OS Project-8 Report

Virtual Memory Manager

Zhou Yiyuan

516021910270
1025152261@qq.com

1 Data Structure

The following data structure is used to implement the virtual memory manager:

```
#define TLB_SIZE 16
#define PageTableSize 256
#define PhysicalMemorySize 128
#define FRAME_SIZE 256

int TLBTable[TLB_SIZE][2];  //[pageNumber, frameNumber]
int TLBPointer;
int PageTable[PageTableSize];
unsigned char PhysicalMemory[PhysicalMemorySize][FRAME_SIZE];
int PhysicalMemoryPointer;
int UsedPhysicalMemory = 0;
```

TLBPointer is used to implement TLB's FIFO updating algorithm:

```
void updateTLBTable(int pageNumber, int frameNumber){
    TLBTable[TLBPointer][0] = pageNumber;
    TLBTable[TLBPointer][1] = frameNumber;
    TLBPointer = (TLBPointer + 1) % TLB_SIZE;
}
```

Similarly, PhysicalMemoryPointer is used to implement FIFO algorithm when page replacement occurs.

```
PhysicalMemoryPointer = (PhysicalMemoryPointer + 1) % PhysicalMemorySize;
for(int i=0; i<FRAME_SIZE; ++i){
    PhysicalMemory[PhysicalMemoryPointer][i] = buffer[i];
}

if(UsedPhysicalMemory < PhysicalMemorySize){
    UsedPhysicalMemory ++;
}else{ //page replace
    //change TLB Table
    for(int i=0; i<TLB_SIZE; ++i){
        if(TLBTable[i][1] == PhysicalMemoryPointer){
            TLBTable[i][0] = -1;
            break;
        }
    }
    //change page table
    for(int i=0; i<PageTableSize; ++i){
        if(PageTable[i] == PhysicalMemoryPointer){
            PageTable[i] = -1;
            break;
        }
```

```
    }
}

PageTable[pageNumber] = PhysicalMemoryPointer;
```

2 Algorithm

2.1 initialization

```
void init(void){
    n_addrs = 0;
    n_pageFaults = 0;
    n_TLBHits = 0;

    for(int i=0; i<TLB_SIZE; ++i){
        TLBTable[i][0] = -1;
    }
    TLBPointer = 0;

    for(int i=0; i<PageTableSize; ++i){
        PageTable[i] = -1;
    }
    UsedPhysicalMemory = 0;
    PhysicalMemoryPointer = 0;

    f_disk = fopen("BACKING_STORE.bin", "rb");
    if(f_disk == NULL){
        printf("Fail to find the BACKING_STORE.bin\n");
        exit(-1);
    }
}
```

2.2 main()

For a logical address, we can obtain its physical address and its value by the following steps:

(a) use masks to obtain the pageNumber and offset from the logical address

```
    int pageNumber = (logical_addr & 0xFFFF) >> 8;
    int offset = logical_addr & 0xFF;
```

(b) query the frameNumber from TLBTable. If TLB hits, jump to step (d)

```
for(i=0; i<TLB_SIZE; ++i){
    //TLB Hit
    if(TLBTable[i][0] == pageNumber){
        frameNumber = TLBTable[i][1];
        n_TLBHits ++;
        break;
    }
}
```

(c) query the frameNumber from PageTable

```
//TLB miss
if(i == TLB_SIZE){
    frameNumber = PageTable[pageNumber];
}
```

(d) Determine if the frame exists in the physical memory. If existing, jump to step (f)

(e) load the frame into the memory from the disk and modify the TLBTable and PageTable

```
//page faults
if(frameNumber < 0){
    frameNumber = readFromDisk(pageNumber);
    n_pageFaults ++;
}
```

(f) obtain the physical address and the value of the logical address

```
 signed char value = PhysicalMemory[frameNumber][offset];

printf("frame number=%d, offset = %d, value=%d\n", frameNumber, offset,
↪  value);
printf("Virtual address: %d Physical address: %d Value: %d\n",
↪  logical_addr, (frameNumber << 8) | offset, value);
```

