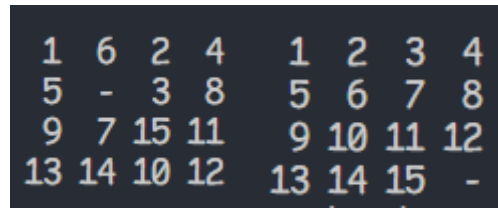


Tugas Kecil 3
IF2211 STRATEGI ALGORITMA
Penyelesaian Persoalan 15-Puzzle dengan Algoritma
Branch and Bound

Oleh
13520153 – Vito Ghifari

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
SEMESTER 2 2021/2022

Algoritma *Branch and Bound* pada Persoalan 15-Puzzle



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

Gambar 1. Kiri: konfigurasi 15-Puzzle sembarang;
Kanan: konfigurasi 15-Puzzle yang terselesaikan.

1. Asumsi konfigurasi awal dari 15-puzzle yang di-*input* dapat diselesaikan. Untuk memeriksa jika suatu konfigurasi puzzle dapat diselesaikan, konfigurasi puzzle tersebut dihitung nilai dari $KURANG(i) + x$. Konfigurasi awal ini disebut dengan simpul akar.
2. Inisialisasi *priority queue* dengan prioritas tertinggi *queue* adalah *node* dengan *cost* terkecil.
3. *Cost* atau $c(P)$ didefinisikan sebagai $f(P) + g(P)$. $f(P)$ yaitu kedalaman atau *step* yang dibutuhkan untuk mencapai simpul tersebut dari simpul akar. Sementara itu, $g(P)$ adalah banyaknya *tile* pada puzzle yang tidak sesuai pada tempatnya.
4. Definisikan *min_node* sebagai simpul akar
5. Bangkitkan simpul anak-anak *min_node*. Simpul anak adalah konfigurasi puzzle dari simpul *min_node* yang diubah posisi *tile* kosongnya. Pengubahan posisi ini bisa ke atas, ke kanan, ke kiri, dan ke bawah. Beberapa *move* mungkin tidak valid (contohnya *tile* kosong berada di kolom paling kanan tetapi *move* nya “right”).
6. Setiap simpul anak dicek terlebih dahulu apakah konfigurasi simpul tersebut sudah ada pada simpul lain. Jika tidak ada, simpul ini valid.
7. Simpul anak yang valid dimasukkan ke dalam *priority queue*.
8. Keluarkan simpul dari *priority queue*. Cek nilai $g(P)$ pada simpul ini.
9. Jika $g(P)$ simpul ini lebih dari nol, definisikan simpul ini sebagai *min_node* dan kembali ke langkah 5. Jika $g(P)$ sama dengan nol, lanjutkan ke langkah berikutnya.
10. Simpul *min_node* adalah solusi dari 15-Puzzle. Program selesai.

Screenshot Input-Output Program

```
>>INPUT 15-PUZZLE DARI FILE<<
Contoh: test/instance1.txt, instance1.txt
Masukkan nama file: test/solvable0.txt
Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:

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

Gambar 2.1 Test case solvable0.txt

```
-- Fungsi KURANG(i) --
1: 0
2: 0
3: 0
4: 0
5: 0
6: 0
7: 0
8: 0
9: 0
10: 0
11: 0
12: 0
13: 1
14: 1
15: 1
16: 5
```

Gambar 2.2 Fungsi KURANG(i) solvable0.txt

Langkah ke-1:

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

Langkah ke-2:

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

Gambar 2.3 Langkah-langkah solvable0.txt

```
PUZZLE BERHASIL DISELESAIKAN
Waktu eksekusi: 0.002000570297241211 s
Kedalaman / step: 2
Node dibangkitkan: 7
Nilai [SIGMA Kurang(i)] + X: 8
```

Gambar 2.4 Hasil solvable0.txt

```
>>INPUT 15-PUZZLE DARI FILE<<
Contoh: test/instance1.txt, instance1.txt
Masukkan nama file: test/solvable2.txt
Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:

