요약: 제12장 - CPU 스케줄링: 비례 배분 (Proportional Share Scheduling)

---

1. 비례 배분 스케줄러 개요

비례 배분 스케줄러는 각 프로세스가 CPU 자원을 정해진 비율로 할당받는 것을 목표로 함.

추첨 스케줄링(lottery scheduling)은 무작위 추첨을 통해 CPU를 할당하며, 비율에 따라 추첨권(ticket)을 부여.

---

2. 추첨 스케줄링 (Lottery Scheduling)

기본 개념: 각 프로세스가 가진 추첨권 수에 비례하여 CPU를 할당받을 확률이 증가.

예제:
A 프로세스가 75개의 티켓, B가 25개의 티켓을 가지면, A

는 75%의 CPU를, B는 25%를 받도록 설계.

장점:

1. 무작위성이 불규칙한 상황을 처리하기 용이.

2. 관리 비용이 적음(상태 정보 최소화).

3. 빠른 결정이 가능(난수 생성 후 추첨권만 확인).

---

3. 구현과 작동 방식

총 추첨권에서 무작위로 하나의 숫자를 뽑아 해당 프로세스를 선택.

각 타임 슬라이스가 끝날 때마다 같은 과정을 반복.

구현 예시 코드:

```
int winner = getrandom(0, totaltickets);
while (current) {
    counter += current->tickets;
    if (counter > winner) break;
    current = current->next;
}
```

---

4. 보폭 스케줄링 (Stride Scheduling)

결정적(deterministic)인 방식으로 각 프로세스에 **보폭(stride)**을 할당.

보폭은 추첨권 수에 반비례(더 적은 추첨권일수록 큰 보폭).

작동 방식: 가장 작은 pass 값을 가진 프로세스를 실행하며, 실행 후 보폭만큼 pass 값을 증가.

장점: 정확한 비율로 CPU 할당을 보장하지만, 새로운 프로

세스 추가 시 복잡도가 높음.

---

5. 추첨권 배분의 어려움과 한계

정확한 비율 보장이 어렵고, 특히 짧은 시간 동안 비율이 맞지 않을 수 있음.

새로운 프로세스가 도착할 때 보폭 스케줄링은 복잡도가 높아지지만, 추첨 스케줄링은 간단히 대응 가능.

---

6. 실제 활용과 한계

추첨 및 보폭 스케줄링은 흥미로운 개념이지만, 입출력(I/O)과 같은 복잡한 환경에서 성능이 떨어질 수 있음.

MLFQ와 같은 스케줄러가 더 직관적이고 널리 사용됨.