LAPORAN TUGAS KECIL

IF2211 STRATEGI ALGORITMA

PENYELESAIAN PERSOALAN 15-PUZZLE DENGAN ALGORITMA BRANCH AND BOUND

Disusun oleh:

Claudia 13520076

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

CARA KERJA PROGRAM BRANCH AND BOUND

Untuk program ini, blok kosong (blok yang dapat dipindahkan) ditandai dengan blok bernomor 16. Cara kerja program 15 puzzle solver dengan menggunakan algoritma branch and bound adalah sebagai berikut:

1. Pertama, tentukan dulu apakah susunan puzzle bisa mencapai final state atau tidak. Jika iya, program akan dilanjutkan.
2. Cek apakah state susunan puzzle merupakan final state, jika iya maka program akan berhenti.
3. Jika bukan final state, maka blok kosong akan dipindahkan ke atas, bawah, kiri, dan kanan jika bisa dipindahkan (tidak out of bound).
4. Setelah itu, hitung cost untuk masing-masing state ketika blok kosong dipindahkan ke atas, bawah, kiri, dan kanan. Masukkan cost dan state ke dalam queue, queue akan disort berdasarkan cost terkecil. Jika ada state yang pernah digenerate sebelumnya, tidak akan dimasukkan ke dalam queue.
5. Ambil state dengan cost terkecil, setelah itu lakukan langkah 2-5 secara berulang.

SCREENSHOT INPUT-OUTPUT PROGRAM

I/O jika menerima input file dan puzzle tidak solvable

Jalankan python main.py pada terminal (input)

```
PS D:\Github\bnb-15-puzzle\src> python main.py

========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
```

Pilih Y (mau menggunakan file eksternal) dan masukkan nama file beserta .txt

```
*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): puzzle.txt
```

Output program

```
========PUZZLE's INITIAL STATE==========
1 3 4 15
2 16 5 12
7 6 11 14
8 9 10 13

KURANG (1) = 0
KURANG (2) = 0
KURANG (3) = 1
KURANG (4) = 1
KURANG (5) = 0
KURANG (6) = 0
KURANG (7) = 1
KURANG (8) = 0
KURANG (9) = 0
KURANG (10) = 0
KURANG (11) = 3
KURANG (12) = 6
KURANG (13) = 0
KURANG (14) = 4
KURANG (15) = 11
KURANG (16) = 10
X = 0
SIGMA of KURANG(i) + X =  37

Loading...
```

I/O jika menerima input file dan puzzle solvable

Jalankan python main.py pada terminal (input)

```
PS D:\Github\bnb-15-puzzle\src> python main.py

========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
```

Pilih Y (mau menggunakan file eksternal) dan masukkan nama file beserta .txt

```
PS D:\Github\bnb-15-puzzle\src> python main.py

========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): puzzle2.txt
```

Output program

```
========PUZZLE's INITIAL STATE==========
1 2 3 4
5 6 16 8
9 10 7 11
13 14 15 12

KURANG (1) = 0
KURANG (2) = 0
KURANG (3) = 0
KURANG (4) = 0
KURANG (5) = 0
KURANG (6) = 0
KURANG (7) = 0
KURANG (8) = 1
KURANG (9) = 1
KURANG (10) = 1
KURANG (11) = 0
KURANG (12) = 0
KURANG (13) = 1
KURANG (14) = 1
KURANG (15) = 1
KURANG (16) = 9
X = 1
SIGMA of KURANG(i) + X =  16

Loading...
```

```
down
1 2 3 4
5 6 7 8
9 10 16 11
13 14 15 12

right
1 2 3 4
5 6 7 8
9 10 11 16
13 14 15 12

down
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

PUZZLE IS SOLVED!
TOTAL STEPS :  3
TOTAL NODE GENERATED:  9
TIME TAKEN:  0.00099945068359375 s
```

I/O jika tidak menerima input file (puzzle dirandom)

Jalankan python main.py pada terminal (input)

```
PS D:\Github\bnb-15-puzzle\src> python main.py

========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
```

Pilih N (tidak mau menggunakan file eksternal)

```
*** DO YOU WANT TO USE INPUT FILE? (Y/N)
N
```

