Laporan Tugas Kecil 1 IF2211 Strategi Algortima

SEMESTER II 2024/2025

Penyelesaian IQ Puzzle Pro dengan Algoritma Brute Force

Oleh

Faqih Muhammad Syuhada

NIM : 13523057

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

Februari 2025

DAFTAR ISI

BAB I

DESKRIPSI TUGAS

I.1 IQ Puzzler Pro

IQ Puzzler Pro adalah permainan papan yang diproduksi oleh perusahaan Smarts Games. Tujuan dari permainan ini adalah pemain harus dapat mengisi seluruh papan dengan piece atau blok puzzle yang telah disediakan.

Komponen penting dari permainan IQ Puzzler Pro ini terdiri dari :

1. Board (Papan Permainan) - Board merupakan komponen utama yang menjadi tempat dan tujuan permainan dimana pemain harus bisa mengisi seluruh area papan menggunakan blok atau piece yang telah disediakan.
2. Piece/Blok - Blok merupakan komponen yang digunakan pemain untuk mengisi papan kosong hingga terisi penuh. Setiap blok memiliki bentuk yang unik dan semua blok harus digunakan untuk menyelesaikan puzzle permainan ini.

Permainan dimulai dengan papan yang kosong. Pemain dapat meletakkan blok puzzle sedemikian sehingga tidak ada blok yang tumpang tindih kecuali kasus 3D. Setiap blok puzzle dapat di rotasi maupun di cerimankan. Puzzle dinyatakan selesai jika papan terisi penuh dan seluruh blok puzzle berhasil di letakkan di papan tanpa terkecuali.

Tugas dari program ini adalah menemukan cukup satu solusi dari permainan IQ Puzzler Pro dengan menggunakan Algoritma Brute Force, atau menampilkan bahwa solusi tidak ditemukan jika tidak ada solusi yang mungkin dari puzzle.

I.2 Algoritma Brute Force

Algoritma Brute Force adalah algoritma penyelesaian suatu masalah berbasis komputasi dengan pendekatan yang lempang atau *straightforward* dan sederhana.

Algoritma Brute Force adalah algoritma natural yang biasanya pertama kali terpikirkan oleh problem solver karena implementasinya yang mudah. Algoritma ini bekerja dengan cara mencoba seluruh kemungkinan jalan penyelesaian yang ada tanpa pikir panjang.

Hampir semua persoalan dapat diselesaikan dengan algoritma ini, namun kurang efisien. dan umumnya lambat untuk masukan yang besar

BAB II

STRATEGI ALGORITMA

II.1 Rancangan Algoritma

Dengan algoritma Brute Force. Program ini mencoba semua kemungkinan hingga menemukan satu solusi yang valid. Berikut alur penyelesaiannya :

1. Program mulai dengan meminta pengguna untuk memasukkan nama file tersebut. File tersebut berisikan :
    a. Ukuran Papan (N x M)
    b. Jumlah potongan puzzle (P)
    c. Bentuk potongan huruf yang akan diisi ke dalam papan.
    d. Program membaca dari file yang diberikan dan mengekstrak dimensi papan beserta jumlah potongan puzzle.
    e. Kemudian, program membaca potongan puzzle yang ada dan menyimpannya dalam bentuk lsit (List<String>).
    f. Objek Board : Papan berukuran N x M yang diinisialisasi. Setiap sel pada papan diisi dengan karakter ‘0’ dengan anggapan ruang kosong (untuk kasus default).
    g. Objek PuzzleSolver : Menerima jumlah potongan puzzle dan objek Board. Dalam konstruktor, sebuah list puzzles diinisialisasi untuk menyimpan semua bentuk potongan yang dibaca dari file.
    h. Metode processPuzzle di PuzzleSolver digunakan untuk membaca potongan-potongan puzzle dan menyimpannya ke dalam list puzzles. Setiap potongan disimpan dengan karakter pembeda yaitu AlphabetID(AID) dan bentuknya (shape) dalam bentuk array 2D.
2. Algoritma bruteforcenya mulai di solvePuzzle:
    a. Basis Rekursi :

i. Jika indeks y sudah melampui kolom papan. Jika ya, lakukan transisi ke baris berikutnya.
ii. jika indeks X susah melampui baris papan. Pastikan potongan telah digunakan atau validasi solusi.

b. Pencarian Indeks Kosong

Jika sel papan pada posisi (x,y) tidak kosong, lanjutkan ke sel berikutnya (x,y+1).

c. Mencoba Semua Potongan Puzzle, jika potongan tidak terpakai, cari semua variasi dari potongan tersebut dengan memanggil searchVariations.
d. Untuk Setiap variasi yang ditemukan, diperiksa apakah potongan tersebut bisa dipasang pada posisi (x,y) menggunakan checkPuzzle.
    i. Jika potongan dapat dipasang :
        1. Potongan dipasang menggunakan placePuzzle.
        2. Ditandai sudah digunakan dengan setUsed(true).
        3. Rekursi keposisi berikutnya dengan solvePuzzle(x,y+1).
    ii. Jika tidak ditemukan solusi alias potongan tidak dapat dipasang
        1. Lepaskan potongan dengan removePuzzle dan ditandai sebagai tidak terpakai kembali.

