Praktikum Struktur Data

Pertemuan Ke-12

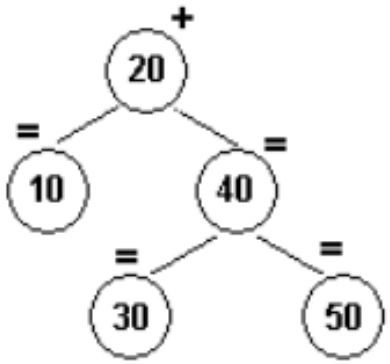
AVL TREE

- AVL Tree adalah Binary Search Tree yang memiliki perbedaan tinggi/ level maksimal 1 antara subtree kiri dan subtree kanan.
- AVL Tree muncul untuk menyeimbangkan Binary Search Tree. Dengan AVL Tree, waktu pencarian dan bentuk tree dapat dipersingkat dan disederhanakan.

Penambahan Node Ke AVL Tree

- Apabila nilai node yang mau ditambahkan lebih kecil daripada nilai node pembanding maka penelusuran pindah ke anak sebelah kiri (left child).
- Apabila nilai node yang mau ditambahkan lebih besar daripada nilai node pembanding maka penelusuran pindah ke node sebelah kanan (right child).
- Setelah node baru dikaitkan pada pohon dilakukan pemeriksaan apakah selisih tinggi pada setiap node tidak lebih dari satu level.
- Apabila selisih tinggi lebih dari satu level maka dilakukan pembenahan dengan algoritma **single rotation** atau **double rotation**.

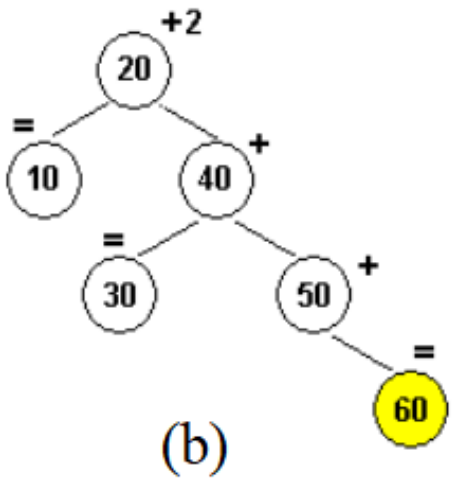
Node Pada AVL Tree



(a)

- Pada setiap node AVL tree pada gambar (a) dilengkapi tanda keseimbangan tinggi (height balance).
- Tanda (=) menyatakan left subtree dan right tree dari node ini mempunyai height yang sama.
- Tanda (–) menyatakan left subtree dari node ini mempunyai height lebih besar dari right subtree-nya. Node yang mempunyai height balance – disebut tallleft.
- Tanda (+) menyatakan right subtree dari node ini mempunyai height lebih besar daripada left subtree-nya. Node yang mempunyai height balance + disebut tallright.

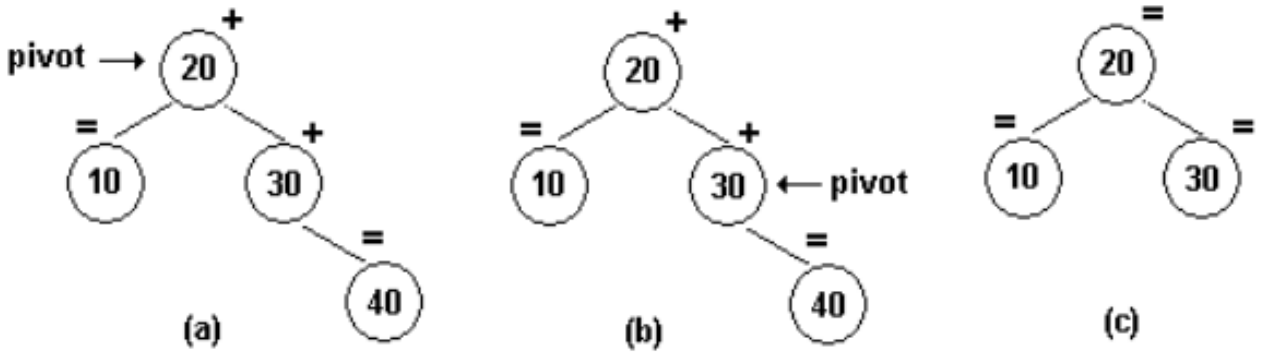
Penambahan Node Ke AVL Tree



(b)

- AVL tree pada gambar (a) akan ditambahkan node dengan nilai kunci 60.
- Penelusuran dimulai dari root, node 40, dan node 50. Jalur ini disebut search path.
- Node baru dengan nilai kunci 60 akan ditambahkan sebagai right child dari node 50.
- Penambahan node baru ini menyebabkan pohon tidak lagi
- bersifat AVL karena node 20 mempunyai selisih height 2 pada right subtree (gambar (b)).

