Tugas Kecil 1 IF2211 Strategi Algoritma

Semester II tahun 2023/2024

Penyelesaian Cyberpunk 2077 Breach Protocol dengan Algoritma Brute Force

Mohammad Akmal Ramadan

13522161

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2024

BAGIAN I

Algoritma Brute Force

```cpp
void findPaths(const std::vector<std::vector<std::string>> &matrix,
               int buffer_size,
               int row,
               int col,
               std::vector<std::pair<std::string, std::pair<int, int>>> &currentPath,
               const std::vector<std::vector<std::string>> &sequences,
               const std::vector<int> &rewards,
               int &maxReward,
               bool isVerticalMove,
               std::vector<std::vector<bool>> &visited,
               std::vector<std::pair<std::string, std::pair<int, int>>> &highestRewardPath)
{
  if (row >= matrix.size() || col < 0 || col >= matrix[0].size() || visited[row][col])
  {
    return;
  }

  visited[row][col] = true; // Mark the current position as visited

  currentPath.push_back({matrix[row][col], {row, col}});

  if (currentPath.size() < buffer_size)
  {
    for (int jump = 1; jump < matrix.size(); jump++)
    {
      if (isVerticalMove)
      {
        findPaths(matrix, buffer_size, row + jump, col, currentPath, sequences, rewards,
                  maxReward, false, visited, highestRewardPath);
        findPaths(matrix, buffer_size, row - jump, col, currentPath, sequences, rewards,
                  maxReward, false, visited, highestRewardPath);
      }
      else
      {
        findPaths(matrix, buffer_size, row, col + jump, currentPath, sequences, rewards,
                  maxReward, true, visited, highestRewardPath);

        findPaths(matrix, buffer_size, row, col - jump, currentPath, sequences, rewards,
                  maxReward, true, visited, highestRewardPath);
      }
    }
  }

  int currentReward = calculateReward(currentPath, sequences, rewards);
  if (currentReward > maxReward)
  {
    maxReward = currentReward;
    highestRewardPath = currentPath;
  }

  currentPath.pop_back();

  visited[row][col] = false;
}
```

Gambar I.1 Algoritma Utama untuk Mencari Sequence pada C++

Pada Gambar I.1 merupakan fungsi untuk mencari semua kemungkinan lintasan yang ada pada matriks menggunakan algoritma rekursif depth first search (DFS), fungsi ini menerima 11 parameter, dengan spesifikasi sebagai berikut:

1. const std::vector<std::vector<std::string>> &matrix
   inisialisasi matrix dua dimensi dengan elemen bertipe string
2. std::vector<std::pair<std::string, std::pair<int, int>>> &currentPath
   inisialisasi variabel untuk menyimpan lintasan sementara berisi dua elemen, yang pertama adalah token dan yang kedua menyimpan koodinat dari token tersebut berbentuk tabel

Berikut cara kerja fungsi tersebut dari awal hingga menemukan lintasan:

1. Pada baris 13 – 16, baris ini akan melakukan validasi apakah indeks baris dan kolom merupakan indeks yang valid, apabila berada di rentang indeks minimum dan maksimum matris, maka program akan lanjut, baris ini juga akan memvalidasi apakah sel matriks tersebut sudah pernah dikunjungi atau belum, apabila belum, program akan lanjut.
2. Apabila pada baris 13 – 16 semua syarat untuk lanjut sudah terpenuhi, maka program akan lanjut, lalu pada baris 18 – 20, program akan membuat sel matriks yang sedang dicek menjadi pernah dikunjungi melalui variabel visited, lalu sel matriks tersebut akan dimasukkan ke variabel currentPath, yang berisi sebuah sekuens token yang membentuk lintasan.
3. Lanjut ke baris 22 – 42, ini merupakan baris yang melakukan rekursif berulang kali untuk mencari lintasan, pertama-tama program akan memastikan jumlah token pada sequence masih di bawah jumlah buffer, lalu program akan melakukan rekursif hingga melakukan jump dari indeks pertama ke indeks terakhir
4. Lalu pada baris 44 – 49 akan dihitung rewards ketika sudah mendapat lintasan yang maksimal dan rewards lintasan yang lebih besar dari yang sebelumnya akan disimpan pada variabel.
5. Yang terakhir pada baris 51 – 53 program akan melakukan penghapusan element terakhir pada array currentPath untuk melakukan backtracking dan mencari path lain, lalu variabel visited akan diset false agar bisa dieksplor kembali di lintasan yang lain.

Dengan algoritma rekursif depth first search ini, dipastikan program akan mencari semua kemungkinan lintasan yang ada pada matriks dengan batasan-batasan yang ada.

Pranala *repository:* https://github.com/akmalrmn/Tucil1_13522161

BAGIAN II

Test Case

1. I/O dari contoh pada spesifikasi

```
7
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
BD E9 1C
15
BD 7A BD
20
BD 1C BD 55
30
```

```
Total Reward: 50
Highest Reward Path: 7A BD 7A BD 1C BD 55
1, 1
1, 4
3, 4
3, 5
6, 5
6, 4
5, 4

Execution time: 52 ms
```

