Bab 8

Basic Block Identification dan Local Optimization

Sub-CPMK yang Dicakup dalam Bab Ini:

- **Sub-CPMK 4.2:** Mengimplementasikan basic block identification

8.1 Dasar Basic Block dan Karakteristiknya

Basic Block adalah sekumpulan instruksi kode antara (*intermediate code*) yang dieksekusi secara berurutan tanpa adanya instruksi lompatan (*jump/branch*) di tengah-tengahnya.

8.1.1 Definisi Formal

Sebuah blok instruksi disebut *Basic Block* jika:

- Hanya memiliki satu titik masuk (*entry point*), yaitu instruksi pertama.
- Hanya memiliki satu titik keluar (*exit point*), yaitu instruksi terakhir.
- Semua instruksi di dalamnya dieksekusi secara sekuensial.

8.1.2 Contoh Basic Block

Perhatikan potongan kode berikut:

```
t1 = a + b
t2 = t1 * c
x = t2
```

Setelah eksekusi dimulai dari baris pertama, semua baris berikutnya dijamin akan dieksekusi secara berurutan. Ini adalah satu *Basic Block*.

8.2 Algoritma Identifikasi Leader

Untuk membagi kode antara menjadi beberapa *basic block*, kita harus mengidentifikasi instruksi-instruksi yang menjadi "pemimpin" (*leader*).

8.2.1 Aturan Identifikasi Leader

Instruksi adalah *leader* jika memenuhi salah satu syarat berikut:

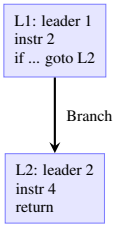
1. Instruksi pertama dalam program/fungsi.

2. Instruksi yang merupakan target dari sebuah lompatan (*conditional jump* atau *unconditional jump*).

3. Instruksi yang muncul tepat setelah instruksi lompatan.

8.2.2 Pembentukan Blok

Setelah semua *leader* ditemukan, setiap *basic block* terdiri dari sebuah *leader* dan semua instruksi berikutnya hingga (tapi tidak termasuk) instruksi *leader* berikutnya.



Gambar 8.1: Identifikasi leader dan pembentukan blok

8.3 Local Optimization: Constant Folding

Constant Folding adalah teknik optimasi yang mengevaluasi ekspresi dengan operan konstanta pada waktu kompilasi (*compile-time*) alih-alih pada waktu eksekusi (*run-time*).

8.3.1 Mekanisme Kerja

Jika kedua operan dalam operasi biner adalah literal (misal: `3 + 5`), kompilator langsung menggantinya dengan hasilnya (`8`).

8.3.2 Keuntungan

Teknik ini mengurangi jumlah instruksi yang dieksekusi oleh prosesor, mempercepat program, dan seringkali membuka peluang optimasi lebih lanjut (seperti eliminasi variabel yang tidak terpakai).

```
// Sebelum optimasi
t1 = 10 * 2
t2 = a + t1

// Sesudah Constant Folding
t1 = 20
t2 = a + 20
```

8.4 Local Optimization: Constant Propagation

Constant Propagation bekerja sama dengan *constant folding* dengan menyebarkan nilai konstanta ke penggunaan variabel berikutnya di dalam blok yang sama.

8.4.1 Prinsip Kerja

Jika sebuah variabel diberikan nilai konstanta (misal: `x = 10`), setiap penggunaan `x` di instruksi berikutnya (sebelum `x` diubah kembali) diganti langsung dengan nilai `10`.

8.4.2 Contoh Kasus

```
x = 5
y = x + 2
z = y * 10
```

Akan dioptimasi secara bertahap menjadi:

1. `y = 5 + 2` (Propagation)
2. `y = 7` (Folding)
3. `z = 7 * 10` (Propagation)
4. `z = 70` (Folding)

8.5 Control Flow Graph (CFG) Construction

Control Flow Graph (CFG) adalah representasi berarah di mana setiap *node* adalah sebuah *basic block* dan setiap *edge* menunjukkan kemungkinan aliran kendali antar blok tersebut.

