

# Operating Systems

## Project #4

컴퓨터학부

2023년 5월 8일

### Reader-Writer 문제

Reader와 writer가 공유자원에 접근할 때, reader는 다른 reader와 동시에 접근할 수 있다. 하지만 writer가 다른 writer나 reader와 동시에 접근하면 문제가 발생할 수 있는 것을 reader-writer 문제라고 한다. 이 문제를 해결하기 위한 해법에는 reader 선호, writer 선호, 공정한 reader-writer, 세 가지가 있다. 어떤 방식이든 모두 writer에게는 상호배타를 보장하고, reader에게는 다른 reader와 최대한 중복을 허용하는 것을 목표로 한다.

### Reader 선호

수업시간에 배운 해법으로 CS(Critical Section; 임계구역)에 reader가 있으면 다른 reader가 중복해서 들어갈 수 있다. 그러나 이 방식은 늦게 온 reader가 기다리고 있는 writer를 앞지를 수 있어서 writer가 굼주릴 수 있다. 특히 reader의 수가 상대적으로 많은 환경에서는 writer가 기회조차 얻지 못하는 경우가 발생할지도 모른다.

### Writer 선호

Reader의 중복을 최대한 허용하되 기다리는 writer가 있으면 reader가 더 이상 writer를 앞지르지 못하게 하는 방식이다. 뿐만 아니라 늦게 온 writer는 기다리고 있는 reader를 앞지를 수 있어서 이번에는 reader가 굼주릴 수 있다. 이러한 방식을 writer 선호라 부르고, writer의 수가 상대적으로 적은 환경에서 유리하다.

### 공정한 reader-writer

Reader 선호 방식이나 writer 선호 방식은 어느 한 쪽이 굼주릴 수 있다. 공정한 reader-writer는 선착순으로 CS에 들어가면서 reader의 중복을 최대한 허용하는 방식이다. 이 방식에서는 늦게 온 reader/writer가 기다리고 있는 다른 reader/writer를 앞지르지 못한다.

### 구현

시스템에 20개의 reader 스레드와 5개의 writer 스레드가 있다. 각 스레드는 CS에 들어가서 필요한 공유자원을 읽거나 쓴다. 스레드가 CS에 들어가면 표준출력이라는 가상 공유자원을 접근한다고 가정한다. 스레드가 CS에 들어왔다는 것을 보이기 위해 다음과 같은 문자나 문자열을 출력한다. (주의: 출력 결과를 올바르게 보려면 흰색 바탕의 큰 화면을 사용해야 한다.)

- Reader가 CS에 들어가면 첫 번째 reader는 A 문자를, 두 번째 reader는 B 문자를, ..., 이런 식으로 각 reader는 같은 문자를 8192개 출력하고 CS를 빠져나온다. Reader는 이 과정을 반복한다.
- Writer가 CS에 들어가면 배열에 저장된 어떤 얼굴 이미지를 출력한다. 출력이 끝나면 CS를 빠져나온다. 한 번 출력이 끝나면 SLEEPTIME 나노초 (nanoseconds) 내에서 랜덤 시간만큼 쉬었다가 다시 이 과정을 반복한다.

앞에서 요구한 reader와 writer는 `reader_writer.skeleton.c`라는 골격파일에 구현되어 있다. 다만 CS 접근을 통제할 수 있는 스레드 동기화 코드가 빠져있다. 이 골격파일을 사용하여 reader-writer 문제를 푸는 다음 네 가지 버전을 설계하고 구현한다.

- **Reader 선호 (조건변수 버전):** POSIX 조건변수를 사용하여 reader 선호 방식을 구현한다. 단, 조건변수를 조회하기 위한 뮤텝락 (mutex lock) 한 개 이외에는 어떠한 뮤텝락이나 세마포를 사용할 수 없다.
- **Writer 선호 (뮤텝락 버전):** POSIX 뮤텝락을 사용하여 writer 선호 방식을 구현한다. 단, POSIX 조건변수나 세마포를 사용할 수 없다.
- **Writer 선호 (조건변수 버전):** POSIX 조건변수를 사용하여 writer 선호 방식을 구현한다. 단, 조건변수를 조회하기 위한 뮤텝락 한 개 이외에는 어떠한 뮤텝락이나 세마포를 사용할 수 없다.
- **공정한 reader-writer (뮤텝락 버전):** POSIX 뮤텝락을 사용하여 공정한 reader-writer 방식을 구현한다. 단, POSIX 조건변수나 세마포를 사용할 수 없다.

## 골격파일

앞에서 언급한 것처럼 골격파일 `reader_writer.skeleton.c`에는 reader와 writer가 구현되어 있다. 이 골격파일을 컴파일해서 실행하면 약 RUNTIME 나노초 동안 25개의 스레드가 동기화 없이 각자 출력을 쏟아내고 종료한다. 컴퓨터 성능에 떨어져서 실행 시간이 부족하다고 생각되면 RUNTIME 값을 환경에 맞게 늘린다. Writer의 출력 스피드를 조절하는 SLEEPTIME도 각자의 컴퓨팅 환경에 맞게 조절할 수 있다.

## 제출물

Reader-writer 문제에 대한 네 가지 해법이 잘 설계되고 구현되었다는 것을 보여주는 자료를 각자가 판단하여 PDF로 묶어서 이름\_학번\_PROJ4.pdf로 제출한다. 여기에는 다음과 같은 것이 반드시 포함되어야 한다.

- ~~본인이 설계한 네 가지 버전에 대한 설명 (총 4쪽 이내)~~
- ~~컴파일 과정을 보여주는 화면 캡처~~
- 실행 결과물의 주요 장면을 발췌해서 그에 대한 상세한 설명 및 차이점
- ~~과제를 수행하면서 경험한 문제점과 느낀점~~
- 프로그램 소스파일 4개 (`reader_prefer_cond.c`, `writer_prefer_mutex.c`, `writer_prefer_cond.c`, `fair_reader_writer_mutex.c`) 별도 제출

- 실행 결과물 4개 (reader\_prefer\_cond.txt, writer\_prefer\_mutex.txt, writer\_prefer\_cond.txt, fair\_reader\_writer\_mutex.txt). 이 결과물은 크기가 크기 때문에 소스파일과 마찬가지로 별도의 파일로 제출한다. 각 출력물은 해법이 올바르게 작동하는 지 판단할 수 있을 만큼 정보가 충분해야 한다. 만일 부족하여 행위를 충분히 관찰할 수 없다면 해법으로 인정하지 않는다.

## 평가

- Correctness 50%: 프로그램이 올바르게 동작하는 지를 보는 것입니다. 여기에는 컴파일 과정은 물론, 과제가 요구하는 기능이 문제없이 잘 작동한다는 것을 보여주어야 합니다.
- Presentation 50%: 자신의 생각과 작성한 프로그램을 다른 사람이 쉽게 이해할 수 있도록 프로그램 내에 적절한 주석을 다는 행위와 같이 자신의 결과를 잘 표현하는 것입니다. 뿐만 아니라, 프로그램의 가독성, 효율성, 확장성, 일관성, 모듈화 등도 여기에 해당합니다. 이 부분은 상당히 주관적이지만 그러면서도 중요한 부분입니다. 컴퓨터과학에서 중요하게 생각하는 best coding practices를 참조하기 바랍니다.

HK