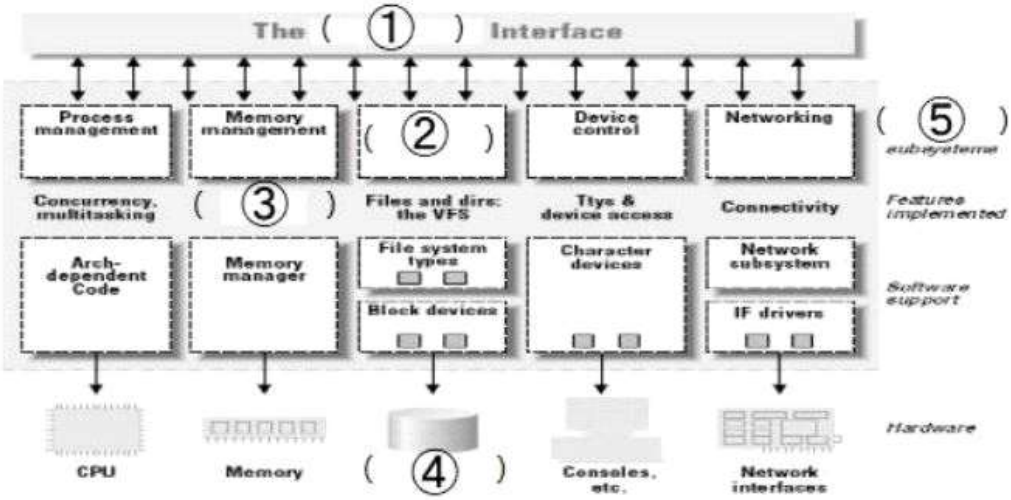
(1장 OS Introduction + 6장 File System Basic + 2장 Processes)

1. ❶ 주어진 표의 빈칸을 채우시오.(1.5점) ❷ 주어진 표에서 [ ]가 의미하는 것이 무엇인지 간략하게 쓰시오.(1점) ❸ 표에 있는 VFS의 fullname과 역할을 서술하시오.(1.5점) ❹ File System에서 device와 cpu사이의 Protocol의 종류 3가지를 쓰고 특징에 대해 서술하시오.(2점) ❺ ⑤번의 protection기술 중 제어의 흐름을 가로채기도 하는 이것의 이름을 쓰고 이것의 전체 전개과정을 간략히 서술하시오.(2점) ❻ Policy와 Mechanism의 정의에 대한 빈칸을 채우고 multiple processes에서 time sharing을 사용할 때의 Mechanism과 Policy를 쓰시오.(2점)



❻ : Policy : (Which(or What)) to do? / Mechanism : (How) to do?

1 ①System call ②File System ③Virtual memory ④Disks & CDs ⑤Kernel

2 module을 나타낸다.

3 VFS(Virtual File System) : Kernel과 같이 File System의 Interface나 규격을 정의한다.

4 device의 status를 check하는 단계는 총 4단계로 구성되며 아래와 같다.

1) idle check 2) data 3) command 4) finish check

- PIO(Programmed I/O) : 위의 4단계를 CPU가 모두 수행하는 Protocol이다. 추가적으로 이러한 과정을 polling이라하며 Spin wait과 비슷하게 CPU를 반납하지 않고 점검을 수행하기 때문에 CPU가 낭비될수 있지만 context switch가 필요하지 않으므로 overhead가 발생하지 않는다.
- Interrupt : device가 idle일 때 device가 직접 CPU에게 알리는 Protocol이다. 때문에 idle check를 CPU가 직접 수행하지 않아 sleep wait과 비슷하게 CPU는 그동안 다른 작업을 수행할 수 있다. 따라서 CPU 낭비를 줄일수 있지만 context switch가 필요하므로 overhead가 발생할 가능성이 있다.
- DMA : PIO에서는 CPU가 직접 memory와 data copy를 수행하기 때문에 CPU가 낭비되지만 DMA controller는 data copy를 CPU대신 수행해주기 때문에 속도가 빠른 CPU(10^{-9}s)와 비교적 매우 느린 Memory(10^{-3}s) 사이에서의 CPU 낭비를 현저하게 줄여준다.

5 Trap

전개과정 : System call → trap → save context and switch stack → jump to the trap table → kernel mode → return from system call → switch stack and restore context → jump to the next instruction of the System call → user mode.

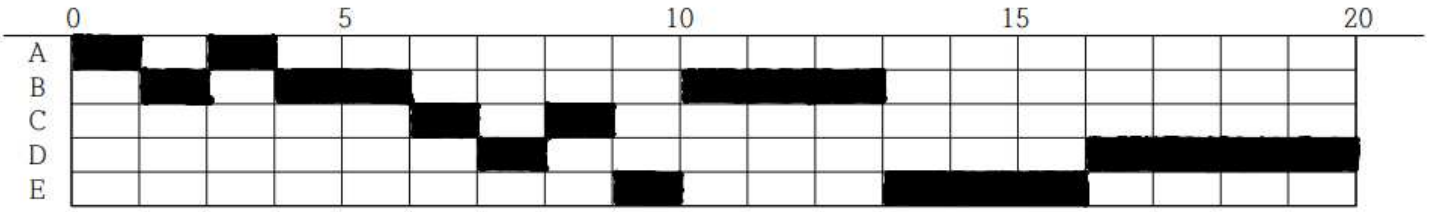
(3장 Scheduling + 9장 Paging and Beyond Physical Memory)