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

Proses puzzle ini (Y/N)? y
15-PUZZLE SOLVER STARTING...
```

Gambar 3.1 Test case solvable2.txt

```
-- Fungsi KURANG(i) --
1: 0
2: 0
3: 0
4: 1
5: 1
6: 4
7: 0
8: 1
9: 1
10: 0
11: 1
12: 0
13: 2
14: 2
15: 5
16: 10
```

Gambar 3.2 Fungsi KURANG(i) solvable2.txt

Langkah ke-1:	Langkah ke-18:
1 6 2 4	1 2 3 4
- 5 3 8	5 6 7 8
9 7 15 11	9 10 11 12
13 14 10 12	13 14 15 -
arah: left	arah: down

Gambar 3.3 Langkah-langkah solvable2.txt

```

PUZZLE BERHASIL DISELESAIKAN
Waktu eksekusi: 0.25036072731018066 s
Kedalaman / step: 18
Node dibangkitkan: 6510
Nilai [SIGMA Kurang(i)] + X: 28

```

Gambar 3.4 Hasil solvable2.txt

```

> type
>>INPUT 15-PUZZLE DARI CONSOLE<<
Pisahkan kolom dengan spasi; Pisahkan baris dengan enter.
Substitusi tile kosong dengan angka 0.
[1]: 5 1 3 4
[2]: 9 2 7 8
[3]: 0 6 15 11
[4]: 13 10 14 12
Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:

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

Proses puzzle ini (Y/N)? y
15-PUZZLE SOLVER STARTING...

```

Gambar 4.1 Test case dari input manual

```

-- Fungsi KURANG(i) --
1: 0
2: 0
3: 1
4: 1
5: 4
6: 0
7: 1
8: 1
9: 4
10: 0
11: 1
12: 0
13: 2
14: 1
15: 5
16: 7

```

Gambar 4.2 Fungsi KURANG(i) test case input manual

Langkah ke-1: 5 1 3 4 - 2 7 8 9 6 15 11 13 10 14 12 arah: up	Langkah ke-3: 1 - 3 4 5 2 7 8 9 6 15 11 13 10 14 12 arah: right	Langkah ke-5: 1 2 3 4 5 6 7 8 9 - 15 11 13 10 14 12 arah: down	Langkah ke-7: 1 2 3 4 5 6 7 8 9 10 15 11 13 14 - 12 arah: right	Langkah ke-9: 1 2 3 4 5 6 7 8 9 10 11 - 13 14 15 12 arah: right
Langkah ke-2: - 1 3 4 5 2 7 8 9 6 15 11 13 10 14 12 arah: up	Langkah ke-4: 1 2 3 4 5 - 7 8 9 6 15 11 13 10 14 12 arah: down	Langkah ke-6: 1 2 3 4 5 6 7 8 9 10 15 11 13 - 14 12 arah: down	Langkah ke-8: 1 2 3 4 5 6 7 8 9 10 - 11 13 14 15 12 arah: up	Langkah ke-10: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 - arah: down

Gambar 4.3 Langkah-langkah 15-puzzle dari input manual

```

PUZZLE BERHASIL DISELESAIKAN
Waktu eksekusi: 0.0030014514923095703 s
Kedalaman / step: 10
Node dibangkitkan: 24
Nilai [SIGMA Kurang(i)] + X: 28

```

Gambar 4.4 Hasil solvable2.txt

```

>>>INPUT 15-PUZZLE DARI FILE<<
Contoh: test/instance1.txt, instance1.txt
Masukkan nama file: test/unsolvable1.txt
Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:

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

Proses puzzle ini (Y/N)? y
15-PUZZLE SOLVER STARTING...

