Laporan

Tugas Kecil 1 IF2211 Strategi Algoritma

Semester II Tahun 2022/2023

Pencarian Solusi Permainan Kartu 24 dengan Algoritma *Brute Force*

Farizki Kurniawan

13521082

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2021

1. Algoritma *Brute Force* pada 24 *Solver*

Permainan kartu 24 adalah permainan dimana akan diambil 4 kartu (dengan nilai dari A, 2—10, J, Q, hingga K) dan dengan menggunakan aritmatika dasar berupa penjumlahan(+), pengurangan(-), perkalian(*), atau pembagian (/) antara nilai dari 4 kartu, akan dicari urutan ekspresi tersebut yang memberikan nilai sebesar 24. Setiap ekspresi dibedakan dengan urutan nilai, operasi aritmatik, dan pengurungannya.

Pada 24 *Solver*, terdapat 3 bagian utama yang perlu diperhatikan, yaitu:

a. Susunan angka
   Dari 4 bilangan yang didapat, dengan asumsi setiap bilangan berbeda dan dengan memerhatikan urutan, terdapat 4! = 24 susunan bilangan yang berbeda.

b. Susunan operator
   Untuk 4 bilangan, akan terdapat 3 buah operasi aritmatika agar didapatkan satu buah hasil akhir. Terdapat 4 buah operator yang mungkin dipakai, sehingga terdapat sebanyak $4^3 = 64$ susunan operator yang dapat digunakan.

c. Susunan tanda kurung
   Untuk 4 bilangan dan 3 operator, secara efektif terdapat 5 cara untuk menyusun tanda kurung agar menghasilkan urutan operasi yang berbeda-beda. Urutan tersebut adalah sebagai berikut.

   (a # b) # (c # d)
   a # ((b # c) # d)
   a # (b # (c # d))
   ((a # b) # c) # d
   (a # (b # c)) # d

   Dimana a,b,c,d adalah urutan dari bilangan dan # adalah operator aritmatika yang memungkinkan (+,-,*,/).

Dengan memerhatikan ketiga aspek di atas, program berjalan sebagai berikut:

a. Program menerima/menggenerasi input berupa 4 buah kartu.

b. Program menghasilkan semua permutasi dari keempat nilai (*exhaustive search)*. Permutasi ini dihasilkan dengan menggunakan algoritma *backtracking*, dimana pada setiap rekurensnya, kita akan memperhatikan pertukaran untuk elemen yang difokuskan (dimulai dari elemen paling kiri) dengan setiap elemen lain, kemudian memanggil fungsi yang sama, tetapi dengan fokus pertukaran pada elemen di kanan dari elemen fokus. Pemanggilan fungsi berakhir saat elemen yang difokuskan berada di paling ujung dari kumpulan elemen tersebut.

c. Program menghasilkan semua kemungkinan susunan operator (*exhaustive search)*.
   Semua susunan kemungkinan dihasilkan dengan menggunakan *triple-nested loop* dengan setiap *loop* melakukan *traversal* nilai dari 1—4, dimana setiap nilai ini berkoresponsi dengan sebuah operator. Susunan operator kemudian dikonstruksi dari ketiga nilai pada *loop* tersebut.

d. Program mengevaluasi setiap susunan bilangan dan operator pada semua kemungkinan susunan tanda kurung (*exhaustive search)*.
   Evaluasi dilakukan dengan melakukan operasi secara bertahap dari kurung yang paling dalam hingga didapatkan nilai akhir.

e. Program mengecek nilai hasil evaluasi.
   Apabila nilai dari sebuah ekspresi adalah 24, maka program akan mencatatnya dan jumlah ekspresi yang bernilai 24.

f. Program mengeluarkan *output*.
   Output program berupa jumlah ekspresi yang memenuhi, seluruh ekspresinya, kemudian waktu yang digunakan untuk menggenerasi solusi.

