TUGAS KECIL I
IF2211 Strategi Algoritma
Penyelesaian Cyberpunk 2077 Breach Protocol dengan Algoritma Brute Force

DISUSUN OLEH:
13522054 - Benjamin Sihombing

Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2023

BAB 1
Deskripsi Singkat Program

Pada tugas kecil ini, program akan mencari solusi dari game Cyberpunk 2077 Breach Protocol. Program ini terinspirasi dari *solver* permainan https://cyberpunk-hacker.com. Untuk menyelesaikan permasalahan, program menggunakan algoritma *brute force*. Di game Cyberpunk 2077 Breach Protocol, terdapat 4 komponen penting. Berikut ini 4 komponen tersebut:

1. Token – terdiri dari dua karakter alfanumerik seperti E9, BD, dan 55.
2. Matriks – terdiri atas token-token yang akan dipilih untuk menyusun urutan kode.
3. Sekuens – sebuah rangkaian token (dua atau lebih) yang harus dicocokkan.
4. Buffer – kumpulan token yang dapat disusun secara sekuensial.

Program ini memiliki beberapa modifikasi. Pada program ini, setiap sekuens memiliki bonus. Bonus bisa bernilai positif maupun negatif. Untuk mendapatkan bonus, sekuens harus terdapat pada buffer. Program akan mencari solusi buffer yang memiliki bonus terbanyak dan ukuran terpendek (terkecil).

Buffer akan dibentuk dari matriks dengan mekanisme berikut:

1. Token pertama akan diambil dari baris teratas matriks
2. Token selanjutnya akan diambil dari token yang berada di atas atau di bawah token sebelumnya.
3. Token selanjutnya akan diambil dari token yang berada di kanan atau di kiri token sebelumnya
4. Proses 2 dan 3 akan diulangi sampai buffer optimal atau penuh

BAB 2
Algoritma

Pada tugas kecil ini, algoritma *brute force* digunakan untuk menyelesaikan Cyberpunk 2077 Breach Protocol. Secara sederhana, program tugas kecil ini bisa dibagi menjadi beberapa bagian. Berikut ini adalah bagian-bagian utama dari program:

1. Input

   Pada bagian ini, pengguna akan diminta untuk menentukan metode input yang diinginkan (input dari file teks atau input dari CLI). Untuk input dari file teks, perlu dimasukkan path file dengan lengkap. Jika file tidak ditemukan, penguna akan diminta memasukkan path file kembali. Di bagian ini, input dari user akan disimpan ke dalam variabel-variabel. Bagian ini juga terdapat pembangkit acak untuk input dari CLI. Input dari pengguna dianggap sudah benar pada program ini.
2. Proses mencari solusi

   Untuk mencari solusi, dilakukan beberapa langkah. Berikut ini langkah-langkah mencari solusi:

   a. Mencari semua kemungkinan buffer

      Pertama, dicari semua kombinasi translasi yang mungkin. Kemudian, kombinasi translasi digunakan untuk membuat buffer dalam bentuk koordinat (x,y). Lalu, setiap titik pada buffer koordinat tersebut akan dicek unik atau tidak. Jika unik, buffer akan dicek pada proses berikutnya.

   b. Mencari buffer paling optimal

      Untuk mencari buffer yang optimal, buffer koordinat akan diubah menjadi buffer token. Selanjutnya, setiap sekuens akan dicek apakah sekuens merupakan sub-buffer (bagian dari buffer). Jika sekuens merupakan sub-buffer, buffer tersebut akan mendapatkan bonus. Setiap bonus yang didapat akan diakumulasikan. Total bonus tersebut akan digunakan mencari buffer mana yang paling optimal. Buffer yang mendapat bonus terbesar akan menjadi buffer terbaik. Selain itu, buffer yang paling optimal adalah buffer yang terkecil atau terpendek.
3. Output

   Pada bagian ini akan ditampilkan solusi paling optimal yang didapatkan. Bonus dari buffer, buffer token, dan buffer koordinat akan ditampilkan pada layar CLI. Pengguna juga bisa memilih untuk menyimpan solusi ke dalam file (.txt).

