Tree
(ادامه)

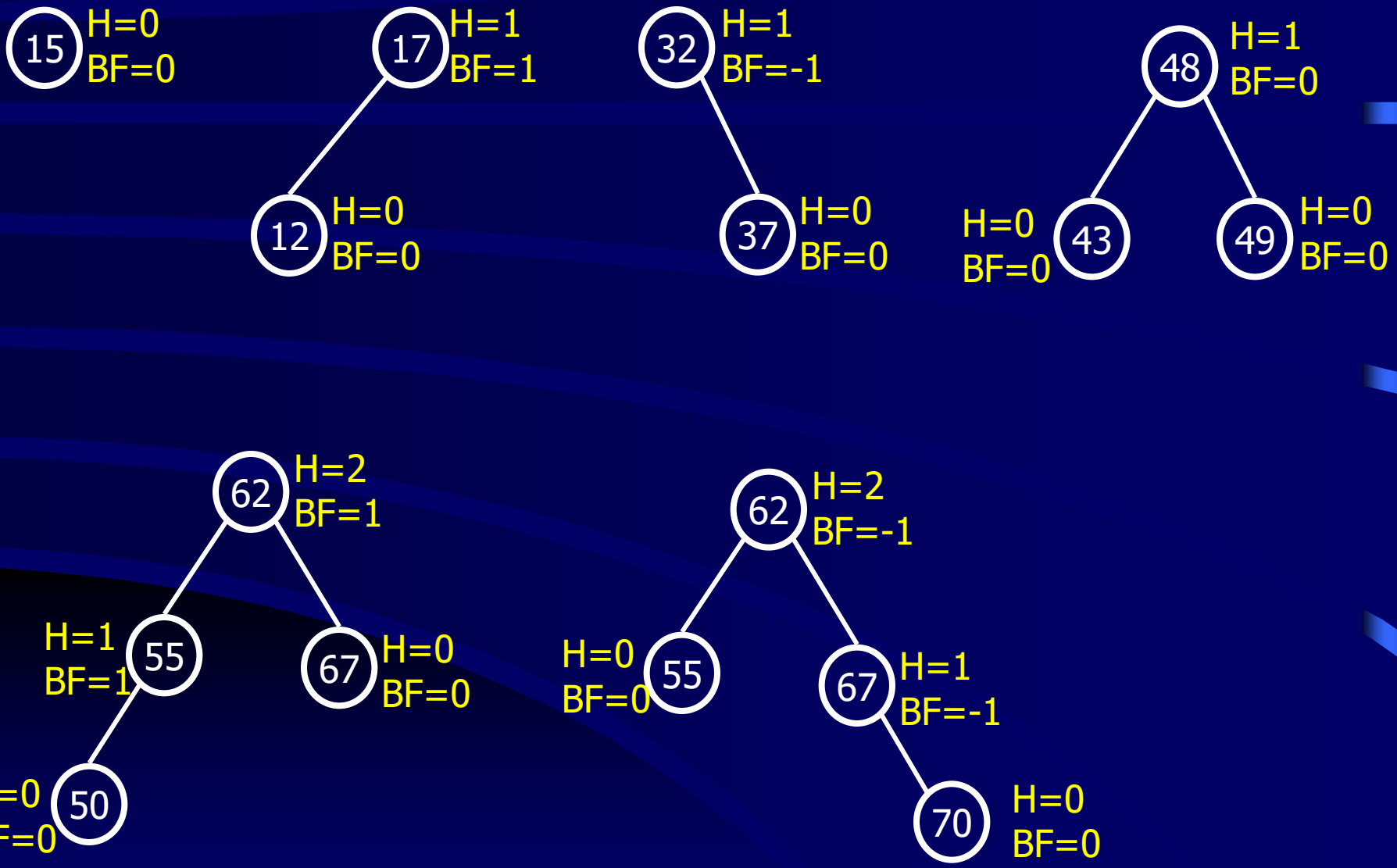
انگيزه

- درخت دودويي جستجو (BST) در حالتي كه كامل باشد ساختار مناسبي براي عمليات ديكشنري است ((O(lgn)
- در بدترين حالت، درخت دودويي جستجو به يك ليست پيوندي تبديل مي شود ((O(n)
- روشهايي وجود دارد كه درخت دودويي جستجو (نزديك به) دودويي كامل نگهداري شود
  - AVL (Adelson, Velskii, Landis)
  - درخت 2-3 (2-3- Tree)
  - درخت 2-3-4 (2-3-4- Tree)
  - درخت قرمز سياه (Red Black Tree)
  - . . .