Output program

```
========PUZZLE's INITIAL STATE==========
14 16 9 4
12 7 13 8
10 3 1 5
2 11 6 15

KURANG (1) = 0
KURANG (2) = 0
KURANG (3) = 2
KURANG (4) = 3
KURANG (5) = 1
KURANG (6) = 0
KURANG (7) = 5
KURANG (8) = 5
KURANG (9) = 8
KURANG (10) = 5
KURANG (11) = 1
KURANG (12) = 9
KURANG (13) = 8
KURANG (14) = 13
KURANG (15) = 0
KURANG (16) = 14
X = 1
SIGMA of KURANG(i) + X =  75

Loading...

PUZZLE IS NOT SOLVABLE :(
```

CHECKLIST

Poin	Ya	Tidak
Program berhasil dikompilasi	Ya	
Program berhasil running	Ya	
Program dapat menerima input dan menuliskan output	Ya	
Luaran sudah benar untuk semua data uji	Ya	
Bonus dibuat		Tidak

KODE PROGRAM

Puzzle.py: menyimpan class Puzzle

```
import random
import copy


class Puzzle:
    node = 0


    # cctor
    def __init__(self) -> None:
        self.puzzle = [0 for i in range(16)]
        self.queue = []
        self.generated = []
        self.path = []
        self.depth = 0


    # to randomize puzzle
    def random(self):
        number = list(range(1, 17))
        random.shuffle(number)
        for i in range (16):
            self.puzzle[i] = number[i]


    # to read puzzle from a txt file
    def readFile(self, fileName):
        f = open(fileName , 'r')
        stringPuzzle = f.read().replace('\n', ' ').split(" ")
        f.close()
        for i in range (16):
            self.puzzle[i] = int(stringPuzzle[i])


    # to print puzzle
    def print(self):
        for i in range(16):
            print(self.puzzle[i], end=" ")
            if (i % 4 == 3):
                print()


    def printup(self):
        idx = self.find16()
        if (idx > 3):
```

```
            self.puzzle[idx] = self.puzzle[idx - 4]
            self.puzzle[idx - 4] = 16
        print('up')
        self.print()
        print()


    def printdown(self):
        idx = self.find16()
        if (idx < 12):
            self.puzzle[idx] = self.puzzle[idx + 4]
            self.puzzle[idx + 4] = 16
        print('down')
        self.print()
        print()


    def printleft(self):
        idx = self.find16()
        if (idx % 4 != 0):
            self.puzzle[idx] = self.puzzle[idx - 1]
            self.puzzle[idx - 1] = 16
        print('left')
        self.print()
        print()


    def printright(self):
        idx = self.find16()
        if ( idx % 4 != 3):
            self.puzzle[idx] = self.puzzle[idx + 1]
            self.puzzle[idx + 1] = 16
        print('right')
        self.print()
        print()

    # to count the number of misplaced puzzle
    def misplaced(self, puz):
        count = 0
        for i in range(16):
            if (puz[i] != (i + 1)):
                if (puz[i] != 16):
                    count += 1
        return count



    # to find KURANG (i)
```

```
    def kurang(self, i):
        count = 0
        for j in range (i + 1, 16):
            if (self.puzzle[j] < self.puzzle[i]):
                if (self.puzzle[j] != 0):
                    count += 1
        return count


    # to check if a puzzle is solvable or not
    def isSolvable(self):
        total = 0
        kurang = [0 for i in range (16)]
        for i in range (16):
            kurang[self.puzzle[i] - 1] = self.kurang(i)
            total += self.kurang(i)


        for i in range (16):
            print("KURANG (" + str(i + 1) + ") = "+ str(kurang[i]))

        idx = [1, 3, 4, 6, 9, 11, 12, 14]
        x = False
        for i in idx:
            if (self.puzzle[i] == 16):
                x = True
                total += 1


        if (x):
            print("X = 1")
        else:
            print("X = 0")


        print("SIGMA of KURANG(i) + X = ", total)
        print()
        print("Loading...")
        print()

        if (total % 2 == 0):
            return True
        else:
            return False


    # to check if it is already the final state
    def isSolution(self):
```

```
        solution = True
        for i in range (16):
            if (self.puzzle[i] != (i + 1)):
                solution = False


        return solution


    # return the index of 16 (blank space)
    def find16(self):
        i = 0
        while (self.puzzle[i] != 16):
            i += 1