-- Fungsi KURANG(i) --
1: 0
2: 0
3: 1
4: 1
5: 0
6: 0
7: 1
8: 0
9: 0
10: 0
11: 3
12: 6
13: 0
14: 4
15: 11
16: 10

Konfigurasi ini tidak dapat diselesaikan.
Nilai [SIGMA Kurang(i)] + X: 37

```

Gambar 5 Hasil unsolvable1.txt

```
> type
>>INPUT 15-PUZZLE DARI CONSOLE<<
Pisahkan kolom dengan spasi; Pisahkan baris dengan enter.
Substitusi tile kosong dengan angka 0.
[1]: 1 2 3 4
[2]: 5 6 7 8
[3]: 9 10 11 12
[4]: 13 15 14 0
Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:

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

Proses puzzle ini (Y/N)? y

15-PUZZLE SOLVER STARTING...

-- Fungsi KURANG(i) --
1: 0
2: 0
3: 0
4: 0
5: 0
6: 0
7: 0
8: 0
9: 0
10: 0
11: 0
12: 0
13: 0
14: 0
15: 1
16: 0

Konfigurasi ini tidak dapat diselesaikan.
Nilai [SIGMA Kurang(i)] + X: 1
```

Gambar 6 Hasil puzzle yang tidak dapat diselesaikan dari input manual

Daftar Periksa Program

Poin	Ya	Tidak
1. Program berhasil dikompilasi	✓	
2. Program berhasil <i>running</i>	✓	
3. Program dapat menerima input dan menuliskan output.	✓	
4. Luaran sudah benar untuk semua data uji	✓	
5. Bonus dibuat		✓

Kode Sumber Program

bnb.py

```
from copy import deepcopy
from time import time
from Node import Node, flatten, display_puzzle
from queue import PriorityQueue

# heuristik untuk tidak membangkitkan node dengan konfigurasi yang sama
seen_combination = dict()
# jumlah node yang dibangkitkan
generated_node = 0

def solvable(puzzle) -> bool:
    kurang = [0 for _ in range(17)]
    x = 0
    flattened = flatten(puzzle)
    for index, i in enumerate(flattened):
        if i == 0:
            i = 16
            # Menentukan sel kosong di tempat arsir atau tidak
            index_row = index // 4
            index_col = index % 4
            if (index_row + index_col % 2) == 1:
                x = 1
        temp = 0
        for j in flattened[index+1:]:
            if j < i and j != 0:
                temp += 1
        kurang[i] = temp
    print("-- Fungsi KURANG(i) -- ")
    for i in range(1, 17):
        print(f"{i}: {kurang[i]}")
    print()
    return sum(kurang) + x

def get_empty_position(puzzle: list) -> tuple:
    for row_index, row in enumerate(puzzle):
        for col_index, cell in enumerate(row):
            if cell == 0:
                return (row_index, col_index)

def move(puzzle: list, empty_position: tuple, direction: str) -> list:
```

```

empty_row = empty_position[0]
empty_col = empty_position[1]

x_mod = 0
y_mod = 0

# Jika direction tidak valid, return None
if direction == "up":
    if empty_row == 0:
        return
    y_mod = -1
elif direction == "down":
    if empty_row == 3:
        return
    y_mod = 1
elif direction == "left":
    if empty_col == 0:
        return
    x_mod = -1
elif direction == "right":
    if empty_col == 3:
        return
    x_mod = 1

puzzle_copy = deepcopy(puzzle)

# Melakukan perubahan posisi antara tile kosong dengan tile
sebelahnya
tmp = puzzle_copy[empty_row + y_mod][empty_col + x_mod]
puzzle_copy[empty_row + y_mod][empty_col + x_mod] = 0
puzzle_copy[empty_row][empty_col] = tmp
return puzzle_copy

def generate_child(puzzle_node: Node) -> list:
    global seen_combination, generated_node
    moved_puzzle = []

    row, col = get_empty_position(puzzle_node.get_puzzle())
    direction_list = ["left", "right", "up", "down"]

