Double Linked-List

(TIB11 – Struktur Data)

Pertemuan 11, 12

Sub-CPMK

- Mahasiswa mampu membuat Double Linked-List dan mengakses data nya (C3, A3)

Materi

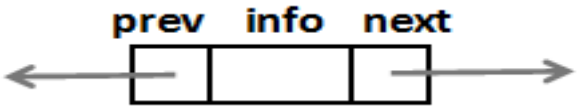
- Konsep Double Linked-List
- Menambahkan Node
- Mencari Node
- Menghapus Node
- Memindahkan Node

1.

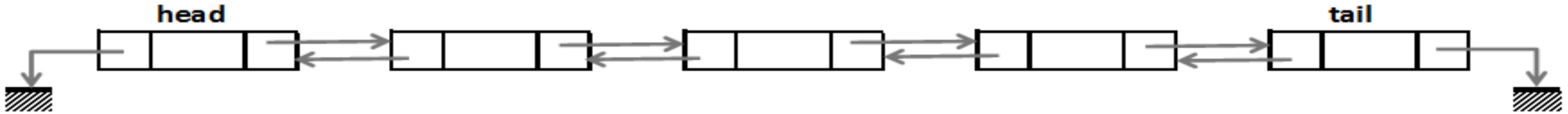
Konsep Double Linked-List

Double Linked List

- Setiap node memiliki dua link



- Previous Link menunjuk ke previous node
- Next Link menunjuk ke next node
- Head → Prev Link Pointed as NULL
- Tail → Next Link Pointed as NULL



Kelebihan Double Linked List

- Ketika kehilagan Head, asalkan Pointer masih menunjuk ke salah satu node/simpul, maka Head masih dapat dicari dengan melakukan Previous terus menerus sampai ditemukan simpul yang link previousnya menunjuk ke NULL

Operasi pada Double Linked List

- Sama seperti pada Single Linked List, Double linked list juga memiliki operasi-operasi cari, sisip/tambah dan hapus dengan mengarahkan previous Link ke previous node dari node yang akan dihapus
- Dengan adanya prev link pada tiap nodenya dapat mengurangi penambahan variabel pointer pada tiap operasinya

2.

Menambahkan Node

Insert pada bagian depan list

- Buat sebuah simpul/node baru
  ptrBaru = malloc(……..)
- Isi informasi simpul/node baru tersebut
- Arahkan next link ke simpul/node kepala
  ptrBaru->next = Head
- Arahkan prev link dari simpul kepala ke node baru
  Head->prev = ptrBaru
- Isi prev link node baru dengan NULL
  ptrBaru->prev = NULL
- Set head pointer ke simpul/node baru tersebut
  Head = ptrBaru

Insert di tengah – setelah current cell

- Arahkan Ptr ke node yang akan dibuat node baru pada next link nya
- Buat sebuah simpul/node baru
  Baru = malloc()
- Isi informasi simpul/node baru tersebut
- Copy next link dari current node/simpul ke next link simpul/node baru
  Baru->next = Ptr->next
- Copy prev link dari node setelah current node/simpul ke prev link simpul/node baru
  Baru->prev = Ptr->next->prev
- Set prev link pada simpul/node setelah current simpul/node kenode baru
  Ptr->next->prev = Baru
- Set next link pada current simpul/node ke simpul/node baru
  Ptr->next = Baru

Insert di tengah – sebelum current cell

- Arahkan Ptr ke node yang akan dibuat node baru pada prev link nya
- Buat sebuah simpul/node baru
  Baru = malloc()
- Isi informasi simpul/node baru tersebut
- Copy prev link dari current node/simpul ke prev link simpul/node baru
  Baru->prev = Ptr->prev
- Copy next link dari node sebelum current node/simpul ke next link simpul/node baru
  Baru->next = Ptr->prev->next
- Set next link pada simpul/node sebelum current simpul/node kenode baru
  Ptr->prev->next = Baru
- Set prev link pada current simpul/node ke simpul/node baru
  Ptr->prev = Baru