2-1. 캐시의 다양한 기법 중 Cache affinity를 고려하여 등장한 Scheduling 기법을 full name으로 작성하고 기법의 장점과 단점(단점은 해결방법까지)을 기술하시오.(2점)

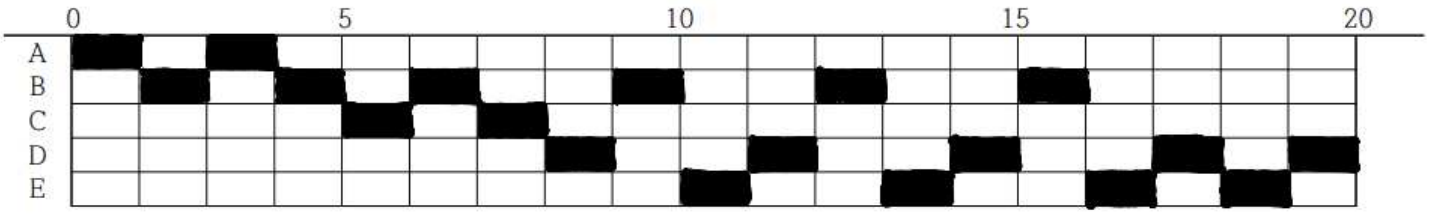
2-2. ❶ MLFQ Scheduling(time quantom = 2^i)을 사용하던 와중 time=10부터 SJF Scheduling으로 바뀌었다. 밑에 그래프에 Workload를 활용하여 해당 칸을 색칠하시오.(2점) ❷ 같은 Workload를 사용하여 RR Scheduling(time quantom = 1)을 사용했을 때의 그래프를 색칠하시오.(2점) ❸ RR Scheduling 그래프에 대한 average response time과 average turnaround time을 계산하고(1점) ❹ 색칠한 RR의 그래프에서 앞에서 부터 색챌된 10개를 page reference string이라고 할 때 paging replacement 기법 중 LRU의 Hit Ratio를 계산하고 계산과정을 보이시오. (Cache Size : 3 frames)(2점) ❺ RR과 MLFQ의 time quantom크기에 따른 tradeoff에 대해 서술하시오.(1점)

Process	Arrival Time	Service Time
A	0	2
B	1	6
C	4	2
D	6	6
E	8	5

❶



❷



2-1. MQMS (Multi Queue Multiprocessor Scheduling)
장점: cache affinity, less lock contention
단점: need to consider load balancing → 해결방법: migration, work stealing

2-2. [3] average response time : (0+0+1+2+2)/5 = 1
average turnaround time : (3+15+4+16+13)/5 = 10

[4]

Access	Hit/Miss	Evict	Resulting Cache State
A	Miss		A
B	Miss		A B
A	Hit		A B
B	Hit		B A
C	Miss		A B C
B	Hit		A C B
C	Hit		A B C
D	Miss	A	B C D
B	Hit		C D B
E	Miss	C	D B E

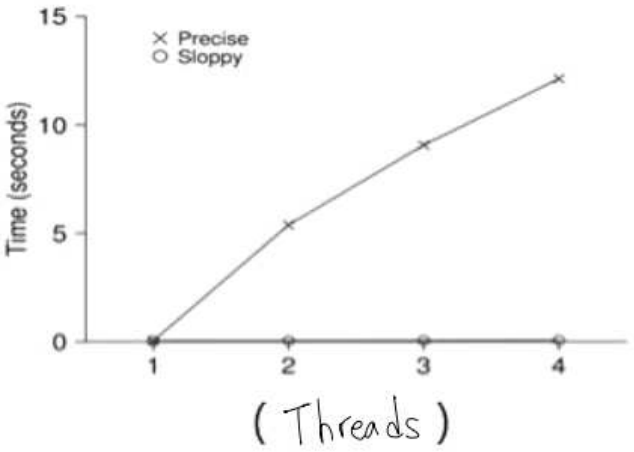
Hit Ratio : 5/10 x 100 = 50 %

[5] Small : response time이 좋아지고(작아지고) context switch overhead가 증가한다.
Large : response time이 나빠지고(커지고) Context switch overhead가 감소한다.

(4장 Thread and Lock + 5장 Semaphore and Deadlock)

3. ❶ pthread_mutex_init(A,B)와 sem_init(C,D,E)에서 A, B, C, D, E를 각각 설명하시오.(a,b(1.5점), c,d,e(1.5점)) ❷ mutex(Mutual Exclusion)과 sem(Semaphore)의 차이를 범위의 측면에서 설명하시오.(2점) ❸ Sleep wait과 Producer/Consumer Problem의 공통점을 한 가지 쓰고 간단하게 설명하시오.(2점) ❹ Sloppy counter에 대한 그래프와 표에 있는 빈칸을 채우고 그래프에서 두 가지의 선이 차이나는 이유를 간단하게 설명하시오.(2점) ❺ (Bonus) 4장. Concurrency : Thread and Lock에서 처음에 교수가 아이들에게 Concurrency에 대해 설명할 때 예로 든 과일은?(1점)

❹



Time	L1	L2	L3	L4	G
0	0	0	0	0	0 ③
1	0	0	1	1	0 ④
2	1	0	2	1	0 ⑤
3	2	0	3	1	0 ⑥
4	3	0	3	2	0 ⑦
5	4	1	3	3	0 ⑧
6	5→0 ①	1	3	4	5 ⑨
7	0	2	4	5→0 ②	10 ⑩

1 A : 초기화 시킬 mutex 객체
B : mutex의 특성을 정의하는 변수, 기본적으로 NULL을 사용
C : 초기화 시킬 sem 객체
D : 프로세스간의 공유여부를 설정하는 값.
E : 초기화된 sem 객체에 할당하는 값.

