

PENDAHULUAN

Pada implementasi permasalahan TSP dengan Dynamic programming, saya menggunakan bahasa rust untuk mengimplementasikan tantangan ini. Dalam Program saya, terdapat 3 file utama yaitu main.rs untuk melakukan penginputan dan juga pemanggilan fungsi, kemudian terdapat juga tsp.rs yang dimana terdapat kode program utama untuk permasalahan tsp, dan terakhir matrix_reader.rs yang berfungsi untuk membaca file txt untuk penginputannya.

IMPLEMENTASI PROGRAM

Main.rs

```
1 mod matrix_reader;
2 mod tsp;
3 use tsp::{tsp_dp};
4 use matrix_reader::read_matrix_from_file;
5
6 use std::io::{self};
7
8 fn main() -> io::Result<()> {
9     println!("Enter the filename containing the matrix:");
10    let mut input = String::new();
11    io::stdin().read_line(&mut input)?;
12    let filename = input.trim();
13
14    let dist = read_matrix_from_file(filename)?;
15    let n = dist.len();
16
17    let (result, route) = tsp_dp(&dist, n);
18
19    if result == i32::MAX {
20        println!("No feasible path found due to infinite or invalid weights.");
21    } else {
22        println!("\nThe minimum cost is: {}", result);
23        println!("Path taken (starting from city 1): {}",
24            route.iter().map(|&x| (x + 1).to_string()).collect::<Vec<_>>().join(" -> "));
25
26        if route.len() > 1 {
27            println!("\nDetailed route:");
28            for i in 0..route.len() - 1 {
29                let from = route[i];
30                let to = route[i + 1];
31                println!("Travel from city {} to city {}: {} units",
32                    from + 1, to + 1, dist[from][to]);
33            }
34        }
35    }
36
37    Ok(())
38 }
```

tsp.rs

```
1  const INFINITY: i32 = i32::MAX;
2
3  pub fn tsp_dp(dist: &[Vec<i32>], n: usize) -> (i32, Vec<usize>) {
4      let mut memo = vec![vec![i32::MAX; 1 << n]; n];
5      let mut path = vec![vec![usize::MAX; 1 << n]; n];
6
7      let min_cost = tsp(0, 1, n, dist, &mut memo, &mut path);
8      let route = find_path(n, &path, 0);
9
10     (min_cost, route)
11 }
12
13 pub fn tsp(pos: usize, visited: usize, n: usize, dist: &[Vec<i32>], memo: &mut Vec<Vec<i32>>, path: &mut Vec<Vec<usize>>) -> i32 {
14     if visited == (1 << n) - 1 {
15         return dist[pos][0];
16     }
17
18     if memo[pos][visited] != i32::MAX {
19         return memo[pos][visited];
20     }
21
22     let mut res = i32::MAX;
23     for i in 0..n {
24         if (visited & (1 << i)) == 0 && dist[pos][i] != INFINITY {
25             let next_cost = tsp(i, visited | (1 << i), n, dist, memo, path);
26             if next_cost != INFINITY {
27                 let cost = dist[pos][i] + next_cost;
28                 if cost < res {
29                     res = cost;
30                     path[pos][visited] = i;
31                 }
32             }
33         }
34     }
35
36     memo[pos][visited] = res;
37     res
38 }
```

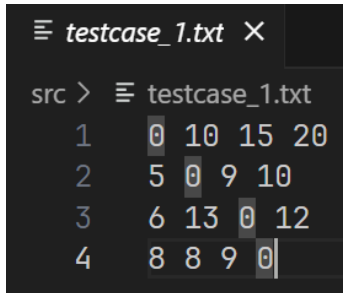
```
40 pub fn find_path(n: usize, path: &[Vec<usize>], start: usize) -> Vec<usize> {
41     let mut route = vec![start];
42     let mut current = start;
43     let mut visited = 1 << start;
44
45     while visited != (1 << n) - 1 {
46         let next = path[current][visited];
47         route.push(next);
48         visited |= 1 << next;
49         current = next;
50     }
51     route.push(start);
52     route
53 }
```

