BAB VII
TEKNIK OPTIMASI DAN TABEL INFORMASI

TUJUAN PRAKTIKUM

1. Mengetahui jenis - jenis teknik optimasi beserta kegunaannya
2. Mengetahui jenis - jenis tabel informasi
3. Mengerti bagaimana interaksi antar tabel

TEORI PENUNJANG

7.1 Teknik Optimasi

Menghasilkan kode program dengan ukuran yang lebih kecil, sehingga lebih cepat eksekusinya. Berdasarkan ketergantungan pada mesin :

- Machine Dependent Optimizer
- Machine Independent Optimizer (Optimasi Lokal dan Optimasi Global)

7.1.1 Optimasi Lokal

Dilakukan hanya pada suatu blok dari *source code*, dengan cara :

1. Folding

Nilai konstanta atau ekspresi pada saat compile time diganti dengan nilai komputasinya.

Contoh instruksi :

A:=2+3+B

diganti menjadi

A:=5+B

2. Redundant – Subexpression Elimination

Menggunakan hasil komputasi terdahulu daripada melakukan komputasi ulang.

Contoh urutan instruksi :

```
A:=B+C
X:=Y+B+C
```

B+C redundan, bisa memanfaatkan hasil komputasi sebelumnya, selama tidak ada perubahan nilai pada variabel.

3. Optimasi dalam sebuah iterasi

- ✓ Loop Unrolling : menggantikan suatu loop dengan menulis statement dalam loop beberapa kali.

  Contoh instruksi :

```
FOR I:=1 to 2 DO
    A[I]:=0;
```

  dioptimasi menjadi

```
A[1] := 0;
A[2] := 0;
```

  Pada instruksi pertama yang menggunakan iterasi perlu dilakukan inisialisasi setiap eksekusi loop, pengetesan, adjustment, dan operasi pada tubuh perulangan. Yang kesemuanya itu menghasilkan banyak instruksi. Karena itu dengan optimasi hanya memerlukan dua instruksi assignment.

- ✓ Frequency Reduction : memindahkan statement ke tempat yang lebih jarang dieksekusi.

  Contoh instruksi :

```
FOR I:=1 TO 10 DO
    BEGIN
        X:=5;
        A:=A+1;
    END;
```

  variabel X dapat dikeluarkan dari iterasi, menjadi :

```
X:=5;
```

```
FOR I:=1 TO 10 DO
BEGIN
        A:=A+1
END;
```

4. Strength Reduction

Mengganti suatu operasi dengan jenis operasi lain yang lebih cepat dieksekusi.

Contoh :

pada beberapa komputer operasi perkalian memerlukan waktu lebih banyak dari pada operasi penjumlahan.

```
A := A + 1
```

Dapat digantikan dengan

```
INC(A)
```

7.1.2 Optimasi Global

Dilakukan dengan analisis flow, yaitu suatu graph berarah yang menunjukkan jalur yang mungkin selama eksekusi program. Ada 2 kegunaan optimasi global, yaitu bagi para programmer dan untuk compiler itu sendiri.

- **Bagi Programmer**
  - Unreachable / dead code : Kode yang tidak pernah dieksekusi

    Misal :

    ```
    X := 5;
    IF X = 0 THEN
            A := A + 1
    ```

    Instruksi

    A := A + 1

    Tidak pernah dikerjakan karena kondisi X tidak pernah menjadi 0, sehingga memperlambat proses.

- Unused parameter : parameter yang tidak pernah digunakan dalam prosedur

  Misal :

```
Procedure penjumlahan(a,b,c; Integer);
Var x : integer;
Begin
        X := a + b;
End
```

  Parameter c tidak pernah digunakan sehingga tidak perlu diikutsertakan dikarenakan pada prosedur penjumlahan c tidak pernah dikenakan suatu proses atau nilai

- Unused variable : variabel yang tidak pernah digunakan

  Misal :

```
Var a,b : Integer;
Begin
        a := 5;
end;
```

  Variabel b tidak pernah digunakan dalam manipulasi, sehingga tidak perlu dideklarasikan

- Variable : variabel yang dipakai tanpa nilai awal

  Misal :

```
Var a,b : Integer;
Begin
        a := 5;
        a := a + b;
end;
```

  Variabel b digunakan tetapi tidak memiliki harga awal

➢ Bagi Compiler

- Meningkatkan efisiensi eksekusi program
- Menghilangkan useless code / kode yang tidak terpakai

7.2 Tabel Informasi

Fungsi Tabel Informasi atau Tabel Simbol :

- ✓ Membantu pemeriksaan kebenaran semantik dari program sumber.
- ✓ Membantu dan mempermudah pembuatan *intermediate code* dan proses pembangkitan kode.