    # Kombinasi arah
    for direction in direction_list:
        move_puzzle = move(puzzle_node.get_puzzle(), (row, col),
direction)
        if move_puzzle is not None:
            # Jika move valid, cek apakah konfigurasi ini pernah
dibangkitkan
            if not seen_combination.get(str(move_puzzle)):

```

```

        generated_node += 1
        seen_combination[str(move_puzzle)] = True
        move_node = Node(move_puzzle, puzzle_node,
                        puzzle_node.get_depth() + 1,
direction)

        moved_puzzle.append(move_node)

    return moved_puzzle

def solve_15_puzzle(puzzle):
    # Me-reset seen_combination dan generated_node
    global seen_combination, generated_node
    seen_combination = dict()
    generated_node = 0

    print("\n15-PUZZLE SOLVER STARTING...\n")

    time_start = time()
    num = solvable(puzzle)
    if not num % 2 == 0:
        print("Konfigurasi ini tidak dapat diselesaikan.")
        print("Nilai [SIGMA Kurang(i)] + X:", num)
        return

    q = PriorityQueue()

    root_node = Node(puzzle, None, 0, None)
    if (root_node.calculate_g() == 0):
        print("Puzzle sudah berada pada posisi solusi!")
        return

    min_node = root_node
    # Selama goal node belum tercapai, lakukan iterasi untuk mencari
node hingga g = 0
    while (min_node.calculate_g() > 0):
        child_nodes = generate_child(min_node)
        for node in child_nodes:
            q.put(node)
        min_node = q.get()

    time_stop = time()

    # Lakukan iterasi dari node akhir ke root node
    steps = [min_node]
    parent_node = min_node.get_parent_node()
    while parent_node is not None:
        steps.append(parent_node)
        parent_node = parent_node.get_parent_node()

```

```

steps.reverse()

for i, node in enumerate(steps):
    print(f"Langkah ke-{i}:")
    display_puzzle(node.get_puzzle())
    print(f"arah: {node.get_previous_move() if
(node.get_previous_move() is not None) else '-'}\n")

print("PUZZLE BERHASIL DISELESAIKAN")
print(f"Waktu eksekusi: {time_stop - time_start} s")
print(f"Kedalaman / step: {min_node.get_depth()}")
print(f"Node dibangkitkan: {generated_node}")
print("Nilai [SIGMA Kurang(i)] + X:", num)

```

main.py

```

from read_input import parse_console, parse_file
from bnb import solve_15_puzzle

def print_help():
    print(">>> Command tersedia <<<")
    print("help: Menampilkan menu ini")
    print("file: Melakukan load puzzle dari file")
    print("type: Melakukan load puzzle dari input console")
    print("exit: Mengakhiri program")

def main():
    print("--- 15-PUZZLE SOLVER ---")
    print("supported by branch and bound algorithm\n")
    print_help()

    cmd = input("\n> ")
    while cmd != "exit":
        if cmd == "file":
            success, puzzle = parse_file()
            if success:
                solve_15_puzzle(puzzle)
        elif cmd == "type":
            success, puzzle = parse_console()
            if success:
                solve_15_puzzle(puzzle)
        elif cmd == "exit":
            pass

```

```

        elif cmd == "help":
            print_help()
        else:
            print("Command tidak diketahui!")
            cmd = input("\n> ")

    print("Terima kasih telah menggunakan program 15-puzzle
solver :)")

main()

```

Node.py

```

class Node:
    def __init__(self, puzzle: list, parent_node, depth: int,
previous_move: str = None):
        self._puzzle = puzzle
        self._parent_node = parent_node
        self._depth = depth
        self._cost = depth + self.calculate_g() #  $c(P) = f(P) + g(P)$ 
        self._previous_move = previous_move

    def calculate_g(self) -> int:
        flattened = flatten(self._puzzle)
        g = 0
        for i in range(16):
            if (flattened[i] != i+1) and (flattened[i] != 0):
                g += 1
        return g

    def get_parent_node(self):
        return self._parent_node

    def get_puzzle(self) -> list:
        return self._puzzle

    def get_cost(self) -> int:
        return self._cost

    def get_previous_move(self) -> str:
        return self._previous_move

    def get_depth(self) -> int:
        return self._depth

    def debug(self) -> None:

```

```

        display_puzzle(self._puzzle)
        print("cost:", self._cost)
        print("prev move:", self._previous_move)
        print()