Pivot Node

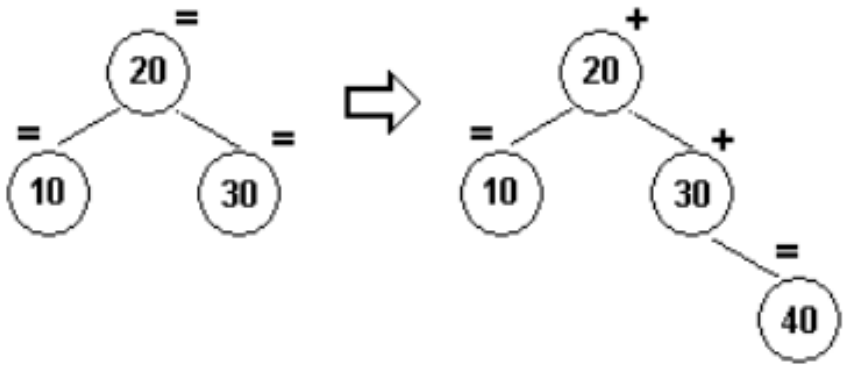


- Pivot node adalah node pada search path yang keseimbangannya tallleft atau tallright dan paling dekat ke posisi node baru yang akan ditambahkan.
- Pada AVL tree gambar (b) pivot node adalah node dengan nilai kunci 20.
- Pada gambar (a) akan ditambahkan node baru dengan nilai kunci 15, maka yang menjadi pivot node adalah node dengan nilai kunci 20.
- Pada gambar (b) akan ditambahkan node baru dengan nilai kunci 45, maka yang menjadi pivot node adalah node dengan nilai kunci 30.
- Pada gambar (c) tidak ada pivot node karena semua node dalam keadaan seimbang.

Kasus yang dapat terjadi pada penambahan node baru ke dalam AVL tree

Kasus 1:

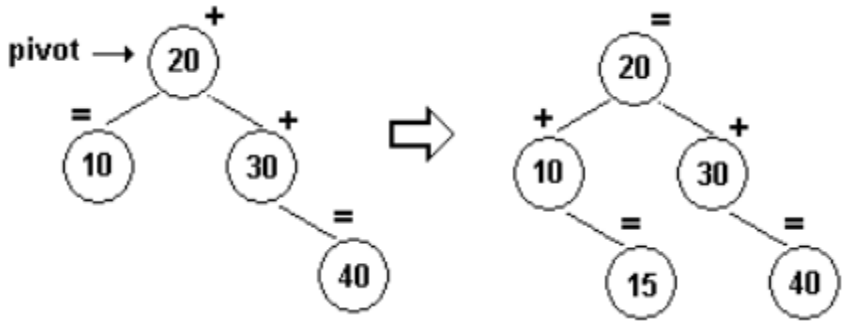
- Tidak ada pivot node pada AVL tree karena semua node dalam keadaan seimbang.
- Penambahan node dapat langsung dilakukan dan kemudian semua node pada search path dihitung kembali height balance-nya.



Kasus yang dapat terjadi pada penambahan node baru ke dalam AVL tree

Kasus 2:

- Pada AVL tree terdapat pivot node dan subtree tempat node baru akan ditambahkan mempunyai height balance yang lebih kecil, artinya pivot node menyatakan tallleft dan node baru akan ditambahkan pada posisi right subtree atau pivot node menyatakan tallright dan node baru akan ditambahkan pada posisi left subtree.
- Penambahan node dapat langsung dilakukan dan kemudian semua node pada search path mulai dari pivot node dihitung kembali height balance-nya.