2 sem은 시스템 범위에 걸쳐있기 때문에 file system 상의 file형태로 존재하는 반면 mutex는 프로세스 범위에 있기 때문에 process가 종료되면 사라진다.

3 Sleep wait과 Producer/Consumer Problem 모두 Queue를 사용해야 한다.
Sleep wait에서는 Sleep 상태에 들어간 process를 깨우기 위해 사용하고
Producer/Consumer Problem은 조건변수의 값을 저장하기 위해 사용한다.

5 peach

(5장 Semaphore and Deadlock + 7장 Advanced File System)

4. ❶ 건욱이는 코로나19사태로 인해 다니던 알바에서 해고당했다. 용돈을 받지 않고 생활해왔기에 당장 급했던 건욱이는 어쩔 수 없이 너무 힘든걸 알았기에 하고 싶지 않았던 물류창고 정리 알바를 하게 되었다. 물건리스트를 받고 물류창고에 들어간 건욱이는 깜짝 놀라고 말았다. 리스트 목록에 있는 물류의 종류에 비해 터무니 없이 많은 상자들과 물류들이 널려있었기 때문이다. Banker's algorithm을 활용해 건욱이를 도와 정리할 방법(상자를 가득 채워 상자를 테이핑)을 기술해보자.(2점) ❷ 건욱이를 기껏 도와주고 나니 사장이 와서 상자가 아직도 이렇게 많이 남았냐고 하면서 당장 상자의 개수를 줄이라고 한다. 하지만 건욱이는 힘이 빠져서 물건을 많이 옮길 수가 없다. Banker's algorithm의 victim기법을 활용해 건욱이를 도와주자.(1.5점) ❸ deadlock과 Banker's algorithm의 unsafe사이의 관계를 설명하시오.(1점) ❹ file system의 기술 중에서 Banker's algorithm과 같이 문제를 방지하는 EXT3과 관련된 기술의 이름을 쓰고 그 기술을 사용하지 않았을 때 일어날 수 있는 문제점의 종류 2가지를 쓰고 간단히 설명하시오.(1.5점) ❺ ❹에서 언급 된 기술에 대한 내용 중 Recovery에 대한 빈칸을 채우고 Transaction 사이에 일어나는 Log 과정에 대해 설명하시오.('-)'는 오류발생지점을 의미한다.)(빈칸2점, Log2점)
(테이핑한 상자 개수의 최대값을 구하는 것이 목적이 아니므로 ❶에서 테이핑한 상자의 최대 개수를 구하는 것은 고려하지 않고 풀어도됨. 하지만 널려있는 물류가 남으면 안됨.)

물류리스트	상자상태	널려있는 물류의 개수
신발	2/8, 3/8, 7/8, 4/8, 5/8	9
모자	4/12, 7/12, 11/12, 6/12, 9/12	8
티셔츠	13/15, 6/15, 7/15, 9/15, 10/15	7
안경	17/20, 6/20, 10/20, 2/20, 18/20	17

❺
redo : (journaling) -> (checkpointing)
undo : (TxB(Transaction Begin)) -> (TxE (Transaction End))

Log: Transaction 사이에 여러 이유들로 비정상적인 종료를 하게 될 경우 log에 기록된 내용을 참고하여 다시 작성하거나 복구하기 위해 log를 작성한다.

1. 신발: 3개가 들어있는 상자에 5개
4개가 들어있는 상자에 4개
모자: 9개가 들어있는 상자에 5개
9개가 들어있는 상자에 3개
티셔츠: 13개가 들어있는 상자에 2개
10개가 들어있는 상자에 5개
안경: 17개가 들어있는 상자에 3개
6개가 들어있는 상자에 14개
를 채워 모두 테이핑한다.

2. 신발 : 남은 2/8, 7/8, 5/8 중에서 2개 짜리 상자를 다른 상자에 각각 1개씩 옮겨 담아 상자의 갯수를 1개 줄인다.
모자 : 남은 4/12, 11/12, 6/12 에서 4개 짜리 상자를 다른 상자에 각각 1개, 3개씩 옮겨 담아 상자의 갯수를 1개 줄인다.
티셔츠 : 남은 6/15, 7/15, 9/15 에서 6개 짜리 상자를 다른 상자에 각각 3개씩 옮겨 담아 상자의 갯수를 1개 줄인다.
안경 : 남은 10/20, 2/20, 18/20 에서 2개 짜리 상자를 18개 짜리 상자에 옮겨 담아 상자의 갯수를 1개 줄인다.

3. unsafe 한 상태라고 해서 무조건 deadlock이 발생하는 것이 아니고 unsafe한 상태의 process들 중에서 deadlock 이 발생한다.