2. *Source Code 24 Solver*

```
#include <bits/stdc++.h>
using namespace std;
using namespace std::chrono;

const double EPS=1e-9;
set<string> permutation;
vector<string> ans;

// used to operate given two numbers and an operator
double oprt(double x1,double x2,char op){
    if(op=='+') return x1+x2;
    if(op=='-') return x1-x2;
    if(op=='*') return x1*x2;
    if(op=='/') return x1/x2;
    else return 0;
}

// change char to integer
double token(char c){
    if(c>='2'&&c<='9') return c-'0';
```

```
    if(c=='A') return 1;
    if(c=='T') return 10;
    if(c=='J') return 11;
    if(c=='Q') return 12;
    if(c=='K') return 13;
    else return -1;
}

// change char to string
string cts(char c){
    if(c=='T')return "10";
    string s="";
    return s+c;
}
// change int to string
string its(int x){
    string s;
    char c=x+'0';
    s=c;
    if(x==1)return "A";
    if(x==10)return "T";
    if(x==11)return "J";
    if(x==12)return "Q";
    if(x==13)return "K";
    else return s;
}

//operator token
string optoken(int x){
    if(x==1)return "+";
    if(x==2)return "-";
    if(x==3)return "*";
    if(x==4)return "/";
    else return 0;
}

void bf(string s1,string s2){
    //bruteforce all 5 possible ordering
    double a,b,c,d;
    char op1,op2,op3;

    a = token(s1[0]);
    b = token(s1[1]);
    c = token(s1[2]);
    d = token(s1[3]);
```

```
    op1 = s2[0];
    op2 = s2[1];
    op3 = s2[2];

    double x1,x2,x3;

    // Evaluate every possible expression,
    // If equal to 24, add to ans vector.
    // (a # b) # (c # d)
    x1 = oprt(a,b,op1);
    x2 = oprt(c,d,op3);
    x3 = oprt(x1,x2,op2);
    if(fabs(x3-24)<EPS){
    ans.push_back("("+cts(s1[0])+cts(op1)+cts(s1[1])+")"+cts(op2)+"("+cts(s1[2])+
cts(op3)+cts(s1[3])+")");
    }

    // a # ((b # c) # d)
    x1 = oprt(b,c,op2);
    x2 = oprt(x1,d,op3);
    x3 = oprt(a,x2,op1);
    if(fabs(x3-24)<EPS){
    ans.push_back(cts(s1[0])+cts(op1)+"(("+cts(s1[1])+cts(op2)+cts(s1[2])+")"+cts
(op3)+cts(s1[3])+")");
    }

    // a # (b # (c # d))
    x1 = oprt(c,d,op3);
    x2 = oprt(b,x1,op2);
    x3 = oprt(a,x2,op1);
    if (fabs(x3-24)<EPS){
    ans.push_back(cts(s1[0])+cts(op1)+"("+cts(s1[1])+cts(op2)+"("+cts(s1[2])+cts(
op3)+cts(s1[3])+"))");
    }

    // (a # (b # c)) # d
    x1 = oprt(b,c,op2);
    x2 = oprt(a,x1,op1);
    x3 = oprt(x2,d,op3);
    if (fabs(x3-24)<EPS){
    ans.push_back("("+cts(s1[0])+cts(op1)+"("+cts(s1[1])+cts(op2)+cts(s1[2])+"))"
+cts(op3)+cts(s1[3]));
    }
```

```
    //  ((a # b) # c) # d
    x1 = oprt(a,b,op1);
    x2 = oprt(x1,c,op2);
    x3 = oprt(x2,d,op3);
    if (fabs(x3-24)<EPS){
    ans.push_back("(("+cts(s1[0])+cts(op1)+cts(s1[1])+")"+cts(op2)+cts(s1[2])+")"
+cts(op3)+cts(s1[3]));
    }
}

//bruteforce all possible operator
void allOp(string s1){
    string s2 = "";
    int i,j,k;
    for(i = 1 ; i <= 4; i++){
        for(j = 1; j <= 4; j++ ){
            for(k = 1; k <= 4; k++){
                s2 = optoken(i) + optoken(j) + optoken(k);
                //bruteforce all parantheses orderings
                bf(s1,s2);
            }
        }
    }
}

//generate all permutations
void gnrt(string s,int l){
    int i;

    if(l==3) permutation.insert(s); //basis
    else{
        char temp;
        for(i=l;i<=3;i++){
            // try to swap l with all other elements to its right
            temp = s[l];
            s[l] = s[i];
            s[i] = temp;

            //recursive for next l
            gnrt(s,l+1);

            //reswap back (to backtrack)
            temp = s[l];
            s[l] = s[i];
            s[i] = temp;
```

```
        }
    }
}


int main(){
    cout <<
"      ___                                                                 "
<<endl;
    cout << "   //   ) ) //___/
/        ___       ___      //           ___       __       " <<endl;
    cout << "    ___/ / /____  /       ((   ) ) //   ) ) // ||  / / //___) )
//  ) )   " <<endl;
    cout << "  / ____/      / /         \\\\     //   / / //  || / /
//       //         " <<endl;