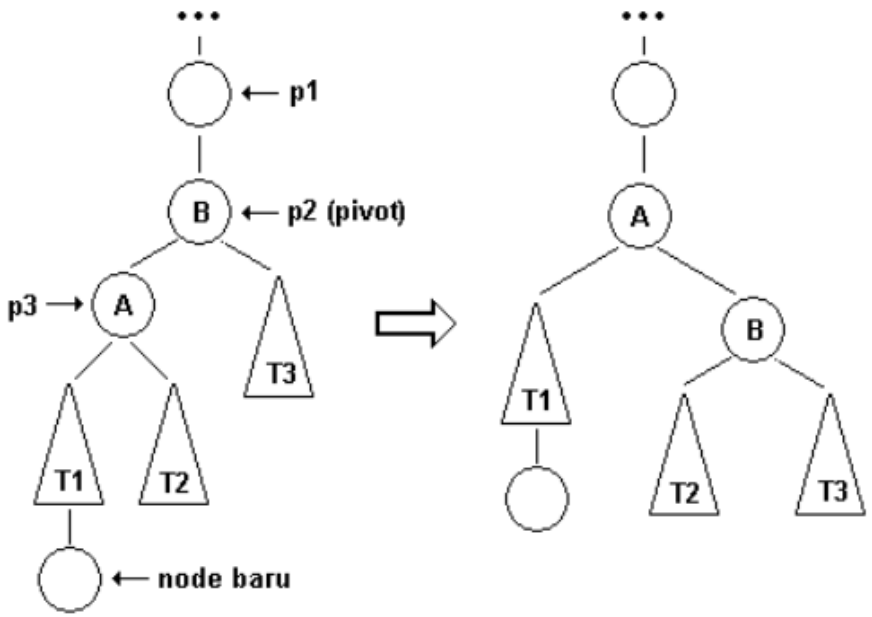


Kasus yang dapat terjadi pada penambahan node baru ke dalam AVL tree

Kasus 3:

- Pada AVL tree terdapat pivot node dan subtree tempat node baru akan ditambahkan mempunyai height balance yang lebih besar, sehingga penambahan node baru membuat subtree semakin tidak seimbang.
- Pemecahan dilakukan dengan rotasi tunggal (single rotation) atau rotasi ganda (double rotation).