(g) update TLBTable

```
//TLB miss
if(i == TLB_SIZE){
    updateTLBTable(pageNumber, frameNumber);
}
```

2.3 readFromDisk()

When page faults occur, we must load the frame into the physical memory from the disk. **fseek()** and **fread()** is used to read the file from the disk:

```
fseek(f_disk, pageNumber * FRAME_SIZE, SEEK_SET);
signed char buffer[FRAME_SIZE];
fread(buffer, sizeof(signed char), FRAME_SIZE, f_disk);
```

If PhysicalMemorySize is smaller than the virtual address space, page replace will occur when the physical memory is full. Then, a page-replace policy should be taken, which means remove an old frame from the physical memory to make room for the new frame. And items related to the old frame in TLBTable and PageTable should also be modified to maintain consistency.

```
int readFromDisk(int pageNumber){
    if(fseek(f_disk, pageNumber * FRAME_SIZE, SEEK_SET) != 0){
        printf("Error seeking in backing store\n");
        exit(-1);
    }

    signed char buffer[FRAME_SIZE];
    if(fread(buffer, sizeof(signed char), FRAME_SIZE, f_disk) == 0){
        printf("Error seeking in backing store\n");
        exit(-1);
    }

    PhysicalMemoryPointer = (PhysicalMemoryPointer + 1) % PhysicalMemorySize;
    for(int i=0; i<FRAME_SIZE; ++i){
        PhysicalMemory[PhysicalMemoryPointer][i] = buffer[i];
    }

    if(UsedPhysicalMemory < PhysicalMemorySize){
        UsedPhysicalMemory ++;
    }else{ //page replace
        //change TLB Table
        for(int i=0; i<TLB_SIZE; ++i){
            if(TLBTable[i][1] == PhysicalMemoryPointer){
                TLBTable[i][0] = -1;
                break;
            }
        }
        //change page table
        for(int i=0; i<PageTableSize; ++i){
            if(PageTable[i] == PhysicalMemoryPointer){
                PageTable[i] = -1;
                break;
            }
        }
```

```
    }

    PageTable[pageNumber] = PhysicalMemoryPointer;

    return PhysicalMemoryPointer;
}
```

3 Result

3.1 Address Translation

I sample the last 30 addresses from the address list in correc.txt to display. When PhysicalMemorySize=256, both of Physical address and Value are same with the groundtruth, because the physical memory is large enough to hold all frames so that no page replace occurs. When PhysicalMemorySize=128, Value are same but Physical address are different, because page replace occurs and changes physical addresses.

```
Virtual address: 23195 Physical address: 35995 Value: -90
Virtual address: 27227 Physical address: 28763 Value: -106
Virtual address: 42816 Physical address: 19520 Value: 0
Virtual address: 58219 Physical address: 34155 Value: -38
Virtual address: 37606 Physical address: 21478 Value: 36
Virtual address: 18426 Physical address: 2554 Value: 17
Virtual address: 21238 Physical address: 37878 Value: 20
Virtual address: 11983 Physical address: 59855 Value: -77
Virtual address: 48394 Physical address: 1802 Value: 47
Virtual address: 11036 Physical address: 39964 Value: 0
Virtual address: 30557 Physical address: 16221 Value: 0
Virtual address: 23453 Physical address: 20637 Value: 0
Virtual address: 49847 Physical address: 31671 Value: -83
Virtual address: 30032 Physical address: 592 Value: 0
Virtual address: 48065 Physical address: 25793 Value: 0
Virtual address: 6957 Physical address: 26413 Value: 0
Virtual address: 2301 Physical address: 35325 Value: 0
Virtual address: 7736 Physical address: 57912 Value: 0
Virtual address: 31260 Physical address: 23324 Value: 0
Virtual address: 17071 Physical address: 175 Value: -85
Virtual address: 8940 Physical address: 46572 Value: 0
Virtual address: 9929 Physical address: 44745 Value: 0
Virtual address: 45563 Physical address: 46075 Value: 126
Virtual address: 12107 Physical address: 2635 Value: -46
```

