Aplikasi OBE Untuk Mengurangi Kompleksitas Algoritma Program Penghitung Determinan Matriks Persegi

Alif Bhaskoro / 13514016
Program Studi Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514016@std.stei.itb.ac.id

***Abstract*—Setiap algoritma memiliki nilai kompleksitasnya masing-masing. Jika sebuah algoritma dengan nilai kompleksitas yang lebih rendah dengan algoritma lain yang menghasilkan hasil yang sama, maka dapat dikatakan bahwa algoritma pertama lebih efisien (mangkus). Operasi baris elementer adalah suatu cara untuk mengubah suatu matriks menjadi matriks eselon / matriks eselon tereduksi. Makalah ini akan membahas tentang penggunaan OBE untuk mengurangi kompleksitas alogritma penghitung determinan.**

***Keywords*—Efisien, Kompleksitas Algoritma, Matriks, OBE.**

I. PENDAHULUAN

Algoritma adalah langkah-langkah penyelesaian suatu masalah secara sistematis. Untuk menyelesaikan suatu masalah, tentu banyak cara untuk menyelesaikannya. Misalkan ada permasalahan yaitu mengharuskan kita bergerak dari A-B. Penyelesaian untuk masalah ini tentu banyak sekali yaitu A-B, A-C-B, A-C-D-B, dll.

Namun tidak semua algoritma bersifat efisien. Sebagai contoh mari kita ambil 2 buah algoritma untuk permasalahan diatas yaitu A-B dan A-C-B. Kita asumsikan untuk berpindah 1 huruf dibutuhkan waktu 1 detik. Jika kita hitung waktu yg diperlukan kedua algoritma maka algoritma kedua memerlukan waktu 1 detik lebih lama dibandingkan algoritma pertama. Mungkin untuk permasalahaan ini, hal ini tidak signifikan karena hanya menambah waktu sebesar 1 detik. Namun jika permasalahannya berubah menjadi berpindah dari A ke B lalu kembali ke A lagi sebanyak 100 kali ? Untuk kasus ini, algoritma kedua membutuhkan waktu 200 detik lebih lama dibandingkan algoritma pertama. Karena itu efisien merupakan salah satu nilai yang harus dipertimbangkan dalam pembuatan sebuah algoritma.

Matriks adalah hal yang lazim di bidang engineering. Matriks memiliki banyak kegunaan di segala macam bidang, seperti untuk representasi graf. Salah satu unsur matriks adalah determinan. Ada banyak algoritma untuk menghitung determinan, namun belum tentu semua algoritma tersebut efisien.

II. DASAR TEORI

Berikut adalah dasar-dasar teori yang digunakan dalam pembahasan makalah ini.

2.1 Kompleksitas Algoritma

Kompleksitas algoritma dibedakan menjadi 2 macam, yaitu kompleksitas ruang dan kompleksitas waktu. Kompleksitas ruang adalah ruang memori yang dibutuhkan sebuah algoritma sebagai fungsi dari ukuran masukan n. Kompleksitas waktu adalah waktu yang dibutukan algoritma untuk menyelesaikan masalah sebagai fungsi dari ukuran masukan n. Contoh pada bab sebelumnya merupakan contoh sebuah kompleksitas waktu.

2.1.1 Kompleksitas Waktu Asimpiotik

Kompleksitas waktu yang presisi sebuah algoritma terkadang tidak perlu untuk kita ketahui. Yang lebih penting adalah bagaimana waktu terbaik dan waktu terburuk tumbuh bersamaan dengan meningkatnya ukuran masukan. Untuk mengetahui kinerja suatu algoritma, kita perlu mengetahui perkiraan kasar kebutuhan waktu algoritma dan seberapa cepat fungsi kebutuhan waktu tersebut akan tumbuh. Notasi kompleksitas waktu asimpiotik disebut juga "O-Besar" (Big-O) yang mempunyai definisi :

$T(n) = O(f(n))$ bila terdapat konstanta C dan n_0 sedemikian rupa sehingga $T(n) <= C(f(n))$ untuk $n >= n_0$.

Arti dari notasi Big-O ini adalah jika sebuah algoritma mempunyai waktu asimpiotik sebanyak O(f(n)), maka jika n dibuat makin besar maka waktu yang diperlukannya tidak akan pernah melebihi suatu konstanta (C) dikali dengan f(n). Jadi bisa dikatakan bahwa f(n) adalah batas atas atau upper bound dari T(n).

