



Multinary Tree

(TIB11 – Struktur Data)

Pertemuan 23, 24

Sub-CPMK

- Mahasiswa mampu menyusun data di dalam simpul-simpul multiary tree (C3, A3)

Materi

- Pengertian Multiary Tree
- B-Tree



1.

Pengertian Multiary Tree

Multiary Tree

- Multiary Tree / N-ary Tree adalah Tree dengan setiap simpul yang dapat memiliki anak lebih dari dua
- Multiary Tree dapat diterapkan dengan cara:
 1. Node dengan sebuah ke anak pertama dan sebuah link ke saudara. Cara ini memiliki penyimpanan yang optimal
 2. Sebuah node dengan banyak Link anak, cara ini memiliki keterbatasan jumlah anak dari suatu node, biasanya di impementasikan pada n-ary tree

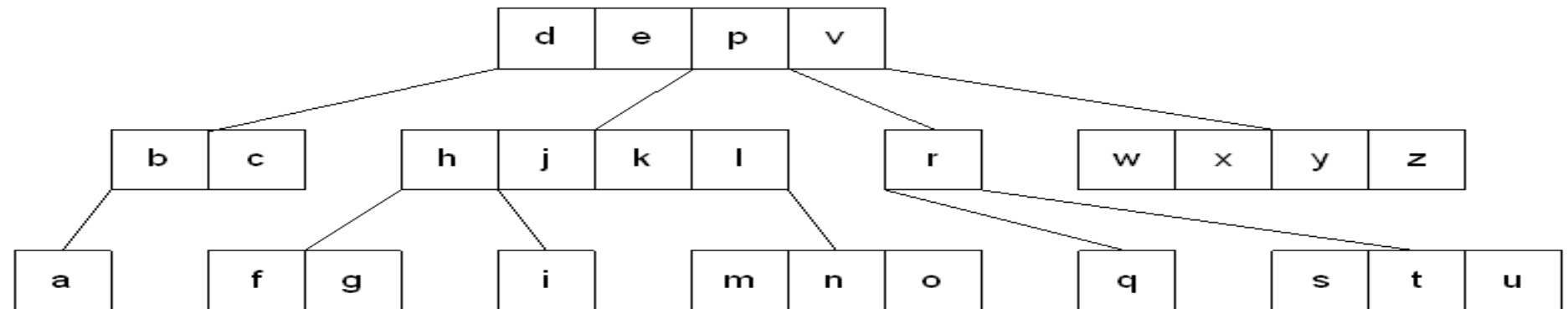
Multiway Tree

- Multiway Tree merupakan salah satu bentuk dari n-ary Tree / multiary tree
 - Salah satu varian dari multiway tree adalah B-Tree
- Catatan: harap diperhatikan B-Tree bukan Binary Tree, melainkan generalisasi dari binary tree untuk merepresentasikan multiary tree sehingga setiap node dapat memiliki anak lebih dari dua dengan setiap key di dalam node tersebut hanya memiliki dua anak agar lebih teratur dan mudah diakses

Contoh Multiway Tree

Contoh

- Multiway Tree dengan Order 5 yg memiliki 4 key dan 5 cabang
- Key d hanya memiliki satu anak kiri
- Key p memiliki dua: anak kiri (h j k l) dan anak kanan (r)
- Key r merupakan anak kanan dari key p dan sekaligus anak kiri dari key v



Multiway Tree

1. Setiap node/simpul memiliki m anak dan $m-1$ key/kunci
2. Key pada setiap node/simpul disusun secara ascending
3. Key pada first- i children lebih kecil dari pada key ke i
4. Key pada last $m-i$ children lebih besar dari pada key ke i



2.

