과제 보고서

차례

I. 서론
Ⅱ. 본론
1. 기아 상태(Starvation)의 정의
2. 다익스트라의 ‘ 식사하는 철학자 문제 ’
3. 교착 상태 해결 방안
Ⅲ. 결론
참고문헌

I. 서론

운영체제는 다수의 프로세스가 제한된 시스템 자원을 효율적으로 공유하도록 설계되어 있다. 그러나 여러 프로세스가 동시에 자원을 요구하는 상황에서 일부 프로세스가 지속적으로 자원을 얻지 못해 대기 상태에 머무는 현상이 발생할 수 있다. 이러한 문제는 시스템의 효율성과 공정성에 직접적인 영향을 미친다. 따라서 자원 경쟁에서 발생하는 기아 상태(Starvation)는 운영체제 설계에서 중요한 고려 요소로 다뤄진다.

본 보고서에서는 먼저 기아 상태의 개념을 정의하고, 다익스트라(Edsger W. Dijkstra)가 제안한 고전적 동기화 문제인 ‘ 식사하는 철학자 문제(Dining Philosophers Problem) ’ 를 분석한다. 이를 통해 교착 상태(Deadlock)의 원인과 해결 방안을 고찰하고, 궁극적으로 효율적이고 공정한 자원 관리의 중요성을 논의하고자 한다.

Ⅱ. 본론

1. 기아 상태(Starvation)의 정의

기아 상태(Starvation)란 운영체제에서 하나 이상의 프로세스가 자원을 요청했음에도 불구하고, 일정 시간 이상 자원을 할당받지 못하는 상태를 말한다. 즉, 시스템의 스케줄링 정책이나 자원 할당 알고리즘에 의해 일부 프로세스가 계속 대기 상태에 머무르며 실행되지 못하는 현상이다. 이러한 상태는 공정성(fairness)을 저해하며, 시스템의 응답 성능에도 부정적인 영향을 미친다.

기아 상태는 주로 다음과 같은 원인으로 발생한다. 첫째, 우선순위 스케줄링에서 낮은 우선순위를 가진 프로세스가 CPU를 계속 할당받지 못하는 경우이다. 둘째, 특정 자원이 반복적으로 동일한 프로세스에 의해 선점되는 경우이다. 셋째, 자원 할당이 비공평하게 이루어질 때 발생한다. 따라서 자원 접근의 균형을 맞추기 위한 정책이 반드시 필요하다.

2. 다익스트라의 ‘ 식사하는 철학자 문제 ’

식사하는 철학자 문제(Dining Philosophers Problem)는 1965년 네덜란드의 컴퓨터 과학자 다익스트라(Edsger W. Dijkstra)에 의해 제안된 동기화(synchronization) 문제이다. 이 문제는 병렬 프로세스 간의 자원 경쟁으로 인해 발생할 수 있는 교착 상태(Deadlock)와 기아 상태를 설명하는 대표적인 예시로 널리 알려져 있다.

문제의 설정은 다섯 명의 철학자가 원형 식탁에 앉아 있고, 철학자들 사이에는 하나씩 포크가 놓여 있는 상황이다. 각 철학자는 ‘ 생각하기(thinking) ’ 또는 ‘ 식사하기(eating) ’ 두 가지 행동 중 하나를 수행한다. 철학자가 식사하려면 자신의 왼쪽과 오른쪽에 놓인 두 개의 포크를 모두 들어야 한다. 그러나 모든 철학자가 동시에 왼쪽 포크를 집고 오른쪽 포크를 기다리면, 아무도 두 개의 포크를 완전히 얻지 못하고

무한히 대기하는 교착 상태가 발생한다.

3. 교착 상태 해결 방안

다익스트라의 식사하는 철학자 문제는 단순한 모델이지만, 교착 상태를 예방하기 위한 다양한 해결책을 탐구할 수 있다. 주요 해결 방안은 다음과 같다.

① **자원 할당 순서 규칙(Resource Ordering Rule)**
모든 자원에 고유한 번호를 부여하고, 프로세스가 항상 번호가 낮은 자원부터 요청하도록 한다. 이를 통해 원형 대기(Circular Wait) 조건을 방지할 수 있다.

② **비대칭 자원 요청(Asymmetric Resource Request)**
철학자들이 자원을 요청하는 순서를 일부 변경하여, 예를 들어 홀수 철학자는 왼쪽 포크부터, 짝수 철학자는 오른쪽 포크부터 집게 한다. 이를 통해 자원 요청의 대칭성을 깨뜨려 교착 상태를 예방한다.

③ **최대 접근 제한(Limiting Concurrent Access)**
식탁에 동시에 앉을 수 있는 철학자의 수를 N - 1명으로 제한함으로써, 모든 철학자가 동시에 두 개의 포크를 요청하는 상황을 방지한다.

④ **우선순위 상승(Priority Aging) 및 타임아웃(Timeout)**
오랫동안 대기한 프로세스의 우선순위를 점차 높이거나, 일정 시간 자원을 얻지 못하면 요청을 포기하도록 하여 기아 상태를 예방할 수 있다.

III. 결론

기아 상태와 교착 상태는 운영체제의 자원 관리에서 가장 중요한 문제 중 하나로, 시스템의 안정성과 효율성을 위협한다. 다익스트라의 ‘ 식사하는 철학자 문제 ’ 는 이러한 자원 경쟁 문제를 단순화하여 표현함으로써, 프로세스 간 동기화의 중요성을 직관적으로 이해하게 한다.

운영체제는 이러한 문제를 방지하기 위해 자원 접근 순서 제어, 공정한 스케줄링, 우선순위 조정 등의 정책을 통해 시스템의 안정성을 확보해야 한다. 궁극적으로는 모든 프로세스가 공평하게 자원을 이용할 수 있는 구조를 설계함으로써 효율적이고 신뢰할 수 있는 운영체제를 구현할 수 있다.

참고문헌

1) Silberschatz, A., Galvin, P. B., & Gagne, G. (2020). Operating System Concepts (10th Edition). Wiley.
2) 다익스트라(E. W. Dijkstra), “Cooperating Sequential Processes,” in Programming Languages, 1968.
3) Stallings, W. (2018). Operating Systems: Internals and Design Principles (9th Edition). Pearson.

연구자 정보

이름 : 강우현
전공 : 인공지능융합공학부
이메일 : kangwoohyun999@gmail.com
전화번호 : 010 - 2872 - 9096