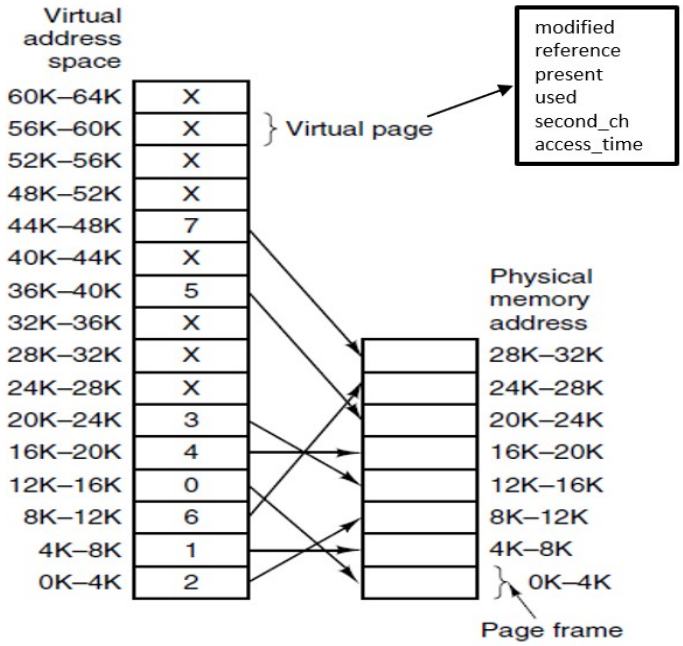
CSE 312
OPERATING SYSTEM
HOMEWORK 3 REPORT

1801042005
OĞUZHAN SEZGİN

Page Table Structure

There are two types of page tables in the assignment, the first is the virtual page table and the second is the physical page table. I used physical to access the virtual page table located in physical memory. Virtual page tables create virtual memory and simulate disk. These tables point to the first index of the segment of the corresponding page in the virtual memory array. The relationship between physical and virtual memory is shown below;



Virtual memory capacity is equal to disk capacity so all pages in the virtual memory is also has a disk address. When user want to access a disk address which does not contain in the physical memory, the corresponding virtual page table is loaded with data from the disk and that page table is loaded into physical memory with the replacement algorithms.

Page tables has some bits. This bits shown below;

```
typedef struct
{
    int modified;
    int reference;
    int present;
    int used;
    int second_ch;
    double access_time;
} page_entry;
```

Modified : When the user change the value of an data which exist in the page, modified bit will be 1 and when page replacement event occurs this page data writing to disk, if the modified bit is 0 operating system does not write to disk, she only replace the page.

Reference : This bit holds the starting index of the segment corresponding to the page in virtual memory. Indexes up to the frame size after this index are considered the data of this page.

Present : This bit determines the page is in the physical memory or not.

Used: This bit determines the page is used or not. This bit for used WSClock(line 909) and LRU(line 852) algorithms.

Second_ch: This bit used for the SC algorithm. When page access more than one without replacement then this bit will be 1(line 736-796).

Access_time: This bit used for WSClock algorithm. It stores the last access time to page (line 756-820).

Get/Set Functions

Get function(line 772) : This function first finds the page of coming index. Then checks the page is present in the physical page or not . If it is present in the physical memory then it returns the data which exist in the coming address, if it is not then it calls the page replacement function load the requested address page. After that it returns the address value.

Set function(line 691) : This function sets the address of value with given value. It works like a get function, first checks the existences in the physical page. If page does not exist in physical page it calls the page replacement function for load the requested address page. After that it changes the given address value.

Page Replacement Algorithms

When the requested address pages does not exist in the physical memory(page fault) operating systems uses this algorithm to load requested address pages. In the program , every time a data in page is changed, it is not written directly to the disk, instead the modified bit is changed and when the pages with this bit of 1 are to be replaced, the current values in the page are written to the disk. If the modified bit is 0 then page replaced without writing to the disk(line 998).

Also there is two function for read(line 1037) and write(line 1059) to the disk. This function writes or reads the value with given address(line number). In disk each line is treated.

as an address.

In program I implement 3 different page replacement algorithm. These are explained below

Last Recently Used (LRU – line 875)

This algorithm checks the used number of pages when it replace. In get and set function every access of the pages operating system increment the used bit value. Then when system has to replace page with LRU algorithm it first finds the least used page (line 841). When it finds then it replaces least used page with requested address page. Also this algorithm has a period(line 75). This global variable thread as a clock and every calling of the LRU algorithm system increments this variable by 1 . I define a access period(line 26), when the the period variable reach the this access period, system resets the all pages used bit and period variable. In this way all used bits resets as a clock cycle.

Second Chance (SC – line 875)

This algorithm checks the more than one usage of the page. There is a circular index(line 76) in this algorithm. This index iterates the pages which exist in the physical memory. When the page exist in the physical memory and user wants to access this page again, system makes the second_ch bit 1 of the corresponding page. Then user wants to access another page and if this page does not exist in the memory than system iterates the pages with phy_index. If the second_ch bit of the page this index is pointing to is 1, it makes this bit 0 and moves to the next page and continues until it finds the page with the second_ch bit 0. When it finds this page, it loads the page it wants to load instead.