    cout << " / /____       / /       //   ) ) ((___/ / //   ||/ /
((____   //          " <<endl;
    cout << endl<<endl;

    cout<< "Choose input method:" <<endl;
    cout<< "1.Randomized" <<endl;
    cout<< "2.User input" <<endl;

    string cards="";
    string inp;
    getline(cin,inp);

    while(!(inp[0]=='1'||inp[0]=='2')){ //input validation
        cout<<"False input! Please input 1 or 2"<<endl;
        getline(cin,inp);
    }

    if(inp[0]=='1'){
        srand((unsigned) time(NULL)); //randomizer

        int i;
        for(i=0;i<=3;i++){
            int random= 1 + (rand()%13);
            cards+=its(random);
        }
    }
    else{
        cout<<"Please input 4 cards (A,2-10,J,Q,K)"<<endl;
        while(true){ // loop until valid input
        cards = "";
```

```
        getline(cin,inp);
        inp += "."; //add mark at the end of string

        //input validation purposes
        int it = 0;
        int cnt = 0;
        bool mustblank = false;
        bool mustzero = false;
        bool finish = false;
        bool valid = true;
        while( inp[it] != '.'){
            char cc = inp[it];
            cc = toupper(cc);
            if( isspace(cc) ){ //skip blanks
                if(mustzero) { valid = false; break;}
                mustblank = false;
            }
            else if((cc>='2'&&cc<='9') || cc=='A' || cc=='J' || cc=='Q' ||
cc=='K' ){
                if(mustblank || mustzero || cnt==4){valid=false; break;}
                cards += cts(cc);
                cnt++;
                mustblank = true;
            }
            else if(cc == '1'){
                if(mustblank || mustzero || cnt==4){valid=false; break;}
                mustzero = true;
            }
            else if(cc='0'){
                if(!mustzero || mustblank || cnt==4){valid=false; break;}
                mustblank = true;
                mustzero = false;
                cards += "T";
                cnt++;
            }
            else{
                valid = false; break;
            }
            it++;
        }
        if(valid && cnt==4)break; //if 4 cards are obtained, proceed
        else { // invalid input
        cout<<"Invalid input. Please input 4 cards in the format (A,2-10,J,Q,K)
separated by space.";
        cout<<endl;
```

```
        }
      }
    }

    cout<<"Cards chosen are:"<<endl;
    for(int i=0;i<=3;i++){
        if(cards[i]=='T')cout<<"10 ";
        else cout<<cards[i]<<" ";
    }
    cout<<endl<<endl;

    cout<<"Processing..."<<endl<<endl;
    auto start = high_resolution_clock::now();

    // generate all permutation
    gnrt(cards,0);
    // generate all operation and parentheses configuration for every permutation
    for(auto u:permutation)allOp(u);

    // all answers generated, stop the timer
    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<microseconds>(stop - start);

    //output solutions
    int sze = ans.size();
    if(sze>0){
        cout << sze << " solutions found" << endl;
        for(auto u:ans) cout<<u<<endl;
    }
    else cout<<"No solutions found"<<endl;

    //output time elapsed
    cout << "Time elapsed: "<< duration.count()/1e3 << " milliseconds" << endl <<
endl;

    cout<< "Do you want to save the solution?" <<endl;
    cout<< "1.Yes" <<endl;
    cout<< "2.No" <<endl;
    getline(cin,inp);

    while(!(inp[0]=='1'||inp[0]=='2')){
        cout<<"False input! Please input 1 or 2"<<endl;
        getline(cin,inp);
    }
```

```
    if(inp[0]=='1'){
        cout<<"Input file name: ";
        cin>>inp;
        string filename=inp+".txt";
        //create and write to new file
        ofstream File(filename);

        File << sze << " solutions found" <<endl<<endl;
        for(auto u:ans) File << u << endl;
        File << "Time elapsed: " << duration.count()/1e3 << " milliseconds" <<
endl;
        File.close();
    }

    cout<<"End of program. Press enter to close. Relaunch program to use
again."<<endl;
    getline(cin,inp);
    return 0;
}
```