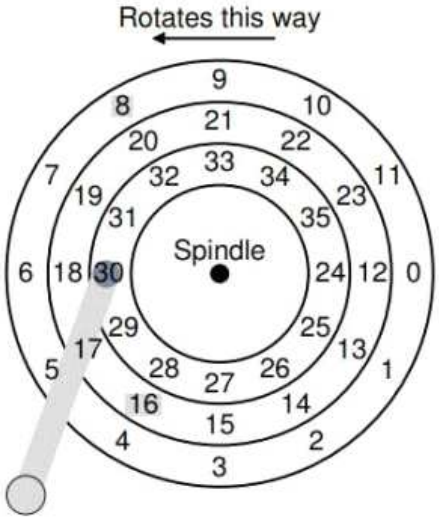
4. 이름: Journaling
문제점: inconsistency – Bitmap과 Inode 중 하나만 쓰여졌을 때 발생하고 두 개가 가지고 있는 정보가 다름을 의미한다.
garbage read – 올바른 data가 아닌 쓰레기 값을 읽는다.

(6장 File System Basic + 7장 Advanced File System)

5. ❶ Disk Scheduling 기법에서 SCAN과 비교해 C-SCAN의 장단점을 쓰고 설명하라.(1.5점) ❷ SPTF를 사용하고 그림을 참고하여 request가 3과 33일 때 가장 시간이 적게 걸리는 경로에 대해 전개과정을 보여라. (사용되는 모든 계산과정을 보여라)(1점) ❸ lseek()의 매개변수 중 whence의 종류 3가지를 쓰시오.(각 0.5점) ❹ EXT4의 최대 file size를 계산과정을 보여 계산하고 indirect block을 사용했을 때의 단점을 간단하게 쓰시오.(1점) ❺ 밑의 표는 disk에 file을 쓰는 과정을 나타낸다. 이 과정에 Caching 기법과 Write buffering 기법을 적용했을 때 지워지는(memory 접근을 하지 않는) read와 write에 선을 그어 지우고 Write buffering의 장단점과 단점의 해결방법을 간략하게 서술하라.(1.5점) ❻ FFS(Fast File System)에서 Cylinder Group에 file과 directory를 allocation할 때 지켜야할 Rule 3가지에 대해 Rule 1과 2는 각각 두 단어로 표현하고 Rule 3은 빈칸을 채우고 단점으로 인해 생기는 현상을 쓰시오.(2.5점) ❼ Flash Memory에 대해 빈칸을 채워라.(1점)

Write buffering의 장점 : memory 접근이 줄어 전체적인 성능이 향상되고 seek overhead가 줄어든다.

❷ : Head Position : 30



❺ 단점: 비정상적인 종료로 인한 데이터 손실 가능성이 있다. → 해결방법: fsync()

	data bitmap	inode bitmap	root inode	foo inode	bar inode	root data	foo data	bar data[0]	bar data[1]	bar data[2]
			read			read				
				read			read			
create (/foo/bar)		read write								
							write			
					read					
				write	write					
write()	read									
	write							write		
					write					
write()	read				read					
	write								write	
					write					
write()	read				read					
	write									write
					write					

❻

Pros) locality (among) files,
Cons) locality (in) files. → seek time 증가 ⇒ Rule 1. load balancing
Rule 2. namespace locality

❼

Flash Memory는 정보의 입출력이 자유로워 (휴대폰, 디지털카메라, 게임기, MP3 등 2차저) 등에 이용된다.
Flash Memory는 (bit) 단위의 정보를 저장하는 (Cell)로 구성된 배열 내에 정보를 저장한다.
Flash Memory를 Disk로 추상화시키기 위해 사용하는 여러 기술들이 있는데 address translation은 block/page 기반의 Flash를 (Sector) 기반의 Disk로 변환시키는 기술이고 (Wear leveling) 기술은 재기록 가능 횟수가 정해져 있는 Flash Memory의 단점을 보완해주는 기술이다.

1 장점: SCAN에 비해 형평성이 좋다.
단점: 안쪽이나 바깥쪽으로 처리할 요청이 없어도 끝까지 이동하기 때문에 비효율적이다.

2 3 then 33 : 2seek → 9/12 rotation → 2seek → 6/12 rotation >> 63/12
33 then 3 : 3/12 rotation → 2seek → 6/12 rotation >> 33/12
따라서 33을 먼저 탐색한 뒤 3을 탐색하는 것이 효율적이다.

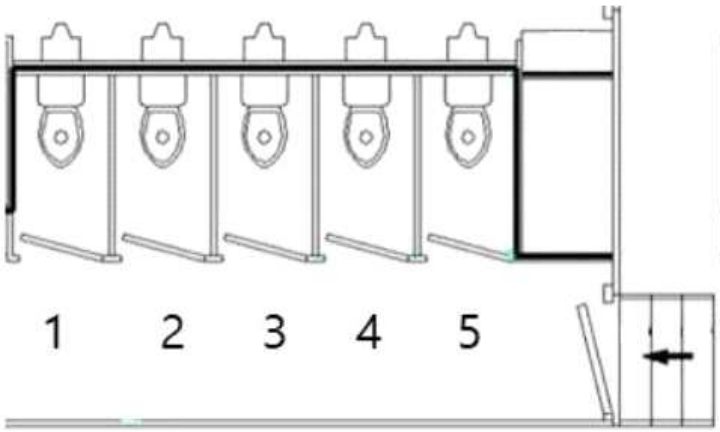
3 SEEK_SET, SEEK_CUR, SEEK_END

4 Direct block pointer: 12 × 4KB = 48KB
Single Indirect block pointer: 1 × 1024 × 4KB = 4MB
Double Indirect block pointer: 1 × 1024 × 1024 × 4KB = 4GB
Triple Indirect block pointer: 1 × 1024 × 1024 × 1024 × 4KB = 4TB
SUM: 48KB + 4MB + 4GB + 4TB
Indirect block의 단점: 추가적인 disk I/O가 필요하다.