B-Tree

B-Tree

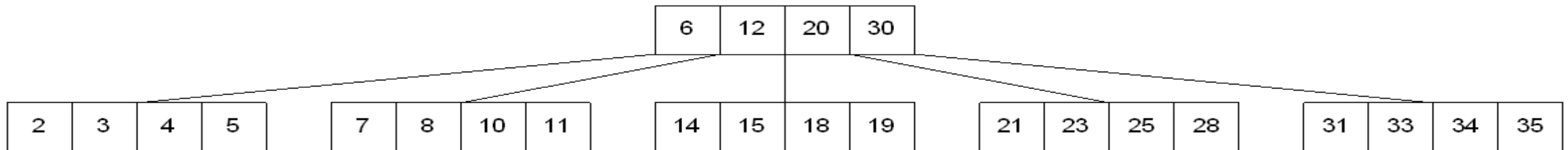
- Salah satu varian dari multiway Tree adalah B-Tree.
- Size pada setiap node dapat dibuat sebagai size dari block
- Jumlah key pada setiap node dapat bervariasi tergantung ukuran key
- Size dari sebuah block bervariasi untuk setiap system

B-tree order

B-tree dengan order m adalah sebuah sebuah multiways search tree

- Setiap root selain leaf memiliki paling tidak dua subtree
- Setiap nonroot dan setiap each nonleaf memiliki $k-1$ keys dan k pointers ke subtrees dimana $\lceil m/2 \rceil \leq k \leq m$
- Setiap leaf node memiliki $k-1$ keys where $\lceil m/2 \rceil \leq k \leq m$
- Semua leaf selalu pada level yang sama

Contoh B-Tree



Insertion

- Dibentuk secara bottom to up
- Root selalu berubah
- Jika ada sebuah ruang pada leaf tujuan, node dapat langsung disisipkan
- Jika semua leaf terisi
 - Bentuk leaf lain
 - Keys dibagi antara diantara leaf tersebut dan sebuah key diproposikan menjadi parent.
 - Dalam hal ini jika parent penuh, ulangi proses sampai mencapai root dan sebuah root dibentuk

Kondisi Insertion

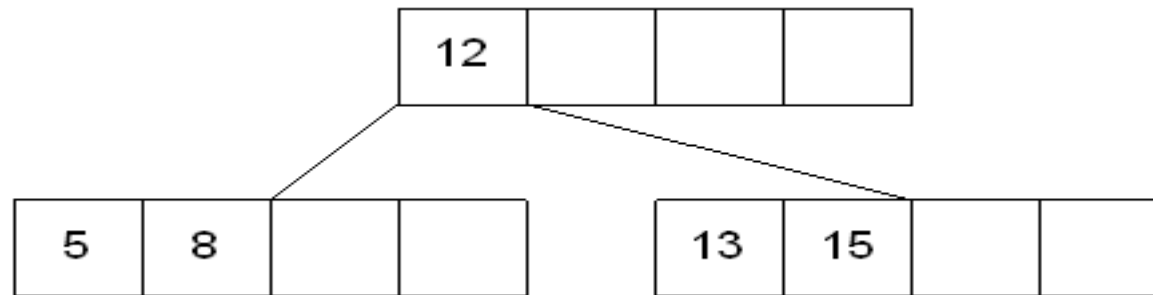
- Masih terdapat ruang untuk menempatkan sebuah leaf
- Leaf yang akan ditempatkan bisa jadi sudah penuh
- Root dari B-tree sudah penuh

Menyisipkan Key pada B-Tree

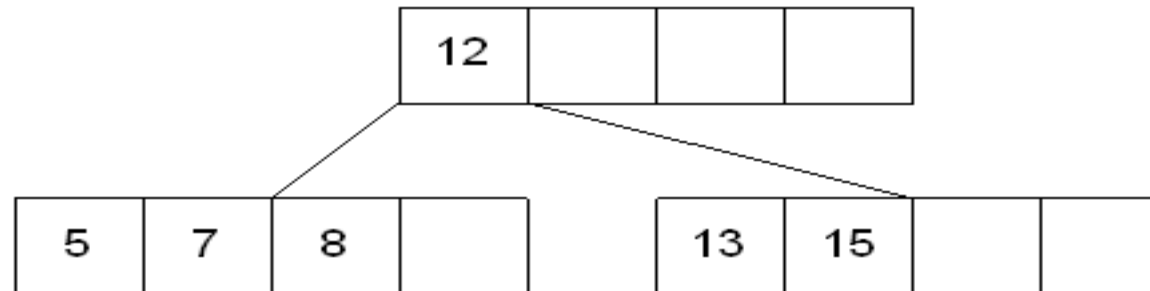
Tiga situasi yang umum terjadi

- Key di tempatkan pada leaf leaf yang masih memiliki ruang (dengan tetap menjaga ascending order)
- Leaf yang akan diisi key sudah penuh
- Special case terjadi jika root B-tree sudah penuh