BAB 3
Source Program

Pada tugas kecil ini, bahasa pemrograman yang saya pilih adalah python. Berikut ini adalah program python Penyelesaian Cyberpunk 2077 Breach Protocol dengan Algoritma Brute Force.

```
import random
import time
import os

### Input data
## Pilih metode input
print("Metode input:")
print("1. Input file (.txt)")
print("2. Input CLI")
print("Pilih metode:" )
opsi = int(input())
while(opsi != 1 and opsi != 2):
    print("Masukkan 1 atau 2!")
    opsi = int(input())
print("=======================================")
if(opsi == 1):
    # Input lewat teks
    print("Masukkan path file dengan lengkap!")
    print("Contoh:")
    print("C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\src\input.txt")
    print("Path file: ")
    namaFile = str(input())
    while(not os.path.isfile(namaFile)):
        print("FILE TIDAK DITEMUKAN!!!")
        print("=======================================")
        print("Masukkan path file dengan lengkap!")
        print("Contoh:")
        print("C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\test\input.txt")
        print("Path file: ")
        namaFile = str(input())
    print("Membaca file ...")
    test = ""
    f = open(namaFile, "r")
    for x in f:
        test += x
    f.close()
    test = test.split("\n")
    nBuffer = int(test[0])
    test.remove(str(nBuffer))
```

```
    sizeMatrix = str(test[0])
    test.remove(str(sizeMatrix))
    sizeMatrix = sizeMatrix.split(" ")
    xMatrix = int(sizeMatrix[0])
    yMatrix = int(sizeMatrix[1])

    mainMatrix = []
    for i in range(yMatrix):
        temp = test[0]
        test.remove(temp)
        temp = temp.split(" ")
        mainMatrix.append(temp)

    nSeq = int(test[0])
    test.remove(str(nSeq))

    seqs = []
    bonus = []
    for i in range(nSeq):
        temp = test[0]
        test.remove(temp)
        temp = temp.split(" ")
        seqs.append(temp)
        temp = test[0]
        test.remove(temp)
        bonus.append(int(temp))
```

```
else:
    # Input dari CLI
    print("Masukkan command dengan format:")
    print("jumlah_token_unik")
    print("token")
    print("ukuran_buffer")
    print("ukuran_matriks")
    print("jumlah_sekuens")
    print("ukuran_maksimal_sekuens")
    print("")
    print("Contoh:\n5\nBD 1C 7A 55 E9\n7\n6 6\n3\n4")
    print("")
    nToken = int(input())
    tokens = str(input())
    tokens = tokens.split(" ")
    nBuffer = int(input())
    sizeMatrix = str(input())
    sizeMatrix = sizeMatrix.split(" ")
    xMatrix = int(sizeMatrix[0])
    yMatrix = int(sizeMatrix[1])
    nSeq = int(input())
    lenSeq = int(input())
    mainMatrix = []
    for i in range(yMatrix):
        temp = []
        for j in range(xMatrix):
            temp.append(random.choice(tokens))
        mainMatrix.append(temp)
    seqs = []
    bonus = []
    for i in range(nSeq):
        temp = []
        for j in range(random.randint(2,lenSeq)):
            temp.append(random.choice(tokens))
        seqs.append(temp)
        bonus.append(random.randint(-100,100))
    print("Matriks: ")
    for i in range(yMatrix):
        print(" ".join(mainMatrix[i]))
    print("Sequences beserta bonusnya:")
    for i in range(nSeq):
        print(" ".join(seqs[i]))
        print(bonus[i])
    print("")
    # print("debug")
```

```
resultPoints = []
resultTokens = []
resultBonus = 0
### Fungsi-fungsi

def convertPoint(arr):
    # merubah buffer translasi menjadi point buffer
    temp = [0,0]
    points = []
    for i in range(len(arr)):
        if(i % 2 == 0):
            temp[0] += arr[i]
            temp[0] %= xMatrix
        else:
            temp[1] += arr[i]
            temp[1] %= yMatrix
        points.append(temp.copy()) # biar temp di points tidak diganggu
