10주차 예비보고서

전공: 아트&테크놀로지학과 학년: 4학년 학번: 20191048 이름: 김도솔

**1.**

1-1)

추천 시스템은 tree 구조를 사용해 만든다. tree의 각 노드는 블록의 ID, 회전 수, 위치, 필드 상태, 누적 점수 등으로 구성된다. tree의 level 1은 현재 블록의 정보를 다루고 tree의 level 2에서는 다음 블록의 정보를 다루며 tree의 level 3에서는 그 다음 블록의 정보를 다루는 식으로 구현된다. 좀 더 자세히 설명하자면 tree의 level 1에서는 현재 블록이 필드에 놓여질 수 있는 모든 경우의 수를 고려하고, level 2에서는 level 1의 각 필드 상태 + 다음 블록이 필드에 놓여질 수 있는 모든 경우의 수를 고려해 자식 노드들을 만든다. 마찬가지로 level 3에서는 level 2의 각 필드 상태 + 다음 블록이 필드에 놓여질 수 있는 모든 경우의 수를 고려해 자식 노드들을 만든다. 이렇게 모든 블록과 그 회전 수를 고려해 만들어진 트리를 탐색하며 누적 점수가 가장 높은 leaf node를 추천할 블록의 위치로 결정한다.

1-2)

장점: tree는 계층이 존재하는 자료구조이기 때문에 블록 순서에 따라 가능한 미래(필드 상태)를 확장시키고, 그것을 바탕으로 모든 미래를 고려해 추천하는 시스템과 어울린다. 따라서 다른 자료구조들에 비해 추천 데이터를 비교적 잘 정돈된 상태로 다룰 수 있다는 점에서 효율적이다. 또한 현재와 확실한 미래의 정보를 동시에 고려해 완성시킨 tree는 가능한 범위 내에서 모든 경우의 수를 탐색할 수 있게 해준다. 따라서 주어진 정보 내에서 최선의 선택을 할 수 있게 도와준다.

단점: 최악의 경우 각 노드에 대해 생성되는 child 노드의 수는 34가지이다. 이때 고려할 블록의 수가 1개 증가할 때마다 tree의 총 노드 수는 34배씩 증가하게 된다. 따라서 n개의 블록을 고려할 경우 트리의 총 노드 수는 개로 고려되는 블록의 수가 증가함에 따라 노드의 개수는 기하급수적으로 증가한다. 이는 더 높은 점수를 얻을 수 있는 블록 위치를 추천하기 위해 더 많은 블록 수를 고려하게 되면 프로그램 수행 시간이 기하급수적으로 증가한다는 것을 의미한다. 마찬가지로 더 많은 블록 수를 고려할수록 메모리 공간의 사용이 기하급수적으로 증가한다. 따라서 고려할 블록 수가 늘어날수록 시간과 공간 두 측면에서 모두 비효율성을 보인다.

**2.**

2-1) Pruning tree

시간의 비효율성을 개선하기 위해 모든 경우의 수를 고려하지 않고 일정한 기준에 따라 tree의 branch를 쳐내 최선의 경우의 수만 고려하는 방법이다. 따라서 가장 좋은 추천 위치를 찾기 위해 소모하는 시간을 줄일 수 있다. 예를 들어 특정 level에서 낮은 누적 점수를 지니는 branch를 자르는 방법이 있다. 이때 주의해야할 점은 가장 높은 점수를 지니는 노드를 포함한 branch를 자르는 경우가 발생하지 않도록 branch를 잘라야 한다. 즉, 해당 branch가 좋지 않은 성능을 가질 것이 명확한 경우에만 pruning을 진행해야 한다.

2-2) Data Simplication

공간의 비효율성을 개선하기 위해 각 노드에 저장된 정보를 단순화하는 방법이다. tree의 노드에 포함된 정보 중에서 가장 큰 공간을 차지하는 것은 필드의 정보를 저장하는 부분이다. 모든 필드의 정보를 배열로 저장한다는 점에서 메모리 소모적이다. 이러한 큰 데이터를 필드의 높이만 고려해 저장하는 등 더 작은 데이터로 줄이는 방법이 데이터 단순화이다. 이때 단순화 과정에서 중요한 정보를 손실할 수도 있기에 신중하게 단순화를 수행해야 한다.