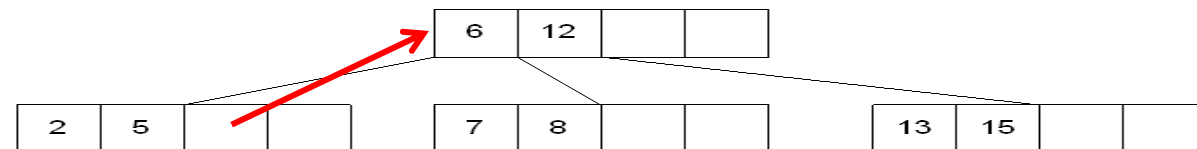
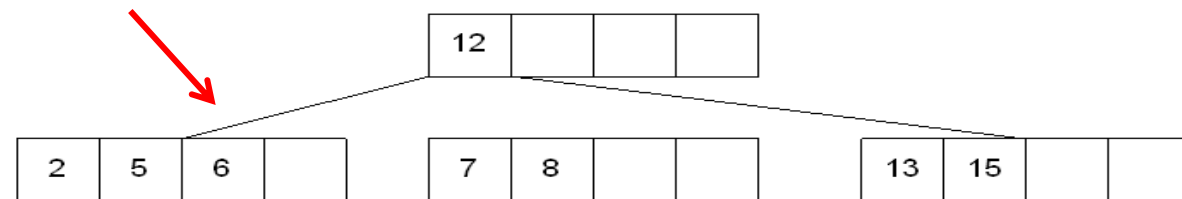
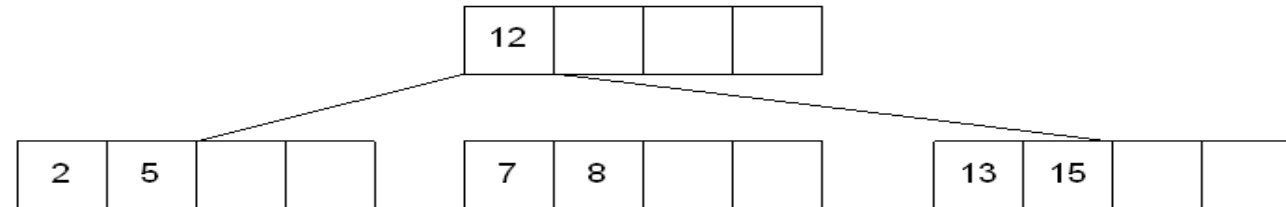
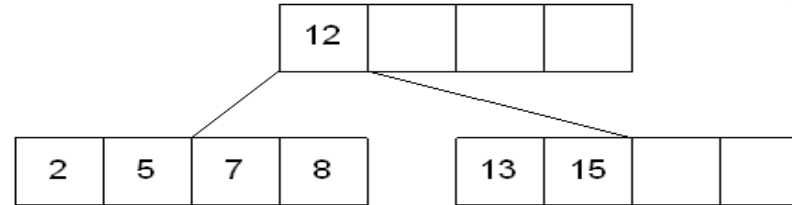
Contoh Penyisipan