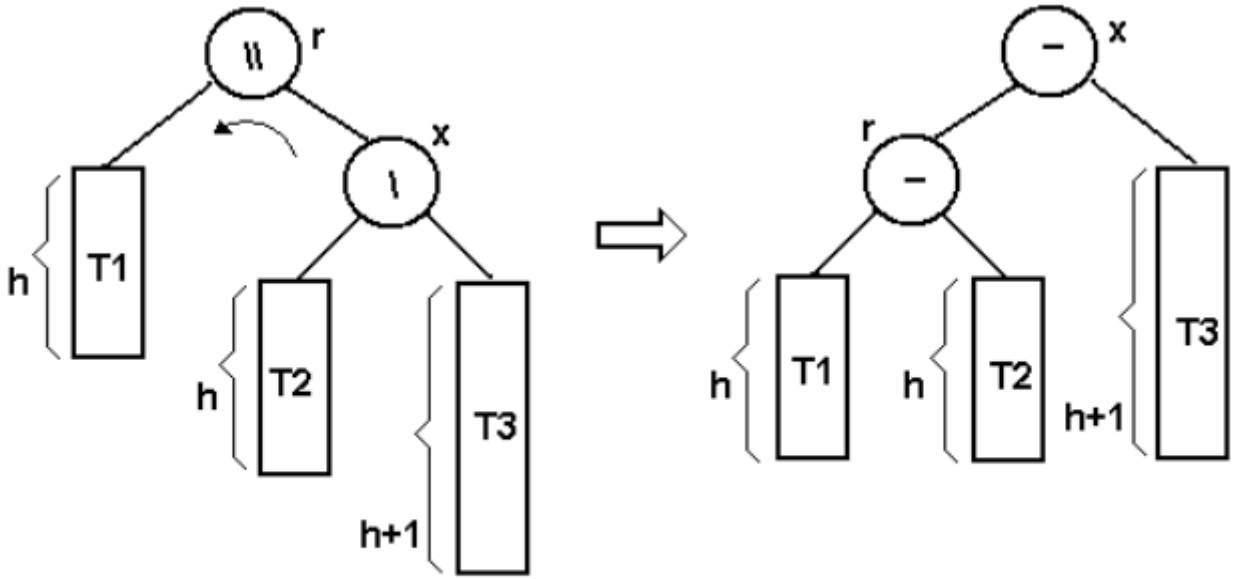
Single Rotation



Gambar 1

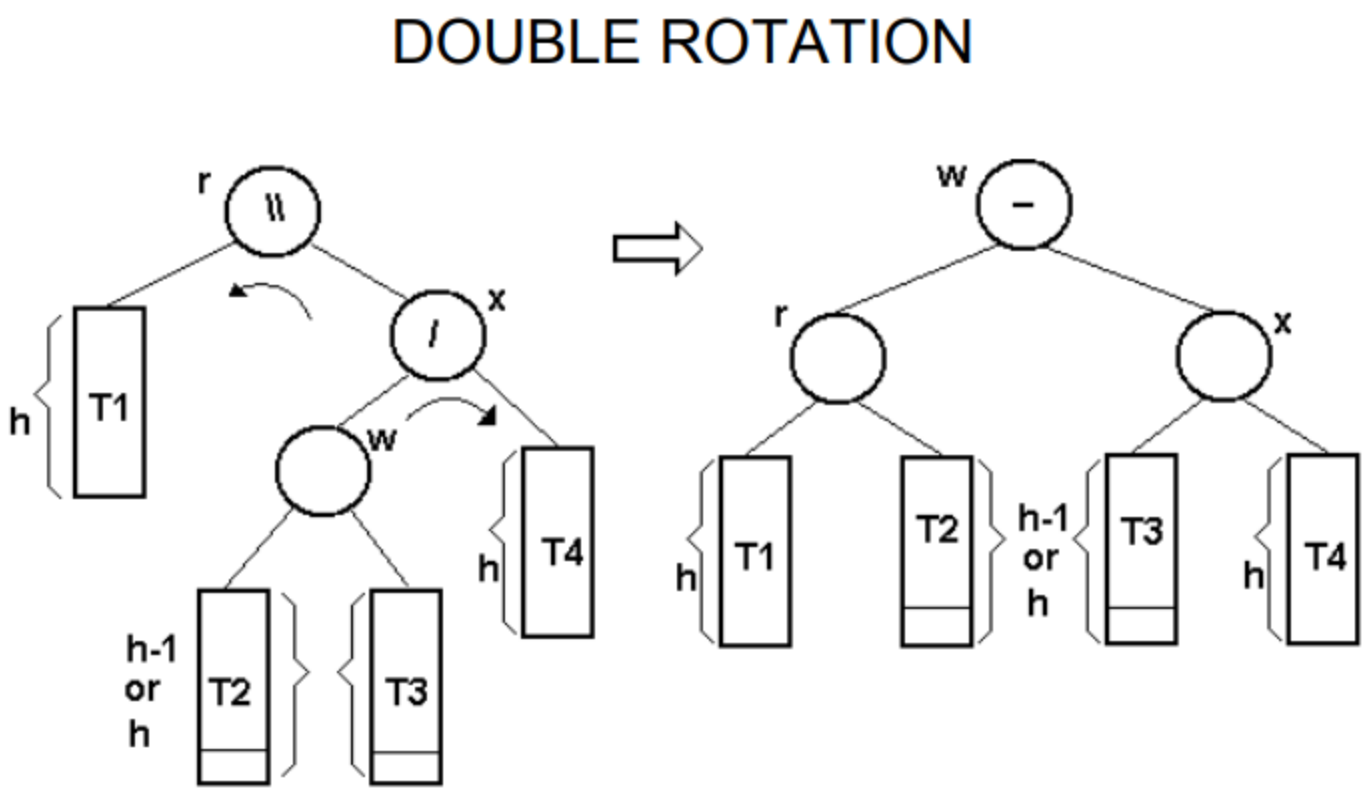
- Single rotation dilakukan bila kondisi AVL tree waktu akan ditambahkan node baru dan posisi node baru seperti pada gambar 1.
- T1, T2, dan T3 adalah subtree yang urutannya harus seperti demikian serta heightnya harus sama (≥ 0). Hal ini juga berlaku untuk AVL tree yang merupakan citra cermin (mirror image) gambar 1.