matrix_reader.rs

```
1 use std::fs::File;
2 use std::io::{self, BufRead, BufReader};
3
4 const INFINITY: i32 = i32::MAX;
5
6 pub fn read_matrix_from_file(filename: &str) -> io::Result<Vec<Vec<i32>>> {
7     let file = File::open(filename)?;
8     let reader = BufReader::new(file);
9     let mut matrix = Vec::new();
10
11     for line in reader.lines() {
12         let line = line?.trim().to_string();
13         let numbers: Vec<i32> = line.split_whitespace()
14             .map(|num| {
15                 if num == "∞" {
16                     INFINITY
17                 } else {
18                     num.parse().unwrap_or_else(|_| {
19                         eprintln!("Error parsing number: {}", num);
20                         std::process::exit(1);
21                     })
22                 }
23             })
24             .collect();
25         matrix.push(numbers);
26     }
27     Ok(matrix)
28 }
```

PENGUJIAN

1. Test case 1



testcase_1.txt

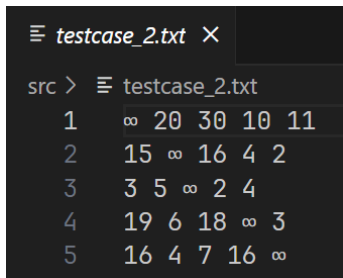
src >	≡	testcase_1.txt
1	0	10 15 20
2	5	0 9 10
3	6	13 0 12
4	8	8 9 0

```
PS C:\Users\User\Documents\ITB\Sem 4\Stima\TSP_with_DP\src> cargo run
Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.01s
Running `C:\Users\User\Documents\ITB\Sem 4\Stima\TSP_with_DP\target\debug\tsp_algo.exe`
Enter the filename containing the matrix:
testcase_1.txt

The minimum cost is: 35
Path taken (starting from city 1): 1 -> 2 -> 4 -> 3 -> 1

Detailed route:
Travel from city 1 to city 2: 10 units
Travel from city 2 to city 4: 10 units
Travel from city 4 to city 3: 9 units
Travel from city 3 to city 1: 6 units
```

2. Test case 2



testcase_2.txt

src >	≡	testcase_2.txt
1	∞	20 30 10 11
2	15	∞ 16 4 2
3	3	5 ∞ 2 4
4	19	6 18 ∞ 3
5	16	4 7 16 ∞

```
PS C:\Users\User\Documents\ITB\Sem 4\Stima\TSP_with_DP\src> cargo run
Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.01s
Running `C:\Users\User\Documents\ITB\Sem 4\Stima\TSP_with_DP\target\debug\tsp_algo.exe`
Enter the filename containing the matrix:
testcase_2.txt

The minimum cost is: 28
Path taken (starting from city 1): 1 -> 4 -> 2 -> 5 -> 3 -> 1

Detailed route:
Travel from city 1 to city 4: 10 units
Travel from city 4 to city 2: 6 units
Travel from city 2 to city 5: 2 units
Travel from city 5 to city 3: 7 units
Travel from city 3 to city 1: 3 units
```

3. Test case 3

	testcase_3.txt						
src >	testcase_3.txt						
1	0	12	10	∞	∞	∞	12
2	12	0	8	11	∞	∞	∞
3	10	8	0	∞	3	∞	9
4	∞	11	∞	0	11	10	∞
5	∞	∞	3	11	0	6	7
6	∞	∞	∞	10	6	0	9
7	12	∞	9	∞	7	9	0

```
PS C:\Users\User\Documents\ITB\Sem 4\Stima\TSP_with_DP\src> cargo run
Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.01s
Running `C:\Users\User\Documents\ITB\Sem 4\Stima\TSP_with_DP\target\debug\tsp_algo.exe`
Enter the filename containing the matrix:
testcase_3.txt

The minimum cost is: 62
Path taken (starting from city 1): 1 -> 2 -> 4 -> 6 -> 7 -> 5 -> 3 -> 1

Detailed route:
Travel from city 1 to city 2: 12 units
Travel from city 2 to city 4: 11 units
Travel from city 4 to city 6: 10 units
Travel from city 6 to city 7: 9 units
Travel from city 7 to city 5: 7 units
Travel from city 5 to city 3: 3 units
Travel from city 3 to city 1: 10 units
```

LAMPIRAN

Link Github :

https://github.com/Filbert88/TSP_with_DP