    return points

def isCorrectBuffer(buffer):
    # mengecek apakah points buffer adalah sebuah set
    tempBuffer = []
    for i in range(len(buffer)):
        if(buffer[i] in tempBuffer):
            pass
        else:
            tempBuffer.append(buffer[i])
    return len(buffer) == len(tempBuffer)

def convertBuffer(arr):
    # mengubah points buffer menjadi token buffer
    tokenBuffer = []
    for point in (arr):
        tokenBuffer.append(mainMatrix[point[1]][point[0]])
    return tokenBuffer

def findBonus(seqs, bonus, bufferPoints):
    result = 0
    for i in range(len(seqs)):
        if(subBuffer(seqs[i], bufferPoints)):
            result += bonus[i]
    return result
```

```python
def subBuffer(sub, main):
    # check sebuah seq adalah sub-buffer (bagian dari buffer)
    sub = " ".join(sub)
    main = " ".join(main)
    return sub in main

def makeBuffer(digit, lenBuffer, buffer, slot):
    # membuat buffer
    if digit == lenBuffer:
        points = convertPoint(buffer)
        if(isCorrectBuffer(points)):
            tokens = convertBuffer(points)
            tempBonus = findBonus(seqs, bonus, tokens)
            # print(tempBonus)
            global resultBonus
            global resultPoints
            global resultTokens
            if(tempBonus > resultBonus):
                resultPoints = points
                resultTokens = tokens
                resultBonus = tempBonus
    else:
        if (digit == 0):
            for i in range(slot[digit]):
                buffer[digit] = i
                makeBuffer(digit + 1, lenBuffer, buffer,slot)
        else:
            for i in range(1,slot[digit]):
                buffer[digit] = i
                makeBuffer(digit + 1, lenBuffer, buffer,slot)
```

```python
### Memproses fungsi dan output
start = time.time() # Waktu awal
for x in range(nBuffer):
    buffer = [0] * (x+1)  # Initialize loop counters
    slot = []
    for i in range(x+1):
        if(i % 2 == 0):
            slot.append(xMatrix)
        else:
            slot.append(yMatrix)

    makeBuffer(0, x+1, buffer, slot)
end = time.time() # waktu akhir

# Output CLI
print("Result: ")
print(resultBonus)
print(" ".join(resultTokens))
for i in range(len(resultPoints)):
    temp = resultPoints[i].copy()
    temp[0] += 1
    temp[1] += 1
    print(temp)
print("Time: ",(end-start) * 10**3, "ms")
```

```
# Output file
print("Apakah ingin menyimpan solusi? (y/n)")
keluar = str(input())
while(keluar != 'y' and keluar != 'n'):
    print("Masukkan y atau n!")
    keluar = str(input())
if(keluar == 'y'):
    fileOut = open("hasil.txt", "w")
    if(opsi == 2):
        fileOut.write("Matriks: ")
        fileOut.write("\n")
        for i in range(yMatrix):
            fileOut.write(" ".join(mainMatrix[i]))
            fileOut.write("\n")
        fileOut.write("Sequences beserta bonusnya:")
        fileOut.write("\n")
        for i in range(nSeq):
            fileOut.write(" ".join(seqs[i]))
            fileOut.write("\n")
            fileOut.write(str(bonus[i]))
            fileOut.write("\n")
    fileOut.write("")
    fileOut.write("Result: ")
    fileOut.write("\n")
    fileOut.write(str(resultBonus))
    fileOut.write("\n")
    fileOut.write(" ".join(resultTokens))
    fileOut.write("\n")
    for i in range(len(resultPoints)):
        temp = resultPoints[i].copy()
        temp[0] += 1
        temp[1] += 1
        fileOut.write(str(temp))
        fileOut.write("\n")
    duration = "Time: " + str((end-start) * 10**3) + " ms"
    fileOut.write(duration)
    fileOut.close()
    print("File disimpan dengan nama hasil.txt")
### Exit
print("Press enter to exit")
input() #agar program tidak tertutup

# Notes:
# 1. Ada beberapa input dari user yang dianggap benar
```