2.1.2 Teorema O-Besar

Misalkan $T_1(n) = O(f(n))$ dan $T_2(n) = O(g(n))$ maka

a. $T_1(n) + T_2(n) = O(\max(f(n),g(n))$
b. $T_1(n)\ T_2(n) = O(f(n)g(n))$
c. $O(cf(n)) = O(f(n))$

d. F(n) = O(f(n))

2.1.3 Nilai O-Besar untuk Instruksi di Algoritma

1. Assignment, perbandingan, operasi aritmetik, read, write membutuhkan waktu O(1)
2. Pengisian elemen array membutuhkan waktu O(1).
3. If C then S1 else S2 membutuhkan waktu $T_c + \max(T_{s1},T_{s2})$.
4. For membutuhkan waktu sebanyak kompleksitas waktu body dari for tersebut dikalikan dengan banyaknya pengulangan.
5. While C do S membutuhkan waktu sebanyak kompleksitas waktu dari body C dan S dikali dengan jumlah pengulangan.
6. Prosedur dan fungsi membutuhkan waktu sebanyak O(1).

2.2 Matriks

Matriks adalah susunan skalar elemen-elemen dalam bentuk baris dan kolom. Ada beberapa matriks khusus yaitu :

1. Matriks Diagonal
   Matriks diagonal adalah matriks bujur sangkar yang untuk setiap elemen yang berada di posisi i =/ j bernilai 0.

$$\begin{bmatrix} 7 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$

Matriks Diagonal

2. Matriks Identitas
   Matriks identitas adalah matriks diagonal dengan semua elemen diagonalnya bernilai 1.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Matriks Identitas

3. Matriks segitiga atas/bawah
   Matriks segitiga atas / bawah adalah matriks yang semua elemen di atas / di bawah diagonal bernilai 0.

$$\begin{bmatrix} 3 & 2 & 2 \\ 0 & 4 & 6 \\ 0 & 0 & 5 \end{bmatrix}$$

Matriks Segitiga Atas

$$\begin{bmatrix} 3 & 0 & 0 \\ 2 & 4 & 0 \\ -3 & 7 & 5 \end{bmatrix}$$

Matriks Segitiga Bawah

4. Matriks Transpose
   Matriks tranpose adalah matriks yang didapatkan dengan transpose suatu matriks, yaitu jika ada suatu matriks berukuran m x n maka matriks transposenya akan berukuran n x m. Misalkan matriks transpose dari A adalah B, maka setiap B[i][j] adalah A[j][i].

$$\begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix} \xrightarrow{\text{Transpose}} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Matriks Transpose

5. Matriks Simetri
   Matriks simetri adalah matriks yang jika di transpose akan menghasilkan matriks yang sama seperti matriks awalnya. $A^T=A$.

$$\begin{bmatrix} 5 & 2 & 3 \\ 2 & 7 & 4 \\ 3 & 4 & 1 \end{bmatrix}$$

Matriks Simetri

6. Matriks 0/1
   Matriks yang setiap elemennya hanya bernilai 0 atau 1. Matriks ini sering digunakan untuk merepresentasikan graf yaitu matriks ketetanggaan.

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Matriks 0/1

7. Matriks Persegi
   Matriks persegi adalah matriks yang berukuran m x m.

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Matriks 0/1 Persegi

8. Matriks Skalar

   Matriks skalar adalah matriks yang memiliki elemen diagonal yang sama dan elemen lain bernilai 0.

$$F = \begin{pmatrix} 7 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 7 \end{pmatrix}$$

Matriks Skalar

2.2.1 Determinan

Determinan merupakan suatu fungsi yang merepresentasikan suatu matriks. Ada banyak cara untuk menghitung determinan yaitu cara Cramer, Kofaktor, dll.

Menghitung determinan dengan cara kofaktor adalah dengan menjumlahkan $a_{1j}C_{1j} + a_{2j}C_{2j} + \ldots + a_{nj}C_{nj}$ dengan n adalah jumlah baris dan kolom suatu matriks persegi, a_{ij} merupakan elemen matriks baris ke-i dan kolom ke-j, dan C_{ij} adalah kofaktor dari minor (matriks baru dengan ukuran n-1 x n-1 yang tidak memiliki baris i dan kolom j).

