LAPORAN TUGAS KECIL 1

Mata Kuliah Strategi Algoritma IF 2211
Dosen Pengampu : Dr. Nur Ulfa Maulidevi, S.T., M.Sc.

Penyelesaian Cryptarithmetic dengan Brute Force

Disusun Oleh :

Jesson Gosal Yo – 13519079

13519079@std.stei.itb.ac.id

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

A. Algoritma Brute Force

Langkah-langkah untuk mencari solusi dalam permasalahan cryptarithmetic dengan metode bruteforce secara umum adalah dengan melakukan subtitusi terhadap huruf-huruf dalam puzzle. Misal ada r buah huruf dan mengasumsikan bahwa huruf yang berbeda pasti merepresentasikan angka yang berbeda maka hal ini akan sama dengan permutasi r dari 10 (angka 0 sampai 9). Dari hasil-hasil subtitusi ini akan diperiksa solusi mana yang menghasilkan ekspresi penjumlahan yang benar.

Pendekatan penulis dalam mengimplementasi permutasi r dari 10 adalah dengan melakukan permutasi dua kali yaitu terhadap array of boolean yang melambangkan r buah angka yang akan dipilih serta permutasi terhadap r buah angka yang dipilih tersebut. Algoritma yang diimplementasi penulis untuk melakukan permutasi adalah next_permutation yaitu permutasi yang diurutkan berdasarkan nilai lexicographicalnya. Algoritma ini dipilih karena program tidak perlu menyimpan semua kemungkinan permutasi dan juga tidak perlu mencari semua kemungkinan permutasi sehingga algoritma ini lebih efisien secara ruang dan sedikit lebih efisien dalam segi waktu jika dibandingkan dengan algoritma yang menghitung semua kemungkinan permutasi terlebih dahulu.

Penulis mengasumsikan huruf dalam file ditulis menggunakan huruf kapital, namun jika file ditulis menggunakan huruf kecil, hal ini dapat dengan mudah ditangani dengan mengubah huruf kecil ke huruf besar terlebih dahulu.Secara sederhana, cara kerja program adalah sebagai berikut:

1. Membaca file dan menyimpan operand dalam vector sambil menghitung jumlah variabel unik , untuk kejelasan penjelasan akan dimisalkan dengan r, variabel yang dimaksud adalah huruf-huruf yang ada dalam puzzle. Perhitungan r akan dicatat oleh array statis variables berukuran 26 yang setiap indeksnya melambangkan huruf ‘A’ hingga ‘Z’ dan nilai pada indeks adalah besar nilai yang akan disubtitusikan ke huruf tersebut. Nilai -1 melambangkan bahwa tidak ada kemunculan huruf tertentu pada puzzle.

2. Menginisialisasi array boolean dengan panjang konstan 10 yang merepresentasikan angka sesuai dengan indeksnya. Beberapa elemen dari array ini akan diinisialisasi dengan *true,* tepatnya sejumlah r. Contohnya [1,1,1,0,0,0,0,0,0,1] akan merepresentasikan ada 4 variabel unik dengan percobaan pilihan angka 0,1,2, dan 9.

3. Selain array boolean, program juga akan menginisialisasi array combinations. dengan panjang sebesar r. Array ini akan digunakan untuk menyimpan r buah angka yang didapat dari array boolean. Array ini disebut combinations karena nilai yang didapat saat pengisian pertama adalah *sorted permutation* yang ekuivalen dengan kombinasi.

4. Kemudian program akan menginisialisasi array boolean dengan r buah *true* yang dilambangkan dengan angka 1 sedemikian cara sehingga nilai lexicographicalnya paling kecil.

5. Setelah itu program akan menyimpan r buah angka yang didapat dari array boolean ke array kombinasi sekaligus mencatatnya dalam array variables

6. Nilai-nilai array akan disubtitusikan ke puzzle. Jika hasil subtitusi valid maka program berhenti dan hasil akan dicetak ke layar. Jika hasil subtitusi belum benar, program akan melakukan next_permute terhadap array kombinasi dan mencatatnya lagi ke array variables.

7. Langkah 6 diulang terus hingga semua kemungkinan permutasi array kombinasi dicapai atau jawaban didapat.

8. Jika langkah 6 sudah diulang hingga kemungkinan permutasi habis, maka sekarang array boolean akan dipermutasi dan program kembali ke langkah 5.

9. Langkah-langkah di atas akan diulang terus hingga jawaban tercapai atau kedua array yaitu array boolean dan array combinatorics sudah mencapai semua kemungkinan permutasinya.

10. Jika jawaban ditemukan, maka program akan mencetak jawaban. Sedangkan jika jawaban tidak ditemukan setelah semua kemungkinan sudah dicoba, maka program akan menceetak “No solutions found”.