BAB 4
Uji Coba

Berikut ini adalah kasus uji coba yang digunakan beserta output yang dihasilkan dari program:

1. Uji Coba 1

Tucil1_13522054 > test > 1.txt

```
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
```

```
Metode input:
1. Input file (.txt)
2. Input CLI
Pilih metode:
1
=====================================
Masukkan path file dengan lengkap!
Contoh:
C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\src\input.txt
Path file:
C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\test\1.txt
Membaca file ...
Result:
50
7A BD 7A BD 1C BD 55
[1, 1]
[1, 4]
[3, 4]
[3, 5]
[6, 5]
[6, 3]
[1, 3]
Time:  457.0755958557129 ms
Apakah ingin menyimpan solusi? (y/n)
y
File disimpan dengan nama hasil.txt
Press enter to exit
```

Tucil1_13522054 > bin > hasil.txt

```
Result:
50
7A BD 7A BD 1C BD 55
[1, 1]
[1, 4]
[3, 4]
[3, 5]
[6, 5]
[6, 3]
[1, 3]
Time: 457.0755958557129 ms
```

2. Uji Coba 2

```
Metode input:
1. Input file (.txt)
2. Input CLI
Pilih metode:
2
=====================================
Masukkan command dengan format:
jumlah_token_unik
token
ukuran_buffer
ukuran_matriks
jumlah_sekuens
ukuran_maksimal_sekuens

Contoh:
5
BD 1C 7A 55 E9
7
6 6
3
4

5
BD 1C 7A 55 E9
7
6 6
3
4
Matriks:
E9 BD 7A E9 7A 55
1C 1C 55 55 55 E9
55 BD 55 E9 BD 1C
1C 1C 1C 55 55 55
E9 BD 55 55 1C 55
7A 55 1C 55 E9 7A
Sequences beserta bonusnya:
BD E9 55 BD
20
55 7A E9 E9
49
E9 E9 55 7A
6
```

```
Result:
55
E9 E9 55 7A E9 E9
[1, 1]
[1, 5]
[3, 5]
[3, 1]
[4, 1]
[4, 3]
Time:  519.6926593780518 ms
Apakah ingin menyimpan solusi? (y/n)
n
Press enter to exit
```

3. Uji Coba 3

Tucil1_13522054 > test > 2.txt

```
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
2
7A 7A 7A 7A 7A 7A 7A
15
1C 7A 7A 7A 7A 7A 7A
30
```

```
Metode input:
1. Input file (.txt)
2. Input CLI
Pilih metode:
1
=====================================
Masukkan path file dengan lengkap!
Contoh:
C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\src\input.txt
Path file:
C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\test\2.txt
Membaca file ...
Result:
0

Time:  451.45559310913086 ms
Apakah ingin menyimpan solusi? (y/n)
y
File disimpan dengan nama hasil.txt
Press enter to exit
```

```
Tucil1_13522054 > bin > hasil.txt
1   Result:
2   0
3
4   Time: 451.45559310913086 ms
```

4. Uji Coba 4

```
Metode input:
1. Input file (.txt)
2. Input CLI
Pilih metode:
2
=====================================
Masukkan command dengan format:
jumlah_token_unik
token
ukuran_buffer
ukuran_matriks
jumlah_sekuens
ukuran_maksimal_sekuens

Contoh:
5
BD 1C 7A 55 E9
7
6 6
3
4

7
r1 T1 op 00 GG g8 fx
8
8 6
4
5
Matriks:
g8 00 op r1 T1 GG op GG
op g8 g8 00 op T1 g8 T1
fx fx fx g8 T1 op GG GG
g8 g8 00 GG GG 00 g8 op
g8 op 00 00 r1 fx T1 fx
fx op op r1 fx 00 fx r1
Sequences beserta bonusnya:
op fx T1 r1 00
-80
T1 r1 fx T1 T1
35
r1 g8 op fx r1
71
r1 op g8
-8
```

```
Result:
71
r1 g8 op fx r1
[4, 1]
[4, 3]
[6, 3]
[6, 5]
[5, 5]
Time:  9561.644554138184 ms
Apakah ingin menyimpan solusi? (y/n)
n
Press enter to exit
```

