國立臺灣大學 National Taiwan University

Nachos Project Assignment 3
Memory Management / CPU part 2

EE 5173 Operating System

TA: Hsueh-Yi, Chen. Advisor: Farn, Wang.

2023/11/23

Project 3 – part1

Memory Management

What will you learn from this project?

1. Memory Swap management
2. Swap replacement algorithm
3. Demand paging

Why we need swap management

1. It expands the amount of memory a process may use.
2. A significant number of the pages referenced by a process early in its life may only be used for initialization and then never used again

Disadvantages:

1. Performance decrease (Disk is slow.)

Swap Management

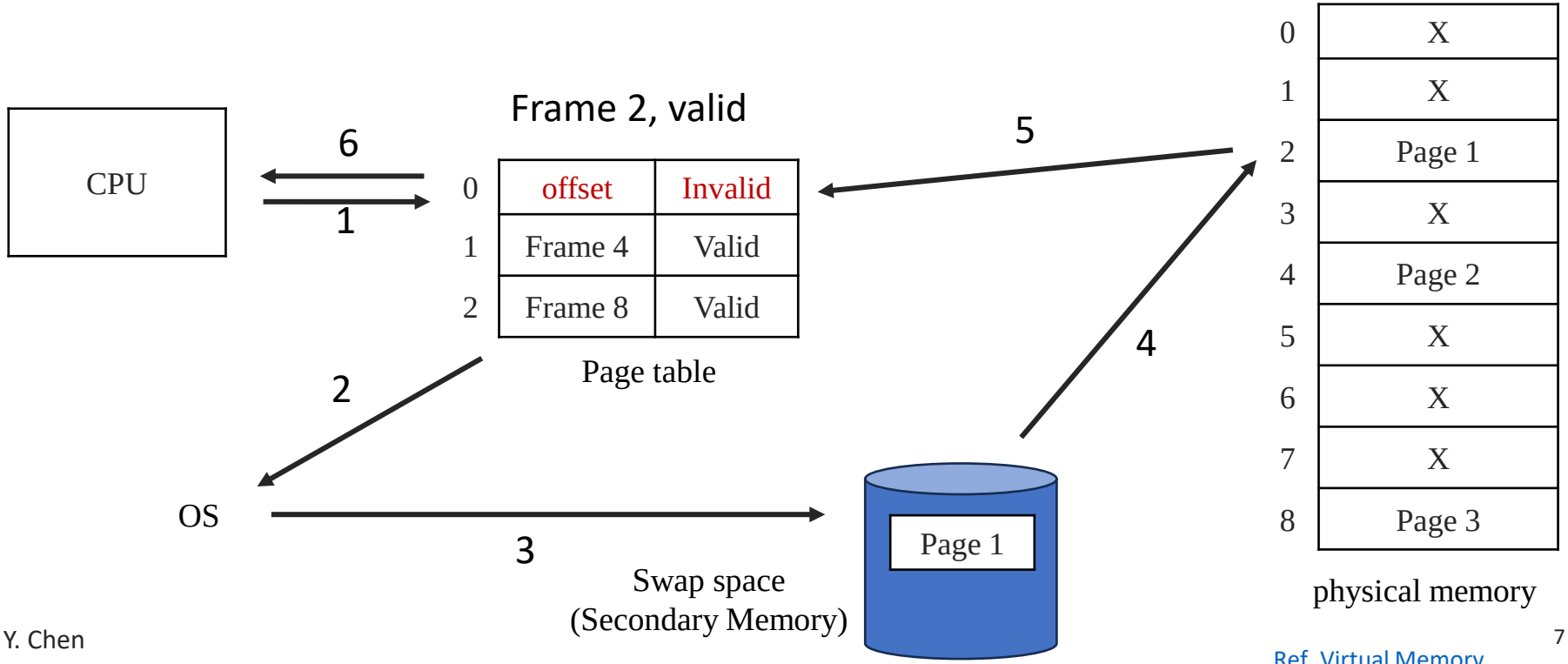
Goal:

- Successfully execute the following test cases, concurrently.
  - /test/matmalt.c
  - /test/sort.c
- Both cases use require lots of memory
  - Larger than the physical memory

Demand Paging

- Only load pages into memory when necessary. (Actually, we loads some pages into memory to avoid lots of page fault).
- Strictly follow above definition which is called pure demand paging.
- You can decide how many preload pages.
- When page not in memory (present bit off), trap to OS via page fault.