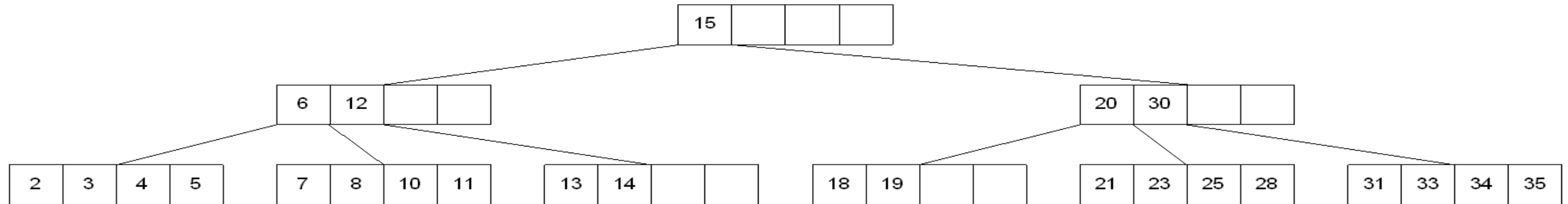
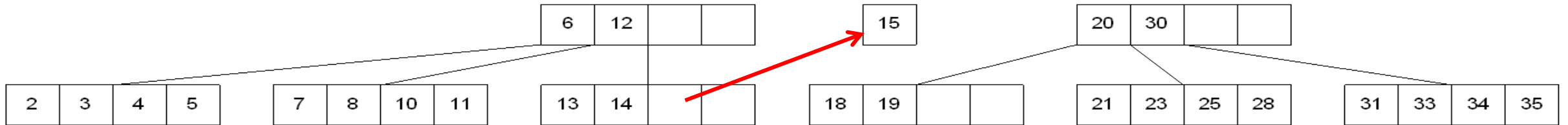
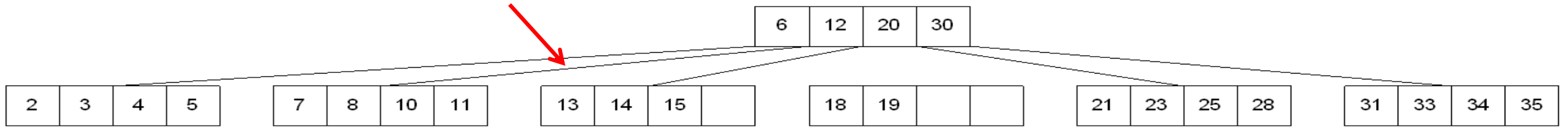
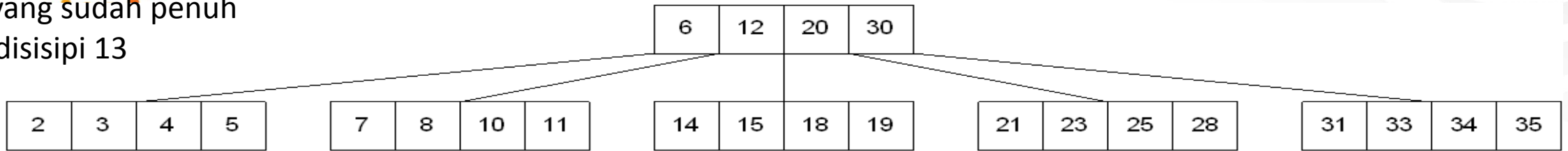
Menyisipkan data 7



Menyisipkan 6 Pada tree berikut ini



Tree yang sudah penuh
akan disisipi 13



Menghapus Key dari B-Tree

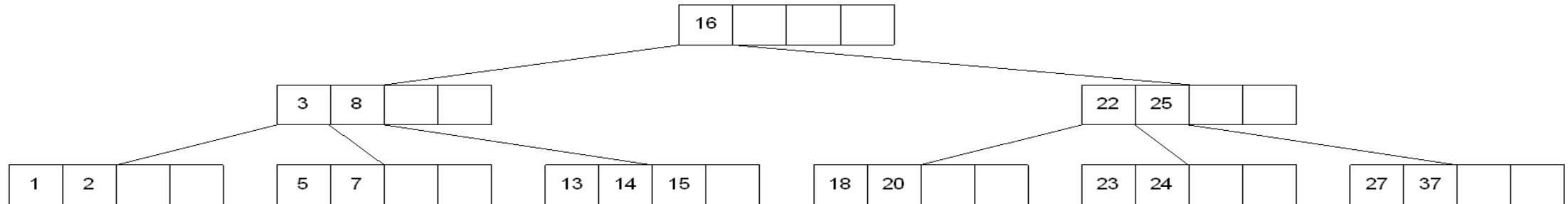
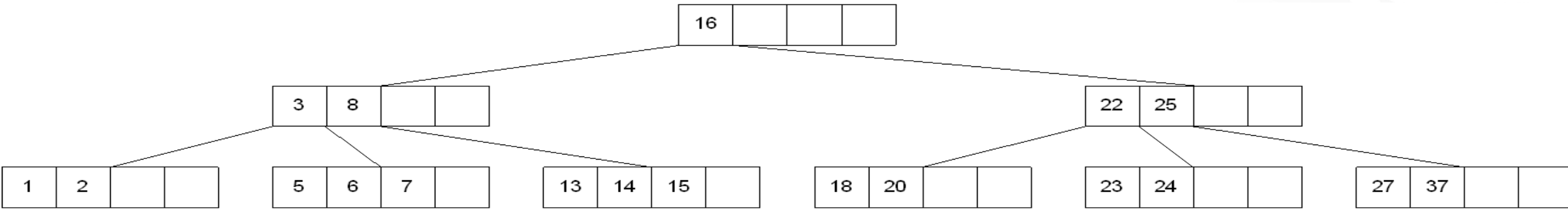
Ada dua kasus