2.2.2 Matriks Eselon

Matriks eselon adalah salah satu bentuk matriks dengan karateristik :

1. Jika elemen suatu baris tidak bernilai 0 semua, maka elemen pertama yang bukan 0 harus bernilai 1 atau disebut leading 1.
2. Jika ada baris yang seluruh elemennya bernilai 0, maka baris tersebut bisa kita letakan di baris terakhir.
3. Jika ada 2 baris berurutan yang seluruh elemennya tidak bernilai 0 semua, maka 1 utama yang berada di baris yang di bawah harus berada di sebelah kanan 1 utama yang berada di baris yang di atas.
4. Setiap kolom yang memiliki 1 utama harus memiliki nilai 0 di tempat lainnya.

Jika suatu matriks memenuhi ketiga syarat di atas (1-3) maka matriks tersebut disebut sebagai matriks eselon. Jika suatu matriks eselon memenuhi syarat ke-4, maka matriks tersebut dapat kita sebut sebagai matriks eselon tereduksi. Salah satu cara untuk membentuk matriks eselon dan matriks eselon tereduksi adalah dengan operasi baris elementer (OBE).

Proses membentuk matriks menjadi bentuk matriks eselon dinamakan proses eliminasi Gauss, sedangkan proses untuk membentuk matriks eselon tereduksi dinamakan proses Gauss-Jordan.

$$\begin{bmatrix} 1 & 2 & 1 & 6 \\ 0 & 1 & 1 & 3 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Matriks Eselon

$$\begin{bmatrix} 1 & 0 & 8 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 0 & 3 & 0 & 4 \\ 0 & 0 & 1 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Matriks Eselon Tereduksi

Matriks persegi jika dirubah menjadi matrik eselon tereduksi akan menghasilkan matriks diagonal. Sehingga untuk mencari determinannya kita hanya perlu mengalikan elemen-elemen diagonalnya saja.

2.2.3 Operasi Baris Elementer

Operasi baris elementer adalah suatu proses pengulangan ketiga langkah di bawah ini :

1. Mengalikan suatu baris dengan konstanta yang bernilai tidak 0.
2. Menukar letak 2 buah baris.
3. Menambahkan suatu baris dengan baris lain yang dikalikan konstanta.

III. APLIKASI OBE DALAM PENGURANGAN NILAI KOMPLEKSITAS WAKTU ALGORITMA PENGHITUNG DETERMINAN

Berikut sebuah algoritma menghitung determinan sebuah matriks dalam bahasa C. Matriks disini merupakan tipe bentukan yang sudah di deklarasikan sebelumnya yaitu int[Nmax][Nmax] dengan Nmax=100. Fungsi LastIdxBrs dan LastIdxKol juga sudah di deklarasikan sebelumnya dengan kegunaan untuk mencari indeks baris terakhir dan indeks kolom terakhir suatu matriks.

```
int Determinan (MATRIKS M)
/*  Prekondisi: IsBujurSangkar(M)
    Menghitung nilai determinan M
*/
{
    /*Kamus*/
    MATRIKS Mx;
    int det;
    int n;/*jumlah pangkat untuk -1*/
    int p;/*pengali*/
    indeks i,j;
    indeks ix,jx;/*indeks untuk matriks bantu*/
    indeks ic,jc;
    /*Algoritma*/
    if(LastIdxBrs(M)==1)
    {
        det=M.Mem[1][1];
    }
    else if(LastIdxBrs(M)==2)
    {
        det=M.Mem[1][1]*M.Mem[2][2]-M.Mem[2][1]*M.Mem[1][2];
    }
    else
    {
```

```
        Mx.NBrsEff=LastIdxBrs(M)-1;
        Mx.NKolEff=LastIdxKol(M)-1;
        i=1;det=0;j=1;
        while(j<=LastIdxKol(M))
        {
            ix=1;ic=1;
            while(ix<=LastIdxBrs(M) && ic<=LastIdxBrs(Mx))
            {
                if(ix!=i)
                {
                    jx=1;
                    jc=1;
                    while(jx<=LastIdxKol(M) && jc<=LastIdxKol(Mx))
                    {
                        if(jx!=j)
                        {
                            Mx.Mem[ic][jc]=M.Mem[ix][jx];
                            jc++;
                        }
                        jx++;
                    }
                ic++;
                }
                ix++;
            }
            /*Matriks Mx terisi*/
            n=1;
            p=1;
            while(n<=i+j)
            {
                p=p*(-1);
                n++;
            }
            det=det+p*Determinan(Mx)*M.Mem[i][j];
            j++;
        }
    }
    return det;
}
```