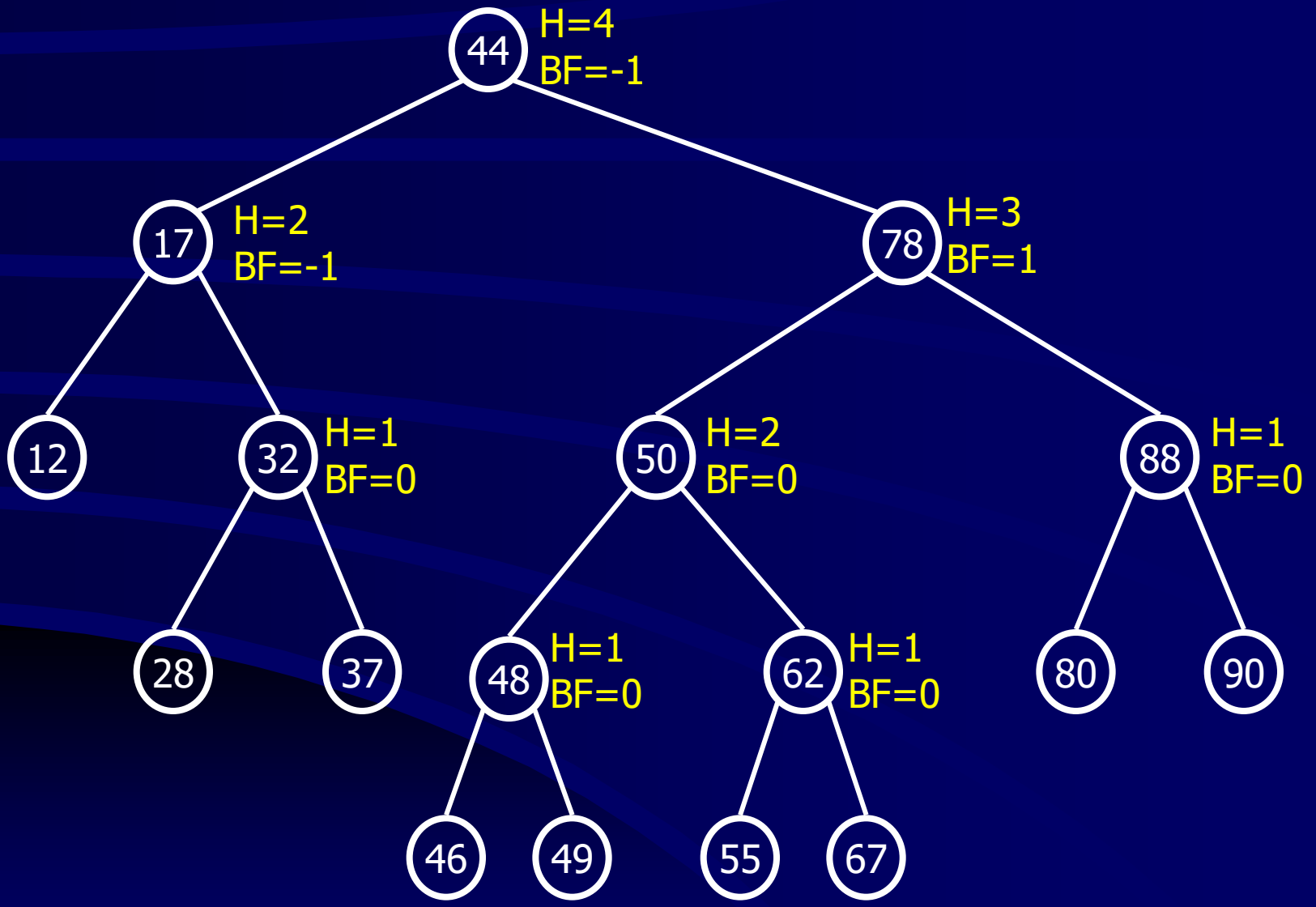
AVL tree

- طبق تعريف: ضريب تعادل (BF:Balance Factor) براي هر گره T در درخت دودويي برابر است با $BF=h_L-h_R$ كه در آن h_L و h_R به ترتيب ارتفاع زيردرختان چپ و راست T است.
- در AVL tree ، BF براي هر گره درخت يكي از مقادير 1،0، يا 1- است.

مثال درخت AVL

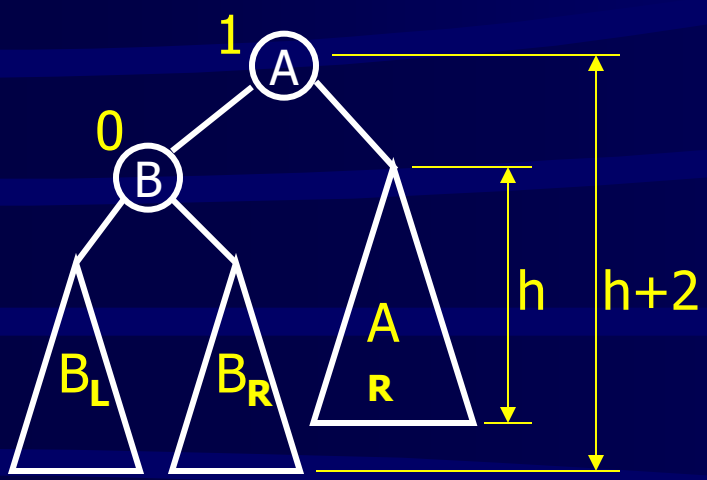


مثال درخت AVL

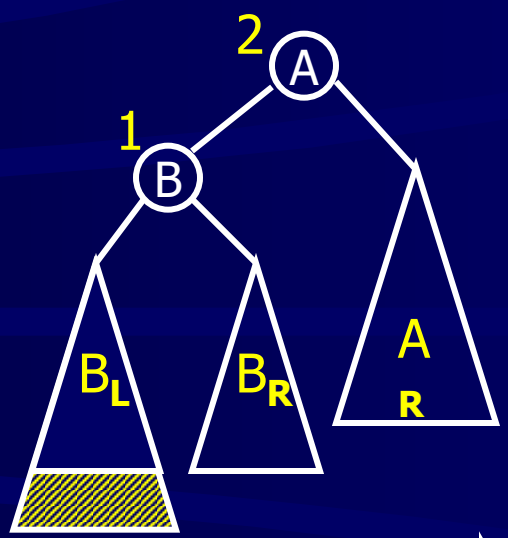


انواع چرخش در AVL

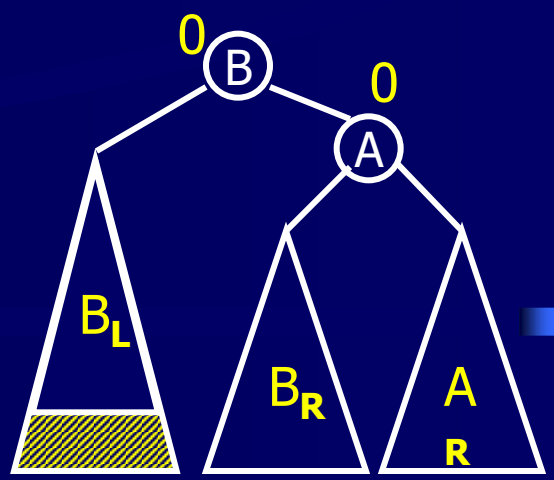
با اضافه شدن گره اي به درخت AVL ممکن است تعادل درخت از بين برود (گره اي با BF= ±2 به وجود آيد)
براي متعادل ساختن درخت پس از اضافه شدن گره اي چرخش (rotation) انجام مي شود.
انواع چرخش (rotation) در درخت AVL بر اساس محل اضافه شده گره Y نسبت به گره A است (گره A نزديکترين جد به گره اضافه شده Y است که پس از اضافه شدن Y، BF براي آن گره ±2 مي شود)



