설계 보고서

운영체제(노삼혁 교수님) 1분반 **A411088 나 승 훈**

1단계 구현의 문제점

1단계에서 제시된 사항만으로 구현하였을 경우 writer가 임계구역에 접근할 수 없는 starvation이 발생한다. reader와 writer는 사용자가 만든 개수에 따라 각각 여러 개의 스레드를 생성하게 된다. reader는 writer가 현재 공유되는 데이터에 쓰기를 하고 있지만 않다면 여러개의 reader들은 마음껏 공유되는 데이터를 읽어볼 수 있다. 그러나 반면 writer는 하나의 reader만 공유되는 데이터를 읽고 있어도 쓸 수가 없다. 왜냐하면 reader가 읽는 도중에 데이터가 바뀐다면 reader는 잘못된 정보를 가져가는 것이기 때문이다. 따라서 writer의 입장에서는 공유데이터에 접근하고 있는 스레드가 아무도 없어야지만 쓸 수 있는 것이다.

starvation발생의 원인분석

그런데 운영체제는 여러 개의 스레드를 운영함에 있어서 나름대로의 스케줄 방식에 따라 실행할 스레드의 순서를 배정한다. 그런데 writer가 배정되어서 공유데이터에 접근하려고 하지만 reader가 읽고 있어서 실행되지 못하고 잠드는 현상이 발생할 수 있다. 일정시간 잠들어 있다가 스케줄러에 의해서 다시 배정되었지만 또 다시 reader가 읽고 있는 상황이 반복될 수 있다. 이러한 반복이 계속된다면, writer는 계속해서 스케줄 됨에도 불구하고 계속해서 다시 잠들면서 쓰기를 수행하지 못하는 상황이 발생할 수 있다. 이것이 writer의 기아현상이다. 물론 이러한 현상이 자주 발생되지는 않겠지만, reader가 많은 경우라면 reader1이 종료되기 전에 context switch가 일어나서 reader2가 수행되고, 또 도중에 reader3이 들어오고 하면서 계속해서 reader들이 공유데이터를 차지하고 있을 수 있기 때문이다. reader가 많은 경우 몇몇의 reader들이 수행을 마치고 빠져나온다고 하더라도, 그 이후에도 계속 다른 reader들이 배정된다면 reader들이 공유데이터를 차지하는 시간이 많아지게 된다. 이에 따라 writer는 스케줄 될 때마다 목적을 이루지 못하고 다시 잠들며 영원히 공유데이터에 접근하지 못하는 현상이 발생할 수 있는 것이다. 이것이 writer의 기아현상이다.

starvation현상의 해결책

writer의 기아현상을 해결하려면 계속해서 reader가 스케줄 되는 현상을 막으면 가능하다. 즉 , writer의 요청이 들어온다면 writer의 요청 뒤에 들어오는 reader들의 요청은 writer뒤로 스케줄 시키는 것이다. writer의 요청이 들어오면 writer가 실행을 마치기까지 다른 reader들은 요청은 뒤로 미루고, 지금까지 공유데이터를 읽고 있던 reader들을 모두 먼저 처리해서 writer가 쓰기를 수행할 수 있도록 해준다. 이렇게

프로그램을 구현한다면 writer가 영원히 공유데이터에 접근하지 못했던 문제를 해결할 수 있다.

실제적인 구현방법

writer가 스케줄 되었는데 공유데이터에 접근할 수 없는 상황이라면 조건변수를 기다리면서 잠들도록 구현한다. 만약 reader들이 모두 공유데이터를 빠져나오는 상황이 발생하면 조건변수에 대하여 시그널을 날려서 조건변수를 기다리며 잠들어있던 writer를 깨운다. 깨어난 writer는 공유데이터가 비어있기 때문에 쓰기를 수행할 수 있게 된다.

* writer가 쓰기를 수행하기 위해 공유데이터에 접근할 때 :
  : pthread_cond_wait(&cond)사용
* reader가 읽기 수행을 마치고 read counter가 0이되면 :
  : pthread_cond_signal(&cond)사용