Berikut adalah sebuah algoritma untuk menghitung determinan sebuah matriks eselon tereduksi persegi beserta dengan kompleksitas waktunya.

```
int Determinan (MATRIKS M)
{
    /*Kamus*/
    int i;
    int det;
    /*Algoritma*/
    i=1;
    det=1;
    while(i<=LastIdxBrs(M))
    {
        det=det*M[i][i];
        i++;
    }
    return det;
}
```

Algoritma ke-2 ini memiliki kompleksitias waktu yang cukup rendah. Assign nilai i dan determinan masing-masing memiliki nilai O(1). Untuk loop while dibawahnya, bodynya dilakukan sebanyak LastIdxBrs(M) atau indeks baris terakhirnya dikurangi dengan 1. Karena matriks yang digunakan adalah matriks persegi, maka jumlah baris dan kolomnya pun sama. Kita misalkan saja dengan n. Body while tersebut memiliki kompleksitas waktu sebanyak O(1), karena hanya berisi assignment-assignment saja. Sehingga total kompleksitas waktu while tersebut adalah (n-1)*O(1) atau disederhanakan menjadi O(n-1). Sehingga kompleksitas waktu totalnya adalah O(n).

Untuk mencari kompleksitas waktu algoritma diatas (algoritma 1)tidaklah mudah karena adanya fungsi rekursif dan banyaknya loop. Karena hasil yang ingin kita dapatkan adalah perbandingan antara kompleksitas waktu algoritma 1 dengan kompleksitas waktu algoritma 2, maka kompleksitas waktu presisi dari algoritma 1 tidak terlalu penting. Namun kita dapat menerka kompleksitas waktu algoritma 1. Jika kita lihat dari bagian if then else nya, maka kita dapat menyimpulkan bahwa kompleksitas waktu algoritma 1 akan sama dengan kompleksitas waktu else terakhir. Hal ini disebabkan if then dan else pertama hanya memiliki kompleksitas waktu O(1), sehingga jika dicari maksimumnya dengan kompleksitas waktu else akan menghasilkan kompleksitas waktu else. Di dalam else ada loop while sekali dengan banyak pengulangan sebanyak LastIdxBrs(M) – 1. Seperti algoritma 2, dalam algoritma ini yang akan diuji adalah matriks persegi, sehingga ukuran baris dan kolomnya sama sehingga dapat kita misalkan sebagai n, maka loop while ini akan diulangin sebanyak n-1 kali.

Di dalam loop while itu sendiri, terdapat loop while lagi yang akan diulangi sebanyak n-1 kali juga. Sehingga kompleksitas waktu minimum untuk else terakhir adalah (n-1)*O(n-1) atau bisa kita sederhanakan menjadi $O(n^2)$. Melihat hal ini, kita dapat menyimpulkan bahwa algoritma 2 memiliki kompleksitas waktu yang lebih kecil daripada kompleksitas waktu algoritma 1.

IV. KESIMPULAN

Algoritma untuk menghitung determinan dari matriks biasa memiliki kompleksitas waktu yang lebih tinggi dibandingkan dengan algoritma untuk menghitung determinan matriks persegi yang sudah dikenakan OBE sehingga menjadi matriks eselon tereduksi. Namun untuk mengaplikasikan OBE pada suatu matriks persegi sehingga dapat menjadi matriks eselon tereduksi tentu membutuhkan algoritma lain. Sehingga jika hanya diaplikasikan sekali tentu kompleksitas waktunya akan sama.

Tetapi untuk mengubah matriks menjadi matriks eselon tereduksi hanya perlu dilakukan sekali. Sehingga jika dibandingkan dengan melakukan perhitungan determinan matriks biasa sebanyak 3 kali dengan merubah sebuah

matriks menjadi matriks eselon tereduksi lalu dihitung determinannya 3 kali, tentu cara kedua akan memiliki kompleksitas waktu yang jauh lebih kecil.

REFERENCES

[1] Hausner, Melvin, A Vector Space approach to Geometry, Dover. 2010.

[2] Munir, Rinaldi, Matematika Diskrit. Bandung : Penerbit Informatika, Palasari.

[3] http://bahanbelajarsekolah.blogspot.co.id/2014/10/pengertian-dan-jenis-jenis-matriks.html? Diakses pada 15 Desember 2015, 19.00

[4] http://t1nez.blogspot.co.id/2009/01/microsoft-excel-matriks-function.html Diakses pada 15 Desember 2015, 19.30

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 16 Desember 2015

Alif Bhaskoro 13514016