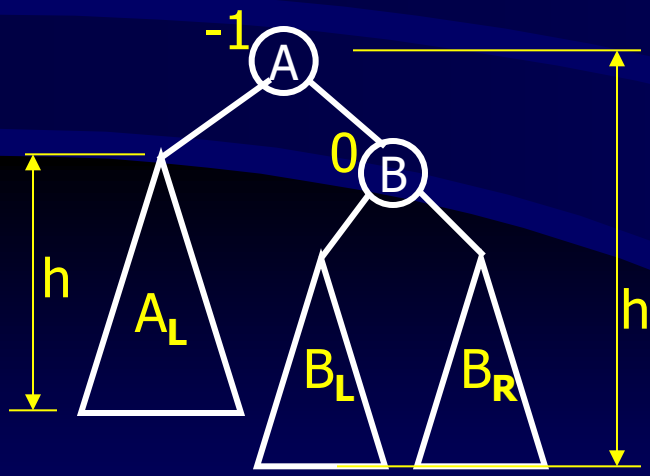
درخت متعادل



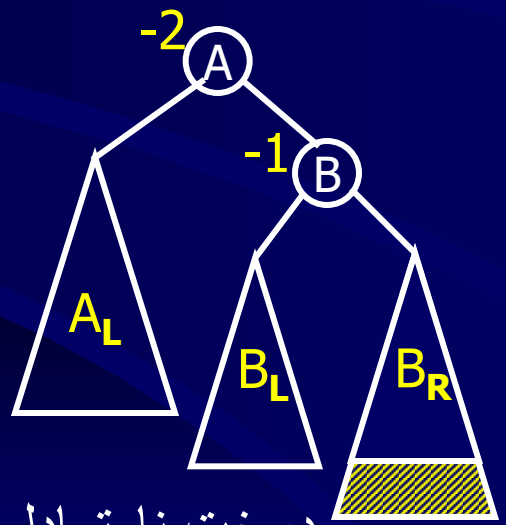
درخت نامتعادل
بعد از insert



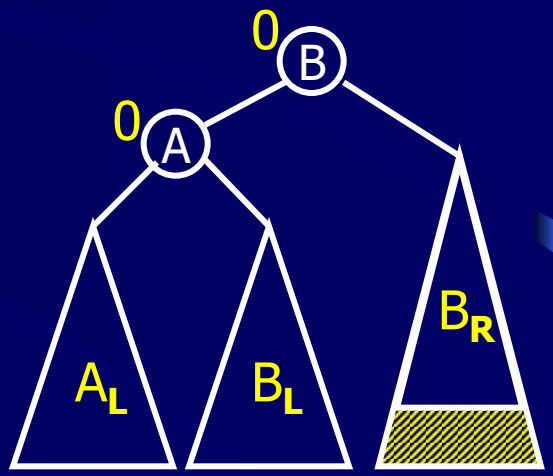
درخت متعادل
بعد از چرخش LL (LL rotation)



