OS2022 Project3

Name : 郭忠翔

Department : 機械所控制組

ID : R10522845

Motivation

Motivation and the problem analysis

個別執行sort.c和matmult.c都可以有正確的輸出，但在同時執行沒有得到正確的輸出，原因是因為執行時需要大量的記憶體，超出系統上限造成core dumped。

Plan to deal with the problem

實作virtual memory，當實體記憶體不足時可以做為輔助，再來是做swap的動作，決定哪個page該被swap到disk上，這邊實作`FIFO`(First In First Out)和`LRU`(Least Recent Used)，FIFO按照page順序當作victim，而LRU是每次將使用最少的page當作victimm。

Implementation

先利用SyncDisk當作輔助記憶體

```
// code/userprog/userkernel.h
class UserProgKernel : public ThreadedKernel {
    public:
    ...
    int vmtype;                  // victim type
    Machine *machine;
    FileSystem *fileSystem;
    SynchDisk *virtualMemoryDisk;
    #ifdef FILESYS
        SynchDisk *synchDisk;
    #endif // FILESYS
    ...
```

配合前次作業的排成，修改Initialize

```
 // code/userprog/userkernel.cc
void
UserProgKernel::Initialize()
{
    //Initialize(RR);
    ThreadedKernel::Initialize();   // init multithreading
    machine = new Machine(debugUserProg);
    fileSystem = new FileSystem();
    virtualMemoryDisk = new SynchDisk("Virtual Memory");
    #ifdef FILESYS
        synchDisk = new SynchDisk("New SynchDisk");
    #endif // FILESYS
}
```

```
void
UserProgKernel::Initialize(SchedulerType type)
{
    ThreadedKernel::Initialize(type);   // init multithreading
    machine = new Machine(debugUserProg);
    fileSystem = new FileSystem();
    virtualMemoryDisk = new SynchDisk("Virtual Memory");
    #ifdef FILESYS
        synchDisk = new SynchDisk("New SynchDisk");
    #endif // FILESYSvu.
}
```

再加入需要用到的變數

```
// code/machine/machine.h
class Machine {
  public:
    ...
    bool usedPhyPage[NumPhysPages];     // record used physical memory
    bool usedvirPage[NumPhysPages];     // record used virtual memory
    int  ID_num;                        // ID for thread
    int PhyPageName[NumPhysPages];
    int count[NumPhysPages];            // counter for LRU

    TranslationEntry *main_tab[NumPhysPages];
    static int fifo;
    ...
// code/userprog/addrspace.h
class AddrSpace {
    public:
        int ID;                         // record thread's ID
    private:
        bool pageTableLoaded;
}
```

接著當程式Load後，開始往下找可用的記憶體，找到最後都沒有時代表記憶體不夠用，若不夠就需要用到virtual memory

```
// code/userprog/addrspace.cc
AddrSpace::AddrSpace()
{
    (kernel->machine->ID_num)++;
    ID = kernel->machine->ID_num;
}
bool
AddrSpace::Load(char *fileName)
{
    ...
    // creat new pageTable
    pageTable = new TranslationEntry[numPages];
    for(unsigned int i = 0, j = 0; i < numPages; i++) {
        pageTable[i].virtualPage = i;
        pageTable[i].physicalPage = i;
        pageTable[i].valid = true;
        pageTable[i].use = false;
        pageTable[i].dirty = false;
        pageTable[i].readOnly = false;
        pageTable[i].count = 0;
    }
    if (noffH.code.size > 0) {
        for(unsigned int j=0,i=0;i < numPages ;i++) {
            j=0;
            while(kernel->machine->usedPhyPage[j] != false && j < NumPhysPages)
j++;
            if(j < NumPhysPages) {
                kernel->machine->usedPhyPage[j] = true;
                kernel->machine->PhyPageName[j] = ID;
                kernel->machine->main_tab[j] = &pageTable[i];
                pageTable[i].physicalPage = j;
                pageTable[i].valid = true;
                pageTable[i].use = false;
                pageTable[i].dirty = false;
                pageTable[i].readOnly = false;
                pageTable[i].ID = ID;
                kernel->machine->totalcount ++;   // LRU count
                pageTable[i].count = kernel->machine->totalcount;
                executable->ReadAt( &(kernel->machine->mainMemory[j *
PageSize]), PageSize,
                    noffH.code.inFileAddr + (i*PageSize));
            }
            // out of memory
            else {
                char *buffer;
                buffer = new char[PageSize];
                j = 0;
                // find unused virtual page
                while(kernel->machine->usedvirPage[j] != false){ j++; }

                kernel->machine->usedvirPage[j]=true;
                pageTable[i].virtualPage = j;
                pageTable[i].valid = false;
                pageTable[i].use = false;
                pageTable[i].dirty = false;
                pageTable[i].readOnly = false;
                pageTable[i].ID = ID;
                executable->ReadAt(buffer, PageSize, noffH.code.inFileAddr + (i
* PageSize));
                // write into Disk
                kernel->virtualMemoryDisk->WriteSector(j, buffer);
            }
        }
    }
}
void AddrSpace::SaveState()
{
    if(pageTableLoaded) {
        pageTable=kernel->machine->pageTable;
        numPages=kernel->machine->pageTableSize;
    }
```

```
}
```

