

## 운영체제론

### 2014년 시행 5급(기술) 공채 제2차시험

응시번호 :

성명 :

제 1 문. 현대의 대부분의 컴퓨터는 가상 메모리 기법을 지원하기 위하여 메모리 관리 유닛(MMU)을 제공하고 있다. MMU는 내부에 TLB가 있어서 최근에 사용한 가상 주소 페이지와 물리 페이지의 매핑 정보를 캐싱하므로 주소 변환을 가속화하는 기능을 제공한다. 다음 물음에 답하시오. (총 25점)

- 1) 일반적으로 운영체제를 설계할 때, 커널과 응용 프로그램을 하나의 주소 공간에 매핑하는 방식을 사용하고 있다. 만일 커널 주소 공간을 분리한다면 TLB 관점에서 어떤 문제가 발생하는가? (10점)
- 2) Copy-on-write가 수행될 때, TLB에 주는 영향에 대하여 기술하시오. (10점)
- 3) 하드웨어에 의해서 관리되는 기존의 TLB가 가상화 환경에서 사용되는 경우의 문제점을 기술하시오. (5점)

제 2 문. 페이징 기법으로 메모리를 관리하는 시스템의 경우, 내부단편화(Internal Fragmentation)로 인한 메모리 낭비가 발생한다. 내부단편화로 인한 메모리 낭비를 정량화하기 위해 공간과 시간의 곱(STP: Space Time Product)의 개념을 사용한다. 예를 들어 100 바이트의 메모리가 내부단편화로 인해 5단위 시간 동안 낭비되었다면, 이 때의 STP는  $100 \times 5 = 500$ 이 된다.

응용프로그램들이 사용할 수 있는 메모리 공간이 3개의 페이지이고, 한 페이지의 크기가 1024 바이트라고 가정하자. 현재 시스템에는 3개의 프로세스 ( $P_A$ ,  $P_B$ ,  $P_C$ )가 수행중이며, 각 프로세스가 차지하는 메모리의 크기는 각각 3800 바이트, 2560 바이트, 3012 바이트이다. 각 프로세스는 가상 메모리 상에서 0번지부터 연속된 공간에 위치한다고 가정한다. 현재 시각 ( $t = 5$ )의 페이지 탑재 상태는 아래와 같다. ( $P_i^j$ : 프로세스  $i$ 의  $j$ 번째 페이지)

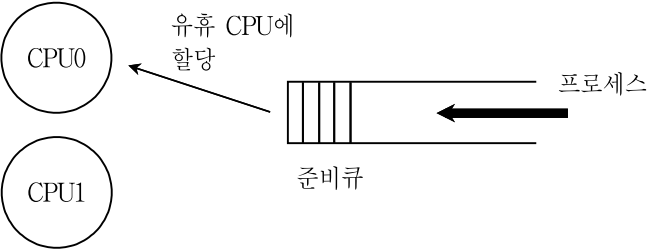
물리페이지	탑재된 페이지	탑재 시각	최근 참조 시각
Page 1	$P_A^1$	0	4
Page 2	$P_B^1$	2	3
Page 3	$P_C^1$	5	5

$t = 6$ 부터의 페이지 참조열이 아래와 같다고 가정하자. (단, 매 1단위시간마다 한 번의 참조가 발생한다)

시각( $t$ )	6	7	8	9	10	11	12	13	14	15	16
참조 페이지	$P_A^2$	$P_A^3$	$P_C^2$	$P_B^2$	$P_A^4$	$P_B^3$	$P_C^3$	$P_B^2$	$P_A^4$	$P_B^3$	$P_C^3$

LRU 페이지 교체 기법을 사용하였을 경우,  $t = 6$ 부터  $t = 17$ 까지 내부단편화로 인해 낭비된 총 메모리량을 STP 값으로 표현하시오. (20점)

제 3 문. 어떤 시스템이 멀티프로세서 스케줄링을 지원하기 위하여 FCFS(First-Come First-Served), RR(Round-Robin), SRT(Shortest Remaining Time)의 3가지 기법을 지원한다. RR의 시간할당량(time quantum) 크기는 2이다. 시스템에 CPU0, CPU1 등 2개의 CPU가 있고, 준비큐(Ready queue)는 전역적으로 하나만 존재한다. 어떤 CPU가 유휴(idle) 상태가 되면 준비큐의 작업을 그 유휴 CPU에서 수행시키는 CPU 할당방식을 따른다. (단, 2개의 CPU가 동시에 유휴 상태가 되면 준비큐의 프로세스가 CPU0부터 할당된다고 가정한다)