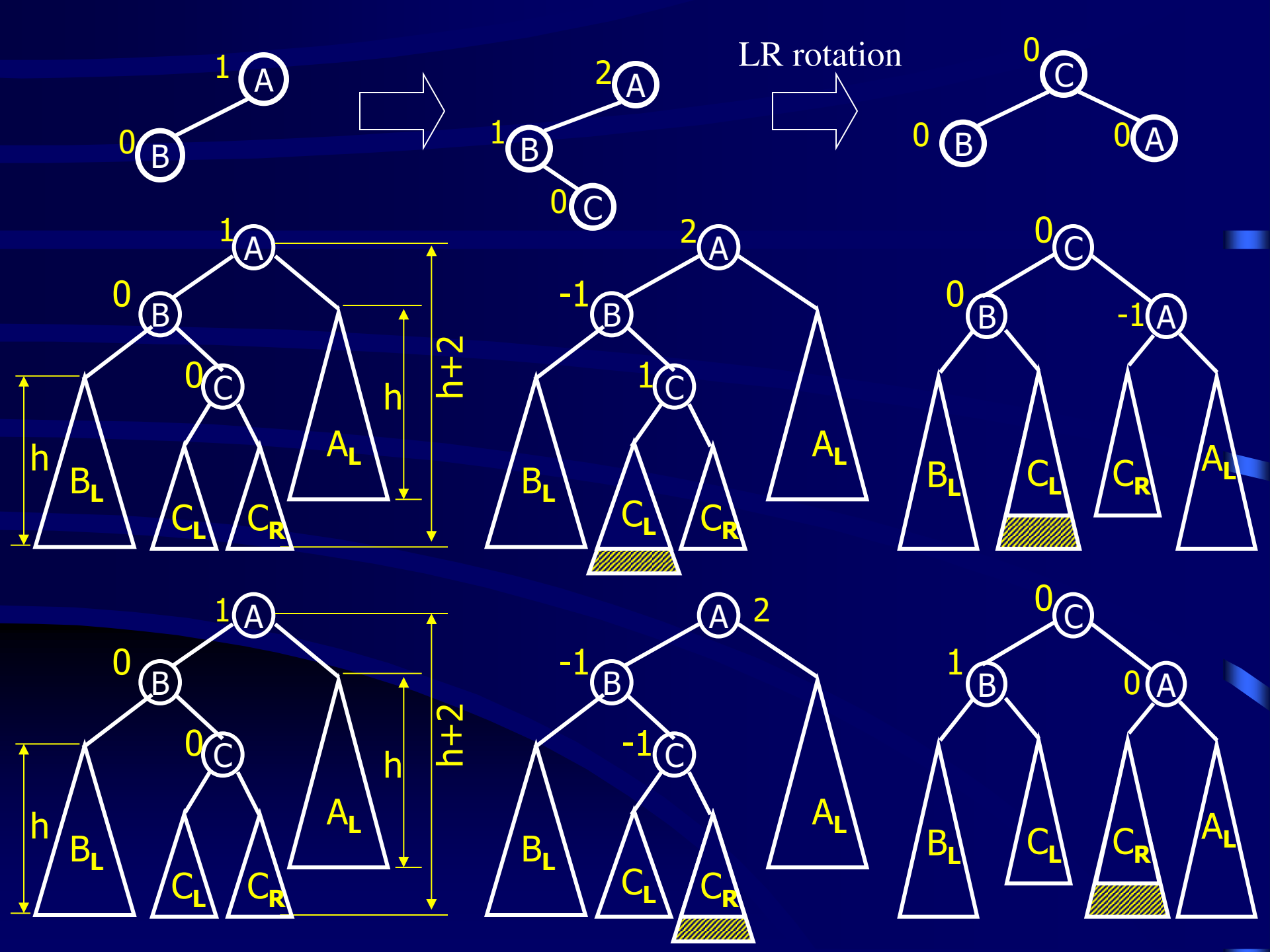
درخت متعادل



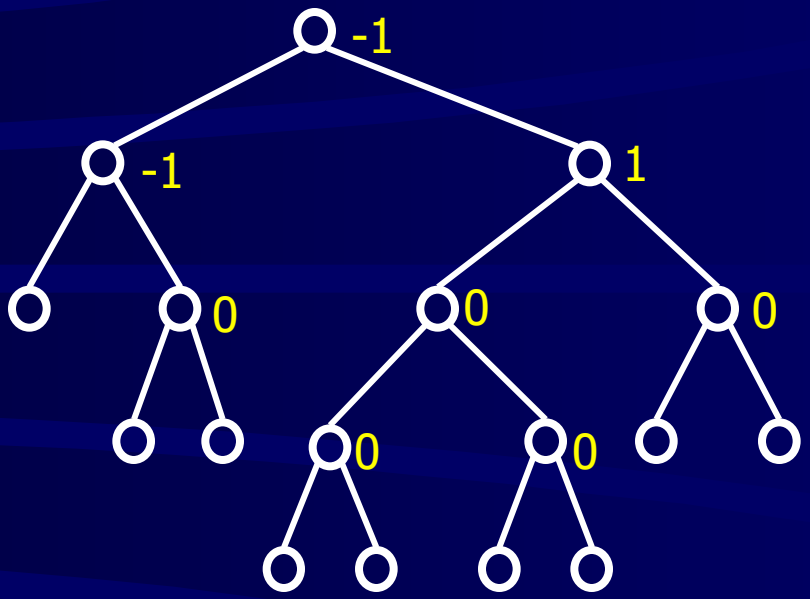
درخت نامتعادل
بعد از insert

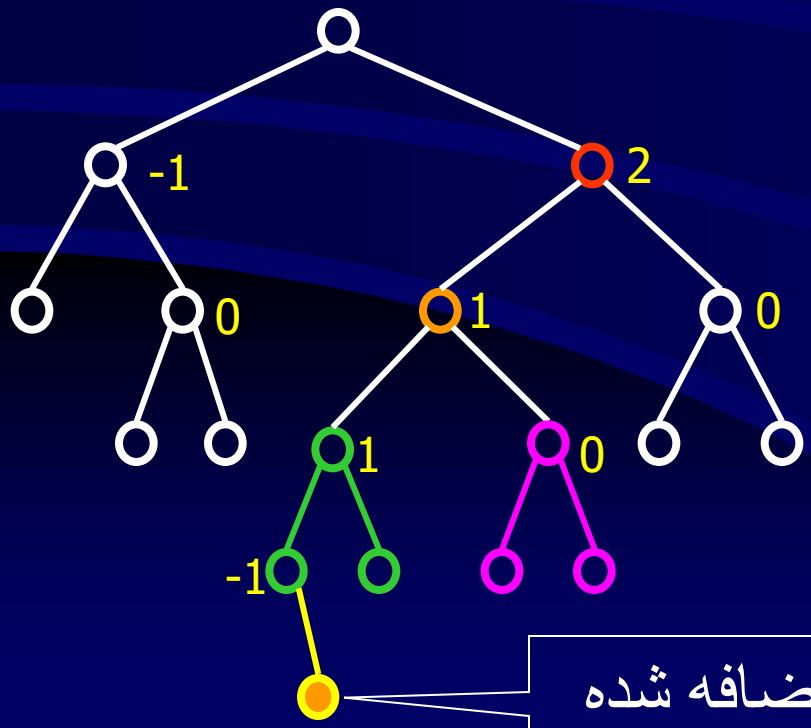


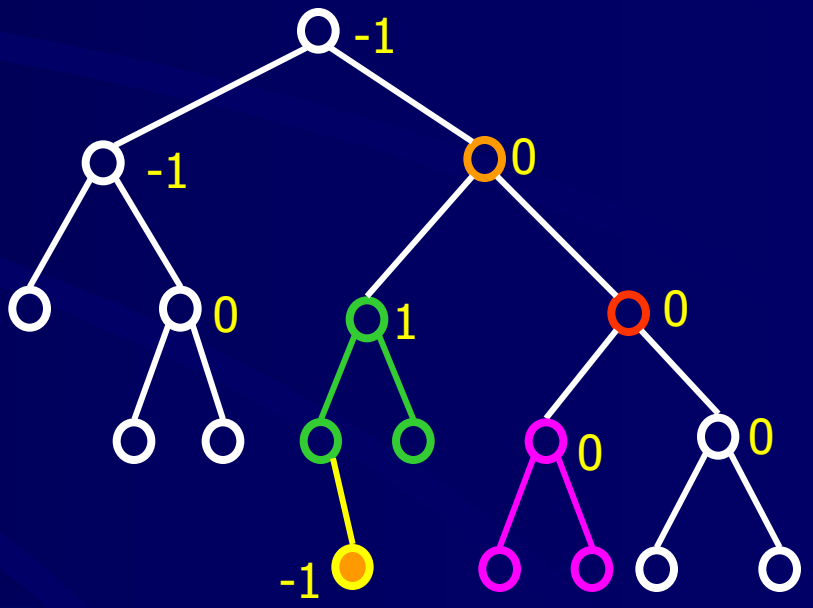
درخت متعادل
بعد از چرخش RR (RR rotation)


LR rotation
1
A
0
B
2
A
1
B
0
C
0
C
0
B
0
A
1
A
0
B
0
C
h
h+2
h
B_L
C_L
C_R
A_L
2
A
-1
B
1
C
0
C
0
B
-1
A
1
A
0
B
0
C
h
h+2
h
B_L
C_L
C_R
A_L
A
2
-1
B
-1
C
0
C
1
B
0
A

مثال





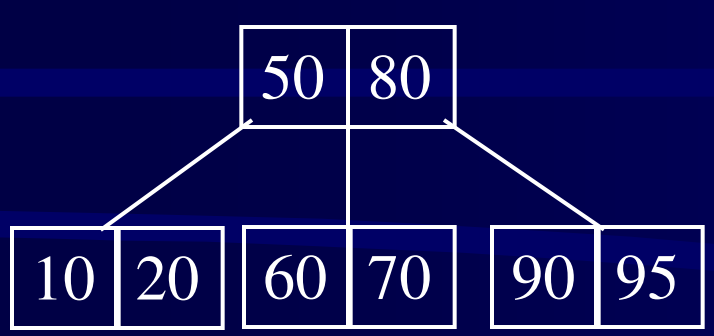


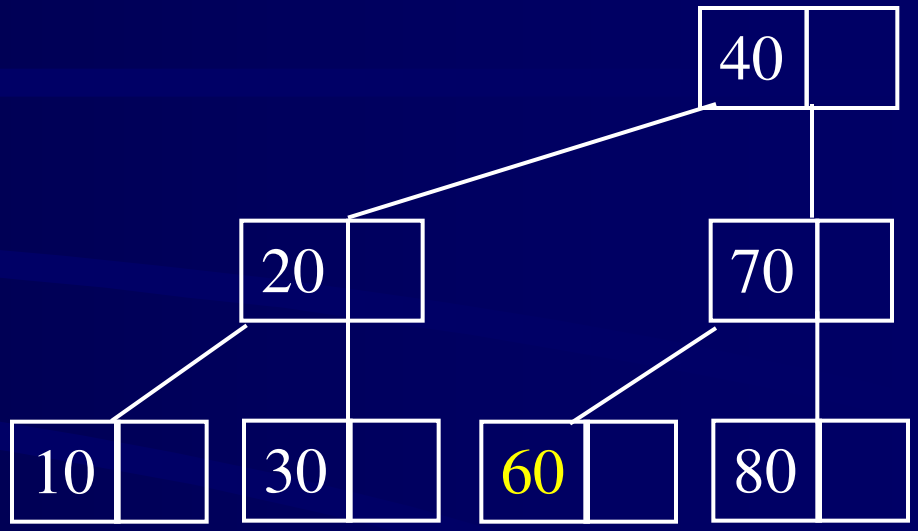
درخت 2-3 (2-3 Tree)