2. I/O dengan tinggi lebih besar dibanding lebarnya

```
Enter the matrix size (W x H): 5 7
Enter the buffer size: 6
Enter the token size: 4
Enter the token: B3 J9 K1 OP
Enter number of sequences: 4
Enter the maximal sequence length: 4

The matrix:
OP OP J9 J9 J9
K1 OP OP B3 K1
K1 B3 K1 B3 J9
J9 J9 J9 K1 OP
J9 B3 OP J9 K1
K1 K1 K1 OP OP
B3 OP J9 OP J9

Sequence 1: K1 B3 K1 OP
Reward 1: 23

Sequence 2: OP K1 J9
Reward 2: 83

Sequence 3: K1 K1 B3
Reward 3: 13

Sequence 4: OP K1 K1 B3
Reward 4: 31
```

```
Total Reward: 96
Highest Reward Path: OP K1 J9 K1 K1 B3
1, 1
1, 3
5, 3
5, 2
1, 2
1, 7

Execution time: 12 ms
```

3. I/O dengan lebar lebih besar dibanding tingginya

```
Enter the matrix size (W x H): 8 5
Enter the buffer size: 11
Enter the token size: 5
Enter the token: BF GO K1 R7 J8
Enter number of sequences: 6
Enter the maximal sequence length: 5

The matrix:
BF R7 J8 BF J8 BF J8 GO
GO GO BF GO BF GO R7 R7
GO J8 R7 R7 J8 GO BF BF
K1 K1 BF J8 BF BF K1 K1
R7 K1 K1 GO J8 BF BF BF

Sequence 1: R7 GO
Reward 1: 45

Sequence 2: R7 GO R7
Reward 2: 85

Sequence 3: K1 K1 R7
Reward 3: 99

Sequence 4: R7 J8 GO R7
Reward 4: 82

Sequence 5: GO R7 GO
Reward 5: 78

Sequence 6: GO R7 GO GO
Reward 6: 15
```

```
Total Reward: 389
Highest Reward Path: R7 J8 GO R7 GO R7 BF K1 K1 R7
2, 1
2, 3
1, 3
1, 5
4, 5
4, 3
7, 3
7, 4
8, 4
8, 2

Execution time: 40477 ms
```

4. I/O jika jumlah buffer satu

```
Enter the matrix size (W x H): 6 6
Enter the buffer size: 1
Enter the token size: 3
Enter the token: GK O0 R3
Enter number of sequences: 3
Enter the maximal sequence length: 4

The matrix:
O0 R3 R3 O0 O0 O0
GK R3 O0 R3 R3 O0
GK R3 O0 GK GK GK
R3 O0 GK GK GK O0
R3 O0 GK R3 R3 GK
O0 GK O0 GK GK GK

Sequence 1: R3 GK
Reward 1: 100

Sequence 2: O0 O0 R3
Reward 2: 94

Sequence 3: R3 R3 O0
Reward 3: 47
```

```
Total Reward: 0
Highest Reward Path:

Execution time: 0 ms
```

5. I/O jika hanya terdapat satu token

```
6
3 3
E9 E9 E9
E9 E9 E9
E9 E9 E9
1
E9 E9 E9 E9
50
```

```
Total Reward: 50
Highest Reward Path: E9 E9 E9 E9
1, 1
1, 2
2, 2
2, 3

Execution time: 0 ms
```

6. I/O jika jumlah sequence lebih dari buffer

```
6
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
BD E9 1C BD E9 1C BD
15
BD 7A BD 1C BD E9 1C
20
BD 1C BD 55 7A BD 1C
30
```

```
Total Reward: 0
Highest Reward Path:

Execution time: 3 ms
```

7. I/O jika terdapat lintasan yang benar tanpa ada keanehan pada input

```
8
6 6
7A 55 E9 E9 1C 55
55 7A 1C 7A E9 55
55 1C 1C 55 E9 BD
BD 1C 7A 1C 55 BD
BD 55 BD 7A 1C 1C
1C 55 55 7A 55 7A
3
BD E9 1C BD E9 1C BD
15
BD 7A BD BD E9 1C
20
BD 1C 7A BD 1C
30
```

```
Total Reward: 30
Highest Reward Path: 7A 55 55 BD 1C 7A BD 1C
1, 1
1, 2
6, 2
6, 3
3, 3
3, 4
1, 4
1, 6

Execution time: 173 ms
```

8. I/O jika jumlah buffer tergolong besar

```
12
7 6
7A 55 E9 E9 1C 55 BD
55 7A 1C 7A E9 55 1C
55 1C 1C 55 E9 BD 1C
BD 1C 7A 1C 55 BD 7A
BD 55 BD 7A 1C 1C BD
1C 55 55 7A 55 7A 55
3
BD E9 1C
15
BD 7A BD
20
BD 1C BD 55
30
```

```
Total Reward: 65
Highest Reward Path: 7A 55 55 BD E9 1C BD 7A BD 1C BD 55
1, 1
1, 2
6, 2
6, 3
5, 3
5, 5
7, 5
7, 4
6, 4
6, 5
3, 5
3, 6

Execution time: 281708 ms
```

9. I/O jika tidak terdapat lintasan yang maksimum walaupun input sudah benar

```
3
5 5
55 1C 1C 1C 55
BD 7A 55 1C 7A
7A BD 1C BD 1C
E9 1C 55 BD 1C
1C 1C BD E9 BD
1
55 BD 7A BD
160
```

```
Total Reward: 0
Highest Reward Path:

Execution time: 0 ms
Do you want to write the output to a file? (y/n): y

Enter the name of the file: solusi9.txt

Thank you for playing this minigame!
#MenujuPerubahan
```

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil dijalankan	✓	
3. Program dapat membaca masukan berkas .txt	✓	
4. Program dapat menghasilkan masukan secara acak	✓	
5. Solusi yang diberikan program optimal	✓	
6. Program dapat menyimpan solusi dalam berkas .txt	✓	
7. Program memiliki GUI		✓