        return i


    # check is it possible to move up
    def isUp(self):
        idx = self.find16()
        if ( idx > 3):
            return True
        else:
            return False

    # move up
    def up(self):
        upPuzzle = copy.deepcopy(self.puzzle)
        idx = self.find16()
        if (idx > 3):
            upPuzzle[idx] = upPuzzle[idx - 4]
            upPuzzle[idx - 4] = 16
        return upPuzzle


    # check is it possible to move down
    def isDown(self):
        idx = self.find16()
        if ( idx < 12):
            return True
        else:
            return False


    # move down
    def down(self):
```

```
        downPuzzle = copy.deepcopy(self.puzzle)
        idx = self.find16()
        if (idx < 12):
            downPuzzle[idx] = downPuzzle[idx + 4]
            downPuzzle[idx + 4] = 16
        return downPuzzle


    # check is it possible to move left
    def isLeft(self):
        idx = self.find16()
        if (idx % 4 != 0):
            return True
        else:
            return False


    # move left
    def left(self):
        leftPuzzle = copy.deepcopy(self.puzzle)
        idx = self.find16()
        if (idx % 4 != 0):
            leftPuzzle[idx] = leftPuzzle[idx - 1]
            leftPuzzle[idx - 1] = 16
        return leftPuzzle

    # check is it possible to move right
    def isRight(self):
        idx = self.find16()
        if (idx % 4 != 3):
            return True
        else:
            return False


    # move right
    def right(self):
        rightPuzzle = copy.deepcopy(self.puzzle)
        idx = self.find16()
        if ( idx % 4 != 3):
            rightPuzzle[idx] = rightPuzzle[idx + 1]
            rightPuzzle[idx + 1] = 16
        return rightPuzzle


    # to check if current puzzle has been generated before
    def isGenerated(self, puzzle):
        if puzzle in self.generated:
```

```
            return True
        else:
            return False

    # branch and bound
    def solvePuzzle(self):

        self.depth += 1


        # check if it is possible to move up
        if (self.isUp()):
            upPuzzle = self.up()
            # if the state has not been generated before
            if (not self.isGenerated(upPuzzle)):
                path = copy.deepcopy(self.path)
                path.append('up')
                self.generated.append(upPuzzle)
                self.queue.insert(0,[self.misplaced(upPuzzle) + self.depth,
upPuzzle, path, self.depth])
                Puzzle.node += 1


        # check if it is possible to move right
        if (self.isRight()):
            rightPuzzle = self.right()
            # if the state has not been generated before
            if (not self.isGenerated(rightPuzzle)):
                path = copy.deepcopy(self.path)
                path.append('right')
                self.generated.append(rightPuzzle)
                self.queue.insert(0,[self.misplaced(rightPuzzle) +
self.depth, rightPuzzle, path, self.depth])
                Puzzle.node += 1


        # check if it is possible to move down
        if (self.isDown()):
            downPuzzle = self.down()
            # if the state has not been generated before
            if (not self.isGenerated(downPuzzle)):
                path = copy.deepcopy(self.path)
                path.append('down')
                self.generated.append(downPuzzle)
                self.queue.insert(0,[self.misplaced(downPuzzle) +
self.depth, downPuzzle, path, self.depth])
                Puzzle.node += 1
```

```
        # check if it is possible to move left
        if (self.isLeft()):
            leftPuzzle = self.left()
            # if the state has not been generated before
            if (not self.isGenerated(leftPuzzle)):
                path = copy.deepcopy(self.path)
                path.append('left')
                self.generated.append(leftPuzzle)
                self.queue.insert(0,[self.misplaced(leftPuzzle) +
self.depth, leftPuzzle, path, self.depth])
                Puzzle.node += 1


        self.queue.sort()


        current = self.queue.pop(0)


        self.puzzle = current[1]
        self.path = current[2]
        self.depth = current[3]
```