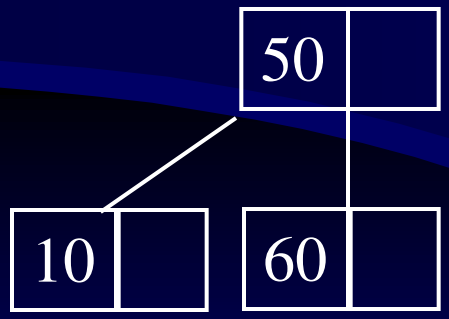
درخت 2-3 درختي است كه يا تهي است يا شرايط زير را دارد:

- هر گره يا 2-node است يا 3-node (در 2-node يك عنصر و در 3-node دو عنصر قرار مي گيرد)
- در 2-node: عنصر Lkey و دو زيردرخت LChild و MChild كه كليدهاي زيردرخت LChild از Lkey كوچكتر و كليدهاي زيردرخت MChild از LKey بزرگتر هستند.
- در 3-node: دو عنصر Lkey و Rkey و سه زيردرخت LChild، MChild، و RChild با شرايط زير:
  - + Lkey < RKey
  - + كليدهاي زيردرخت LChild از LKey كوچكتر هستند
  - + كليدهاي زيردرخت MChild از LKey بزرگتر و از RKey كوچكتر هستند
  - + كليدهاي زيردرخت RChild از Rkey بزرگتر هستند
- همه برگها در يك سطح قرار دارند

2-3 Tree مثال

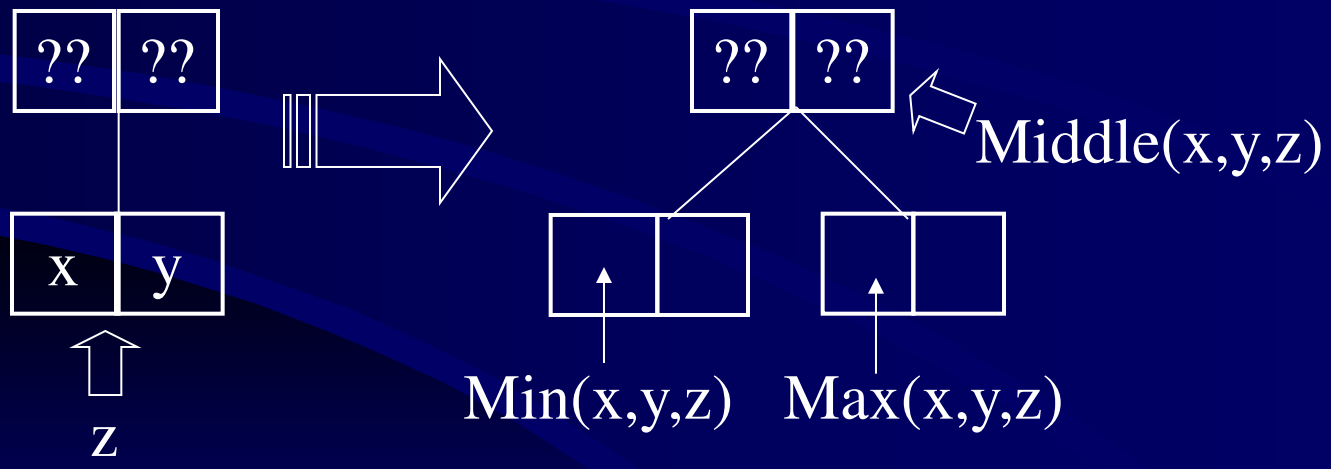






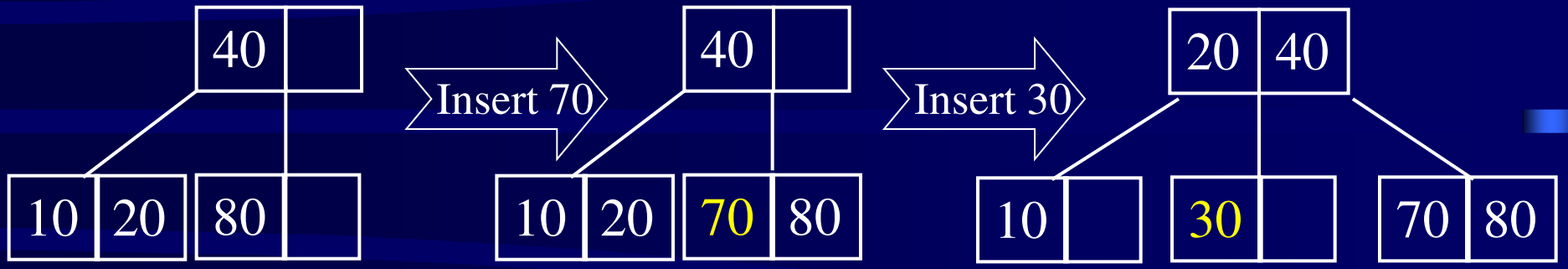
درج عنصر در 2-3 Tree

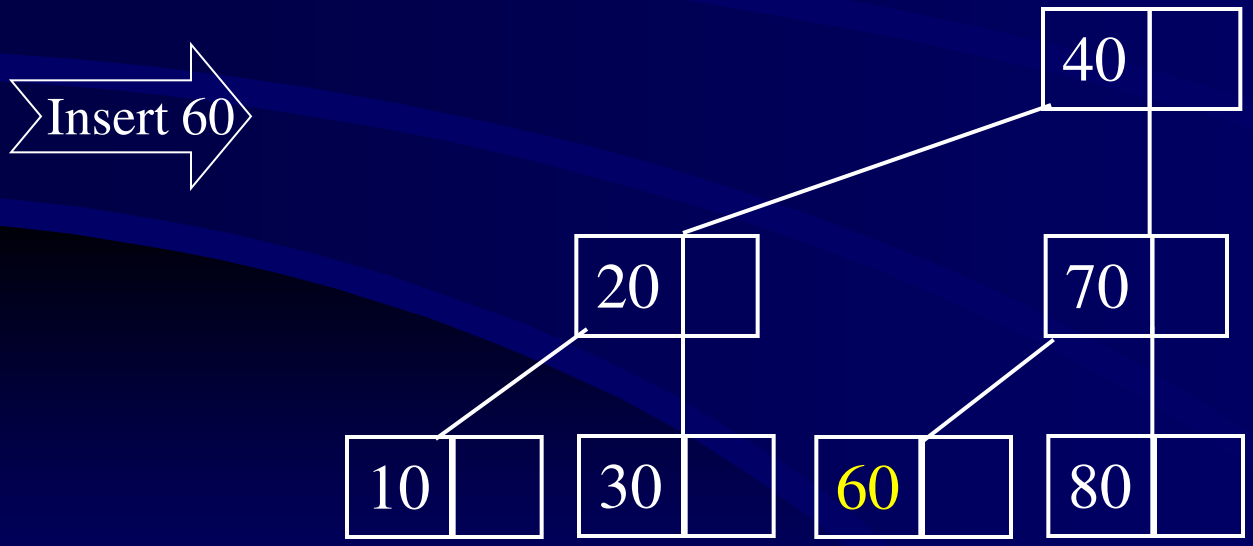
- عنصر جديد به يك برگ اضافه مي شود. اين برگ با جستجوي عنصر جديد پيدا مي شود.
- اگر برگ، يك 2-node باشد، به 3-node تبديل مي شود و عنصر جديد به آن اضافه مي شود (با رعايت ترتيب نسبي عناصر 3-node)
- اگر برگ، يك 3-node باشد، تقسيم (split) اتفاق مي افتد و يك عنصر (عنصر با مقدار كليد مياني بين سه كليد) به گره پدر اضافه مي شود:



- اگر ريشه درخت تقسيم (split) شود، يك ريشه جديد اضافه مي شود و ارتفاع درخت يك واحد اضافه مي شود.

مثال درج عنصر در 2-3 Tree

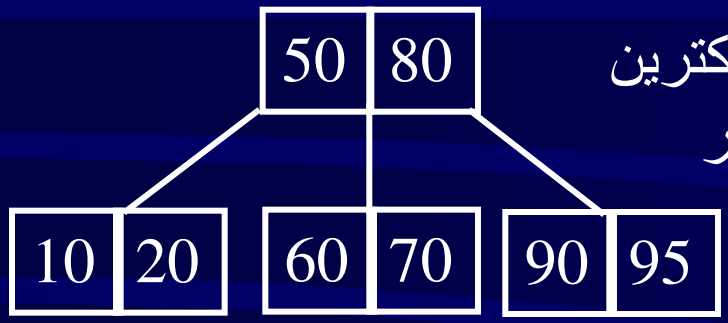




ارتفاع درخت يك واحد اضافه مي شود

حذف عنصر در 2-3 Tree

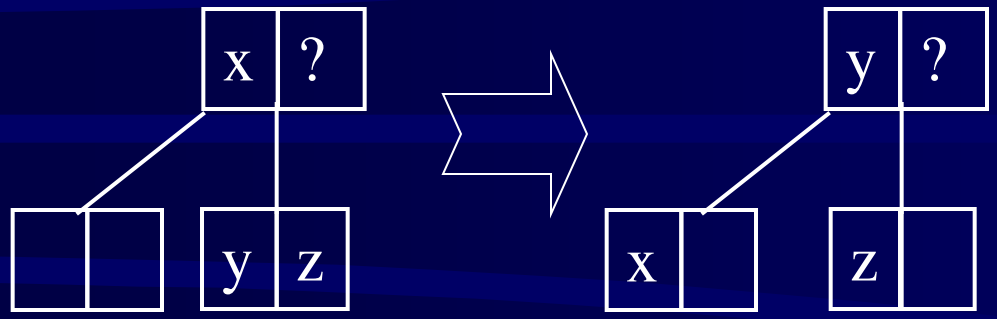
- اگر عنصري كه بايد حذف شود در برگ نيست با يك عنصر در يكي از برگها تعويض مي شود

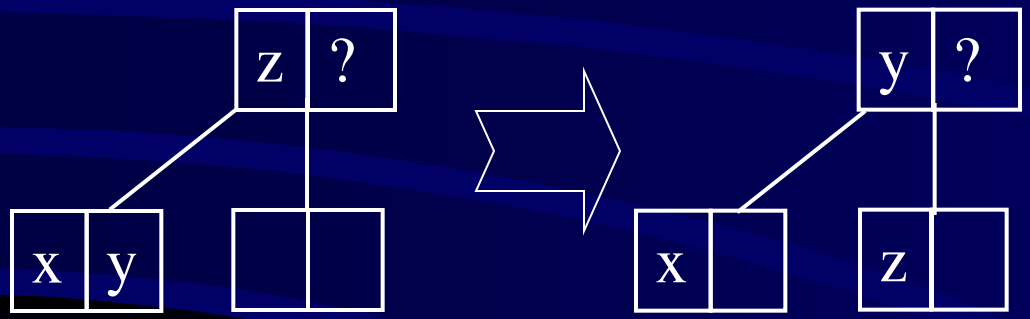


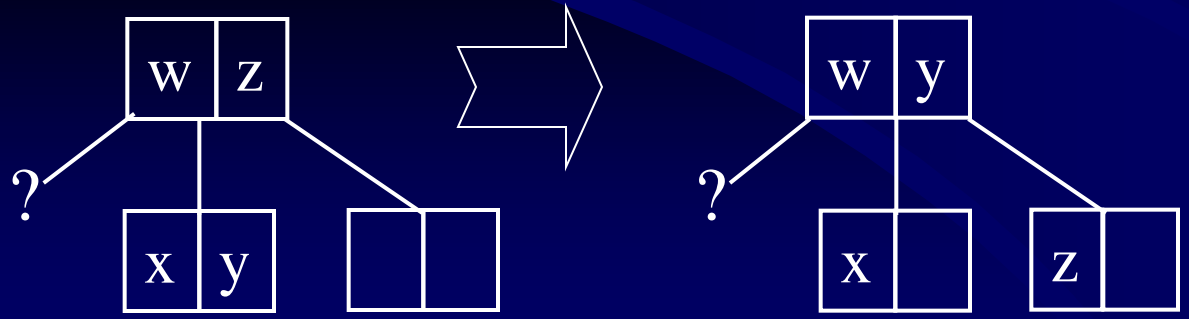
مثلا در درخت مقابل براي حذف 50، با 60 (كوچكترين عنصر زيردرخت راست) يا 20 (بزرگترين عنصر زيردرخت چپ) تعويض مي شود

- اگر عنصر از يك 3-node حذف شود، 3-node به 2-node تبديل مي شود.
- اگر عنصر از يك 2-node حذف مي شود، چرخش (rotation) يا تركيب (combine) انجام مي شود.