先做後綴，在下指令時可以選擇victim type

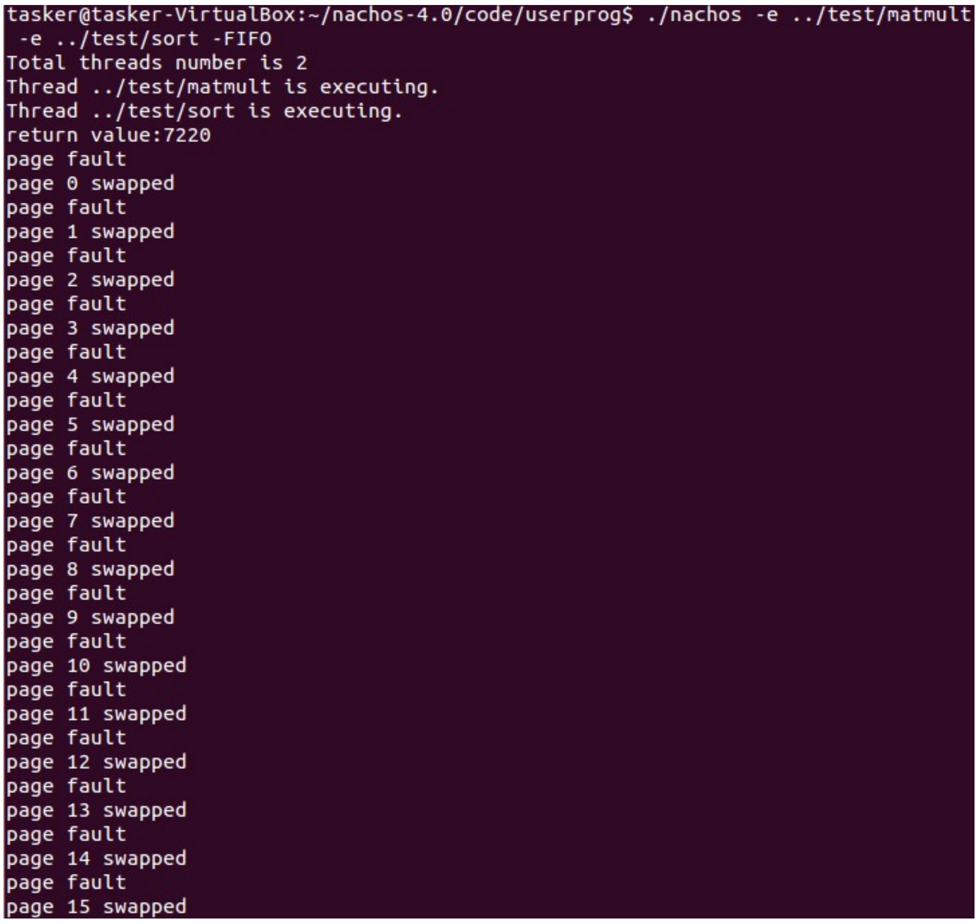
```
// code/userprog/userkernel.cc
UserProgKernel::UserProgKernel(int argc, char **argv)
        : ThreadedKernel(argc, argv)
{
            ...
            else if (strcmp(argv[i], "-LRU") == 0) {
                vmtype = 1;
            }
            else if (strcmp(argv[i], "-FIFO") == 0) {
                vmtype = 0;
            }
}
```