Figure 1: The groundtruth in correct.txt

```
Virtual address: 23195 Physical address: 35995 Value: -90
Virtual address: 27227 Physical address: 28763 Value: -106
Virtual address: 42816 Physical address: 19520 Value: 0
Virtual address: 58219 Physical address: 34155 Value: -38
Virtual address: 37606 Physical address: 21478 Value: 36
Virtual address: 18426 Physical address: 2554 Value: 17
Virtual address: 21238 Physical address: 37878 Value: 20
Virtual address: 11983 Physical address: 59855 Value: -77
Virtual address: 48394 Physical address: 1802 Value: 47
Virtual address: 11036 Physical address: 39964 Value: 0
Virtual address: 30557 Physical address: 16221 Value: 0
Virtual address: 23453 Physical address: 20637 Value: 0
Virtual address: 49847 Physical address: 31671 Value: -83
Virtual address: 30032 Physical address: 592 Value: 0
Virtual address: 48065 Physical address: 25793 Value: 0
Virtual address: 6957 Physical address: 26413 Value: 0
Virtual address: 2301 Physical address: 35325 Value: 0
Virtual address: 7736 Physical address: 57912 Value: 0
Virtual address: 31260 Physical address: 23324 Value: 0
Virtual address: 17071 Physical address: 175 Value: -85
Virtual address: 8940 Physical address: 46572 Value: 0
Virtual address: 9929 Physical address: 44745 Value: 0
Virtual address: 45563 Physical address: 46075 Value: 126
Virtual address: 12107 Physical address: 2635 Value: -46
```

Figure 2: PhysicalMemorySize=256

```
Virtual address: 23195 Physical address: 15003 Value: -90
Virtual address: 27227 Physical address: 31323 Value: -106
Virtual address: 42816 Physical address: 4416 Value: 0
Virtual address: 58219 Physical address: 3179 Value: -38
Virtual address: 37606 Physical address: 10470 Value: 36
Virtual address: 18426 Physical address: 30202 Value: 17
Virtual address: 21238 Physical address: 20982 Value: 20
Virtual address: 11983 Physical address: 3535 Value: -77
Virtual address: 48394 Physical address: 3594 Value: 47
Virtual address: 11036 Physical address: 3868 Value: 0
Virtual address: 30557 Physical address: 4189 Value: 0
Virtual address: 23453 Physical address: 4509 Value: 0
Virtual address: 49847 Physical address: 4791 Value: -83
Virtual address: 30032 Physical address: 4944 Value: 0
Virtual address: 48065 Physical address: 18113 Value: 0
Virtual address: 6957 Physical address: 27693 Value: 0
Virtual address: 2301 Physical address: 21245 Value: 0
Virtual address: 7736 Physical address: 13112 Value: 0
Virtual address: 31260 Physical address: 5148 Value: 0
Virtual address: 17071 Physical address: 5551 Value: -85
Virtual address: 8940 Physical address: 5868 Value: 0
Virtual address: 9929 Physical address: 6089 Value: 0
Virtual address: 45563 Physical address: 6395 Value: 126
Virtual address: 12107 Physical address: 6475 Value: -46
```

Figure 3: PhysicalMemorySize=128

3.2 Statistics

The Page Faults Rate increases when PhysicalMemorySize decreases.

```
Number of addresses = 1000
Page Faults = 244
Page Faults Rate = 0.244
TLB Hits = 54
TLB Hit Rate = 0.054
```

Figure 4: PhysicalMemorySize=256

```
Number of addresses = 1000
Page Faults = 538
Page Faults Rate = 0.538
TLB Hits = 54
TLB Hit Rate = 0.054
```

Figure 5: PhysicalMemorySize=128