- Hapus key dari leaf node
- Hapus key dari nonleaf node
 - Similar dengan deletingByCopying()

Menghapus Dari Leaf

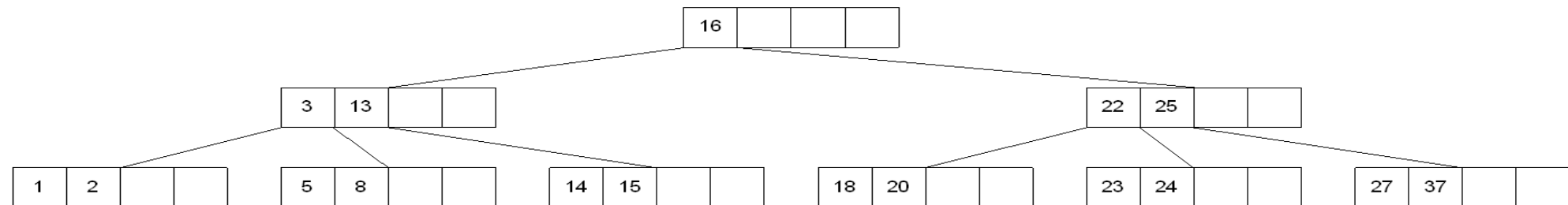
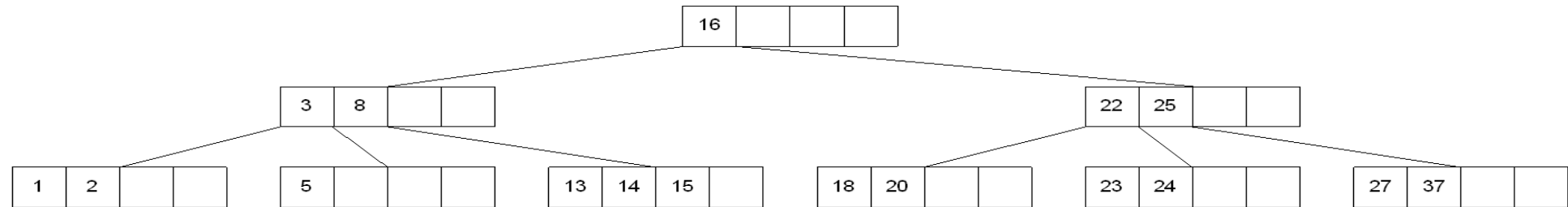
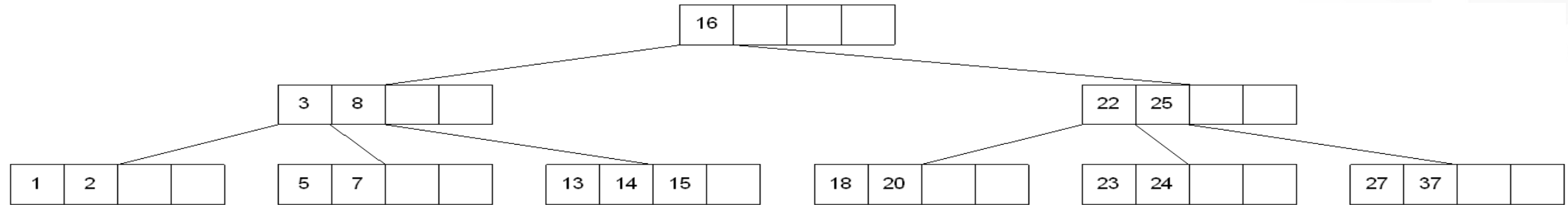
Kondisi

- Jika jumlah key setelah dihapus lebih besar atau sama dengan $m/2$
 - Hapus key
 - Sisipkan key dari sebelah key yang dihapus untuk mengisi sel yang kosong



