

TUGAS KECIL 1
PENYELESAIAN CRYPTARITHMETICS DENGAN ALGORITMA
BRUTE FORCE

IF2211 STRATEGI ALGORITMA



Oleh

RIZKY ANGGITA S SIREGAR

13519132

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2020

ALGORITMA BRUTE FORCE

Setelah menginput nama file yang berisi kata-kata yang akan dicari solusinya, maka selanjutnya proses pencarian solusi terbagi menjadi beberapa tahap :

1. Persiapan

Kata-kata operand yang terdapat pada file, setiap huruf yang unik akan dimasukkan ke sebuah array of unique character. Misal pada inputan SEND + MORE = MONEY, maka kata SEND dan MORE akan dipecah menjadi karakter-karakter, dan karakter yang unik akan dimasukkan ke array of unique character.

Kata SEND dan MORE memiliki 7 huruf yang unik dan kemudian array of unique character akan berisi ['S', 'E', 'N', 'D', 'M', 'O', 'R']. Kemudian dibuat sebuah array of number yang isi dan panjangnya akan disesuaikan mengikuti array of unique character. Pada contoh di atas, maka akan terbentuk sebuah array of number yang berisi [0, 1, 2, 3, 4, 5, 6]. Setiap elemen pada array of number berkorespondensi langsung dengan array of unique character, seperti di bawah ini.

['S', 'E', 'N', 'D', 'M', 'O', 'R']
[0, 1, 2, 3, 4, 5, 6]
Ket: S=0, E=1, N=2, dst

Karena kata pertama pada setiap huruf tidak boleh bernilai 0, maka dilakukan *swap* antara elemen pertama dan elemen kedua, sehingga array of number menjadi [1, 0, 2, 3, 4, 5, 6]. Sehingga sampai tahap ini, kita sudah memiliki array of unique character dan array of number yang saling berhubungan.

2. Proses Brute Force

Tahap selanjutnya ialah menemukan penjumlahan yang sesuai dan tidak melanggar aturan yang sudah dispesifikasikan. Pada algoritma yang saya buat, program akan mencari solusi yang pertama kali ditemukan atau mengeluarkan output tidak ada solusi jika tidak ada solusi yang memenuhi.

Pada setiap iterasi, akan dilakukan percobaan secara terus menerus hingga didapatkan solusi yang memenuhi. Misalnya pada contoh SEND + MORE = MONEY, dengan menggunakan korespondensi antara array of unique character dan array of number, maka pada iterasi pertama akan dilakukan percobaan

-Percobaan pertama

S E N D = 1 0 2 3

M O R E = 4 5 6 0

SEND + MORE = 1023 + 4560 = 5583

Angka 5583 memiliki jumlah digit 4, hal ini tidak sesuai dengan panjang kata hasil MONEY yang memiliki panjang 5.

Kemudian akan dilakukan percobaan berikutnya, yaitu menambahkan satu angka pada array of number. Pada saat ini, array of number berisi [1, 0, 2, 3, 4, 5, 6] yang jika diolah akan menjadi 1023456. Kemudian dicoba kemungkinan berikutnya, yaitu $1023456 + 1 = 1023457$, yang jika diolah kembali menjadi array of number [1, 0, 2, 3, 4, 5, 7].

- Percobaan Kedua

SEND = 1 0 2 3

MORE = 4 5 7 0

SEND + MORE = 1023 + 4570 = 5593 (masih salah karena jumlah digit bilangan hasil < panjang kata hasil)

Kemudian begitu seterusnya, bilangan dari array of number akan terus meningkat, dari 1023457, 1023458, 1023459, 1023460, dst hingga ditemukan kecocokan. Pada setiap iterasi akan dilakukan beberapa pengecekan, yaitu

- Huruf pertama dari sebuah kata tidak boleh berkorespondensi dengan angka 0
- Huruf yang sama harus memiliki angka korespondensi yang sama
- Huruf yang berbeda tidak boleh memiliki angka korespondensi yang sama
- Jumlah digit bilangan hasil harus sama dengan panjang kata hasil