3. Hasil Pengetesan Program

a. 10 10 5 5

Input

```
Please input 4 cards (A,2-10,J,Q,K)
10 10 5 5
```

Output

```
Cards chosen are:
10 10 5 5

Processing...

1 solutions found

(5*5)-(10/10)

Time elapsed: 0.488 milliseconds
```

Output pada file

```
Cards chosen are: 10 10 5 5

1 solutions found

(5*5)-(10/10)

Time elapsed: 0.488 milliseconds
```

b. 3 2 6 8 (input random)

Input

```
Please input 4 cards (A,2-10,J,Q,K)
3 2 6 8
```

Output

```
Cards chosen are:
3 2 6 8

Processing...

26 solutions found
((2+8)*3)-6
(3*(2+8))-6
((3*6)-2)+8
(3*6)-(2-8)
(3*6)+(8-2)
((3*6)+8)-2
(3*(8+2))-6
(3*(8-2))+6
6-((2-8)*3)
6-(3*(2-8))
((6*3)-2)+8
(6*3)-(2-8)
6+(3*(8-2))
(6*3)+(8-2)
((6*3)+8)-2
6+((8-2)*3)
((8+2)*3)-6
(8-2)+(3*6)
8-(2-(3*6))
((8-2)*3)+6
(8-2)+(6*3)
8-(2-(6*3))
8+((3*6)-2)
(8+(3*6))-2
8+((6*3)-2)
(8+(6*3))-2
Time elapsed: 0.472 milliseconds
```