11. Terakhir program akan mencetak jumlah tes dan waktu yang diperlukan untuk melakukan brute force.

B. Source Code Program

```
#include <iostream>
#include <fstream>
#include <chrono>
#include <vector>
#include <array>
#include <iomanip>

using namespace std;
using namespace std::chrono;

int power(int a,int b){
    int total = 1;
    for(int i=0;i<b;i++){
        total*=a;
    }
    return total;
}

void reverse(int arr[],int start,int end){
    int temp;
    if(start<end){
        while(start<end){
            temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;
            start++;
            end--;
```

```
        }
    }
}

int next_permute(int arr[],int size){
    int inv_point = size - 2;
    int temp;

    while(inv_point>=0 && arr[inv_point]>=arr[inv_point+1]){
        inv_point--;
    }
    if(inv_point<0){
        return 0;
    }
    for(int i=size-1;i>inv_point;--i){
        if(arr[i]>arr[inv_point]){
            temp = arr[i];
            arr[i] = arr[inv_point];
            arr[inv_point] = temp;
            break;
        }
    }
    reverse(arr,inv_point+1,size-1);
    return 1;
}

int subtituteOperandsToInt(string str,int variables[]){
    int sum = 0;
    for(int i=0;i < str.size();i++){
        sum += variables[str[i]-65]*power(10,str.size()-i-1);
    }
    return sum;
}

bool isValid(vector<string> operands,int variables[]){
    int sum=0;

    for(int i=0;i<operands.size();i++){
        if(operands[i][0] != '-' && variables[operands[i][0]-
65]==0){ // If first letter of a word is zero we assume it's false
            return false;
        }
        if(i<operands.size()-2){
            sum += subtituteOperandsToInt(operands[i],variables);
        }
    }
```

```
    return sum == subtituteOperandsToInt(operands[operands.size()-
1], variables);
}

bool permute(int combinations[],vector<string> operands,int variables[
], int &counter, int size){
    int j;
    do{
        j = 0;
        counter++;
        for (int i=0;i<26;i++)
        {
            if(variables[i]!=-1){
                variables[i] = combinations[j];
                j++;
            }
        }
        if(isValid(operands,variables)){
            return true;
        }
    } while(next_permute(combinations,size));

    return false;
}


int main(){
    fstream arith_file;
    string file_name;

    int variables[26];
    std::fill(variables,variables+(end(variables) - begin(variables)),
-1);
    int variables_count=0;
    vector<string> operands;
    printf("Input file name : ");
    cin >> file_name;
    arith_file.open(file_name, ios::in);
    if(!arith_file){
        cout << "File not found!";
    } else{
        string line;

        while(getline(arith_file,line)){
            operands.push_back(line);
            for(char &c : line){
                if(c!='-' && c!='+' && variables[(int)c-65]==-1){
                    variables[(int)c-65] = 0;
                    variables_count++;
```

```
                }
            }
        }

        bool found=false;
        int counter=0;

        int v[10] ={0}; //bitmask
        int combinations[variables_count];
        for(int i=10;i>=10-variables_count;i--){
            v[i] = 1;
        }

        int t;

        operands[operands.size()-3].pop_back(); //Removes the + sign

        auto start = high_resolution_clock::now();

        do {
            t=0;
            for (int i = 0; i < 10; ++i) {
                if (v[i]) {
                    combinations[t]=i;
                    t++;
                }
            }
            if(permute(combinations,operands,variables,counter,variabl
es_count)){
                found = true;
                break;
            }
        } while(next_permute(v,10));

        if(found){
            for(int i=0;i<operands.size()-2;i++){
                cout << operands[i] << "\n";
            }
            cout << "---------+" << "\n";
            cout << operands[operands.size()-1] << "\n";

            cout << "\n";

            for(int i=0;i<operands.size()-2;i++){
                cout << subtituteOperandsToInt(operands[i],variables)
<< "\n";
            }
            cout << "---------+" << "\n";
```

```
            cout << subtituteOperandsToInt(operands[operands.size()-
1],variables) << "\n";
        } else{
            cout << "No solutions found" << "\n";
        }

        auto stop = high_resolution_clock::now();
        auto duration = duration_cast<milliseconds>(stop - start);

        cout << "Time taken for brute force : " << duration.count() <<
 " milliseconds" << "\n";
        cout << "Jumlah tes : " << counter << "\n";

        cout << "---------------" << endl;

    } // END OF ELSE STATEMENT

    arith_file.close();
    return 0;
}
```

C. Screenshot Output

```
Input file name : a.txt
FORTY
TEN
TEN
---------+
SIXTY

29786
850
850
---------+
31486
Time taken for brute force : 349 milliseconds
Jumlah tes : 1900757
---------------
```

```
C:\Users\user\Documents\cryptarithmetic\src>a
Input file name : b.txt
SEND
MORE
---------+
MONEY

9567
1085
---------+
10652
Time taken for brute force : 184 milliseconds
Jumlah tes : 995212
---------------
```

```
C:\Users\user\Documents\cryptarithmetic\src>a
Input file name : c.txt
THREE
THREE
TWO
TWO
ONE
---------+
ELEVEN

84611
84611
803
803
391
---------+
171219
Time taken for brute force : 414 milliseconds
Jumlah tes : 1873386
---------------
```

```
Input file name : d.txt
NUMBER
NUMBER
---------+
PUZZLE

201689
201689
---------+
403378
Time taken for brute force : 389 milliseconds
Jumlah tes : 2050001
---------------
```

```
Input file name : e.txt
TILES
PUZZLES
---------+
PICTURE

91542
3077542
---------+
3169084
Time taken for brute force : 434 milliseconds
Jumlah tes : 2346083
---------------
```

```
Input file name : f.txt
FORTY
TEN
TEN
---------+
SIXTY

29786
850
850
---------+
31486
Time taken for brute force : 346 milliseconds
Jumlah tes : 1900757
---------------
```

```
Input file name : g.txt
CLOCK
TICK
TOCK
---------+
PLANET

90892
6592
6892
---------+
104376
Time taken for brute force : 348 milliseconds
Jumlah tes : 1807451
---------------
```

```
Input file name : h.txt
COCA
COLA
---------+
OASIS

8186
8106
---------+
16292
Time taken for brute force : 27 milliseconds
Jumlah tes : 127171
---------------
```

```
Input file name : i.txt
HERE
SHE
---------+
COMES

9454
894
---------+
10348
Time taken for brute force : 78 milliseconds
Jumlah tes : 399223
---------------
```

D. Link Alamat Drive

https://drive.google.com/drive/u/1/folders/1MFkX7DoHpqM-LeVnlz-cXS9p-cZL1ZI9

E. Tabel Ceklist:

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