Insert pada bagian akhir list

- Arahkan Ptr ke node terakhir
- Buat sebuah node baru

  Baru = malloc()
- Isi informasi node baru tersebut
- Set next link dari baru node sebagai NULL

  Baru->next = NULL
- Isi prev link dari node baru ke Tail

  Baru->prev = TAIL
- Arahkan next link pada node terakhir atau tail ke node baru

  Ptr->next = Baru
- Jika memiliki variabel Tail, Jangan lupa memindahkan Tail ke node baru

  Tail = Baru

3.

Mencari Node

Mencari node pada Double LL

- Mencari node pada Double LL dapat dilakukan dengan cara maju menggunakan next link atau mundur menggunakan prev link
- Jika maju maka Ptr diarahkan ke Head
- Jika mundur maka Ptr diarahkan ke Tail

Pencarian Maju

- Proses sama seperti pada single LL
- Assign PointerCell sebagai Head

```
PointerCell = Head;
```

- Bergerak maju dengan mengarahkan PointerCell ke next PointerCell sampai ditemukan node/simpul yang sesuai.

```
PointerCell = PointerCell->Next;
```

Pencarian Mundur

- Assign PointerCell sebagai Head

```
PointerCell = Tail;
```

- Bergerak mundur dengan mengarahkan PointerCell ke prev PointerCell sampai ditemukan node/simpul yang sesuai.

```
PointerCell = PointerCell->prev;
```

4.

Menghapus Node

Delete Operation

Penghapusan dapat dilakukan:

- Pada bagian depan / delete head (WARNING!!!: don't lose the head!)
- Pada bagian tengah
- Pada bagian akhir / delete tail

- Karena ini adalah double linked-list, maka jangan lupa mengarahkan prev link dari node setelah node yang akan dihapus ke node pada prev link dari node yang akan dihapus

Hapus Pada bagian depan / delete head

- (WARNING!!!: don’t lose the head!)
- Arahkan pointer Ptr ke Head node sebagai current node
- Set variabel Head ke next dari current node

  Head = Head->Next
- Set prev link dari Head dengan NULL

  Head->prev = NULL
- Hapus current Node

  Free(Ptr)

Hapus Pada bagian tengah

- Arahkan Pointer Ptr ke node yang akan dihapus
- Copy link prev node dari node yangakan dihapus ke prev link node setelah node yang akan dihapus

  Ptr->next->prev = Ptr->Prev
- Copy link next node dari node yang akan dihapus ke next link node sebelum node yang akan dihapus

  Ptr->prev->next = Ptr->prev
- Hapus Current Node

  Free(Ptr)

Hapus Pada bagian akhir / delete tail

- Arahkan Pointer Ptr ke node yang akan dihapus Atau dalam hal ini adalah node terakhir
- Ubahlah next link dari node sebelum node yang akan dihapus dengan NULL

  Ptr->prev->next = NULL
- Hapus current node

  Free(Ptr)

5.

Memindahkan Node

Memindahkan Node pada Head ke Tengah

- Memindahkan Node dapat dilakukan dengan bantuan dua buah variabel pointer yaitu ptrTujuan serta variabel pointer Ptr.
- simpan alamat node di depan tujuan pemindahan ke variabel pointer ptrTujuan (gunakan operasi pencarian node)
- Simpan alamat node yang akan dipindahkan ke variabel pointer Ptr,

  Ptr = Head
- Pindahkan Head ke node berikutnya

  Head = Head->next
- Isi prev Head yagn sekarang dengan NULL

  Head->prev = NULL
- copykan next link dari node yang ditunjuk oleh ptrTujuan ke next link dari node yang ditunjuk oleh Ptr

  Ptr->next = ptrTujuan->next
- copykan prev link dari node setelah Node Tujuan ke node yang ditunjuk oleh Ptr