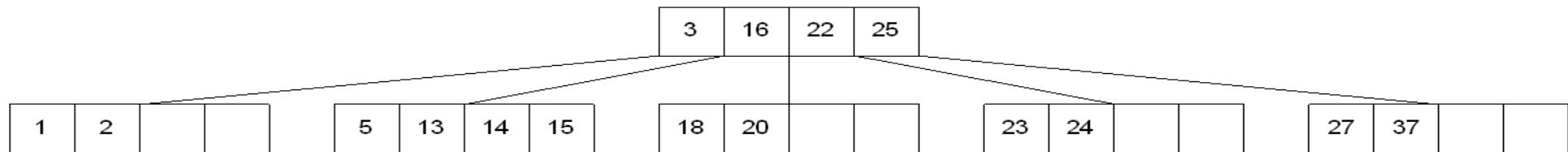
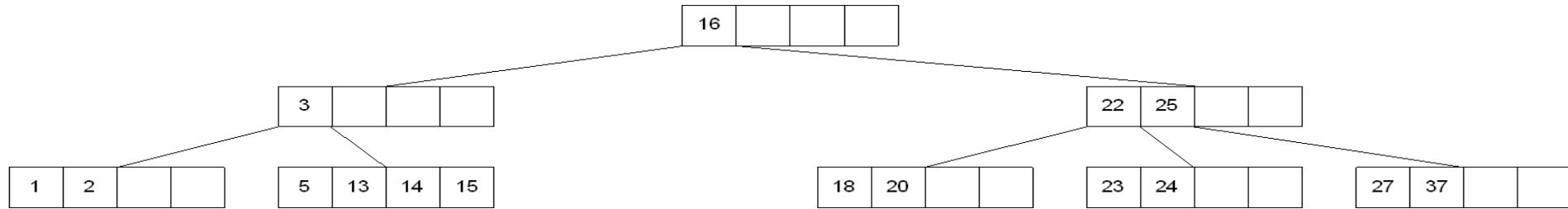
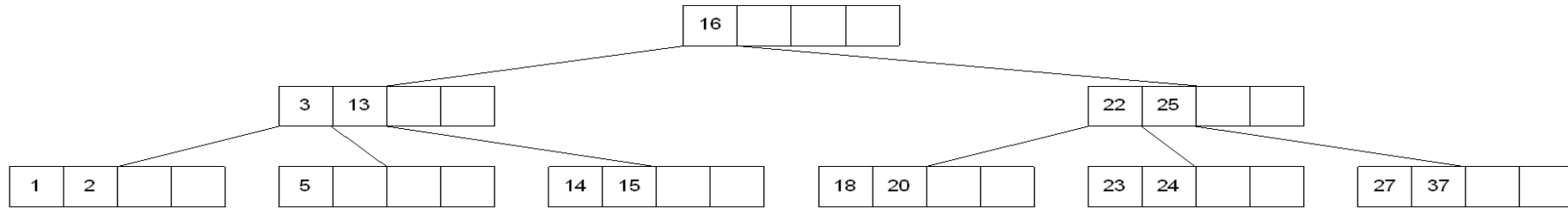
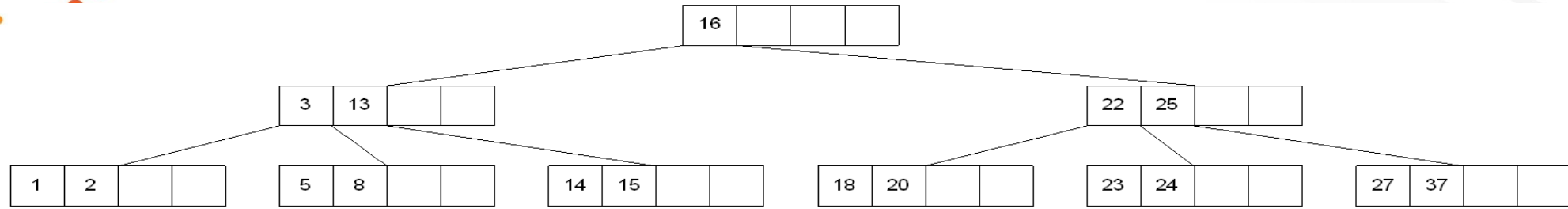
Menghapus Dari Leaf (cont.)