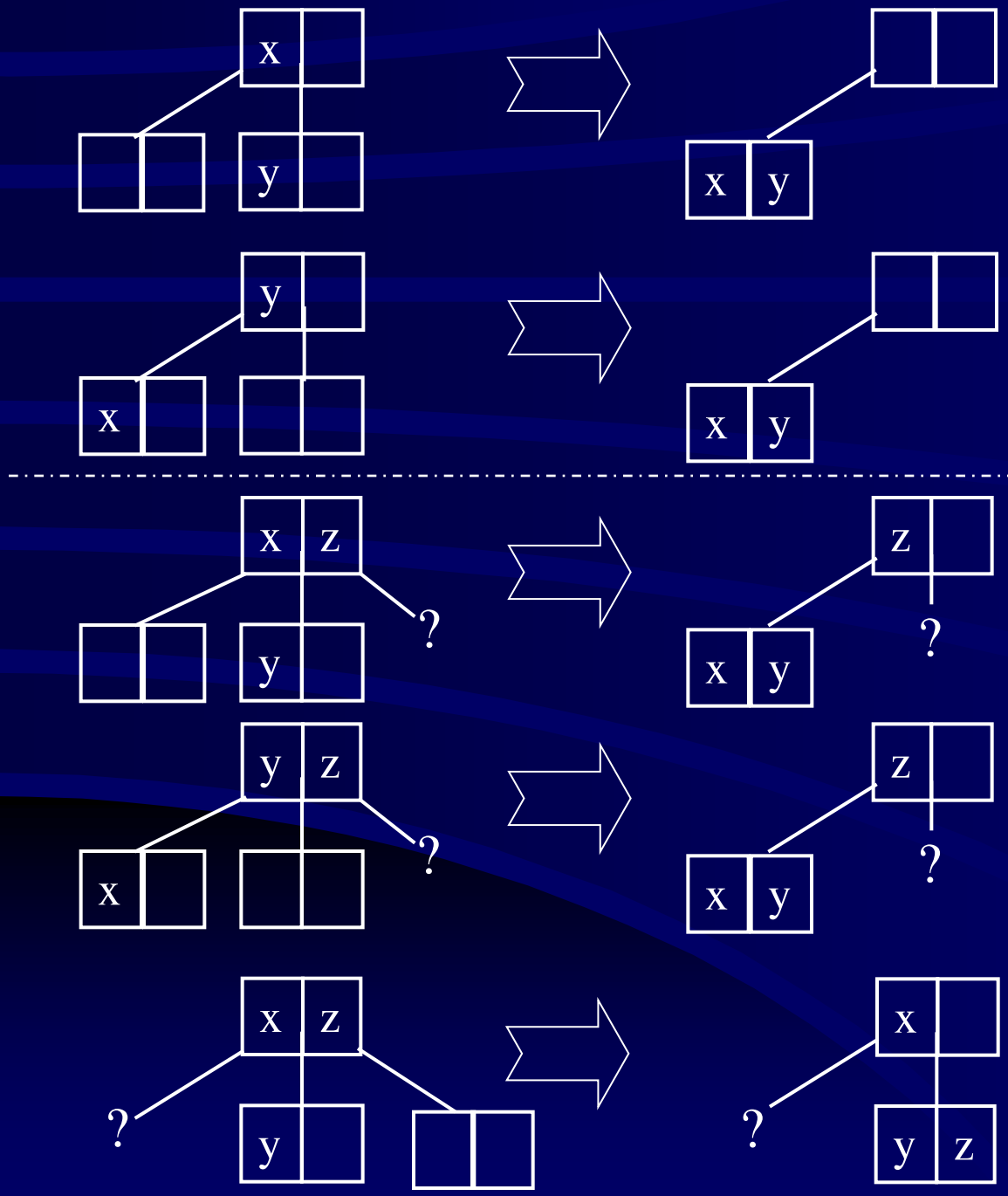
انواع چرخش در 2-3 Tree



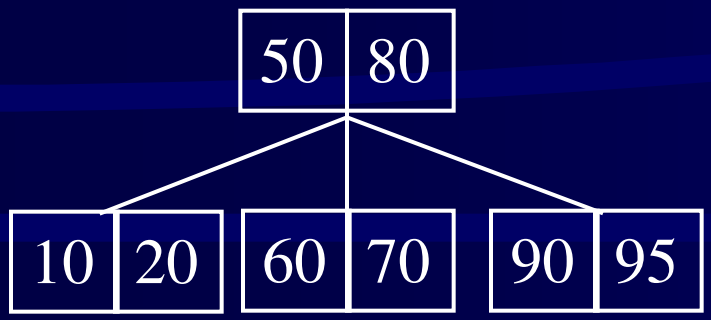


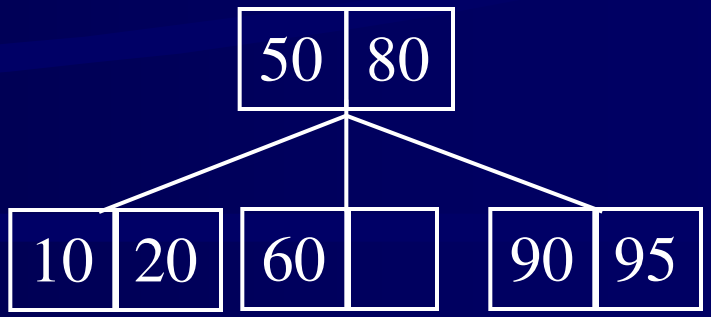


انواع ترکیب در 2-3 Tree

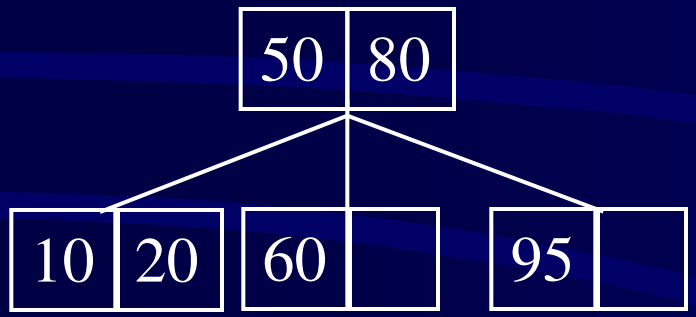


مثال حذف عنصر در 2-3 Tree

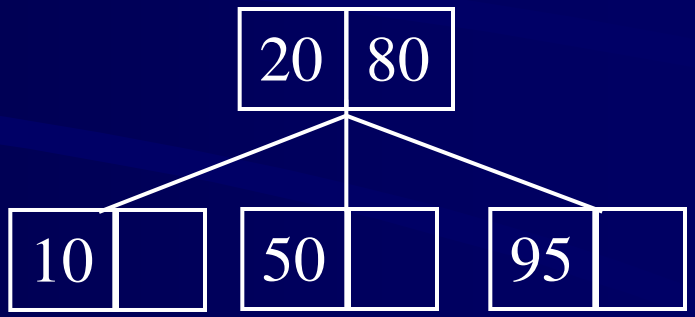




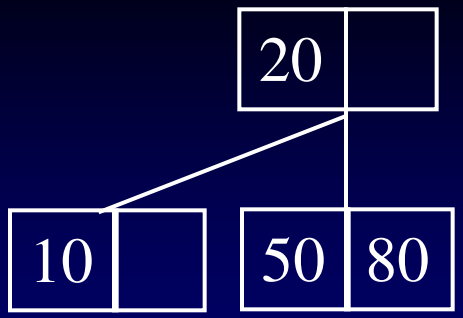
Delete 70



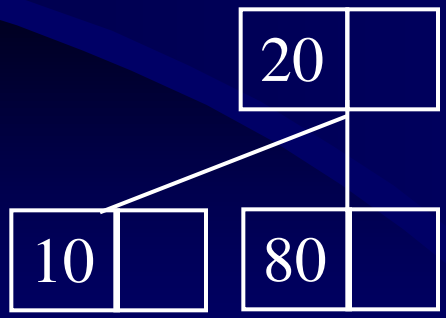