Pada contoh SEND + MORE = MONEY, ditemukan solusi $9567 + 1085 = 10652$. Maka pada kondisi ini, loop akan berhenti dan array of character dan array of number berisi

['S', 'E', 'N', 'D', 'M', 'O', 'R']

[9, 5, 6, 7, 1, 0, 8]

Ket: S=9, E=5, N=6, dst

Kemudian kata hasil yaitu MONEY akan berkorespondensi dengan hasil penjumlahan 10652 sebagai berikut:

['M', 'O', 'N', 'E', 'Y']

[1, 0, 6, 5, 2]

Pada kasus tidak adanya solusi yang dapat ditemukan, program akan berhenti pada angka $10^{(\text{banyaknya huruf unik}) - 1}$. Pada kasus di atas, banyaknya huruf unik adalah 7, maka program akan berhenti pada percobaan $10^7 - 1 = 9999999$. Kondisi tersebut tercapai ketika array of number berisi

[9, 9, 9, 9, 9, 9, 9]

Jika sampai kondisi ini tidak ditemukan solusi, maka program akan berhenti.

Source Code Program

```
# Nama      : Rizky Anggita S Siregar
# NIM       : 13519132
# Tanggal   : 25 Januari 2021
# Deskripsi : Cryptarithm Solver with Brute Force Algorithm
# TUCIL 1 STRATEGI ALGORITMA

from timeit import default_timer as timer
import os

def wordToNum(word, unique_huruf, num_huruf):
    #Mengubah dari kata menjadi angka yang akan dijumlahkan
    sum = 0
    pangkat = len(word)-1
    for i in range(len(word)):
        for j in range(len(unique_huruf)):
            if word[i]==unique_huruf[j] :
                sum = sum + (10**pangkat) * num_huruf[j]
                pangkat = pangkat - 1
    return sum

def numUnique(num_huruf):
    #Mengecek apakah ada angka yang duplikat pada sebuah bilangan
    for i in range (len(num_huruf)-1):
        # print(num_huruf[i+1:])
        if (num_huruf[i]) in num_huruf[i+1:]:
            return False
    return True

def wordUnique(word):
    #Mengecek apakah ada huruf yang duplikat di sebuah kata
    for i in range(len(word)-1):
        if(word[i] in word[i+1:]):
            return False
    return True

def arrnum_hurufToNum(num_huruf):
    # Mengubah array of number to number
    sum = 0
    pangkat = len(num_huruf)-1
    for i in range(len(num_huruf)):
        sum = sum + (10**pangkat) * num_huruf[i]
        pangkat = pangkat - 1
    return sum

def numToarrnum(curr_num):
    # Mengubah number to array of number
    temp = str(curr_num)
    arr_temp = [int(a) for a in temp ]
    return(arr_temp)

#-----/
# MAIN PROGRAM
```

```

print("\n\nCryptarithms Solver with Brute Force Algorithm\n\n")

# PROSES TXT
print("Contoh input nama file: input1.txt")
nama_file = input("Masukkan nama file: ")
start = timer()
a = os.path.abspath(os.curdir)

if os.name=='nt':
    file_path = os.path.join("../test", nama_file)
else:
    file_path = os.path.join(a, "test", nama_file)

input_file = open(file_path, "r")

isi_file = input_file.readlines()

jumlah_operand = len(isi_file)-2          #Jumlah kata yang akan dioperasikan

array_operand = [i.rstrip("\n") for i in isi_file] #Mengambil operand sekaligus menghapus newline character
array_operand = array_operand[:-2] # Buang 2 line terakhir, yaitu garis dan hasil operasi
array_operand[-1] = array_operand[-1].rstrip("+") #Menghapus char +

kata_hasil = isi_file[-1]

# Membentuk array huruf yang unik
unique_huruf = []
for i in range(jumlah_operand):
    for j in range(len(array_operand[i])):
        if not(array_operand[i][j]) in unique_huruf:
            unique_huruf.append(array_operand[i][j])

# Membentuk kemungkinan permutasi pertama kali
num_huruf = [i for i in range(len(unique_huruf))]

#Karena huruf pertama tidak boleh 0, maka di swap dengan elemen ke-2
num_huruf[0], num_huruf[1] = num_huruf[1], num_huruf[0]

found = False
end = False
iterate = 0
awal = arrnum_hurufToNum(num_huruf)

while (not(found) and not(end)):
    arr_angka_operand = []
    summ = 0
    for i in range(jumlah_operand):
        temp = wordToNum(array_operand[i], unique_huruf, num_huruf)
        arr_angka_operand.append(temp)
        summ = summ + temp

    summ = str(summ)
    # print(num_huruf)
    finished = True