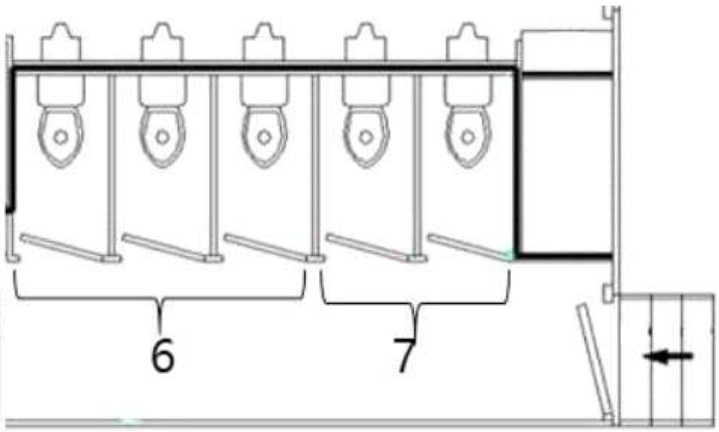
(8장 Memory Management + 2장 Process + 5장 Semaphore and Deadlock)

6. ❶ 키가 157인 건욱이는 백화점에서 쇼핑을 하던 와중 볼 일을 보기 위해 화장실에 들어갔다. 1번 화장실은 키가 151~155, 2번 화장실은 키가156~160, 3번 화장실은 키가 161~165, 4번 화장실은 키가 166~170, 5번 화장실은 키가 171~175인 사람들이 들어갈만한 크기로 되어있어 건욱이는 2번 화장실에 들어가야 하지만 다른 사람이 이미 들어가 있어 기다리고 있는 상태다. 현재 상황을 Memory Management으로 보고 process와 resource를 정하고 2번 화장실의 상태를 RAG로 표현하시오.(2.5점) ❷ 6번 7번 화장실은 모두 차 있고 6번에는 3명이 각각 1칸씩 7번은 2명이 모두 오른쪽 칸을 기다리고 있다. 이 상황을 RAG로 표현하여라. (기다리는 사람들은 알파벳(안에 있는 사람 소문자, 기다리는 사람 대문자)으로 표현)(2점) ❸ 7번(7번 화장실은 한 칸당 크기가 1) 화장실을 paging의 기법 중 Buddy allocation과 접목시킨다면 어떠한 변화가 일어날지 예상하여라.(2.5점) ❹ 건욱이(Process)는 백화점(Memory)에서 쇼핑이 끝나자 백화점을 나왔다. Process States 변화에 대한 그림을 그리고 각각의 상태변화에 대한 명칭을 쓰고 건욱이가 백화점을 나온 상황을 Process States 측면에서의 명칭과 이 명칭과 반대되는 명칭 두 개를 쓰고 설명하여라.(3점)

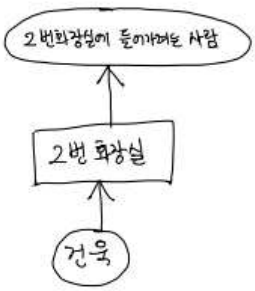
❶



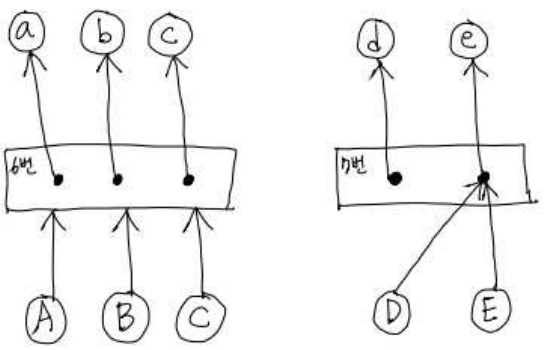
❷



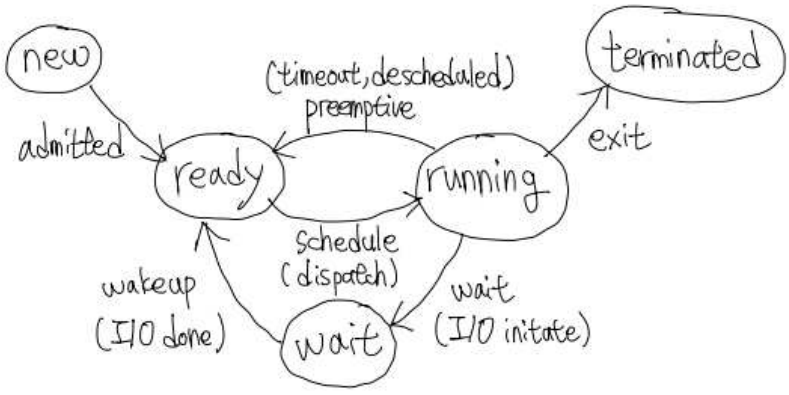
1



2



4



suspend : 사용하지 않는 process를 RAM에서 disk로 내린다.
resume : 다시 RAM으로 올라온다.