Untuk mencapai fungsi tersebut dilakukan dengan menambah dan mengambil atribut variabel yang dipergunakan pada program dari tabel. Atribut, misalnya nama, tipe, ukuran variabel.

Tabel Simbol berisi daftar dan informasi *identifier* pokok yang terdapat dalam program sumber, disebut Tabel Pokok / Utama.

Tabel Pokok belum mengcover semua informasi, untuk itu disediakan tabel lagi sebagai pelengkap Tabel Pokok.

Untuk mengacu pada tabel simbol yang bersesuaian dengan suatu *indentifier* tertentu, maka pada Tabel Pokok harus disediakan field yang bisa menjembatani *identifier* dari Tabel Pokok ke tabel-tabel lain yang bersesuaian. Untuk itu, pemilihan elemen tabel pada Tabel Pokok maupun tabel lainnya, merupakan sesuatu yang sangat penting.

7.2.1 Elemen Tabel Informasi

Elemen pada Tabel Simbol bermacam-macam, tergantung pada jenis bahasanya. Misalnya :

1. No urut *identifier* : Menentukan nomor urut *identifier* dalam tabel simbol.
2. Nama *identifier* : Berisi nama-nama *identifier* (nama variabel, nama tipe, nama konstanta, nama procedure, nama fungsi, dll) yang terdapat pada program sumber. Nama-nama ini akan dijadikan referensi pada waktu analisa semantik, pembuatan intermediate code, serta pembangkitan kode.
3. Tipe *identifier* : Berisi keterangan/informasi tipe dari record dan string, maupun procedure dan function.

4. Object time address : address yang mengacu ke alamat tertentu.
5. Dimensi dari *identifier* yang bersangkutan.
6. Nomor baris variabel dideklarasikan.
7. Nomor baris variabel direferensikan.
8. Field link.

7.2.2 Implementasi Tabel Informasi

Beberapa jenis :

1. Tabel *Identifier* : Berfungsi menampung semua *identifier* yang terdapat dalam program.
2. Tabel *Array* : Berfungsi menampung informasi tambahan untuk sebuah *array*.
3. Tabel Blok : Mencatat variabel-variabel yang ada pada blok yang sama.
4. Tabel Real : menyimpan elemen tabel bernilai real.
5. Tabel String : Menyimpan informasi string.
6. Tabel *Display* : Mencatat blok yang aktif.

7.2.2.1 Tabel Identifier

Memiliki field :

- ✓ No urut *identifier* dalam tabel
- ✓ Nama *identifier*
- ✓ Jenis/obyektif dari *identifier :* Prosedur, fungsi, tipe, variabel, konstanta
- ✓ Tipe dari *identifier* yang bersangkutan : integer, char, boolean, array, record, file, no-type
- ✓ Level : Kedalaman *identifier* tertentu, hal ini menyangkut letak *identifier* dalam program. Konsepnya sama dengan pembentukan tree, misal *main program* = level 0. Field ini digunakan pada *run time* untuk mengetahui current activation record dan variabel yang bisa diakses.

Untuk *identifier* yang butuh penyimpanan dicatat pula :

- Alamat relatif/address dari *identifier* untuk implementasi

- Informasi referensi (acuan) tertentu ke alamat tabel lain yang digunakan untuk mencatat informasi-informasi yang diperlukan yang menerangkannya.
- *Link* : Menghubungkan *identifier* ke *identifier* lainnya, atau yang dideklarasikan pada level yang sama.
- Normal : Diperlukan pada pemanggilan parameter, untuk membedakan parameter *by value* dan *reference* (berupa suatu variabel boolean)

Contoh :

Terdapat listing program sebagai berikut :

```
Program A;
var B : integer;
        Procedure X(Z:char);
        var C : integer
        Begin
        …….
```

