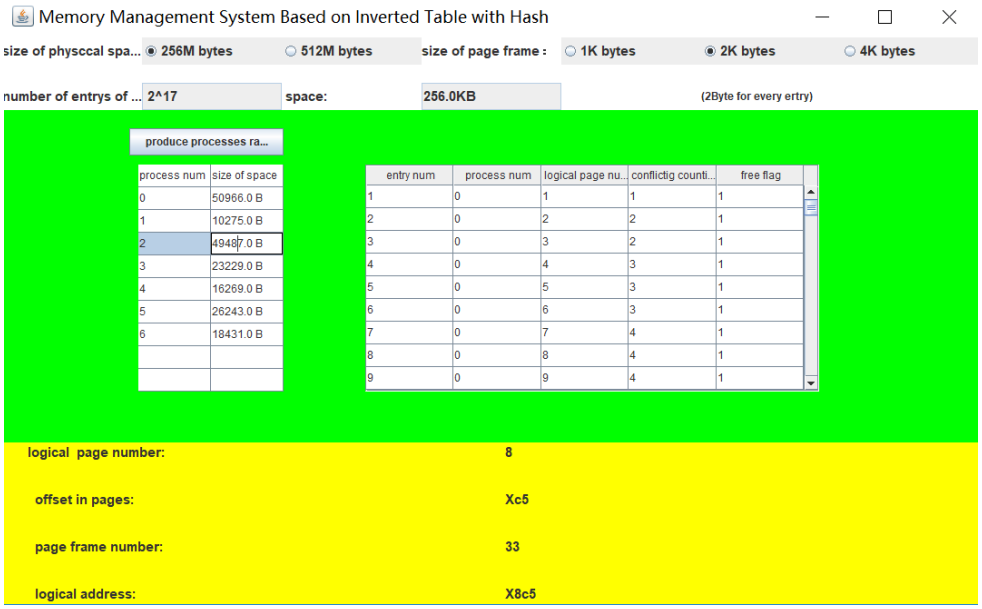

Fig. 5 Implement of memory management system with JAVA

References

[1]. Tiwari, D., Lee, S., Tuck, J., & Solihin, D. (2010). MMT: Exploiting fine-grained parallelism in dynamic memory management. IEEE International Symposium on Parallel & Distributed Processing (pp.1-12). IEEE.

[2]. Harvey-Lees-Green, Nicholas, et al. "A Dynamic Memory Management Unit for Real Time Systems." IEEE, International Symposium on Real-Time Distributed Computing IEEE, 2017.

[3]. Information on: https://en.wikipedia.org/wiki/Operating_system

[4]. Zhao Gang. "Discrete Allocation Storage Management - Pagination Management Analysis." Academic Conference on Building Technologies and Management 2015.

[5]. Luo Yonglong, Zhao Cheng. Simulation of Partitioned Storage Management Process in Multiprogramming System [J] .Journal of West Anhui University, 1999 (4): 49-51.

[6]. Song Jinhua, Ma Chuanqi. Linux Memory Management Buddy algorithm [J]. Fujian Computer, 2009, 25 (1): 65-65.