3. Di dalam kelas Piece, status potongan dikelola menggunakan variable boolean used :
    a. Setiap potongan memiliki status apakah telah digunakan atau belum.
    b. Ini sangat penting untuk menghindari penggunaan potongan yang sama lebih dari sekali didalam satu buah solusi. Status ini diperiksa diawal setiap iterasi potongan dalam solvePuzzle.
4. Metode searchVariations digunakan untuk mendapatkan dan menciptakan semua variasi dari potongan dengan menggunakan rotasi 90 derajat hingga 360 derajat dan cerminan horizontal untuk menghasilkan semua variasi.

5. Setelah semua posisi papan diperiksa program melakukan validasi dengan metode isValidSolution yang memastikan setiap potongan digunakan hanya satu kali.
6. Jika ditemukan solusi maka akan ditanyakan pertanyaan untuk menjalankan saveSolution di kelas board.

BAB III

IMPLEMENTASI DALAM BAHASA JAVA

III.1 Source Code

III.1.1 Main.java

```java
import object.*;
import java.util.*;
import java.io.*;

public class Main {
    Run | Debug
    public static void main(String[] args) {
        System.out.print(s:"Masukkan nama file test case (.txt): ");
        Scanner ScanInput= new Scanner(System.in);
        String filename= ScanInput.nextLine();
        int N=0,M=0,P=0;

        try (Scanner scanFile= new Scanner(new File("../test/"+filename))) {
            String firstLine= scanFile.nextLine();
            String[] tcLine= firstLine.split(regex:" ");
            if (tcLine.length< 3) {
                throw new IllegalArgumentException(s:"Baris pertama harus memiliki 3 angka.");
            }
            try {
                N= Integer.parseInt(tcLine[0]);
                M= Integer.parseInt(tcLine[1]);
                P= Integer.parseInt(tcLine[2]);
            }catch(NumberFormatException e) {
                throw new IllegalArgumentException(s:"Baris pertama harus berisi angka integer yang valid.");
            }
            String boardShape= scanFile.nextLine().trim();
            Board board= new Board(N,M);
            PuzzleSolver puzzleSolver= new PuzzleSolver(P,board);
            String line;
            List<String> curLine= new ArrayList<>();
            char curAid= ' ';
            while (scanFile.hasNextLine()) {
                line= scanFile.nextLine();
                char firstChar= line.charAt(index:0);
                if (Character.isUpperCase(firstChar)) {
                    if (firstChar != curAid && curAid != ' ') {
                        puzzleSolver.processPuzzle(curAid,curLine);
                        curLine.clear();
                    }
                    curAid= firstChar;
                }
                curLine.add(line);
            }
            // puzzleSolver.printPuzzles();
            if (!curLine.isEmpty()) {
                puzzleSolver.processPuzzle(curAid,curLine);
            }
            // System.err.println(puzzleSolver.getLength());
```

```
            long startTime= System.currentTimeMillis();
            if (puzzleSolver.solvePuzzle(x:0,y:0)&&puzzleSolver.getLength()==P) {
                board.print();
                long endTime= System.currentTimeMillis();
                System.out.println("Waktu pencarian: "+ (endTime - startTime)+ " ms");
                System.out.println("Banyak kasus yang ditinjau: "+ puzzleSolver.getSolutionCount());
                System.out.print(s:"Apakah anda ingin menyimpan solusi? (ya/tidak): ");
                String save= ScanInput.nextLine();
                if (save=="ya") {
                board.saveSolution(filename);
            }
            } else if (puzzleSolver.getLength()==P){
                System.out.println(x:"Jumlah piece tidak sesuai masukkan.");
            }
            else {
                System.out.println(x:"solusi tidak ditemukan.");
            }
            ScanInput.close();
        } catch (IOException read) {
            System.out.println("Kesalahan saat membaca file: "+ read.getMessage());
        }
    }
}
```

III.1.2 Piece.java

```
package object;
import java.util.*;
import java.io.*;

public class Piece {
    char[][] shape;
    char aid;
    boolean used;

    public Piece(char aid,char[][] shape) {
        this.shape= shape;
        this.aid= aid;
        this.used= false;
    }

    public char getaid() {
        return aid;
    }

    public char[][] getShape() {
        return shape;
    }

    public boolean isUsed() {
        return used;
    }

    public void setUsed(boolean used) {
        this.used= used;
    }
}
```