```

```

if (len(summ) < len(kata_hasil)) :
    finished = False
    # print("Tes") #continue, karna yang dihasilkan masih kurang

elif len(summ)==len(kata_hasil):
    #Terbagi dua kasus, jika terdapat huruf duplikat pada kata_hasil
    atau tidak

    if (wordUnique(kata_hasil)):
        #TIDAK terdapat huruf duplikat pada kata_hasil

        if (not(numUnique(summ)) or not(numUnique(num_huruf))):
            finished = False

    else:
        #Terdapat huruf duplikat pada kata_hasil
        if (numUnique(summ)):
            finished = False
        else:
            #Mengecek dua huruf yang sama tetapi berbeda angka yang
            berkorespondennya
            for i in range (len(kata_hasil)):
                for j in range(len(kata_hasil)):
                    if kata_hasil[i]==kata_hasil[j]:
                        if(int(summ[i]) != int(summ[j])) ):
                            finished = False
                            break
                    if(not(finished)):
                        break

            if (not(numUnique(num_huruf))):
                finished = False

        for i in range (len(kata_hasil)):
            for j in range(len(num_huruf)):
                if int(summ[i])==num_huruf[j]:          #jika angka sama
                    if(kata_hasil[i]!=unique_huruf[j]): # jika hurufnya
berbeda, maka belum benar
                        finished = False
                        break
                    if kata_hasil[i]==unique_huruf[j]: # jika hurufnya sama
                        if(int(summ[i]) != num_huruf[j]): #jika angkanya
berbeda, maka belum benar
                            finished = False
                            break
            if (not(finished)):
                break

        # Mengecek awalan kata tidak boleh 0
        for i in range(len(arr_angka_operand)):
            temp = str(arr_angka_operand[i])
            if(temp[0]==0):
                finished = False
                break

        if(str(summ)[0]==0):

```

```

        finished = False

    if finished and (len(summ) == len(kata_hasil)):
        found = True
        end = True
    else:
        #Increment
        iterate += 1
        curr_num = arrnum_hurufToNum(num_huruf) + 1
        num_huruf = numToarrnum(curr_num)

        maks = 10**((len(unique_huruf)) - 1)
        if (curr_num==maks):
            end = True
            found = False

if(end):

    print("\n\nDONE\n")
    print("Array of Unique Character: ", unique_huruf)
    print("Array of Number: ", num_huruf)
    print("\n")

    print("Input\n")
    for i in range(len(isi_file)):
        print(isi_file[i], end="")

    print("\n\nSolusi\n")

    if found:
        for i in range(len(arr_angka_operand)):
            print(arr_angka_operand[i])
            print("----- +")
            print(summ)
            print()
        else:
            print("Solusi tidak ada!\n")

    end = timer()
    print("Time elapsed : ", end-start, end=" seconds\n")

    #Perhitungan awal dimulai dari 1023.....x, dengan x adalah n-1
    (n=jumlah huruf unik pada input kata)
    print("Jumlah total tes: ", iterate+awal, "-", awal, "=", iterate)
    print()

```

Screenshot

```
Contoh input nama file: input1.txt
Masukkan nama file: input1.txt

DONE

Array of Unique Character: ['S', 'E', 'N', 'D', 'M', 'O', 'R']
Array of Number: [9, 5, 6, 7, 1, 0, 8]

Input

SEND
MORE+
-----
MONEY

Solusi

9567
1085
----- +
10652

Time elapsed : 190.72285313700002 seconds
Jumlah total tes: 9567108 - 1023456 = 8543652
```

