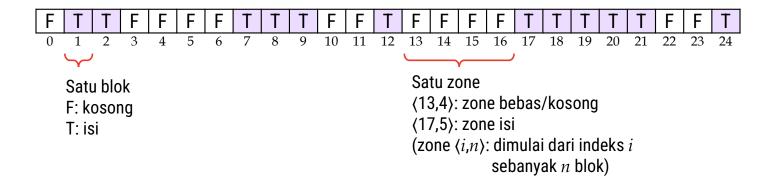
Studi Kasus: Pengelolaan Memori Representasi Kontigu

IF2110 – Algoritma dan Struktur Data Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung

Memori

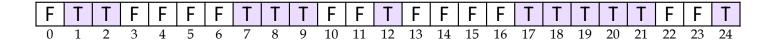




Deskripsi Persoalan

Memori dinyatakan sebagai N_BLOCK buah blok kontigu

- F: KOSONG
- T: ISI



Zone Bebas: blok-blok berurutan yang berstatus F (KOSONG)

Dinyatakan oleh indeks blok KOSONG pertama dan ukurannya

- Zone bebas I: (0,1)
- Zone bebas II: (3,4)
- Zone bebas III: (10,2)
- Zone bebas IV: <13,4>
- Zone bebas V: (22,2)

Deskripsi Persoalan

Alokasi dan dealokasi memori menyebabkan perubahan terhadap zone bebas

Realisasikan prosedur-prosedur sebagai berikut:

- Prosedur initMem mengeset semua blok menjadi blok KOSONG
- Prosedur **allocBlock** melakukan alokasi: membuat blok KOSONG sejumlah x menjadi ISI; menghasilkan indeks awal (startIdx) di mana alokasi dilakukan
- Prosedur **deallocBlock** "membebaskan" zone ISI: membuat blok ISI sejumlah x yang berawal di indeks startIdx menjadi KOSONG
- Prosedur compaction (memory compaction) memampatkan memori sehingga semua blok KOSONG berada di bagian kiri memori dan semua blok ISI berada di kanan memori

Representasi secara kontigu dengan array

Representasi dengan Array: Struktur Data

KAMUS

```
constant UNDEF: integer = -1
constant N_BLOCK: integer = 100
STATMEM: array [0..N_BLOCK-1] of boolean
{ tabel status memori: true jika ISI, false jika kosong }
```

Representasi dengan Array: Prosedur initMem

Secara umum: set seluruh elemen STATMEM menjadi bernilai false

Algoritma: Diktat hlm. 167

```
procedure initMem
{ I.S.: Sembarang }
{ F.S.: Semua blok memori dinyatakan KOSONG }
{ Proses: Semua status blok dengan indeks [0..N_BLOCK-1] dijadikan KOSONG dengan traversal blok [0..N_BLOCK-1].
    Proses sekuensial tanpa penanganan kasus kosong (N_BLOCK ≥ 0) }
```

Representasi dengan Array: Prosedur allocBlock (1)

Representasi dengan Array: Prosedur allocBlock (2)

Strategi First Fit: Alokasi dilakukan pada zone yang memenuhi syarat yang **pertama kali ditemukan**.

Diktat hlm. 168

Sketsa umum algoritma:

```
repeat

cari blok kosong pertama (skema search)

if ketemu blok kosong then

catat indeks sebagai NAwal, yaitu blok awal zone kosong

hitung NKosong, yaitu banyaknya blok dlm zone kosong tersebut

until semua blok sudah diperiksa or NKosong ≥ x

terminasi

if ada zone memenuhi syarat then

ubah status semua blok pada zone tersebut [startIdx..startIdx+x−1] menjadi ISI
```

Representasi dengan Array: Prosedur allocBlock (3)

Strategi Best Fit: Alokasi dilakukan terhadap zone yang memenuhi syarat dan berukuran sama dengan x, atau jika tidak ada zone berukuran x, maka diambil yang ukurannya minimal. Keuntungan: blok tidak terpartisi kecil-kecil

Diktat Hlm. 169

```
repeat
cari zone kosong pertama (skema search)
if ketemu blok kosong then
hitung banyaknya blok dlm zone kosong tsb.
if ada zone kosong dan memenuhi syarat then
if zone kosong pertama then
inisialisasi
ukuran zone Minimum: NBMin dan Posisi Awal NAwal
else cek apakah lebih baik, jika ya, update NBMin dan startIdx
until semua blok diperiksa or blok berukuran = x
terminasi
if ada zone memenuhi syarat then
ubah status blok pada zone tersebut [startIdx...startIdx+x-1] menjadi ISI
```

Representasi dengan Array: Prosedur deallocBlock

Secara umum: Pemrosesan sekuensial (traversal) untuk membuat status blok [startIdx..startIdx+x-1] KOSONG

Diktat hlm. 170

Representasi dengan Array: Prosedur compaction (1)

Representasi dengan Array: Prosedur compaction (2)

Proses: "menggeser" elemen tabel yang berstatus KOSONG ke kiri

Dua pass (diktat hlm. 170):

- Traversal untuk mencacah banyaknya blok kosong, misalnya NKosong
- Traversal untuk memberi status [0..NKosong-1] dengan KOSONG dan [NKosong..N_BLOCK-1] dengan ISI

Satu pass:

• Traversal untuk mengelompokkan KOSONG di kiri dan ISI di kanan dengan menukarkan dua elemen.