main.py: main program

```
from puzzle import Puzzle
import time
import copy


# Main Program


print()
print("=========================================")
print("======WELCOME TO 15 PUZZLE SOLVER!======")
print("=========================================")
print()


print("*** DO YOU WANT TO USE INPUT FILE? (Y/N)")
```

```
# Asumsi input user selalu benar
inputUser = input()
print()


puzzle = Puzzle()


if (inputUser == "Y"):
    inputFile = input("*** INPUT THE FILE NAME (with .txt): ")
    puzzle.readFile(inputFile)
else:
    puzzle.random()


printpuzzle = copy.deepcopy(puzzle)


print()
print("========PUZZLE's INITIAL STATE==========")
puzzle.print()
print()


if (puzzle.isSolvable()):


    startTime = time.time()


    puzzle.generated.append(puzzle.puzzle)


    while(not puzzle.isSolution()):
       puzzle.solvePuzzle()


    endTime = time.time()
    n = len(puzzle.path)


    for path in puzzle.path:
        if (path == 'up'):
            printpuzzle.printup()
        elif (path == 'down'):
            printpuzzle.printdown()
```

```
        elif (path == 'left'):
            printpuzzle.printleft()
        else:
            printpuzzle.printright()



    print("PUZZLE IS SOLVED!")
    print("TOTAL STEPS : ", n)
    print("TOTAL NODE GENERATED: ", Puzzle.node)
    print("TIME TAKEN: ",endTime - startTime,"s")


else:
    print("PUZZLE IS NOT SOLVABLE :(")
```

BERKAS TEKS INSTANSIASI 5 BUAH PERSOALAN 15-PUZZLE

Perhatikan bahwa tidak boleh ada spasi di akhir setiap line!!!

Solvable 1

16 2 3 4
1 6 7 8
5 10 11 12
9 13 14 15

```
========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): solvable1.txt

========PUZZLE's INITIAL STATE==========
16 2 3 4
1 6 7 8
5 10 11 12
9 13 14 15

KURANG (1) = 0
KURANG (2) = 1
KURANG (3) = 1
KURANG (4) = 1
KURANG (5) = 0
KURANG (6) = 1
KURANG (7) = 1
KURANG (8) = 1
KURANG (9) = 0
KURANG (10) = 1
KURANG (11) = 1
KURANG (12) = 1
KURANG (13) = 0
KURANG (14) = 0
KURANG (15) = 0
KURANG (16) = 15
X = 0
SIGMA of KURANG(i) + X =  24

Loading...
```

```
down
1 2 3 4
16 6 7 8
5 10 11 12
9 13 14 15

down
1 2 3 4
5 6 7 8
16 10 11 12
9 13 14 15

down
1 2 3 4
5 6 7 8
9 10 11 12
16 13 14 15

right
1 2 3 4
5 6 7 8
9 10 11 12
13 16 14 15

right
1 2 3 4
5 6 7 8
9 10 11 12
13 14 16 15

right
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

PUZZLE IS SOLVED!
TOTAL STEPS :  6
TOTAL NODE GENERATED:  11
TIME TAKEN:  0.0009975433349609375 s
```

Solvable 2

5 1 3 4
9 2 7 8
16 6 15 11
13 10 14 12

```
========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): solvable2.txt

========PUZZLE's INITIAL STATE==========
5 1 3 4
9 2 7 8
16 6 15 11
13 10 14 12

KURANG (1) = 0
KURANG (2) = 0
KURANG (3) = 1
KURANG (4) = 1
KURANG (5) = 4
KURANG (6) = 0
KURANG (7) = 1
KURANG (8) = 1
KURANG (9) = 4
KURANG (10) = 0
KURANG (11) = 1
KURANG (12) = 0
KURANG (13) = 2
KURANG (14) = 1
KURANG (15) = 5
KURANG (16) = 7
X = 0
SIGMA of KURANG(i) + X =  28

Loading...
```

```
up
5 1 3 4
16 2 7 8
9 6 15 11
13 10 14 12

up
16 1 3 4
5 2 7 8
9 6 15 11
13 10 14 12

right
1 16 3 4
5 2 7 8
9 6 15 11
13 10 14 12

down
1 2 3 4
5 16 7 8
9 6 15 11
13 10 14 12

down
1 2 3 4
5 6 7 8
9 16 15 11
13 10 14 12

down
1 2 3 4
5 6 7 8
9 10 15 11
13 16 14 12

right
1 2 3 4
5 6 7 8
9 10 15 11
13 14 16 12
```

```
up
1 2 3 4
5 6 7 8
9 10 16 11
13 14 15 12

right
1 2 3 4
5 6 7 8
9 10 11 16
13 14 15 12

down
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

PUZZLE IS SOLVED!
TOTAL STEPS :  10
TOTAL NODE GENERATED:  23
TIME TAKEN:  0.0009982585906982422 s
```