3 2.5점 : 크기가 1.5보다 크고 2보다 작거나 같은 사람이 들어갈 수 있다.
1점 : 크기가 2 이하인 사람이 들어갈 수 있다.

(8장 Memory Management + 9장 Paging and Beyond Physical Memory)

7. ❶ External Fragmentation과 Internal Fragmentation에 대해 segmentation과 paging을 사용하여 간단하게 설명하시오.(1점) ❷ Memory와 Disk에서의 Compaction은 각각 어떤 의미를 가지고 있는지 간단하게 설명하시오.(1.5점) ❸ segmentation fault와 protection fault는 각각 어떤 경우에 발생하는지 서술하라.(1.5점) ❹ Heap과 Stack에 대한 설명이다. ( )안에 맞는 것을 선택하여라.(각 0.5점) ❺ virtual memory, physical memory와 process의 관계에 대하여 빈칸을 채워라.(각 0.5점) ❻ A : Program Code, B : Heap, C : Stack일 때 표의 빈칸을 채우고 그림을 참고하여 virtual address(14KB)를 physical address로 변환하고 계산과정도 나타내어라.(2점) ❼ ❻번 표 중 우측 2개의 표(Address Space, Page Table)을 활용하여 2과 5의 physical address를 계산하여라. (page table을 이용한 physical address 계산 시 virtual memory의 6KB까지만 고려하고 단위는 KB -> B로 생각하고 계산하여라)(2점)

❹

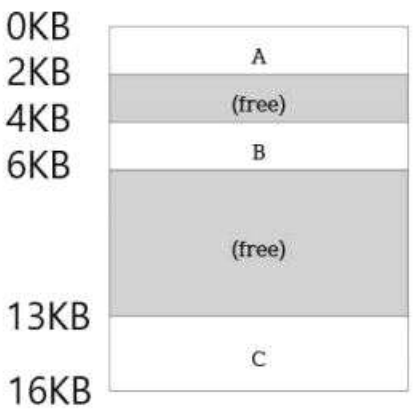
Stack은 (암묵적 / 명시적)이다.
Heap은 (Short lived memory / Long lived memory) 이다.

❺

virtual memory : (per) process
physical memory : (shared by) process

❻

Segment	Base	Size (max 4K)	Grows Positive?
A(00)	32K	2K	(/)
B(01)	34K	2K	(/)
C(11)	28K	3K	(O)



Page Table

7
3
10

1 Internal fragmentation : 할당된 공간보다 작은 크기의 Program이 들어가 할당된 공간을 Program이 차지한 공간을 제외한 공간이 빈 상태로 남아있는 단편화 현상. Paging에서 발생한다.

External fragmentation : Free한 영역들의 크기가 할당을 필요해하는 Program보다 작아 Free한 상태로 계속 남아있는 단편화 현상. Segmentation에서 발생한다.

2 Memory : free space를 아껴쓴다.
Disk : seek time을 줄인다.

3 Segmentation fault : program이 허용되지 않은 memory 영역에 접근을 시도하거나 허용되지 않은 방법으로 memory에 접근을 시도할 때 발생한다.

Protection fault : Program이 OS의 공간에 접근하려고 할때 발생한다.

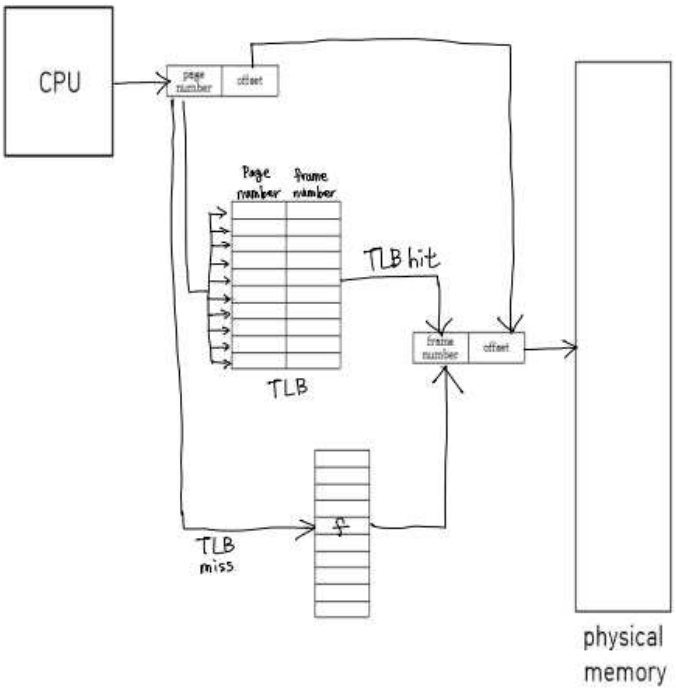
6 virtual address (14KB) : 11 1000 0000 0000
-> segment number : 11 (Stack)
-> offset : 1000 0000 0000 (2KB)
⇒ physical address : 28KB + (2KB - 4KB) = 26KB

7 virtual address : 2 -> physical address : 7 x 2B + 2 = 16
virtual address : 5 -> physical address : 10 x 2B + 1 = 21