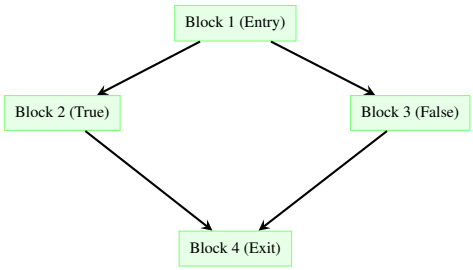
8.5.1 Membangun Tepi (Edges)

Sebuah tepi ditambahkan dari blok B_1 ke blok B_2 jika:

- Terdapat instruksi lompatan (jump) dari akhir B_1 ke awal B_2.
- B_2 terletak tepat setelah B_1 dalam urutan program, dan B_1 tidak diakhiri dengan lompatan mutlak (*unconditional jump*).

8.5.2 Pentingnya CFG

CFG adalah struktur data utama untuk melakukan *Global Optimization* dan analisis aliran data (*Data Flow Analysis*) yang mencakup seluruh fungsi/program.



Gambar 8.2: Contoh Control Flow Graph sederhana

Aktivitas Pembelajaran

1. **Block Identification**: Implementasikan algoritma basic block identification.
2. **CFG Construction**: Bangun control flow graph dari three-address code.
3. **Local Optimizations**: Implementasikan constant folding dan algebraic simplification.
4. **Data Flow**: Implementasikan reaching definitions analysis.
5. **Optimization Pipeline**: Buat pipeline untuk multiple local optimizations.

Latihan dan Refleksi

1. Identifikasi basic blocks dalam potongan three-address code yang diberikan!

2. Gambarkan CFG untuk program dengan nested if-else dan while loops!

3. Implementasikan constant folding untuk ekspresi aritmatika kompleks!

4. Analisis reaching definitions untuk program sederhana!

5. Identifikasi dead code yang dapat dieliminasi!

6. **Refleksi**: Bagaimana basic blocks menyederhanakan optimasi compiler?

Asesmen (Evaluasi Kinerja)

Instrumen Penilaian untuk Sub-CPMK 4.2

A. Pilihan Ganda

1. Basic block memiliki:
   (a) Multiple entry points
   (b) Single entry point
   (c) Multiple exit points
   (d) No exit points

2. Leader adalah instruksi yang:
   (a) Pertama dalam program
   (b) Target dari jump
   (c) Setelah conditional jump
   (d) Semua jawaban benar

3. Constant folding adalah:
   (a) Menghapus konstanta
   (b) Menggabungkan konstanta
   (c) Mengidentifikasi konstanta
   (d) Mengoptimasi loops

B. Essay

1. Jelaskan algoritma basic block identification dengan contoh konkret!

2. Implementasikan local optimization pipeline dengan constant folding, copy propagation, dan dead code elimination!

Rubrik Penilaian: Lihat Lampiran A

Checklist Pencapaian Kompetensi

Centang item berikut setelah Anda yakin telah menguasainya:

☐ Saya dapat mengimplementasikan basic block identification

☐ Saya dapat membangun control flow graph

☐ Saya dapat melakukan constant folding

☐ Saya dapat menerapkan algebraic simplification

☐ Saya dapat mengimplementasikan copy propagation

☐ Saya dapat melakukan dead code elimination

Rangkuman

Bab ini membahas basic block identification dan local optimization, termasuk algoritma identifikasi, CFG construction, dan berbagai teknik optimasi lokal. Mahasiswa belajar membangun fondasi untuk global optimizations.

Poin Kunci:

- Basic blocks adalah unit analisis untuk optimasi compiler
- Leaders menandai awal dari setiap basic block
- CFG merepresentasikan alur kontrol antar basic blocks
- Local optimizations bekerja dalam satu basic block
- Data flow analysis menyediakan informasi untuk optimasi

Kata Kunci: ***Basic Block***, ***Control Flow Graph***, ***Local Optimization***, ***Constant Folding***, ***Copy Propagation***, ***Dead Code Elimination***, ***Data Flow Analysis***