III.1.3 Board.java

```java
package object;
import java.util.*;
import java.io.*;

public class Board {
    int rows;
    int cols;
    private char[][] board;
    public Board(int N,int M) {
        this.rows= N;
        this.cols= M;
        this.board= new char[N][M];
        for (int i=0;i < N;i++) {
            for (int j=0;j < M;j++) {
                this.board[i][j]= '0';
            }
        }
    }

    public void setCustom(int row,char[] pattern) {
        this.board[row]= pattern;
    }

    public char[][] getBoard() {
        return board;
    }

    public char[][] getCustom() {
        return board;
    }

    public void placePuzzle(int x,int y,char aid,char[][] puzzleShape) {
        for (int i=0;i < puzzleShape.length;i++) {
            for (int j=0;j < puzzleShape[i].length;j++) {
                if (puzzleShape[i][j]== aid) {
                    board[x+ i][y+ j]= aid;
                }
            }
        }
    }

    public void removePuzzle(int x,int y,char[][] puzzleShape) {
        for (int i=0;i < puzzleShape.length;i++) {
            for (int j=0;j < puzzleShape[i].length;j++) {
                if (puzzleShape[i][j]!= '0' && puzzleShape[i][j]!='.') {
                    board[x+ i][y+ j]= '0';
                }
            }
        }
    }
}
```

III.1.4 puzzleSolver.java

```java
package object;
import java.util.*;
import java.io.*;

public class PuzzleSolver {
    private Board board;
    private static List<Piece> puzzles;
        private int count;
        public PuzzleSolver(int P,Board board) {
            this.board= board;
            this.puzzles= new ArrayList<>();
            this.count=0;
        }
        public int getLength() {
            return puzzles.size();
        }

        public void processPuzzle(char aid,List<String> lines) {
            int height= lines.size();
            int width=0;
            for (String line: lines) {
                width= Math.max(width,line.length());
            }

            char[][] shape= new char[height][width];
            for (int i=0;i< height;i++) {
                char[] chars= lines.get(i).toCharArray();
                for (int j=0;j < chars.length;j++) {
                    if (chars[j]== ' ') {
                        shape[i][j]= '.';
                    } else {
                        shape[i][j]= chars[j];
                    }
                }
                for (int j= chars.length;j < width;j++) {
                    shape[i][j]= '.';
                }
            }
            puzzles.add(new Piece(aid,shape));
        }

        public static void printPuzzles() {
            for (Piece puzzle : puzzles) {
                for (char[] row : puzzle.shape) {
                    System.out.println(row);
                }
                System.out.println();
            }
```