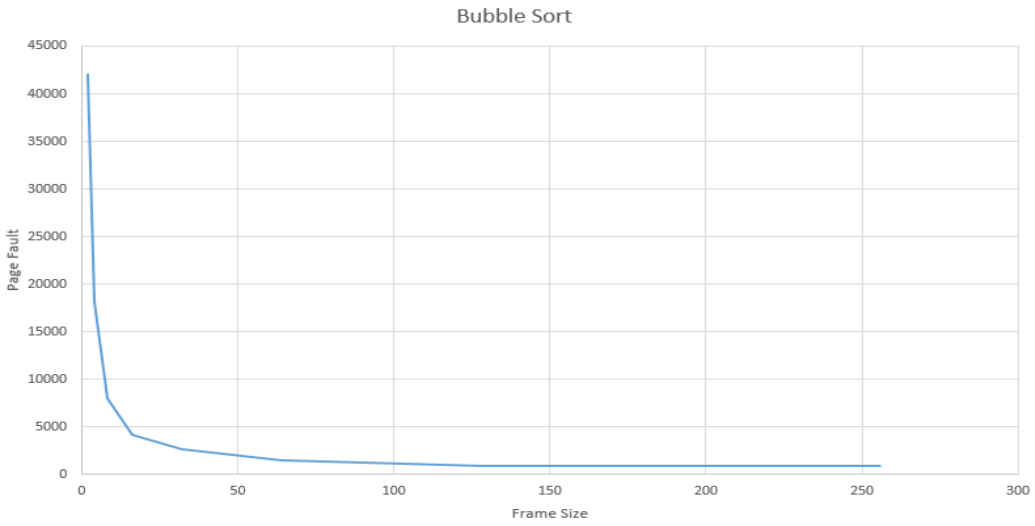
Working Set Clock(WSClock – line 902)

This algorithm similar to the SC algorithm. It checks the used of the pages and last access time of page. There is a circular index as SC algorithm for iterates pages which exist in the physical memory. It checks used bit as SC algorithm but it also checks the time. There is the threshold time value(line 25) and this algorithm compare this time and with the page access time. If the last access time between current time period is the grater than threshold value and the used bit is 0 then algorithm replace this page. It iterates until you find the page that satisfies the condition. Then it also has a period for reset the used bit similar to LRU algorithm.

Best Frame Size

In this part ı usd bubble sort function, quick sort function and LRU page replacement algorithm for finding best frame size. In the find_best_frame function(part3.c-line 369) program increments the t_frame_size and decrements the t_num_phy and t_num_virt variables in a loop. In every increment and decrement event physical memory, virtual memory and disk loads new random values. Then bubble and quick sort algorithms works then program finds the best frame size(least page replacement count) and choose. For find best frame size , I use different virtual and physical memory number because of the execution time. Graphics and other informations shown below.

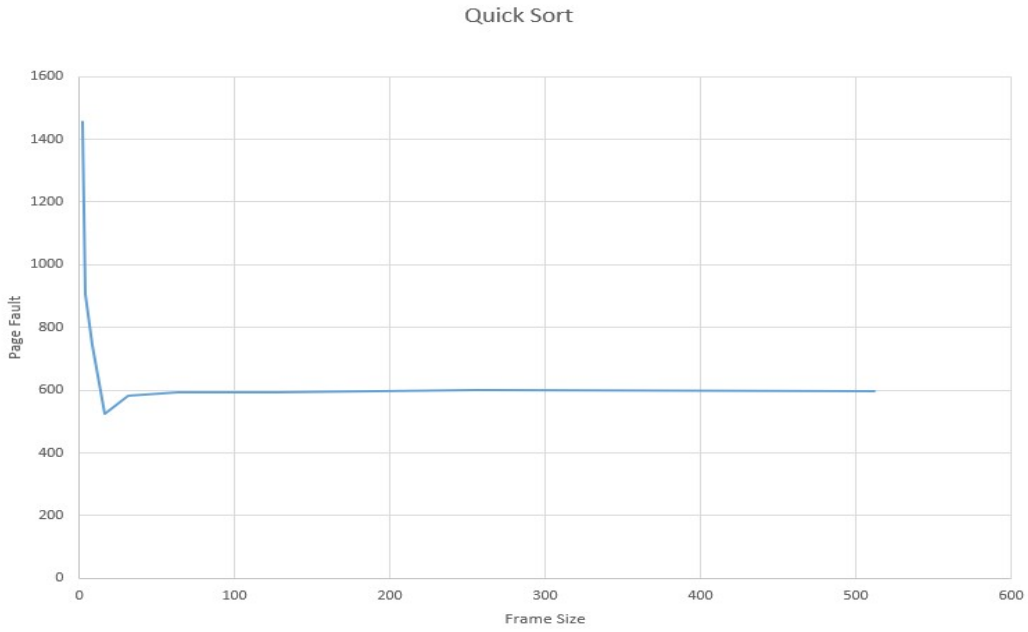
Bubble Sort Graphic



Bubble Sort Page Fault/Frame Size Graphic

Physical memory size is 256 integer and virtual memory size 512 integer. Best frame size is 256. When page frame size increased by 2 times, page fault number decrease by half.