아래와 같은 프로세스들이 시스템에 들어올 때 다음 물음에 답하시오.  
(총 30점)

프로세스	도착시간	수행시간
A	1	3
B	2	8
C	3	5
D	4	6

1) FCFS, RR, SRT로 수행하는 경우, 매 시간 어느 프로세스가 어떤 CPU에서 스케줄링되는지 표시하시오. (단, RR의 경우에는 여러 프로세스가 동시에 준비큐에 삽입될 때에는 프로세스 이름의 알파벳 순서가 빠를수록 준비큐의 앞쪽(front)에 놓인다고 가정한다) (12점)

FCFS

시간 \ CPU	1	2	3	4	5	6	7	8	9	10	11	12	13	14
CPU0														
CPU1														

RR

시간 \ CPU	1	2	3	4	5	6	7	8	9	10	11	12	13	14
CPU0														
CPU1														

SRT

시간 \ CPU	1	2	3	4	5	6	7	8	9	10	11	12	13	14
CPU0														
CPU1														

- 2) 이 프로세스들을 스케줄링함에 있어 FCFS, RR, SRT의 평균 반환시간(turnaround time)은 각각 얼마인가? (9점)
- 3) 위의 결과를 관찰했을 때, 일반적으로 단일 처리기 시스템에서 사용하는 대기 시간이나 반환시간을 멀티프로세서 시스템의 스케줄링에 대한 평가 척도로 사용할 때의 문제점은 무엇인가? (9점)

제 4 문. 다음은 n개의 프로세스가 있는 환경에서 프로세스 i의 임계 구역(critical section) 진입 여부를 결정하는 Bakery 알고리즘을 나타낸 것이다. 이 알고리즘에서 choosing[]과 number[]는 모든 프로세스가 공유하는 자료 구조로 각각 false와 0으로 초기화되어 있다고 가정한다. 또한  $(a, b) < (c, d)$ 가 참이 되기 위해서는  $a < c$ 이거나,  $a=c$ 이고  $b < d$ 이어야 한다고 가정한다. 다음 물음에 답하시오.

(총 25점)

```

1  choosing[i] = true;
2  number[i] = max(number[0], number[1], ..., number[n-1]) + 1;
3  choosing[i] = false;
4  for (j = 0; j < n; j++) {
5      while (choosing[j]);
6      while ((number[j] != 0)&& ((number[j], j) < (number[i], i)));
7  }
8
9  // critical section
10
11 number[i] = 0;
12
13 // remainder section

```

- 1) 위 알고리즘의 2번 줄에서 보듯이 임계 구역에 진입하고자 하는 프로세스는 이미 발급된 번호보다 1이 큰 번호를 부여받으므로, 임계 구역에 진입하기 위해 대기하고 있는 프로세스는 모두 다른 번호를 부여받을 것으로 예상된다. 그러나 위 알고리즘의 6번 줄을 보면 다음과 같이 비교를 하고 있다. (여기서 i와 j는 프로세스 번호이며  $i \neq j$ 이다)

(number[j], j) < (number[i], i)

임계 구역에 진입하기 위해 대기 중인 프로세스는 모두 다른 번호를 받을 것으로 예상되므로 아래와 같이 단순 비교를 해도 될 것 같은데, 이와 같이 하지 않고 위와 같이 비교를 하는 이유는 무엇인가? (12점)

number[j] < number[i]

- 2) 5번 줄에서는 프로세스 j가 번호를 부여받는 동안 기다리고 있는데, 프로세스 i는 이미 번호를 부여받은 상태이기 때문에 프로세스 j가 부여받을 번호는 프로세스 i가 부여받은 번호보다 클 것이다. 그렇다면 이런 경우에는 굳이 6번 줄의 while 문을 수행할 필요가 없으므로 5번 줄을 다음과 같이 고칠 수 있을 것이다.

if (choosing[j]) continue;

그럼에도 불구하고 이렇게 하지 않은 이유는 무엇인가? (13점)

## 안전행정부 시험출제과장