        # operasi less than (<) untuk priority queue
        def __lt__(self, other) -> bool:
            return self.get_cost() < other.get_cost()

def flatten(puzzle) -> list:
    # Mengubah puzzle 2D menjadi 1D
    flattened = []
    for content in puzzle:
        flattened.extend(content)
    return flattened

def display_puzzle(puzzle) -> None:
    for row in puzzle:
        for cell in row:
            if cell < 10:
                if cell == 0:
                    print("-", end=" ")
                else:
                    print(f" {cell}", end=" ")
            else:
                print(cell, end=" ")
        print()

```

read_input.py

```

from Node import display_puzzle

def parse_puzzle(puzzle_raw: list):
    valid = True
    puzzle = []

    # Validasi puzzle dengan dictionary yang me-mapping dari 0 hingga
    16
    number_map = dict([(str(x), 0) for x in range(16)])

    if len(puzzle_raw) != 4:
        valid = False
    else:
        for k in puzzle_raw:
            row = k.strip().split(" ")

```

```

        for cell in row:
            res = number_map.get(cell)
            if res is None or res > 0:
                valid = False
                break
            else:
                number_map[cell] = 1
        puzzle.append([int(x) for x in row])
    return (valid, puzzle)

def parse_file():
    print("\n>>INPUT 15-PUZZLE DARI FILE<<")
    print("Contoh: test/instance1.txt, instance1.txt")
    filename = input("Masukkan nama file: ")

    success = False
    puzzle = []
    try:
        with open(filename, "r") as f:
            valid, puzzle = parse_puzzle(f.readlines())
            if valid:
                print("Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:\n")
                display_puzzle(puzzle)
                confirm = input("\nProses puzzle ini (Y/N)? ")
                if confirm.lower() == "y":
                    success = True
                else:
                    print("Proses dibatalkan")
            else:
                print("Konfigurasi puzzle tidak valid!")
    except FileNotFoundError:
        print("File tidak ditemukan!")

    return (success, puzzle)

def parse_console():
    print(">>INPUT 15-PUZZLE DARI CONSOLE<<")
    print("Pisahkan kolom dengan spasi; Pisahkan baris dengan enter.")
    print("Substitusi tile kosong dengan angka 0.")

    puzzle = []
    puzzle_raw = []
    success = False
    for i in range(4):
        row = input(f"[{i+1}]: ")
        puzzle_raw.append(row)

```

```
valid, puzzle = parse_puzzle(puzzle_raw)

if not valid:
    print("Konfigurasi puzzle yang di-input tidak valid!")
else:
    print("Puzzle berhasil di-parse! Berikut konfigurasi puzzle yang terdeteksi:\n")
    display_puzzle(puzzle)
    confirm = input("\nProses puzzle ini (Y/N)? ")
    if confirm.lower() == "y":
        success = True
    else:
        print("Proses dibatalkan")

return (success, puzzle)
```


Instansiasi Persoalan 15-Puzzle

Test case 1 (solvable):

1 2 3 4

5 6 7 8

9 10 - 11

13 14 15 12

Test case 2 (solvable):

1 2 3 4

5 6 - 8

9 10 7 11

13 14 15 12

Test case 3 (solvable):

1 6 2 4

5 - 3 8

9 7 15 11

13 14 10 12

Test case 4 (unsolvable):

1 3 4 15

2 - 5 12

7 6 11 14

8 9 10 13

Test case 5 (unsolvable):

1 2 3 4

5 6 7 8

9 10 11 12

13 15 14 -

Link Repository Tugas Kecil 2 Strategi Algoritma

https://github.com/VanillaMacchiato/Tucil3_13520153