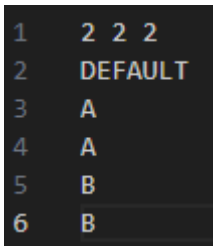
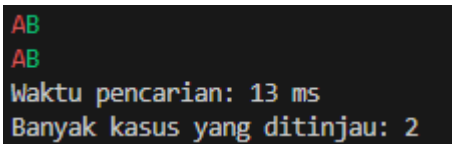
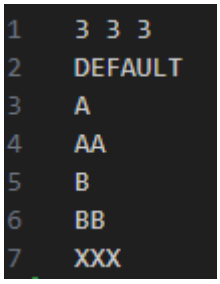
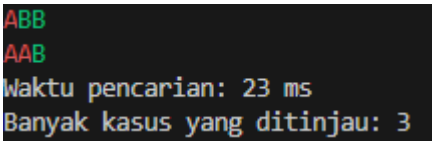
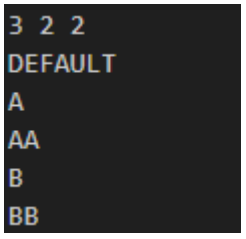
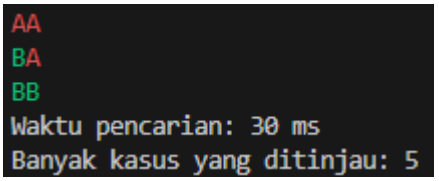
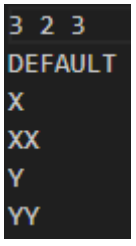
```java
public boolean solvePuzzle(int x,int y) {
    if (y== board.cols) {
        x++;
        y= 0;
    }
    if (x== board.rows) {
        return isValidSolution();
    }

    if (board.getBoard()[x][y] != '0') {
        return solvePuzzle(x,y+ 1);
    }

    for (Piece piece : puzzles) {
        if (piece.isUsed()) continue;
        List<char[][]> variations= searchVariations(piece.getShape());
        for (char[][] variation : variations) {
            if (checkPuzzle(x,y,variation,piece.getaid())) {
                board.placePuzzle(x,y,piece.getaid(),variation);
                piece.setUsed(used:true);
                count++;
                if (solvePuzzle(x,y+ 1)) {
                    return true;
                }

                board.removePuzzle(x,y,variation);
                piece.setUsed(used:false);
            }
        }
    }

    return false;
}

private boolean isValidSolution() {
    for (Piece Piece : puzzles) {
        if (!Piece.isUsed()) return false;
    }
    return true;
}
```

```java
private static List<char[][]> searchVariations(char[][] original) {
    List<char[][]> variations= new ArrayList<>();
    Set<String> newVariation= new HashSet<>();
    char[][] current= original;
    for (int i= 0;i < 4;i++) {
        addNew(current,newVariation,variations);
        addNew(reflect(current),newVariation,variations);
        current= rotate90(current);
    }
    // for (int index= 0;index < variations.size();index++) {
    //     char[][] grid= variations.get(index);
    //     for (char[] row : grid) {
    //         System.out.println(new String(row));
    //     }
    //     System.out.println();
    // }
    return variations;
}

private static void addNew(char[][] variation,Set<String> newVariation,List<char[][]> variations) {
    String key= Arrays.deepToString(variation);//note : pengganti stringBuilder
    if (newVariation.add(key)) {
        variations.add(variation);
    }
}

private static char[][] rotate90(char[][] piece) {
    int rows= piece.length;
    int cols= piece[0].length;
    char[][] rotated= new char[cols][rows];
    for (int r= 0;r < rows;r++) {
        for (int c= 0;c < cols;c++) {
            rotated[c][rows - 1 - r]= piece[r][c];
        }
    }
    return rotated;
}

private static char[][] reflect(char[][] piece) {
    int rows= piece.length;
    int cols= piece[0].length;
    char[][] reflected= new char[rows][cols];
    for (int r= 0;r < rows;r++) {
        for (int c= 0;c < cols;c++) {
            reflected[r][cols - 1 - c]= piece[r][c];
        }
    }
    return reflected;
}
```

```
    private boolean checkPuzzle(int x,int y,char[][] Piece,char aid) {
        int cols= Piece[0].length;
        int rows= Piece.length;
        for (int i= 0;i < rows;i++) {
            for (int j= 0;j < cols;j++) {
                if (Piece[i][j]== aid) {
                    int newY= y+ j;
                    int newX= x+ i;
                    if (newX >= board.rows || newY >= board.cols || board.getCustom()[newX][newY] != '0' || board.getBoard()[newX][newY] != '0') {
                        return false;
                    }
                }
            }
        }
        return true;
    }

    public int getSolutionCount() {
        return count;
    }
}
```

BAB IV

TESTING

NO	Test Case		Penjelasan
	Input	Output	
1.			Kasus 2x2 Sederhana
2.			Kasus 3x3 valid
3.			Kasus 3x2 valid
4.			Kasus 3x2 banyak piece tidak sesuai

5.	3 3 3 DEFAULT A AAA B BB XXX	Jumlah piece tidak sesuai masukkan.	Kasus 3x3 bentuk piece tidak sesuai
6.	4 4 4 DEFAULT A AA B BB XXXX XXXX YY	ABBY AABY XXXX XXXX Waktu pencarian: 28 ms Banyak kasus yang ditinjau: 4	Kasus 4x4 piece besar
7.	5 4 5 DEFAULT A AA B BB XXXXX YYYYY ZZ ZZ	AAXY BAXY BBXY ZZXY ZZXY Waktu pencarian: 28 ms Banyak kasus yang ditinjau: 52	Kasus 5x4 banyak piece

8.	5 5 7 DEFAULT A AA B BB C CC D DD EE EE E FF FF F GGG	ABBCC AABDC EEEDD EEFFF GGGFF Waktu pencarian: 32 ms Banyak kasus yang ditinjau: 785	Kasus 5x5 dari spesifikasi

BAB V

LAMPIRAN

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa kesalahan	✓	
2	Program berhasil dijalankan	✓	
3	Solusi yang diberikan program benar dan mematuhi aturan permainan	✓	
4	Program dapat membaca masukan berkas .txt serta menyimpan solusi dalam berkas .txt	✓	
5	Program memiliki *Graphical User Interface* (GUI)		✓
6	Program dapat menyimpan solusi dalam bentuk file gambar		✓
7	Program dapat menyelesaikan kasus konfigurasi *custom*		✓
8	Program dapat menyelesaikan kasus konfigurasi Piramida (3D)		✓
9	Program dibuat oleh saya sendiri	✓	

DAFTAR REFERENSI

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/stima24-25.htm

https://www.youtube.com/watch?v=opr5q5tZxLM

https://breder.org/sudoku-solver