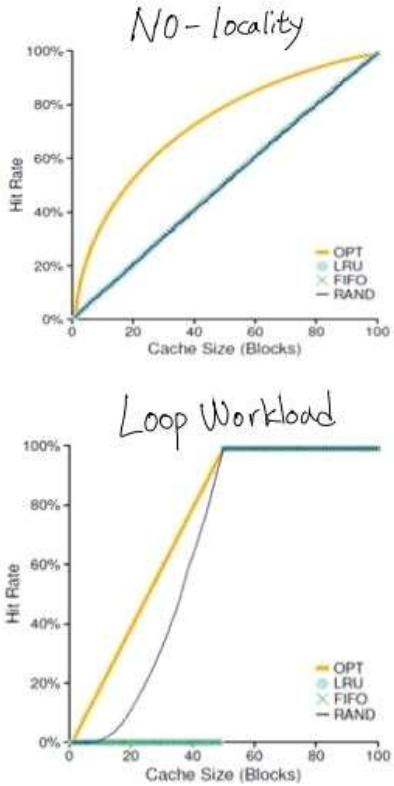
(9장 Paging and Beyond Physical Memory)

8. ❶ PCB에 들어있는 정보의 종류 3가지를 쓰고 설명하시오.(1점) ❷ TLB를 사용한 Address Translation의 전개과정이다. 필요한 화살표를 모두 표시하고 "page number, frame number, TLB, TLB miss, TLB hit" 단어들을 각각 맞는 위치에 위치시켜라.(2점) ❸ ASID의 fullname을 쓰시오.(1점) ❹ Multi level Page Table에서의 Virtual Address는 어디를 가르키는 bit들의 구성을 가지고 있는지 자세하게 기술하시오.(1점) ❺ Swap out을 하는 단위의 종류를 쓰고 어떤 경우에 어떤 단위를 사용하는지 쓰고 PTE에 있는 Present bit가 의미하는 바를 기술하시오.(1.5점) ❻ Paging에서 Replacement Policy 중 FIFO에서는 Belady's anomaly라는 현상이 일어난다. 이 현상은 예를 들어 cache size가 3frames, 4frames 두 종류가 있을 때 3fames일 때 hit ratio = 50%, 4frames 일 때 hit ratio = 20%처럼 frame의 수가 많을 때 오히려 적은 hit ratio를 보이는 것을 말한다. 이 Belady's anomaly의 원인과 이로 인한 영향(Thrashing을 활용하여 작성)은 무엇인지에 대해 기술하고 두 가지의 hit ratio에 대한 AMAT를 계산하시오. (T_M=100ns, T_D=10ms)(2.5점) ❼ 다음의 두 그래프는 OPT, LRU, FIFO, RAND를 비교한 그래프이다. 각각의 그래프가 어떠한 기준으로 비교한 것인지에 대해 예상해 보아라.(1점)

❷



❼



1 Process ID : Process의 고유번호
Process State : ready, run, wait 등의 상태
Process Counter : 다음 실행될 Process의 pointer

3 Address Space IDentifier

4 VPN(Page Directory Index, Page Table Index), offset

5 page : memory의 hungry condition이 커져을때
process : memory의 hungry condition이 우리을때

Present bit == 1 → page가 physical space에 존재
Present bit == 0 → page가 swap space에 존재

6 원인 : Temporal locality를 고려하지 않기 때문이
영향 : 낮은 Hit Ratio로 인해 잦은 page fault로 Thrashing이 발생

AMAT = (Hit x T_M) + (Miss x T_D)

1) 50 %
AMAT = (0.5 x 100) + (0.5 x 10,000,000) = 5,000,050 ≒ 5ms

2) 20 %
AMAT = (0.2 x 100) + (0.2 x 10,000,000) ≒ 8,000,020 ≒ 8ms

(2장, 3장, 7장, 8장)

9. (2장)❶ 다음 그림은 context switch의 store과 restore의 pseudo code의 일부분이다. 둘 중 어떤 코드가 save의 코드일지 선택하고 mode switch에 대해 설명하여라.(1.5점) (3장)❷ $\tau_{n+1} = \alpha t_n + (1-\alpha) \cdot \tau_n$ 이 공식은 다음 CPU burst를 예측하기 위한 공식이다. 이 공식에서 α가 가질 수 있는 값의 범위와 의미하는 바를 쓰시오.(1.5점) (7장)❸ 트리를 참고해서 FFS allocation을 적용한 표의 구성을 완성하시오.(group당 block의 수는 3으로 제한, group의 개수는 답과 관련이 없으므로 추가 및 삭제 가능)(3점) (7장)❹ LFS에 대하여 빈칸을 채우시오.(2점) (8장)❺ malloc을 사용할 때 memory leak과 buffer overflow는 각각 어떤 이유로 발생하는 문제들인지 쓰시오.(2점)

❶

1)

```
movl 4(%esp), %eax
movl 28(%eax), %ebp
movl 24(%eax), %edi
movl 20(%eax), %esi
movl 16(%eax), %edx
movl 12(%eax), %ecx
movl 8(%eax), %ebx
movl 4(%eax), %esp
```

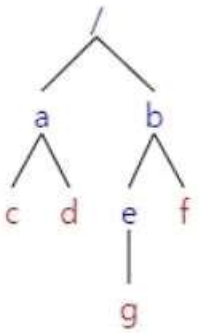
2)

```
movl %esp, 4(%eax)
movl %ebx, 8(%eax)
movl %ecx, 12(%eax)
movl %edx, 16(%eax)
movl %esi, 20(%eax)
movl %edi, 24(%eax)
movl %ebp, 28(%eax)
```

2번이 save의 코드이다.

mode switch : user mode에서 system call이 호출되면 system call interface를 통해 kernel mode로 전환시켜준다.