5. Uji Coba 5

Tucil1_13522054 > test > 3.txt

```
10
9 6
7A 55 E9 E9 1C 55 x8 55 BD
55 7A le 7A E9 55 1C 55 1C
BD x8 1C 55 E9 BD E9 7A x8
le 1C 7A 1C le BD 7A le 7A
BD 55 BD 7A 1C 1C le BD 55
1C 55 x8 7A 55 7A le x8 7A
5
7A 7A 7A 7A 7A 7A 7A
50
1C 7A
-10
7A 7A x8 55 BD 7A 7A le BD
40
1C 7A le 7A 7A x8 55 BD 7A
30
7A x8 le BD
20
```

```
Metode input:
1. Input file (.txt)
2. Input CLI
Pilih metode:
1
=====================================
Masukkan path file dengan lengkap!
Contoh:
C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\src\input.txt
Path file:
C:\Kuliah\Tingkat 2\Semester 4\Stima\Tucil1_13522054\test\3.txt
Membaca file ...
Result:
40
E9 7A 7A x8 55 BD 7A 7A le BD
[3, 1]
[3, 4]
[7, 4]
[7, 1]
[8, 1]
[8, 5]
[4, 5]
[4, 2]
[3, 2]
[3, 5]
Time:  1535172.3232269287 ms
Apakah ingin menyimpan solusi? (y/n)
n
Press enter to exit
```

6. Uji Coba 6

```
Metode input:
1. Input file (.txt)
2. Input CLI
Pilih metode:
2
=====================================
Masukkan command dengan format:
jumlah_token_unik
token
ukuran_buffer
ukuran_matriks
jumlah_sekuens
ukuran_maksimal_sekuens

Contoh:
5
BD 1C 7A 55 E9
7
6 6
3
4
```

```
8
aB 76 CD o7 ef I5 xY He
9
7 9
5
6
Matriks:
o7 ef He I5 I5 CD I5
I5 He CD o7 I5 ef xY
CD CD xY xY o7 aB He
xY He CD He He CD xY
I5 aB He I5 He 76 CD
aB 76 I5 aB I5 xY CD
ef o7 76 aB aB aB ef
aB 76 ef aB 76 xY xY
CD aB aB 76 o7 ef aB
Sequences beserta bonusnya:
aB 76
-7
I5 ef 76 CD
-72
ef 76 I5 ef ef o7
23
76 xY He CD I5 I5
74
xY 76 He CD
-34
```

```
Result:
74
ef 76 xY He CD I5 I5
[2, 1]
[2, 8]
[7, 8]
[7, 3]
[1, 3]
[1, 5]
[4, 5]
Time:  194119.92740631104 ms
Apakah ingin menyimpan solusi? (y/n)
y
File disimpan dengan nama hasil.txt
Press enter to exit
```

Tucil1_13522054 > bin > hasil.txt

```
Matriks:
o7 ef He I5 I5 CD I5
I5 He CD o7 I5 ef xY
CD CD xY xY o7 aB He
xY He CD He He CD xY
I5 aB He I5 He 76 CD
aB 76 I5 aB I5 xY CD
ef o7 76 aB aB aB ef
aB 76 ef aB 76 xY xY
CD aB aB 76 o7 ef aB
Sequences beserta bonusnya:
aB 76
-7
I5 ef 76 CD
-72
ef 76 I5 ef ef o7
23
76 xY He CD I5 I5
74
xY 76 He CD
-34
Result:
74
ef 76 xY He CD I5 I5
[2, 1]
[2, 8]
[7, 8]
[7, 3]
[1, 3]
[1, 5]
[4, 5]
Time: 194119.92740631104 ms
```

BAB 5
Pranala

Link repository dari Tugas Kecil 1 IF 2211 Strategi Algoritma Benjamin Sihombing, https://github.com/Bbennn/Tucil1_13522054.

BAB 6
Lampiran

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		☑