**CPU Scheduler 보고서**

2023131014 은주연

서론

\* CPU 스케줄러의 개념 등

\* 본인이 구현한 스케줄러에 대한 요약 설명(특징 등)

o 본론

\* 다른 CPU 스케줄링 시뮬레이터에 대한 소개

\* 본인이 구현한 시뮬레이터의 시스템 구성도

\* 각 모듈에 대한 설명(알고리즘으로 표현)

\* 시뮬레이터 실행 결과 화면

\* 알고리즘들간의 성능 비교 등

o 결론

\* 구현한 시뮬레이터에 대한 정리

\* 프로젝트 수행 소감 및 향후 발전방향 등

목차

1. **서론**
   1. **CPU 스케줄러의 개념**

CPU 스케줄러는 운영체제의 핵심 구성요소 중 하나로, 시스템 내의 여러 프로세스 중에서 어떤 프로세스에게 CPU 자원을 언제, 얼마나 할당할지를 결정하는 알고리즘적 중재자이다. 이는 단순한 자원 할당 기능을 넘어, 시스템의 전체적인 효율성과 사용자 체감 성능에 직결되는 판단자로서의 역할을 수행한다.

운영체제는 하나의 CPU를 여러 개의 프로세스가 경쟁하는 상황에서 각 프로세스의 상태와 요구사항을 고려하여 CPU를 공정하고 효율적으로 배분해야 한다. 이때 CPU 스케줄러는 ready 상태에 있는 프로세스들 중에서 다음으로 실행될 프로세스를 선택하며, 이 선택의 기준이 되는 것이 바로 스케줄링 알고리즘이다. 알고리즘의 종류에 따라, 사용자의 응답 시간, 시스템 처리량, 대기 시간, 자원 활용률 등 운영체제의 다양한 성능 지표가 달라진다.

CPU 스케줄링은 다음과 같은 네 가지 주요 상황에서 필요하다.

첫째, running → waiting 상태 전환은 프로세스가 스스로 I/O 요청 등의 이유로 CPU를 반납하고 waiting 상태로 이동할 때 발생하며, 이는 비선점(Nonpreemptive) 상황이다.

둘째, running → ready 전환은 선점(Preemptive) 상황으로, 타임 슬라이스 종료나 더 높은 우선순위 프로세스의 등장으로 인해 현재 프로세스를 강제로 중단하고 다른 프로세스에 CPU를 재할당한다.

셋째, waiting → ready 전환은 I/O 완료 시 발생하며, 역시 우선순위나 알고리즘 조건에 따라 선점이 가능하다.

마지막으로, terminate는 작업 완료에 따른 상태 전환으로, ready queue가 비어 있다면 CPU는 idle 상태로 전환된다.

결국 CPU 스케줄러는 동시에 다수의 프로세스를 처리하려는 운영체제의 핵심 메커니즘이다. 본 프로젝트에서는 이 스케줄러가 실제로 어떻게 동작하는지를 다양한 알고리즘 구현을 통해 시뮬레이션하고, 각 방식의 성능 특성과 차이를 evaluation을 통해 분석하고자 한다.