❸



a = "1 1 1 1 1 1"
b = "2 2"
c = "3"
d = "4"
e = "5 5 5 5 5 5"
f = "6 6"
g = "7 7 7 7"

group	inodes	data
0	/	/
1	acd	1 1 1 3 4
2	befg	1 1 1 2 2 5 5 5 6 6 7 7 7
3		5 5 5 7
4		
5		

❹

이름 : LFS의 Fullname : (Log Structured File System)

특징 : (Out)place update

Need to add new mapping information –) (inode map)

Need (garbage collection) for reclaiming invalidated data

[2] $0 \le \alpha \le 1$

determines the weight of each history

10번

- IoT (Internet of Things), 빅데이터(Bigdata), 클라우드(Cloud) 시대에 적합한 미래 운영체제의 새로운 기술을, CPU 가상화, 메모리 가상화, 병행성, 영속성이라는 4가지 측면에서 기술하시오. (전체 10점, 각 측면마다 2.5점씩)

IoT와 Bigdata, Cloud에 대해 검색해보고 찾아보면 와중 새로운 기술의 등장과 발전역시 중요하지만 더 중요한 것은 보안의 확실함 이라는 생각이 들었다. 또, 최근 토스에서 발생했던 해킹 사고에 대해서도 들은 바가 있었기에 보안은 역시나 어떠한 기술이 있어서도 선행되어야 한다는 생각 또한 하였다.

따라서 나는 새로운 보안 기술을 소개해보고 그에 따라 CPU 가상화, Memory 가상화, 병행성, 영속성에서의 측면을 기술해 보았다.

1. CPU 가상화, Persistance

이번 토스의 사고에 대한 내 생각은 보안인증에 있어서 더욱 깊은 안정성이 필요하다고 생각했다. 지금 제안하는 방법으로 인해서 인해서 모든 문제가 해결되진 않겠지만 사고의 가능성을 현저하게 줄일 수 있을 것이라고 생각한다. 내가 제안할 방법은 동시접속자를 막는 것이다. 나는 CPU 가상화를 통해 각각의 ID가 접속하면 접속에 대한 표시를 하는 Flag가 들어있는 CPU를 부여하고 같은 ID에 대해 접속이 시도되고 같은 ID에 대한 CPU의 Flag가 켜지려고 시도될 때 이미 켜져있는 Flag에 의하여 접속을 제한하는 방법이다. 또 금융과 관련된 경우 atomicity에 대한 보장은 로그아웃에 의해 보장된다. 만약 동시접속이 발생하여 강제로 로그아웃이 되고 ID가 상실 경우 Transaction을 사용하여 atomicity를 보상한다. 이러한 방법을 통하여 보안을 한층 강화시킬 수 있다고 생각한다.

2. Memory 가상화

Memory 가상화에서는 어떻게 하면 physical memory를 좀더 효율적으로 사용할 수 있을까 라는 문제에 초점을 맞추어 보았다. virtual memory를 사용함으로써 야기되는 문제중 하나인 External fragmentation에 대해 생각해보았다. External fragmentation에 대한 해결은 paging 기법을 사용하면 되지만 paging에서는 Internal fragmentation이 발생하게도 하고 allocation 에는 시간이 많이 소요되는 것으로 알고 있다. 그에 따라서 나는 반대로 virtual memory per process가 아닌 virtual memory per physical memory를 생각해 보았다. 애초에 process에게 virtual memory를 부여하여 segmentation이나 paging으로 physical memory에 접근하는 것이 아니라 일정 간격으로 나눈 physical memory에 virtual memory를 부여하여 크기가 맞는 process에게 해당 virtual memory를 부여한다면 External fragmentation 뿐만 아니라 Internal fragmentation 또한 해결될 것이라고 생각한다. 또한 paging은 page table의 사용으로인해 2번의 memory 접근이 필요하고 이 접근의 수를 줄이기 위해 TLB등의 기술을 사용하지만 physical memory에 직접 virtual memory를 부여해준다면 1번의 memory 접근은 필수적이겠지만 virtual memory의 주소가 즉 physical memory의 주소이기 때문에 그 이외의 memory 접근은 요구되지 않을것이다.

3. Concurrency

IoT의 발전이 거듭될수록 보안에 대한 문제는 커져만 간다. 따라서 나는 이 문제를 Concurrency에도 이용해보려고 한다. Deadlock을 활용하여 보안을 강화시키는 것이다. 마치 방에 문을 열고 들어가려고 하는데 문 앞에 태풍이 설치되어 있다고 생각해본다면 우리는 태풍이 사라지기 전까지는 문을 열지 못할 것이다. 예를 들어 노트북이 있는 카메라의 process에 해커가 접근을 시도한다고 생각해보자. 카메라에 접근을 시도하려고 할때 카메라에 대한 process는 살아있지만 CPU 가상화를 통한 다른 deadlock 상태인 process 들을 카메라에 대한 process 앞에 쌓아 놓는다면 문을 열기 전의 태풍과 비슷한 형태로 인해 보안이 강화될 것이라고 생각한다.