Tabel *Identifier* akan mencatat semua *identifier* :

0 A
1 B
2 X
3 Z
4 C

Contoh implementasi table identifier :

```
TabId: array [0..tabmax] of
        record
                name   : string;
                link   : integer;
                obj    : objek;
        tipe  :  types;
```

```
                ref       : integer;
                normal : boolean;
                level  :  0..maxlevel;
                address : integer;
        end;
```

Dimana :

objek = (konstant, variabel, prosedur, fungsi)

types = (notipe, int, reals, booleans, chars, arrays, record)

7.2.2.2 Tabel Array

Memiliki *field* :

- ✓ No urut suatu *array* dalam tabel
- ✓ Tipe dari indeks *array* yang bersangkutan
- ✓ Tipe elemen *array*
- ✓ Referensi dari elemen *array*
- ✓ Indeks batas bawah *array*
- ✓ Indeks batas atas *array*
- ✓ Jumlah elemen *array*
- ✓ Ukuran total *array* (*total size* = (atas-bawah+1) x *elemen size*)
- ✓ *Elemen size* (ukuran tiap elemen)

Tabel *Array* diacu dengan field referensi pada Tabel *Identifier*.

Contoh implementasi table array :

```
        TabArray : array  [1...tabmax] of
                record
                        indextype, elementype :  types;
                        elemenref, low, high, elemensize, tabsize : integer
                end;
```

7.2.2.3 Tabel Blok

Memiliki *field* :

- ✓ No urut blok
- ✓ Batas awal blok
- ✓ Batas akhir blok
- ✓ Ukuran parameter / parameter size
- ✓ Ukuran variabel / variabel size
- ✓ Last variable
- ✓ Last parameter

Contoh implementasi table blok :

```
TabBlok: array [1..tabmax]  of
        record
                lastvar, lastpar, parsize, varsize: integer;
        end;
```

Dari contoh listing program berikut :

```
Program a;
var B: integer;
Procedure X(Z:char);
        var C : integer
Begin
…….
```

Akan diperoleh, untuk blok Program A :

last variable = 2

variable size = 2 (dianggap integer butuh dua byte)

last parameter = 0 (tanpa parameter)

parameter size = 0

Untuk blok Procedure X :

last variable = 4

variable size = 2

last parameter = 3

parameter size = 1 (dianggap char butuh satu byte)

7.2.2.4 Tabel Real

Elemen tabel real :

- ✓ No urut elemen
- ✓ Nilai real suatu variabel real yang mengacu ke indeks tabel ini

Contoh implementasi tabel real :

```
TabReal : array [1..tabmax]  of real
```

(pemikiran : setiap tipe yang dimiliki oleh suatu bahasa akan memiliki tabelnya sendiri)

7.2.2.5 Tabel String

Elemennya :

- ✓ No urut elemen
- ✓ Karakter-karakter yang merupakan konstanta

Contoh implementasi tabel string :

```
TabString: array[1..tabmax] of string
```

7.2.2.6 Tabel Display

Elemennya :

- ✓ No urut tabel
- ✓ Blok yang aktif

Pengisian tabel display dilakukan dengan konsep stack.
Urutan pengaksesan : Tabel Display – Tabel Blok – Tabel Simbol.

Contoh implementasi Tabel Display :

TabDisplay: array [1..tabmax]of integer

7.2.3 Interaksi Antar Tabel

Pertama kali tabel display akan menunjuk blok mana yang sedang aktif. Dari blok yang aktif ini, akan diketahui identifier-identifier yang termasuk dalam blok tersebut. Untuk pertama kalinya, yang akan diacu adalah identifier yang paling akhir, kemudian identifier sebelumnya, dan seterusnya. Informasi suatu identifier ini mungkin belum lengkap. Untuk itu dari tabel identifier ini mungkin akan dicari kelengkapan informasi dari suatu identifier ke tabel yang sesuai (tabel real, tabel string, atau tabel array).

LAPORAN PENDAHULUAN

1. Sebutkan dan jelaskan macam – macam teknik optimasi!
2. Sebutkan dan jelaskan macam – macam tabel informasi beserta contohnya!

LAPORAN AKHIR

Buat program dengan menggunakan implementasi Tabel Informasi!