Solvable 3

6 5 2 4
9 1 3 8
10 16 7 15
13 14 12 11

```
========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): solvable3.txt

========PUZZLE's INITIAL STATE==========
6 5 2 4
9 1 3 8
10 16 7 15
13 14 12 11

KURANG (1) = 0
KURANG (2) = 1
KURANG (3) = 0
KURANG (4) = 2
KURANG (5) = 4
KURANG (6) = 5
KURANG (7) = 0
KURANG (8) = 1
KURANG (9) = 4
KURANG (10) = 1
KURANG (11) = 0
KURANG (12) = 1
KURANG (13) = 2
KURANG (14) = 2
KURANG (15) = 4
KURANG (16) = 6
X = 1
SIGMA of KURANG(i) + X =  34

Loading...
```

```
left
6 5 2 4
9 1 3 8
16 10 7 15
13 14 12 11

up
6 5 2 4
16 1 3 8
9 10 7 15
13 14 12 11

up
16 5 2 4
6 1 3 8
9 10 7 15
13 14 12 11

right
5 16 2 4
6 1 3 8
9 10 7 15
13 14 12 11

down
5 1 2 4
6 16 3 8
9 10 7 15
13 14 12 11

left
5 1 2 4
16 6 3 8
9 10 7 15
13 14 12 11
```

```
up
16 1 2 4
5 6 3 8
9 10 7 15
13 14 12 11

right
1 16 2 4
5 6 3 8
9 10 7 15
13 14 12 11

right
1 2 16 4
5 6 3 8
9 10 7 15
13 14 12 11

down
1 2 3 4
5 6 16 8
9 10 7 15
13 14 12 11

down
1 2 3 4
5 6 7 8
9 10 16 15
13 14 12 11

right
1 2 3 4
5 6 7 8
9 10 15 16
13 14 12 11
```

```
down
1 2 3 4
5 6 7 8
9 10 15 11
13 14 12 16

left
1 2 3 4
5 6 7 8
9 10 15 11
13 14 16 12

up
1 2 3 4
5 6 7 8
9 10 16 11
13 14 15 12

right
1 2 3 4
5 6 7 8
9 10 11 16
13 14 15 12

down
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

PUZZLE IS SOLVED!
TOTAL STEPS :  17
TOTAL NODE GENERATED:  626
TIME TAKEN:  0.019998788833618164 s
```

Not Solvable 1

14 12 1 3
4 2 13 10
6 11 15 8
7 9 5 16

```
========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): notsolvable1.txt

========PUZZLE's INITIAL STATE==========
14 12 1 3
4 2 13 10
6 11 15 8
7 9 5 16

KURANG (1) = 0
KURANG (2) = 0
KURANG (3) = 1
KURANG (4) = 1
KURANG (5) = 0
KURANG (6) = 1
KURANG (7) = 1
KURANG (8) = 2
KURANG (9) = 1
KURANG (10) = 5
KURANG (11) = 4
KURANG (12) = 11
KURANG (13) = 7
KURANG (14) = 13
KURANG (15) = 4
KURANG (16) = 0
X = 0
SIGMA of KURANG(i) + X =  51

Loading...

PUZZLE IS NOT SOLVABLE :(
```

Not Solvable 2

12 9 15 13
16 2 7 4
8 3 6 11
1 14 5 10

```
========================================
======WELCOME TO 15 PUZZLE SOLVER!======
========================================

*** DO YOU WANT TO USE INPUT FILE? (Y/N)
Y

*** INPUT THE FILE NAME (with .txt): notsolvable2.txt

========PUZZLE's INITIAL STATE==========
12 9 15 13
16 2 7 4
8 3 6 11
1 14 5 10

KURANG (1) = 0
KURANG (2) = 1
KURANG (3) = 1
KURANG (4) = 2
KURANG (5) = 0
KURANG (6) = 2
KURANG (7) = 5
KURANG (8) = 4
KURANG (9) = 8
KURANG (10) = 0
KURANG (11) = 3
KURANG (12) = 11
KURANG (13) = 10
KURANG (14) = 2
KURANG (15) = 12
KURANG (16) = 11
X = 1
SIGMA of KURANG(i) + X =  73

Loading...

PUZZLE IS NOT SOLVABLE :(
```

ALAMAT DRIVE

https://drive.google.com/drive/folders/1fzks46pmx64xz20z6MEph56x7CB6Ocfv?usp=sharing

ALAMAT GITHUB

https://github.com/clauculus/Tucil3_13520076.git