c. 9 8 3 A

Input

```
Please input 4 cards (A,2-10,J,Q,K)
9 8 3 A
```

Output

```
Cards chosen are:
9 8 3 A

Processing...

151 solutions found
3/((9/8)-A)
(8/3)*(9*A)
((8/3)*9)*A
(8/3)*(9/A)
((8/3)*9)/A
8/((3/9)*A)
8/(3/(9*A))
(8/(3/9))*A
8/((3/9)/A)
8/(3/(9/A))
(8/(3/9))/A
(8/3)*(A*9)
(8/(3*A))*9
((8/3)*A)*9
8/((3*A)/9)
8/(3*(A/9))
8/(3/(A*9))
(8/(3*A))*9
((8/3)*A)*9
(8/3)/(A/9)
8/((3/A)/9)
(8*9)/(3*A)
8*((9/3)*A)
8*(9/(3*A))
(8*(9/3))*A
((8*9)/3)*A
(8*9)/(3/A)
8*((9/3)/A)
8*(9/(3/A))
(8*(9/3))/A
((8*9)/3)/A
(8*9)*(A/3)
8*((9*A)/3)
8*(9*(A/3))
(8*(9*A))/3
((8*9)*A)/3
(8*9)/(A*3)
8*(9/(A*3))
8*((9/A)/3)
(8*(9/A))/3
((8*9)/A)/3
8*((A/3)*9)
(8*(A/3))*9
((8*A)/3)*9
((8*A)/3)*9
(8*A)/(3/9)
8*(A/(3/9))
((8/A)/3)*9
(8/A)/(3/9)
(8*A)*(9/3)
8*((A*9)/3)
8*(A*(9/3))
(8*(A*9))/3
((8*A)*9)/3
(8*A)*(9/3)
((8/A)*9)/3
8/((A/9)*3)
8/(A/(9/3))
(8/(A/9))/3
(9/3)*(8*A)
((9/3)*8)*A
(9/3)*(8/A)
((9/3)*8)/A
9/((3/8)*A)
9/(3/(8*A))
(9/(3/8))*A
9/((3/8)/A)
9/(3/(8/A))
(9/(3/8))/A
(9/3)*(A*8)
(9/(3*A))*8
((9/3)*A)*8
9/((3*A)/8)
9/(3*(A/8))
9/(3/(A*8))
(9/(3*A))*8
((9/3)/A)*8
(9/3)/(A/8)
9/((3/A)/8)
(9*8)/(3*A)
9*((8/3)*A)
9*(8/(3*A))
(9*(8/3))*A
((9*8)/3)*A
(9*8)/(3/A)
9*((8/3)/A)
9*(8/(3/A))
(9*(8/3))/A
((9*8)/3)/A
(9*8)*(A/3)
9*((8*A)/3)
9*((8*A)/3)
(9*(8*A))/3
((9*8)*A)/3
(9*8)/(A*3)
9*(8/(A*3))
9*((8/A)/3)
9*(8/(A/3))
(9*(8/A))/3
((9*8)/A)/3
9*((A/3)*8)
(9*(A/3))*8
((9*A)/3)*8
(9*A)/(3/8)
9*(A/(3/8))
(9/(A*3))*8
9/((A*3)/8)
9/(A*(3/8))
((9/A)/3)*8
(9/A)/(3/8)
(9*A)*(8/3)
9*((A*8)/3)
9*(A*(8/3))
(9*(A*8))/3
((9*A)*8)/3
(9*A)/(8/3)
9*(A/(8/3))
(9/(A*8))/3
9/((A*8)/3)
9/(A*(8/3))
((9/A)*8)/3
9/(A/(8/3))
(9/(A/8))/3
(A/3)*(8*9)
((A/3)*8)*9
A/(3/(8*9))
(A/(3/8))*9
A/((3/8)/9)
(A/3)*(9*8)
((A/3)*9)*8
A/(3/(9*8))
(A/(3/9))*8
A/((3/9)/8)
A*((8/3)*9)
(A*(8/3))*9
((A*8)/3)*9
(A*8)/(3/9)
A*(8/(3/9))
(A*8)*(9/3)
A*((8*9)/3)
A*(8*(9/3))
(A*(8*9))/3
((A*8)*9)/3
A*((9/3)*8)
A*((9/3)*8)
(A*(9/3))*8
((A*9)/3)*8
(A*9)/(3/8)
A*(9/(3/8))
(A*9)*(8/3)
A*((9*8)/3)
A*(9*(8/3))
(A*(9*8))/3
((A*9)*8)/3
Time elapsed: 0.545 milliseconds
```

d. A A A A

Input

```
Please input 4 cards (A,2-10,J,Q,K)
A A A A
```

Output

```
Cards chosen are:
A A A A

Processing...

No solutions found

Time elapsed: 0.041 milliseconds
```

e. 5 5 5 5

Input

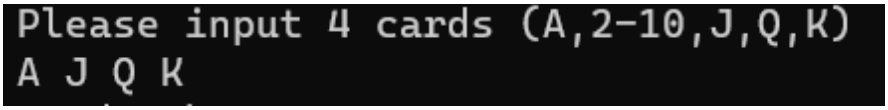
```
Please input 4 cards (A,2-10,J,Q,K)
5 5 5 5
```