Single Rotation



Gambar Pola Single Rotation Menurut buku Robert L. Kruse

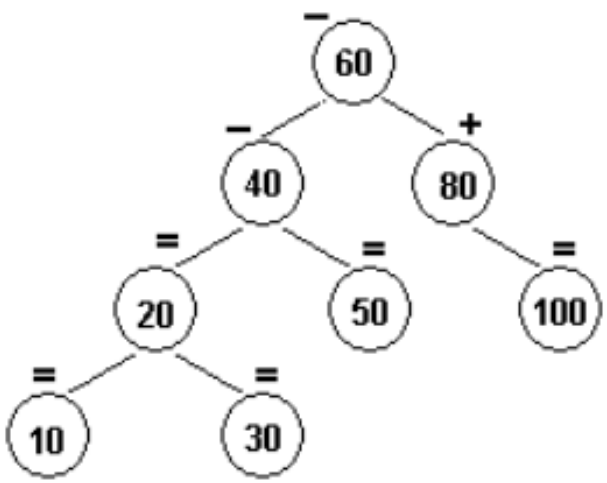
Double Rotation



Gambar Pola Double Rotation Menurut Buku Robert L. Kruse

Contoh Single Rotation

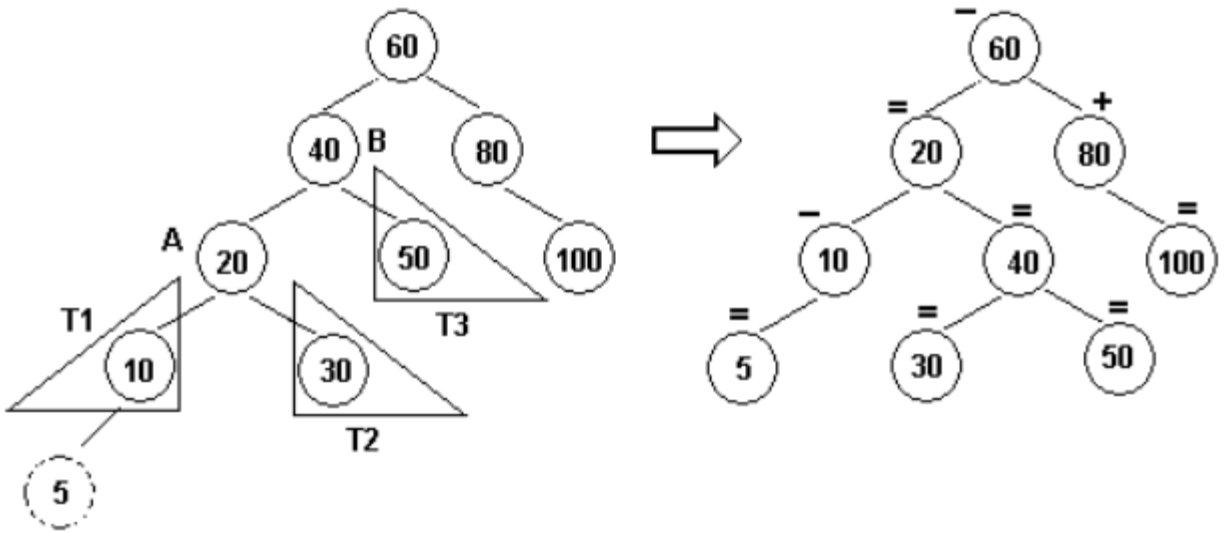
- Misalkan terhadap AVL tree pada gambar 2 akan ditambahkan node bernilai kunci 5.
- Search path melalui node 60, 40, 20, 10 dan pivot node adalah node dengan nilai 40.



Gambar 2

Contoh Single Rotation

- T1 adalah subtree dengan node 20, T2 adalah subtree dengan node 30, dan T3 adalah subtree dengan node 50.
- Ketiga subtree ini tingginya satu level.
- Pada gambar 3 kiri. Maka rotasi membentuk AVL tree seperti gambar 3 kanan.

