Operating Systems
Project #2

컴퓨터학부

2025년 3월 31일

스도쿠 퍼즐

스도쿠 퍼즐은 9×9 격자로 된 보드게임이다. 보드의 모든 가로 줄과 세로 줄은 1부터 9까지 숫자가 중복 없이 나타나야 한다. 이 보드를 다시 9개의 3×3 부분 격자로 분할할 수 있는데, 부분 격자에서도 1부터 9까지 숫자가 중복 없이 나타나야 한다. 아래 그림은 스도쿠 퍼즐이 올바르게 구성된 하나의 해를 보여준다.

6	2	4	5	3	9	1	8	7
5	1	9	7	2	8	6	3	4
8	3	7	6	1	4	2	9	5
1	4	3	8	6	5	7	2	9
9	5	8	2	4	7	3	6	1
7	6	2	3	9	1	4	5	8
3	7	1	9	5	6	8	4	2
4	9	6	1	8	2	5	7	3
2	8	5	4	7	3	9	1	6

퍼즐 검증

이번 과제는 다중 스레드를 생성하고 운영하는 법을 학습한다. 이를 위해 여러 개의 스레드를 생성하여 스도쿠 퍼즐의 해가 올바른지 검증한다. 검증에 사용하는 스레드는 다음과 같이 세 가지 종류가 있다.

- 모든 가로 줄이 올바른지 검증하는 1개의 스레드
- 모든 세로 줄이 올바른지 검증하는 1개의 스레드
- 각각의 부분 격자가 올바른지 검증하는 9개의 스레드

물론 이렇게 많은 11개의 스레드를 생성하여 퍼즐을 검증하는 것은 시간적으로 전혀 도움이 되지 않는다. 이 과제의 목표는 얼마나 빨리 검증하는 것이 아니고, 다중 스레드를 어떻게 생성하고 활용하는 것에 있다. 뿐만 아니라 보드는 여러 개의 스레드가 동시에 접근하는 공유자원이다.

공유자원에 대한 동기화가 전혀 이루어지지 않은 상황에서 검증을 시도하기 때문에 예상하지 못한 데이터 불일치가 발생할 수 있다. 이러한 불일치를 관찰하고 공유자원에 대한 동기화가 왜 필요한지 체험한다.

골격파일

골격파일인 `sudoku.skeleton.c`에는 미완성으로 남아 있는 몇 개의 함수가 있다. 이것을 목적에 맞게 완성해야 하다. 그 함수의 목록은 다음과 같다.

- `void *check_rows(void *arg)`: 스도쿠 퍼즐의 9개 행의 구성이 올바른지 검사하는 함수이다. 하나의 스레드가 이 함수를 수행하게 한다. 매개변수 `arg`는 사용하지 않는다.
- `void *check_columns(void *arg)`: 스도쿠 퍼즐의 9개 열의 구성이 올바른지 검사하는 함수이다. 하나의 스레드가 이 함수를 수행하게 한다. 매개변수 `arg`는 사용하지 않는다.
- `void *check_subgrid(void *arg)`: 스도쿠 퍼즐의 3×3 부분 격자의 구성이 올바른지 9개 부분 격자 중 어느 하나를 검사하는 함수이다. 어떤 위치에 있는 부분 격자를 검사할 것인지는 매개변수 `arg`를 통해 정보를 전달한다. 이 함수를 사용하여 9개의 스레드가 각기 다른 위치에 있는 부분 격자의 구성을 검사하게 한다.
- `void check_sudoku(void)`: 스도쿠 퍼즐이 올바르게 구성되어 있는지 11개의 스레드를 생성하여 검증한다. 한 스레드는 각 행이 올바른지 검사하고, 다른 한 스레드는 각 열이 올바른지 검사한다. 9개의 3x3 부분 격자에 대한 검증은 9개의 스레드를 생성하여 동시에 검사한다. 검사가 끝나면 모든 스레드를 종료하고 결과를 출력한다.
- `void *shuffle_sudoku(void *arg)`: 스도쿠 퍼즐의 값을 3x3 부분 격자 내에서 값을 무작위로 섞는 함수이다. 이 함수는 퍼즐을 검증하는 다른 스레드와 동시에 실행된다. 그러나 공유자원에 대한 동기화가 전혀 이루어지지 않아서 이 함수는 올바른 검증을 방해하는 결과를 초래한다. 어디서 데이터 불일치가 발생했는지 검증 결과를 관찰한다. 이 함수는 완성된 상태로 제공된다.

골격파일과 함께 컴파일 과정을 쉽게 해주는 `Makefile`도 제공한다.

제출물

스도쿠 퍼즐 검증기가 잘 설계되고 구현되었다는 것을 보여주는 자료를 보고서 형식으로 작성한 후 PDF로 변환하여 `이름_학번_PROJ2.pdf`로 제출한다. 여기에는 다음과 같은 것이 반드시 포함되어야 한다.

- 본인이 작성한 함수에 대한 설명
- 컴파일 과정을 보여주는 화면 캡처
- 실행 결과물에 대한 상세한 설명
- 과제를 수행하면서 경험한 문제점과 느낀점
- 프로그램 소스파일 (`sudoku.c`) 별도 제출
- 프로그램 실행 결과 (`sudoku.txt`) 별도 제출

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