  Ptr->prev = ptrTujuan->next->prev
- Arahkan prev link dari node yang ditunjuk oleh node setelah node Tujuan ke Ptr

  ptrTujuan->next->prev=Ptr
- Arahkan next link dari node yang ditunjuk oleh ptrTujuan ke Ptr

  ptrTujuan->next=Ptr

Memindahkan Node pada Head ke Tail

- Memindahkan Node dapat dilakukan dengan bantuan dua buah variabel pointer yaitu ptrTujuan serta variabel pointer Ptr.
- simpan alamat node di depan tujuan pemindahan ke variabel pointer ptrTujuan yaitu node terakhir (gunakan operasi pencarian node atau jika mempunyai variabel pointer Tail dapat juga dengan mengcopykan Tail ke ptrTujuan : ptrTujuan = Tail)
- Simpan alamat node yang akan dipindahkan ke variabel pointer Ptr,

  Ptr = Head
- Pindahkan Head ke node berikutnya

  Head = Head->next
- Isi prev Head yang sekarang dengan NULL

  Head ->prev = NULL
- Arahkan next link dari node yang ditunjuk oleh ptrTujuan ke Ptr

  ptrTujuan->next=Ptr
- Arahkan prev link dari Ptr ke node ditunjuk oleh ptrTujuan

  Ptr->prev = ptrTujuan
- Karena dipindahkan ke Tail, jangan lupa mengubah isi next link dari node yang dipindahkan menjadi NULL

  Ptr->next = NULL

Catatan: Prinsipnya sama seperti memindahkan node dari Head ke Tengah

Memindahkan Node ditengah ke Head

- Memindahkan Node dapat dilakukan dengan bantuan sebuah variabel pointer Ptr.
- Simpan alamat node yang akan dipindahkan ke variabel pointer Ptr, (gunakan operasi pencarian node)
- copykan next link dari node yang ditunjuk oleh Ptr ke next link dari node sebelum dari node tersebut

  Ptr->prev->next = Ptr->next
- copykan prev link dari node yang ditunjuk oleh Ptr ke prev link dari node berikut dari node tersebut

  Ptr->next->prev = Ptr->prev
- Isi next link dari node yang ditunjuk oleh Ptr dengan isi dari variabel Head

  Ptr->next = Head
- Isi prev link dari node yang ditunjuk oleh Ptr dengan NULL

  Ptr->prev = NULL
- Arahkan Head ke node yang ditunjuk oleh Ptr

  Head = Ptr

Memindahkan Node ditengah ke Tengah (setelah current)

- Memindahkan Node dapat dilakukan dengan bantuan variabel pointer ptrTujuan serta sebuah variabel pointer Ptr.
- Simpan alamat node yang akan dipindahkan ke variabel pointer Ptr serta simpan alamat node didepan node tujuan ke variabel pointer ptrTujuan. (gunakan operasi pencarian node untuk mengarahkan masing-masing variabel pointer tersebut)
- copykan next link dari node yang ditunjuk oleh Ptr ke next link dari node sebelum dari node tersebut
  Ptr->prev->next = Ptr->next
- copykan prev link dari node yang ditunjuk oleh Ptr ke prev link dari node berikut dari node tersebut
  Ptr->next->prev = Ptr->prev
- Isi prev link dari node yang ditunjuk oleh Ptr ke node yang ditunjuk oleh ptrTujuan
  Ptr->prev= ptrTujuan
- Isi next link dari node yang ditunjuk oleh Pke dengan isi dari next link node yang ditunjuk oleh variabel ptrTujuan
  Ptr->next = ptrTujuan->next
- Isi prev link dari node yang ditunjuk oleh next link Ptr ke ptr
  ptrTujuan->next->prev=ptr atau Ptr->next->prev = ptr
- Isi next link dari node yang ditunjuk oleh ptrTujuan dengan isi dari next link node yang ditunjuk oleh variabel Ptr
  ptrTujuan->next = ptr