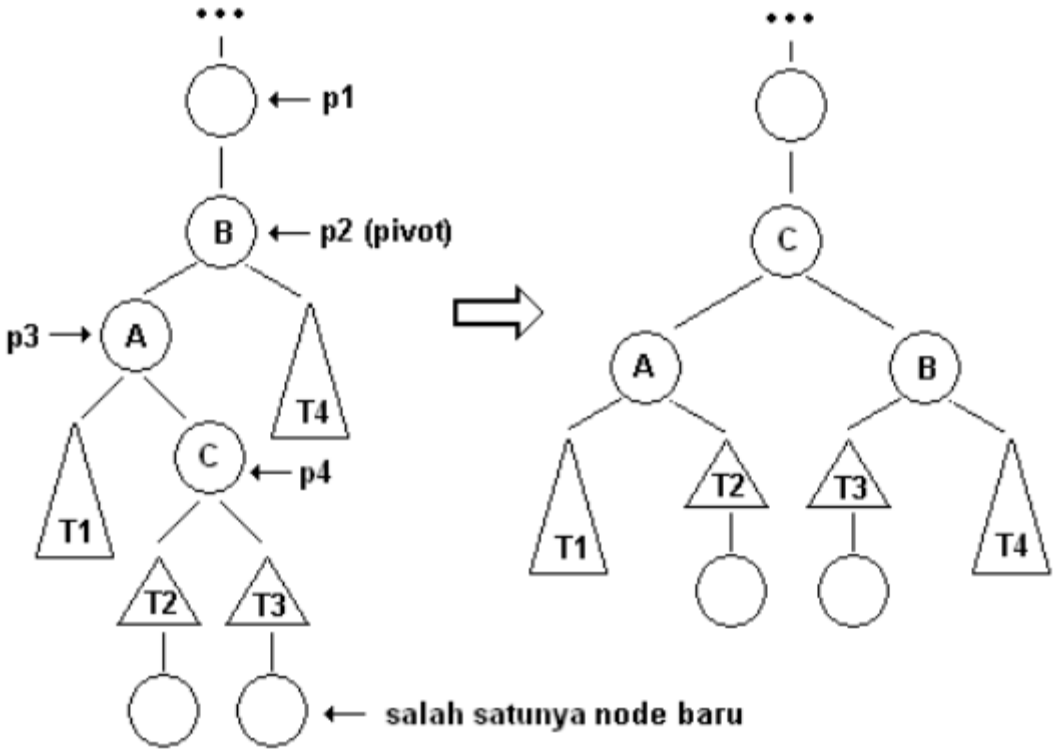


Gambar 3

Double Rotation

- Double rotation dilakukan bila kondisi AVL tree saat akan ditambahkan node baru dan posisi node baru seperti pada gambar 4.
- T1, T2, T3, dan T4 adalah subtree yang urutannya harus seperti demikian.
- Tinggi subtree T1 harus sama dengan T4 (≥ 0), tinggi subtree T2 harus sama dengan T3 (≥ 0). Node yang ditambahkan akan menjadi child dari subtree T2 atau T3.
- Hal ini juga berlaku untuk AVL tree yang merupakan citra cermin (mirror image) gambar 4.

