10주차 결과보고서

전공: 컴퓨터공학과 학년: 4학년 학번: 20212022 이름: 이예준

**1.**

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

Tree 구조를 만들기 위한 Node 구조체를 만들었다. level 변수는 해당 Node의 Depth,

accumulatedScore 변수는 해당 Node의 누적 점수 그리고 recBlockY는 해당 Node의 높이를

저장한다. recfield 2차원 배열은 Block을 놓았을 때의 field 정보를 저장하고, c 포인터 배열은

동적 할당을 하면서 해당 Node의 child Node를 가리킨다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

처음에 recommend 함수가 실행이 되면 RecNode의 child Node 포인터 배열을 동적으로

할당한다. 그 다음 2중 for문을 통해 가능한 모든 블럭의 회전 및 위치에 대해 탐색한다.

ChecktoMove함수를 이용해서 현재 Block이 해당 회전 및 위치로 이동할 수 있는지 확인한다.

블록이 이동 가능한 경우 RecNode를 생성한 뒤 현재 상태들(level, 높이, field 등)을 Node에

저장하고, Block을 가능한 최대까지 아래로 이동시킨다. 그 상태에서 AddBlockToField 함수로

Block을 field에 추가하고, DeleteLine 함수로 제거 가능한 줄을 제거하면서 점수를 갱신한다.

만약 Child Node의 level이 VISIBLE\_BLOCKS보다 작은 경우 recommend 함수를 재귀적으로

호출한다. 재귀 호출을 통해 Child Node의 최대 점수를 구한다. 최대 점수를 가진 Child Node를

찾으면 max값을 갱신하고, root의 child Node의 경우에는 해당 Block의 회전 및 위치를 추가로

저장한다. 사용한 Child Node는 메모리를 해제하여 leak를 방지한다.

시간 복잡도 : 2중 for문은 4가지의 회전수, field의 너비가 각각 for문을 맡고 있고,

재귀호출은 VISIBLE\_BLOCKS에 따라 달라지므로 시간 복잡도는 O(nVISIBLE\_BLOCKS)이다.

공간 복잡도 : 최악의 경우 Tree의 모든 노드의 2차원 배열 field정보를 동시에 저장해야 하므로

공간 복잡도는 O(n2\* nVISIBLE\_BLOCKS)= O(nVISIBLE\_BLOCKS+2)이다.

**2.**

동적 프로그래밍을 활용하여 재귀적 호출을 통해 이전에 계산된 점수를 재사용하면서 중복계산을

줄였고, 필요 없는 Node는 탐색 후 즉시 메모리를 해제하여 모든 Node를 Tree에 저장하지

않음으로써 메모리 사용량을 줄였다. 한계점으로는 여전히 많은 상태를 탐색해야 하므로,

근본적인 시간 복잡도 문제를 해결하지 못한다. 또한 메모리 해제를 정확하게 관리해야 하기

때문에 코드가 복잡해질 수 있으며, Depth가 깊어질수록 현실적으로 모든 상태를 탐색하기

어려워지기 때문에 제한된 실제 게임에서 제한된 Depth로 최적의 결정을 내리지 못할 수 있다.

**3.**

이번 실험과 숙제를 통해 여러 가지 프로그래밍 기술과 알고리즘 최적화에 대한 이해를 깊게 할

수 있었다. 1주차에는 테트리스 게임의 기본적인 구조을 구현하면서 게임의 메커니즘을 이해하게

되었다. 블록의 이동, 회전, 충돌 감지, 라인 삭제 등의 기능을 직접 구현해보면서 게임에 대한

기본 원리를 체득할 수 있었다. 2주차에는 Linked List를 이용하여 유연한 메모리 관리와 삽입 및

삭제의 용이성을 체감했다. 또한, 포인터를 활용한 동적 자료구조 구현 방법과 메모리 관리의

중요성을 배웠다. 마지막 3주차에는 모든 가능한 상태를 탐색하여 최적의 결과를 찾아내는

재귀 알고리즘을 구현함으로써, 탐색 알고리즘의 효율성과 한계를 직접 경험할 수 있었다.

이번 실습은 단순한 게임 구현을 넘어, 복잡한 문제를 단계별로 해결해 나가는 과정을 통해

프로그래밍 실력을 한층 더 향상시키는 계기가 되었고, 앞으로도 이러한 경험을 바탕으로

더 나은 알고리즘과 최적화 기법을 연구하고 적용할 수 있는 능력을 길러 나가고자 한다.