Báo cáo về thời gian chạy và không gian bộ nhớ của thuật toán Alpha-Beta và MiniMax

Nhóm: Đoàn Quốc Đăng và Nguyễn Phương Nhã

Đề tài: Game cờ đam (Checkers)

I. Sơ lược về báo cáo:

- Kết quả thí nghiệm độ phức tạp về thời gian và không gian bộ nhớ được thực hiện ở cả thuật toán MiniMax và Alpha-Beta ở độ sâu (depth) từ 1 đến 7.
- Cấu hình thử nghiệm:
  - Processor: Intel® Pentium® Gold G6405 2 Cores 4 Threads 4.1Ghz.
  - RAM: 16GB DDR4.
  - OS: Windows 11 Pro version 23H2.
  - IDE: IntelliJ IDEA 2023.2.5.
- Đoạn code thực hiện việc đo đạc thời gian chạy và tiêu thụ bộ nhớ:

```
long begin = System.currentTimeMillis();
long memoryBeforeExecute = Runtime.getRuntime().totalMemory() - Runtime.getRuntime().freeMemory();

CheckerBoard bestMove = ai.getBestMove(checkerBoardController.getCheckerBoard(), gameDifficult);
checkerBoardController.setCheckerBoard(bestMove);

long memoryAfterExecute = Runtime.getRuntime().totalMemory() - Runtime.getRuntime().freeMemory();
long end = System.currentTimeMillis();

System.out.println("Depth: " + gameDifficult.getDepth());
System.out.println("Memory used: " + (memoryAfterExecute - memoryBeforeExecute) + "Bytes");
System.out.println("Computed time: " + (end - begin) + "ms");

isBlackTurn = false;
```

- Đoạn code nằm ở phương thức **startGame()** trong lớp **GameController** ở đường dẫn: **com/dangnha/checkers/controller/GameController.java.**

- Trong đó phương thức **ai.getBestMove(CheckerBoard board, GameDiffcult gameDifficult)** gọi lại thuật toán Alpha-Beta để tìm ra bàn cờ tối ưu nhất.

- Do game sẽ tự bắt đầu ở mức **GameDiffcult** là ***EASY*** nên chỉ cần thay đổi giá trị độ sâu của **GameDifficult.*EASY*** để test:

```
ConMuaXaDan
public enum GameDifficult {
    1 usage
    HARD( depth: 7),
    1 usage
    MEDIUM( depth: 5),
    3 usages
    EASY( depth: 3);

    2 usages
    private int depth;
    6 usages  ConMuaXaDan
    private GameDifficult(int depth) { this.depth = depth; }

    1 usage  ConMuaXaDan
    public int getDepth() { return depth; }
}
```

- Thuật toán Alpha-Beta: gồm 2 phương thức **minValue()** và **maxValue()**

```
1 usage  ConMuaXaDan +1
public int minValue(CheckerBoard board, int alpha, int beta, int depth) {
    if (depth == 0) {
        return board.heuristic();
    }

    CheckerBoard cloneBoard = new CheckerBoard(board.getBoardStates(), board.getCheckerList());

    int value = Integer.MAX_VALUE;
    for (CheckerBoard neighbour : cloneBoard.generateNeighbours( isBlackTurn: false)) {
        value = Math.min(value, maxValue(neighbour, alpha, beta,  depth: depth - 1));
        if (alpha >= value) return value;
        beta = Math.min(beta, value);
    }
      System.out.println("min board: \n" + board + ", heuristic: " + board.heuristic() + ", depth: " +
    return value;
}
```

```
3 usages  ConMuaXaDan +1
public int maxValue(CheckerBoard board, int alpha, int beta, int depth) {
    if (depth == 0)
        return board.heuristic();

    CheckerBoard cloneBoard = new CheckerBoard(board.getBoardStates(), board.getCheckerList());

    int value = Integer.MIN_VALUE;
    for (CheckerBoard neighbour : cloneBoard.generateNeighbours( isBlackTurn: true)) {
        value = Math.max(value, minValue(neighbour, alpha, beta,  depth: depth - 1));
        if (value >= beta) return value;
        alpha = Math.max(alpha, value);
    }
//      System.out.println("Max board: \n" + board + ", heuristic: " + board.heuristic() + ", depth: "
    return value;
}
```