Quick Sort Graphic



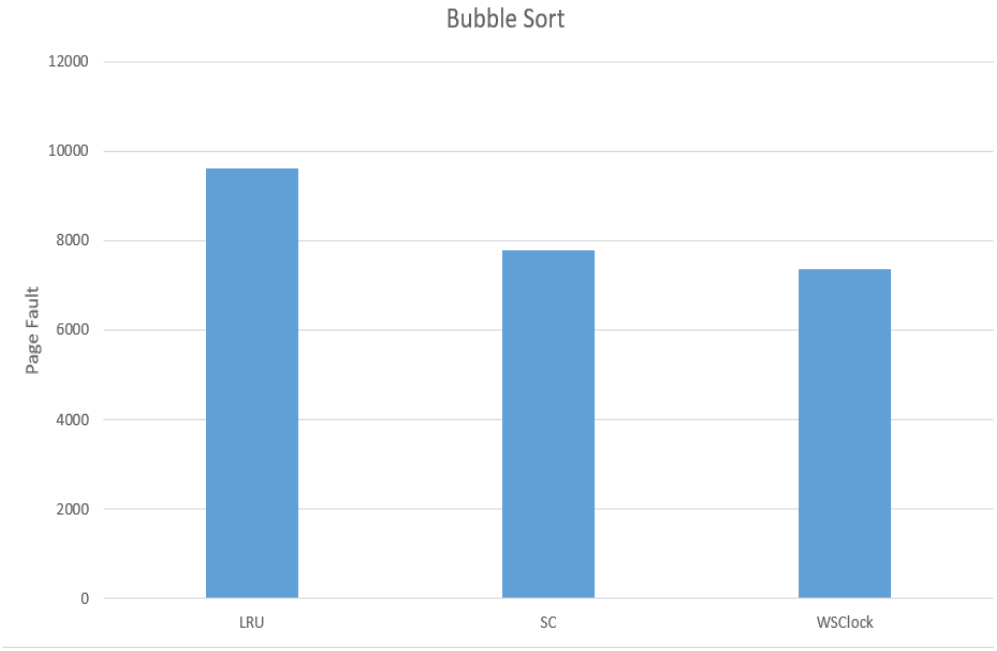
Quick Sort Page Fault/Frame Size Graphic

Physical memory size is 512 integer and virtual memory size 1024 integer. Best frame size is 16. The page fault value, which initially decreased inversely with the frame size, turned to the right proportion as the frame size increased, and the page fault value began to increase.

Best Replace Algorithm(Bonus)

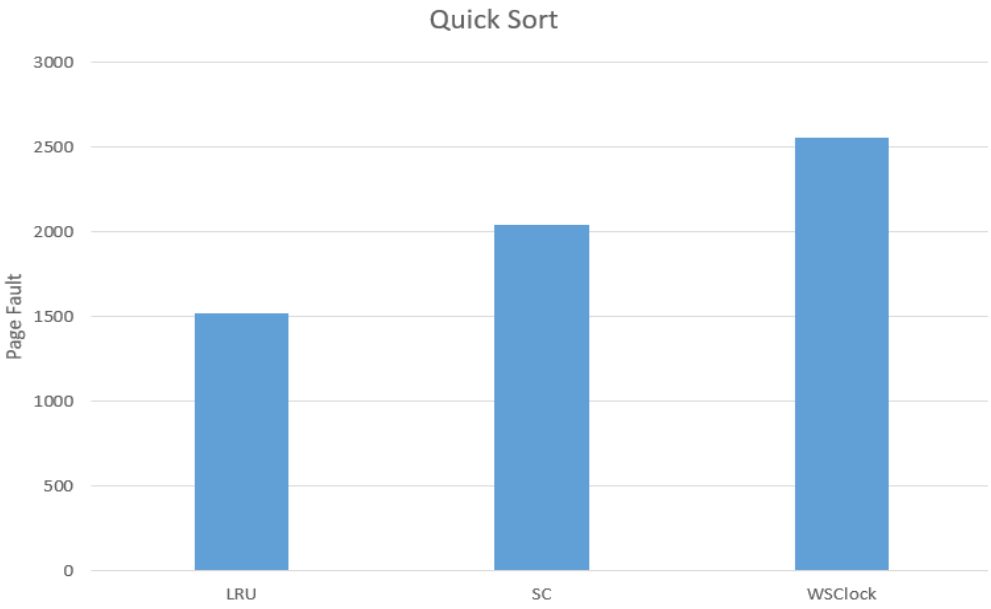
In this part program finds the best page replacement algorithm for each sorting algorithm. System has 4 integer frame size, 64 integer physical size, 256 integer virtual size. The statistics shown below graphics.

Bubble Sort Graphic



Bubble Sort Page Fault/Replace Algorithm Graphic

Quick Sort Graphic



Quick Sort Page Fault/Replace Algorithm Graphic