10주차 결과 보고서

전공: 컴퓨터공학 학년: 2학년 학번: 20221559 이름: 박준우

1. 실습 시간에 작성한 프로그램의 알고리즘과 자료구조를 요약하여 기술하시오.

실습 시간에는 추천 시스템을 구현하기 위해, 앞으로 알고 있는 블럭 수에 대해 모든 위치, 방향을 고려하여 각 단계에서 각 종류를 놓은 필드와, 그 때 얻은 점수를 트리 형태로 만들어 앞으로를 고려했을 때 가장 높은 점수를 얻을 수 있는 경우에서 현재의 선택을 알아본다.

이를 위해서 트리의 각 노드에는 누적 점수, 필드를 저장할 수 있는 배열, 자식노드들의 주소를 저장할 수 있는 배열과, 트리 순회의 종료 조건을 판단해 주기 위한 level을 포함했다.

Modifiy에서는 첫번째로는 돌려서 똑같은 경우를 제거하였다. 각 계층에서 포인터의 배열은 똑같이 선언하나, 실제 해당 포인터 주소에 노드를 생성하지 않게 되므로, 시간, 공간 복잡도가 pruning 된 만큼 줄어들게 된다.

두번째로는 field의 height 가 22로, 32보다 작음을 이용해 bitMask로 각 열의 상태를 저장하였다.

이를 위해 compressfield(), decompressfield()를 구현하였다. 이 방법을 이용해 각 노드의 크기에서 field가 차지하던 부분의 크기를 4/22로 줄일 수 있었다.

또 하나, 점수를 매기는 방법에서, 옆 칸이 벽이거나, 옆 칸에 블럭이 있는 경우에도 추가적인 점수를 주게 변경하여 예측 플레이에서 위로만 쌓이는 경우를 보정해 보았다.

2 - 1. Pruning tree

돌려서 같은 경우를 제거하여 시간적, 공간적 효율성을 확보했다. 어떤 다른 조건 (점수가 다른 것에 비해 낮은 경우)는 1단계로만 판단하면 여러 문제가 생길 수 있어 최소 2단계가 지난 후에 잘라야 하는데, 최대 level 3까지 이용하는 예제에서는 이런 구현의 의미를 크게 찾을 수 없을 것으로 판단하여 별도로 진행하지 않았다.

2-2. Data Simplification

위에서 설명했듯, bitmask를 이용했다. 이 경우에 강의자료에 있던 최대 높이를 저장하는 방식과 비교하자면, 강의자료의 방식은 최대 높이의 블럭 아래에 있던 정보를 손실한다는 단점이 있지만, 개수만 저장해 줌으로서 height가 커져도 관계없다는 장점이 있다.

반면 내가 사용한 방식은 height가 32 이상이 되면 8byte 자료형을 이용하거나, 이것으로도 부족하면 사용하기 힘들어진다는 단점이 있다. 그럼에도 데이터를 잃지 않고, 공간을 획기적으로 줄일 수 있다는 장점은 명확했다

3. 테트리스 3주 과정을 통하여, C 프로그래밍에서 header 파일의 중요성을 깨닫고, 함수로서 작은 부분을 분리하여 구현하는 방법을 배우고, 또 큰 프로그램 소스코드에 익숙해 지는 기회가 되어서 좋았다. 또 포인터를 이용한 자료구조에 익숙해 질 수 있는 기회여서 좋았다.

작성한 코드 중 중복되는 부분이 많아서, 어떠한 Flag를 이용하거나, 함수로 작은 단위로 분리하거나, 더 플레이하기 편한 구현을 해 보고 싶다는 생각도 들었다.