- Thuật toán MiniMax sẽ sử dụng 2 phương thức trên nhưng sẽ comment lại dòng if ở 2 phương thức **minValue()** và **maxValue()**

II. Kết quả thực nghiệm:

Độ sâu (Depth)	*MiniMax*		*Alpha-Beta*	
1	**Memory**: 1.980.200 bytes ~ 1.934 KB **Executed time**: 32ms	Dec 02, 2023 11:02:02 AM WARNING: Loading FXML docu Depth: 1 Memory used: 1980200Bytes Computed time: 32ms	**Memory**: 2.061.280 bytes ~ 2.013 KB **Executed time**: 28ms	Dec 02, 2023 10:47:01 AM WARNING: Loading FXML do Depth: 1 Memory used: 2061280Byte Computed time: 28ms
2	**Memory**: 7.156.248 bytes ~ 6.989 KB **Executed time**: 50ms	Dec 02, 2023 11:02:32 AM WARNING: Loading FXML docu Depth: 2 Memory used: 7156248Bytes Computed time: 50ms	**Memory**: 6.319.264 bytes ~ 6.171 KB **Executed time**: 48ms	Dec 02, 2023 10:49:49 AM j WARNING: Loading FXML docu Depth: 2 Memory used: 6319264Bytes Computed time: 48ms
3	**Memory**: 49.934.320 bytes ~ 48.764 KB **Executed time**: 118ms	Dec 02, 2023 11:03:06 AM ja WARNING: Loading FXML docum Depth: 3 Memory used: 49934320Bytes Computed time: 118ms	**Memory**: 23.457.448 bytes ~ 22.908 KB **Executed time**: 85ms	Dec 02, 2023 10:50:23 AM ja WARNING: Loading FXML docum Depth: 3 Memory used: 23457448Bytes Computed time: 85ms
4	**Memory**: 86.615.792 bytes ~ 84.586 KB **Executed time**: 444ms	Dec 02, 2023 11:03:39 AM ja WARNING: Loading FXML docur Depth: 4 Memory used: 86615792Bytes Computed time: 444ms	**Memory**: 69.914.544 bytes ~ 68.276 KB **Executed time**: 119ms	Dec 02, 2023 10:50:53 AM ja WARNING: Loading FXML docum Depth: 4 Memory used: 69914544Bytes Computed time: 119ms
5	**Memory**: 315.174.896 bytes ~ 307.788 KB **Executed time**: 1.500ms	Dec 02, 2023 11:04:08 AM jav WARNING: Loading FXML docume Depth: 5 Memory used: 315174896Bytes Computed time: 1500ms	**Memory**: 59.487.088 bytes ~ 58.093 KB **Executed time**: 219ms	Dec 02, 2023 10:51:22 AM ja WARNING: Loading FXML docum Depth: 5 Memory used: 59487088Bytes Computed time: 219ms

6	**Memory**: 59.059.152 bytes ~ 57.675 KB **Executed time**: 8.805ms	Dec 02, 2023 11:04:42 AM j WARNING: Loading FXML docu Depth: 6 Memory used: 59059152Bytes Computed time: 8805ms	**Memory**: 112.549.696 bytes ~ 109.912 KB **Executed time**: 407ms	Dec 02, 2023 10:51:55 AM jav WARNING: Loading FXML docume Depth: 6 Memory used: 112549696Bytes Computed time: 407ms
7	**Memory**: 417.179.936 bytes ~ 407.402 KB **Executed time**: 72.713ms	Dec 02, 2023 11:05:37 AM ja WARNING: Loading FXML docum Depth: 7 Memory used: 417179936Bytes Computed time: 72713ms	**Memory**: 109.051.824 bytes ~ 106.496KB **Executed time**: 831ms	Dec 02, 2023 10:52:25 AM java WARNING: Loading FXML documen Depth: 7 Memory used: 109051824Bytes Computed time: 831ms