Steps in handling a page fault



Ref. Virtual Memory

Page fault

- When valid bit set to fault, this will throw a PageFaultException and trap to Kernel

```
213     } else if (!pageTable[vpn].valid) {
214         DEBUG(dbgAddr, "Invalid virtual page # " << virtAddr);
215         return PageFaultException;
216     }
```

/machine/kernel/translate.cc

- We can catch the exception in **ExceptionHandler** and handle the page fault.
- Add some codes in /userprog/exception.cc and create other files if needed.
- BadVAddrReg will store the VA which is not in memory.

Page fault

- You can create a Virtual Memory Manger class to handle page fault.
- Note: Load 1 page
- If memory is enough , get a free frame and put the page in.
- If memory is full, page-replacement algorithm to select victim frame to swap out.
  - Design your own page replacement algorithm to get the frame
  - Specify the method in your report
  - More algorithm more socre

Ref. Virtual Memory

Swap Space – Create Disk

- swap = new SynchDisk in userkernel.cc
- Use the disk as swap space.
- Access the virtual memory in the disk by…
  - kernel->swap->WriteSector
  - kernel->swap->ReadSector
- See “synchdisk.cc” and other files in /filesys for details!
  - Read the header in those files first

SynchDisk

- SynchDisk object resides above the raw disk (Nachos)
- It blocks the calling thread until after the corresponding I/O complete interrupt takes place. (synchronous)
- Nachos is multi-thread and disk cannot service more than one request at a time.
- It provides mutual exclusion, so that multiple threads can safely call the SynchDisk routines concurrently

SynchDisk

SynchDisk provides the following operations

- SynchDisk(char *name):

  Constructor takes the name of Unix file that holds the disk's contents.

- ReadSector(int sectorNumber, char *data)
- WriteSector(int sectorNumber, char *data)

  Invoke the underlying Disk::ReadSector/ WriteSector and then wait for interrupt

- Callback():

  Disk interrupt handler is called when an I/O operation completes

Project Tasks

- Disk – swap space
- Record some necessary information to manage memory swapping
  - PageTable
  - Swap replacement information
  - Memory usage
  - …
- Demand paging , page fault handler and page replacement algorithm (FIFO, LRU…)

Some files that might be useful

- For the disk usage details, see:
  - /filesys/synchdisk.h
  - /filesys/synchdisk.cc
  - Other files in /filesys(Optional)
- For the swap space initialization:
  - /userprog/userkernel.h
  - /userprog/userkernel.cc
- For the page fault handling:
  - /userprog/exception.h
  - /userprog/exception.cc

Some files that might be useful

- For the loading of pages
  - userprog/addrspace.h
  - userprog/addrspace.cc
- Header file (.h) and comments are helpful
- Based on your implementation, there might be different files that your need to see and modify.

Project 3 – part2

CPU scheduling 2

CPU scheduling 2

- Please following the instruction in the assignment
- Assignment will be released on NTU cool.
- TA will explain next week.

Report

Part 1 (50%)

- What is your motivation, problem analysis and plan ?
- Explain the details of code snippet you added or modified.
- Experiment result and some discussion, observation.
- What problem you face and how to tackle it?

Part 2 (50%)

* If your code are more different than reference, more score

Policy

- Please save as [Student ID]_project3_1.pdf, [Student ID]_project3_2.pdf
  - E.g. f10921a18_project3_1.pdf, f10921a18_project3_2.pdf
- Upload to NTU cool. **DDL: 2023/12/28**
- Penalty: decrease 5% per day
- No plagiarism

TAs

- 吳吉加 r12921093@ntu.edu.tw
- 楊冠彥 r11921091@ntu.edu.tw
- 陳學義 f10921a18@ntu.edu.tw

Reference

- Nachos SynchDisk
- Linux Swap Management

Thanks for your attention !