Memindahkan Node ditengah ke Tail

- Memindahkan Node dapat dilakukan dengan bantuan variabel pointer ptrAsal dan ptrTujuan serta sebuah variabel pointer Ptr.
- Simpan alamat node yang akan dipindahkan ke variabel pointer Ptr, serta alamat node didepan node tujuan ke variabel pointer ptrTujuan.
- copykan next link dari node yang ditunjuk oleh Ptr ke next link dari node sebelum dari node tersebut

  Ptr->prev->next = Ptr->next
- copykan prev link dari node yang ditunjuk oleh Ptr ke prev link dari node berikut dari node tersebut

  Ptr->next->prev = Ptr->prev
- Isi next link dari node yang ditunjuk oleh ptrTujuan dengan alamat node yang ditunjuk oleh variabel Ptr

  ptrTujuan->next = Ptr
- Isi prev link dari Ptr node yang ditunjuk oleh ptrTujuan

  Ptr->prev = ptrTujuan
- Karena dipindahkan ke Tail, jangan lupa mengubah isi next link dari node yang dipindahkan menjadi NULL

  Ptr->next = NULL

Memindahkan Node pada Tail ke Head

- Memindahkan Node dapat dilakukan dengan bantuan variabel pointer Ptr.
- Simpan alamat node terakhir (Tail) ke Ptr
- NULL kan next link dari node yang ditunjuk oleh node sebelum node yang akan dipindahkan
  Ptr->prev->next=NULL
- Arahkan next link dari node yang ditunjuk oleh Ptr ke Head
  Ptr->next = Head
- Arahkan prev link dari Head ke node yang ditunjuk oleh Ptr
  Head->prev = Ptr
- Pindahkan Head ke node yang ditunjuk oleh Ptr
  Head = Ptr
- Jangan lupa NULL prev link dari HEAD yang baru
  Head->prev = NULL atau Ptr->prev = NULL
- Jangan lupa jika mempunyai variabel Tail, copykan ptrTujuan ke Tail (Tail=ptrAsal)

Memindahkan Node pada Tail ke tengah

- Memindahkan Node dapat dilakukan dengan bantuan variabel pointer ptrTujuan serta sebuah variabel pointer Ptr.
- Simpan alamat node yang akan dipindahkan (Tail) ke variabel pointer Ptr serta simpan alamat node didepan node tujuan ke variabel pointer ptrTujuan.
- Karena yang dipindahkan ada pada Tail, proses pertama sebelum memindah Tail ke tempat tujuan, NULL kan terlebih dahulu node sebelum Tail

  Ptr->prev->next = NULL
- Isi prev link dari node yang ditunjuk oleh Ptr ke node yang ditunjuk oleh ptrTujuan

  Ptr->prev= ptrTujuan
- Isi next link dari node yang ditunjuk oleh Pke dengan isi dari next link node yang ditunjuk oleh variabel ptrTujuan

  Ptr->next = ptrTujuan->next
- Isi prev link dari node yang ditunjuk oleh next link Ptr ke ptr

  ptrTujuan->next->prev=ptr atau Ptr->next->prev = ptr
- Isi next link dari node yang ditunjuk oleh ptrTujuan dengan isi dari next link node yang ditunjuk oleh variabel Ptr

  ptrTujuan->next = Ptr

Ringkasan

- Penggunaan Double Linked List dapat mengurangi pemakaian variabel yang diperlukan pada proses tambah, hapus dan pindah node
- Proses hapus, pindah dan tambah node pada double linked-list jangan sampa melupakan proses untuk memindahkan previous link dari node-node yang dipindahkan, dan node-node yang berhubungan pemindahan tersebut
- Detail proses penghapusan, penambahan, pencarian dan pemindahan node dapat dilihat pada masing-masing slide, proses tersebut hanya salah satu contoh proses saja, banyak variasi proses yang lain

Terimakasih

TUHAN Memberkati Anda

Teady Matius Surya Mulyana (tmulyana@bundamulia.ac.id)