* 1. **구현된 스케줄러의 요약 (특징 등)**

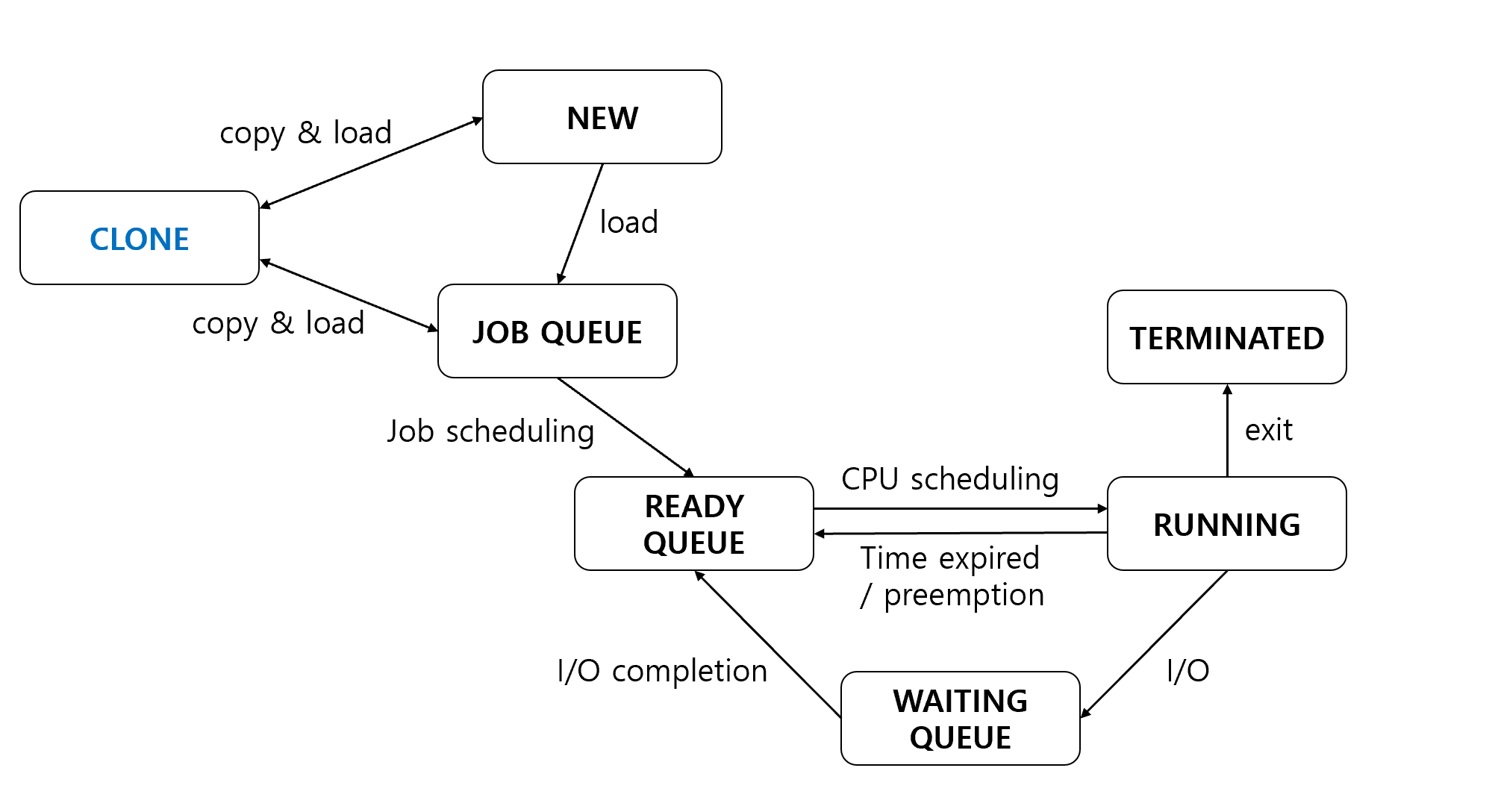
일단 처음에 State로 프로세스의 상태를 { NEW, READY, RUNNING, WAITING, TERMINATED } 라고 정의했다. 각 프로세스는 해당 상태를 가질 수 있다. 그리고 Process 구조체를 만들어서 pid(process ID), arrival time(도착 시간), burst time(전체 CPU 수행 시간), remaining time(남은 CPU 수행 시간), io request times(I/O요청 시점들),

그리고 Process와 job queue를 clone 해서 다음 알고리즘에서도 같은 조건으로 계속 실행될 수 있도록 했다.

1. **본론**
   1. **다른 CPU 스케줄링 시뮬레이터에 대한 소개**

ddd

* 1. **시뮬레이터의 시스템 구성도**



dd

* 1. **각 모듈에 대한 설명 (알고리즘으로 표현)**

1. FCFS
2. SJF
3. SJF preemptive
4. Priority
5. Priority preemptive
6. Round Robin
7. Config
8. Evaluation
9. Print gantt chart
   1. 시뮬레이터 실행 결과 화면
   2. 알고리즘들간의 성능 비교 등