Output

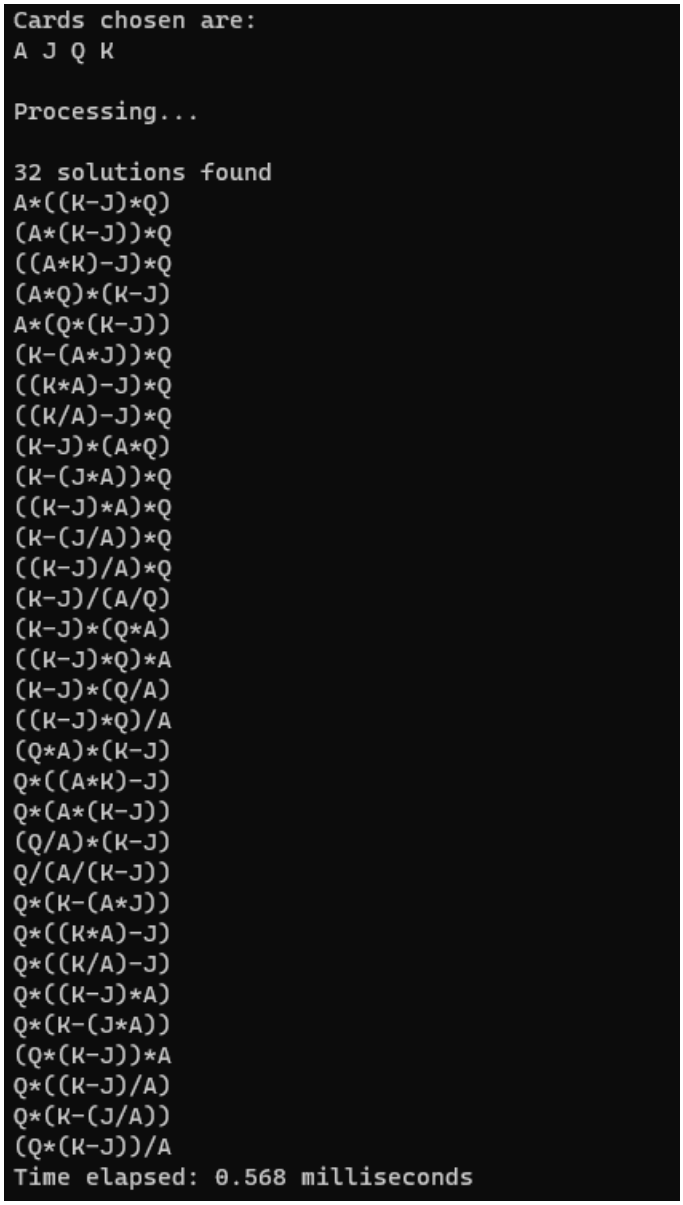
```
Cards chosen are:
5 5 5 5

Processing...

1 solutions found
(5*5)-(5/5)
Time elapsed: 0.04 milliseconds
```

f. A J Q K

Input



Output



g. K 5 J 4 (random input)

Input

```
Choose input method:
1.Randomized
2.User input
1
```

Output

```
Cards chosen are:
K 5 J 4

Processing...

24 solutions found

(5-4)*(J+K)
((5-4)*J)+K
(5-4)*(K+J)
((5-4)*K)+J
J-((4-5)*K)
J+((5-4)*K)
(J*(5-4))+K
(J/(5-4))+K
J-(K*(4-5))
J-(K/(4-5))
(J+K)*(5-4)
J+(K*(5-4))
(J+K)/(5-4)
J+(K/(5-4))
K-((4-5)*J)
K+((5-4)*J)
(K*(5-4))+J
(K/(5-4))+J
K-(J*(4-5))
K-(J/(4-5))
(K+J)*(5-4)
K+(J*(5-4))
(K+J)/(5-4)
K+(J/(5-4))

Time elapsed: 0.48 milliseconds
```

h. Input > 4 kartu

```
Choose input method:
1.Randomized
2.User input
2
Please input 4 cards (A,2-10,J,Q,K)
7 7 7 7 7 7
Invalid input. Please input 4 cards in the format (A,2-10,J,Q,K) separated by space.
```

i. Input < 4 kartu

```
Choose input method:
1.Randomized
2.User input
2
Please input 4 cards (A,2-10,J,Q,K)
7 7 7
Invalid input. Please input 4 cards in the format (A,2-10,J,Q,K) separated by space.
```

j. Input di luar *constraint*

```
Choose input method:
1.Randomized
2.User input
2
Please input 4 cards (A,2-10,J,Q,K)
1029941 934083 92381293 349829
Invalid input. Please input 4 cards in the format (A,2-10,J,Q,K) separated by space.
```

4. *Link Repository*

https://github.com/farizkik/Tucil1_13521082

Poin	Ya	Tidak
1. Program berhasil dikompilasi tanpa kesalahan	✓	
2. Program berhasil *running*	✓	
3. Program dapat membaca *input / generate* sendiri dan memberikan luaran	✓	
4. Solusi yang diberikan program memenuhi (berhasil mencapai 24)	✓	
5. Program dapat menyimpan solusi dalam file teks	✓	