Cryptarithms Solver with Brute Force Algorithm

```
Contoh input nama file: input1.txt
Masukkan nama file: input2.txt

DONE

Array of Unique Character: ['S', 'U', 'N', 'F']
Array of Number: [1, 2, 3, 9]

Input

SUN
FUN+
-----
SWIM

Solusi

123
923
----- +
1046

Time elapsed : 0.0037982009999999455 seconds
Jumlah total tes: 1239 - 1023 = 216
```


Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input3.txt

DONE

Array of Unique Character: ['J', 'U', 'N', 'E', 'L', 'Y']
Array of Number: [5, 4, 8, 6, 3, 7]

Input

JUNE
JULY+

APRIL

Solusi

5486
5437
----- +
10923

Time elapsed : 8.082314032000001 seconds
Jumlah total tes: 548637 - 102345 = 446292

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input4.txt

DONE

Array of Unique Character: ['M', 'E', 'O', 'F', 'R']
Array of Number: [8, 4, 5, 7, 3]

Input

MEMO
FROM+

HOMER

Solusi

8485
7358
----- +
15843

Time elapsed : 1.469323825 seconds
Jumlah total tes: 84573 - 10234 = 74339

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input5.txt

DONE

Array of Unique Character: ['H', 'E', 'R', 'S']
Array of Number: [9, 4, 5, 8]

Input

HERE
SHE+

COMES

Solusi

9454
894
----- +
10348

Time elapsed : 0.13822576399999997 seconds
Jumlah total tes: 9458 - 1023 = 8435

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input6.txt

DONE

Array of Unique Character: ['C', 'O', 'A', 'L']
Array of Number: [8, 1, 6, 0]

Input

COCA
COLA+

OASIS

Solusi

8186
8106
----- +
16292

Time elapsed : 0.15428550099999994 seconds
Jumlah total tes: 8160 - 1023 = 7137

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input7.txt

DONE

Array of Unique Character: ['C', 'R', 'O', 'S', 'A', 'D']
Array of Number: [9, 6, 2, 3, 5, 1]

Input

CROSS
ROADS+

DANGER

Solusi

96233
62513
----- +
158746

Time elapsed : 20.511322243 seconds
Jumlah total tes: 962351 - 102345 = 860006

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input8.txt

DONE

Array of Unique Character: ['N', 'O', 'G', 'U']
Array of Number: [8, 7, 9, 0]

Input

NO
GUN
NO+

HUNT

Solusi

87
908
87
----- +
1082

Time elapsed : 0.19165715399999996 seconds
Jumlah total tes: 8790 - 1023 = 7767

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input9.txt

DONE

Array of Unique Character: ['F', 'O', 'R', 'T', 'Y', 'E', 'N']
Array of Number: [2, 9, 7, 8, 6, 5, 0]

Input

FORTY
TEN
TEN+

SIXTY

Solusi

29786
850
850
----- +
31486

Time elapsed : 57.691369195 seconds
Jumlah total tes: 2978650 - 1023456 = 1955194

Cryptarithms Solver with Brute Force Algorithm

Contoh input nama file: input1.txt
Masukkan nama file: input10.txt

DONE

Array of Unique Character: ['A', 'B', 'C', 'E']
Array of Number: [9, 9, 9, 9]

Input

ABC
EC+

AIJK

Solusi

Solusi tidak ada!

Time elapsed : 0.127748280999999969 seconds
Jumlah total tes: 9999 - 1023 = 8976

ALAMAT DRIVE

<https://drive.google.com/drive/folders/1B6ANurCGmG2-OoqTGZl81pKvqOttDzuQ?usp=sharing>

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan (no syntax error)	✓	
2. Program berhasil running	✓	
3. Program dapat membaca file masukan dan menuliskan luaran.	✓	
4. Solusi cryptarithmic hanya benar untuk persoalan cryptarithmic dengan dua buah operand		✓
5. Solusi cryptarithmic benar untuk persoalan cryptarithmic untuk lebih dari dua buah operand.	✓	