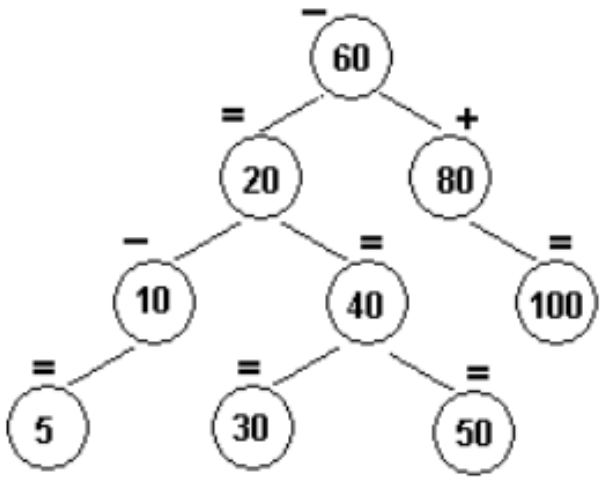
Double Rotation



Gambar 4. Pola Double Rotation

Contoh Double Rotation

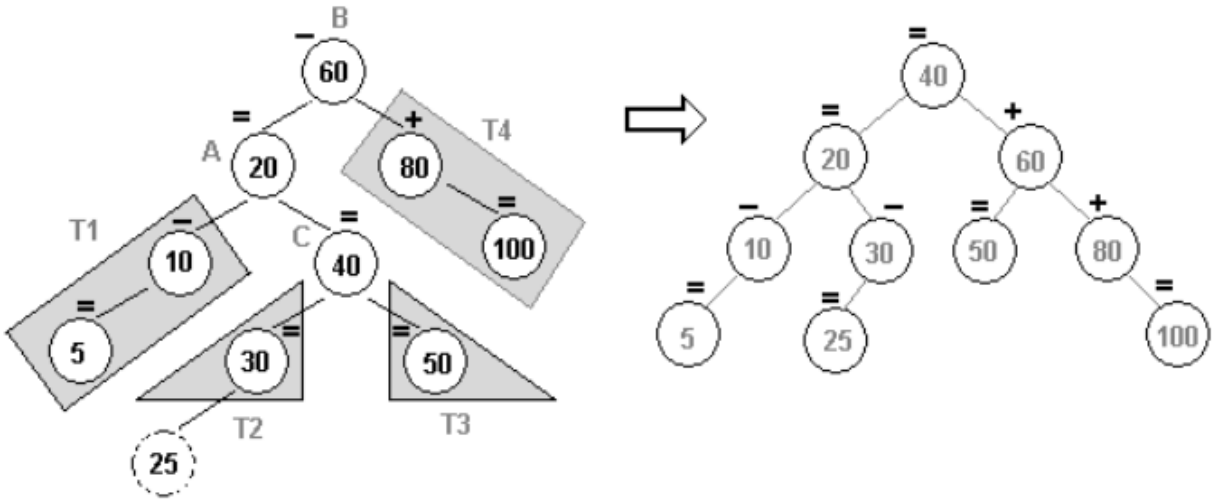
- Misalkan AVL tree pada gambar 5 akan ditambahkan node bernilai kunci 25.
- Search path melalui node 60, 20, 40, 30 dan pivot node adalah node dengan nilai 60.



Gambar 5

Contoh Double Rotation

- T1 adalah subtree dengan node 10 dan 5 (tingginya 2 level), T2 adalah subtree dengan node 30 (tingginya 1 level), T3 adalah subtree dengan node 50 (tingginya 1 level). Dan T4 adalah subtree dengan node 80 dan 100 (tingginya 2 level).
- T1 dan T4 mempunyai tinggi yang sama, demikian pula denan T2 dan T3.
- Node baru yang ditambahkan akan menjadi leftchild dari T2.
- Lihat gambar 6 kiri. Maka rotasi membentuk AVL tree seperti gambar 6 kanan



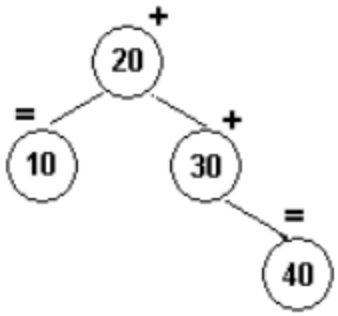
Gambar 6

Menghapus Node di AVL Tree

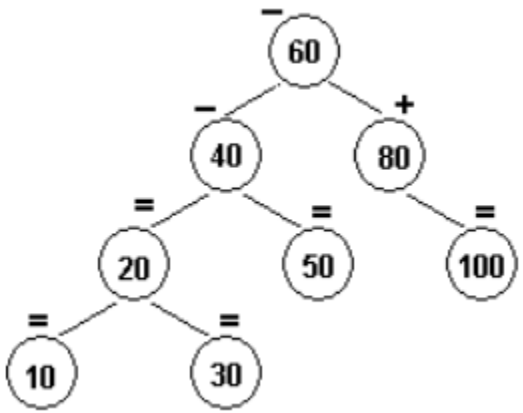
- Proses menghapus sebuah node di AVL tree hampir sama dengan BST.
- Penghapusan sebuah node dapat menyebabkan tree tidak imbang Setelah menghapus sebuah node, lakukan pengecekan dari node yang dihapus → root.
- Gunakan single atau double rotation untuk menyeimbangkan node yang tidak imbang. Pencarian node yang imbalance diteruskan sampai root.

Latihan

1. Tambahkan node 60 pada gambar di samping!



2. Tambahkan node 35 pada gambar di samping!




Thank you!