- Jumlah key setelah penghapusan kurang dari $m/2$
 - Jika jumlah key pada kiri atau kanan sibling cukup untuk mengisi kekurangan node target
 - Ambil key dari parent dan insert menjadi node.
 - Ambil key dari kanan atau kiri sibling untuk mengisi key pada parent node



Menghapus Dari Leaf (cont.)

- Jika jumlah key setelah penghapusan kurang dari $m/2$ (cont.)
 - Jika jumlah key pada kiri atau kanan sibling tidak cukup untuk mengisi kekurangan sebagai target node
 - Merge key sibling dari kiri atau kanan, dengan mengambil key dari parent node meletakkan diantara nya
 - Geser key sebelah kanan parent node untuk mengisi sel kosong setelah key tersebut diambil untuk menggantikan anaknya.
 - Jika jumlah key pada prent kurang dari $m/2$, dapat diisi dengan proses untuk mengisi node kosong pada proses yang sama dengan menghapus non leaf



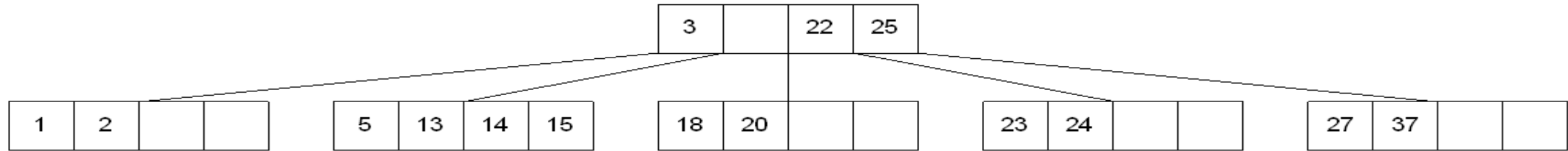
Menghapus dari node non-leaf

- Key yang lebih besar dari yang dihapus dipindah ke kiri untuk menggantikan yang sudah dihapus
- Jumlah key pada leaf harus lebih besar sama dengan $\lceil m/2 \rceil$

Menghapus dari node non-leaf (cont.)

- Penghapusan dari node non-leaf lebih sedikit prosesnya daripada menghapus dari leaf. Deletion from a non-leaf node is reduced to deleting a key from a leaf
- Key yang dihapus direplace dengan immediate predecessor nya (dapat juga dengan successor) yang dapat ditemukan pada sebuah leaf

contoh mengganti simpul terhapus dengan predesesor



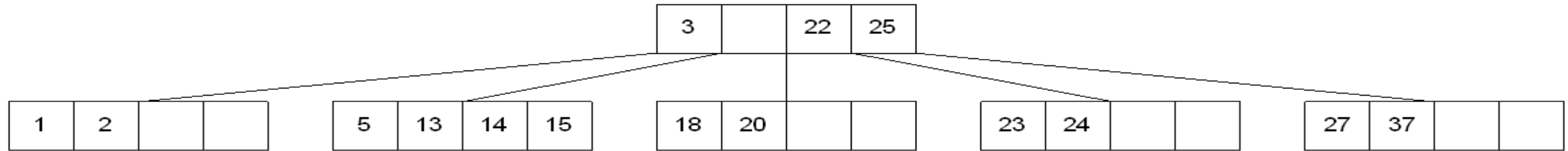
Isi sel terhapus dengan predesesor



immediate predesesor dihapus



contoh mengganti simpul terhapus dengan suksesor



Isi sel terhapus dengan suksesor



Suksesor dihapus sehingga data 20 geser ke kiri



Jika kondisi menjadi tidak seimbang, maka perlu dilakukan penyesuaian lagi



Catatan:

- Contoh-contoh disadur dari:

<http://faculty.cs.niu.edu/~freedman/340/340notes/340multi.htm>

Ringkasan

- N-Ary tree atau Multiary Tree merupakan tree yang dapat memiliki maksimum n anak pada setiap node nya.
- Multiway Tree merupakan salah satu bentuk dari n -ary Tree
- Salah satu varian dari multiway Tree adalah Btree.



Terimakasih

TUHAN Memberkati Anda

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