再來是實作`FIFO`和`LRU`來決定victim page，然後swap到Disk中

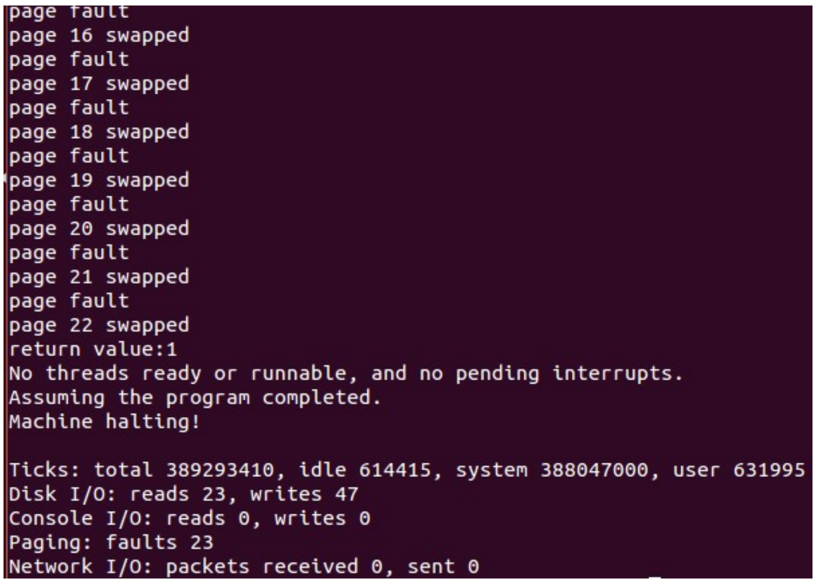
```
// code/machine/tranlate.cc
else {
        char *buffer1;
        buffer1 = new char[PageSize];
        char *buffer2;
        buffer2 = new char[PageSize];
        //FIFO
        if (kernel->vmtype == 0){
            victim = fifo % 32;
        }
        //LRU
        if (kernel->vmtype == 1) {
            int min = pageTable[0].count;
            victim = 0;
            for(int i = 0; i < 32; i++){
                if(min > pageTable[i].count){
                    min = pageTable[i].count;
                    victim = i;
                }
            }
            pageTable[victim].count++;
        }
        printf("page %d swapped\n",victim);
        //write into Disk
        bcopy(&mainMemory[victim * PageSize], buffer1, PageSize);
        kernel->virtualMemoryDisk->ReadSector(pageTable[vpn].virtualPage, 
buffer2);
        bcopy(buffer2, &mainMemory[victim*PageSize], PageSize);
        kernel->virtualMemoryDisk->WriteSector(pageTable[vpn].virtualPage, 
buffer1);

        main_tab[victim]->virtualPage= pageTable[vpn].virtualPage;
        main_tab[victim]->valid = false;
        // load to main memory
        pageTable[vpn].valid = true;
        pageTable[vpn].physicalPage = victim;
        kernel->machine->PhyPageName[victim] = pageTable[vpn].ID;
        main_tab[victim] = &pageTable[vpn];
        fifo ++;
}
```

Result & Discussion

- FIFO



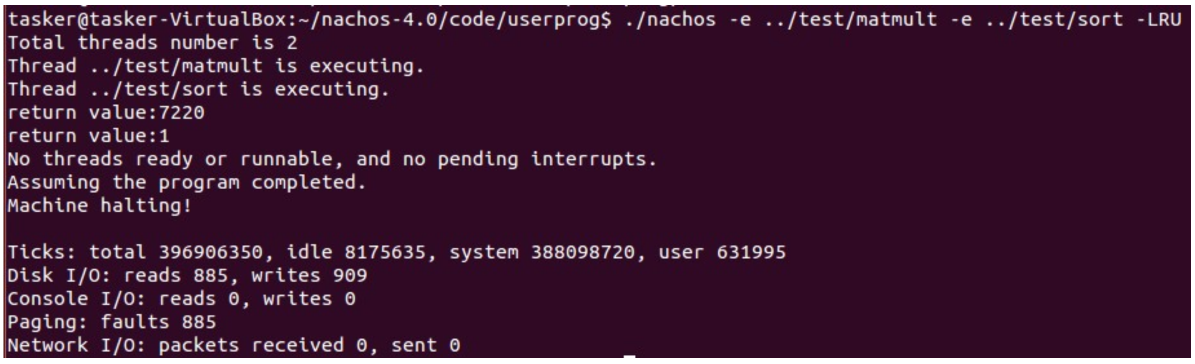
```
tasker@tasker-VirtualBox:~/nachos-4.0/code/userprog$ ./nachos -e ../test/matmult
 -e ../test/sort -FIFO
Total threads number is 2
Thread ../test/matmult is executing.
Thread ../test/sort is executing.
return value:7220
page fault
page 0 swapped
page fault
page 1 swapped
page fault
page 2 swapped
page fault
page 3 swapped
page fault
page 4 swapped
page fault
page 5 swapped
page fault
page 6 swapped
page fault
page 7 swapped
page fault
page 8 swapped
page fault
page 9 swapped
page fault
page 10 swapped
page fault
page 11 swapped
page fault
page 12 swapped
page fault
page 13 swapped
page fault
page 14 swapped
page fault
page 15 swapped
```



```
page fault
page 16 swapped
page fault
page 17 swapped
page fault
page 18 swapped
page fault
page 19 swapped
page fault
page 20 swapped
page fault
page 21 swapped
page fault
page 22 swapped
return value:1
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 389293410, idle 614415, system 388047000, user 631995
Disk I/O: reads 23, writes 47
Console I/O: reads 0, writes 0
Paging: faults 23
Network I/O: packets received 0, sent 0
```

- LRU

  由於Page fault次數較多，將輸出Page fault和第幾個page被swap的輸出註解掉，結果如下


```
tasker@tasker-VirtualBox:~/nachos-4.0/code/userprog$ ./nachos -e ../test/matmult -e ../test/sort -LRU
Total threads number is 2
Thread ../test/matmult is executing.
Thread ../test/sort is executing.
return value:7220
return value:1
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 396906350, idle 8175635, system 388098720, user 631995
Disk I/O: reads 885, writes 909
Console I/O: reads 0, writes 0
Paging: faults 885
Network I/O: packets received 0, sent 0
```