Delete 90



Delete 60



Delete 95



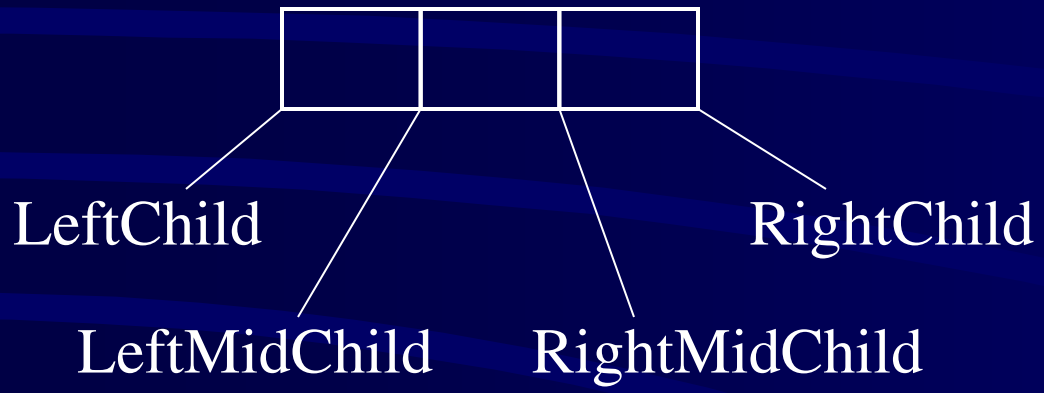
Delete 50

Delete 10

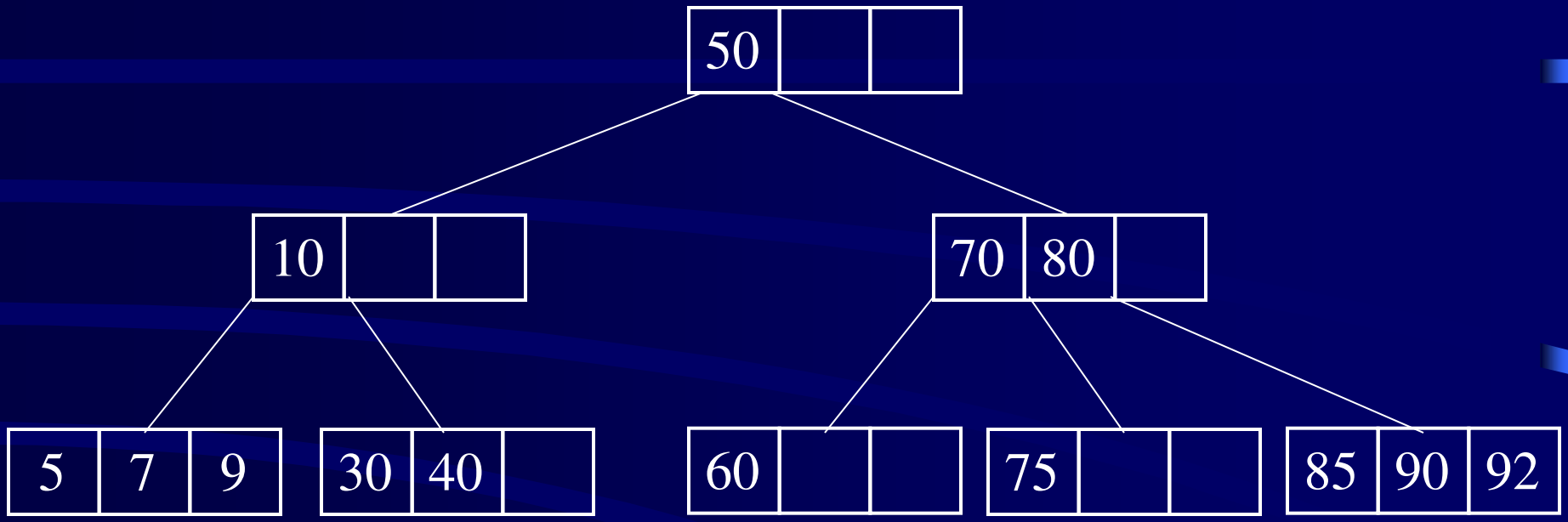
درخت 2-3-4 Tree 2-3-4

درخت 2-3-4 گسترشي از درخت 2-3 است که در آن علاوه بر 2-node و 3-node ، 4-node نیز وجود دارد.

در 4-node سه کلید و چهار زیردرخت وجود دارد:



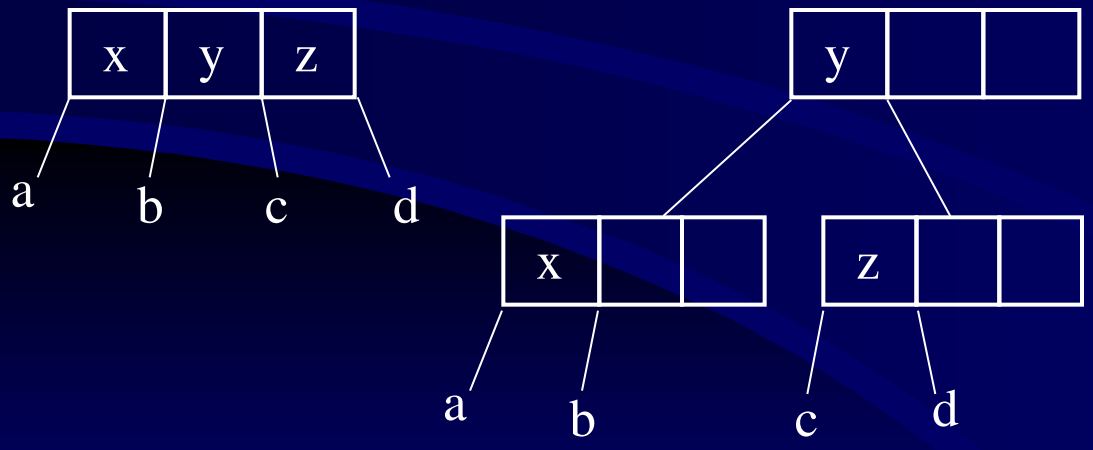
2-3-4 Tree مثال



درج عنصر در 2-3-4 Tree

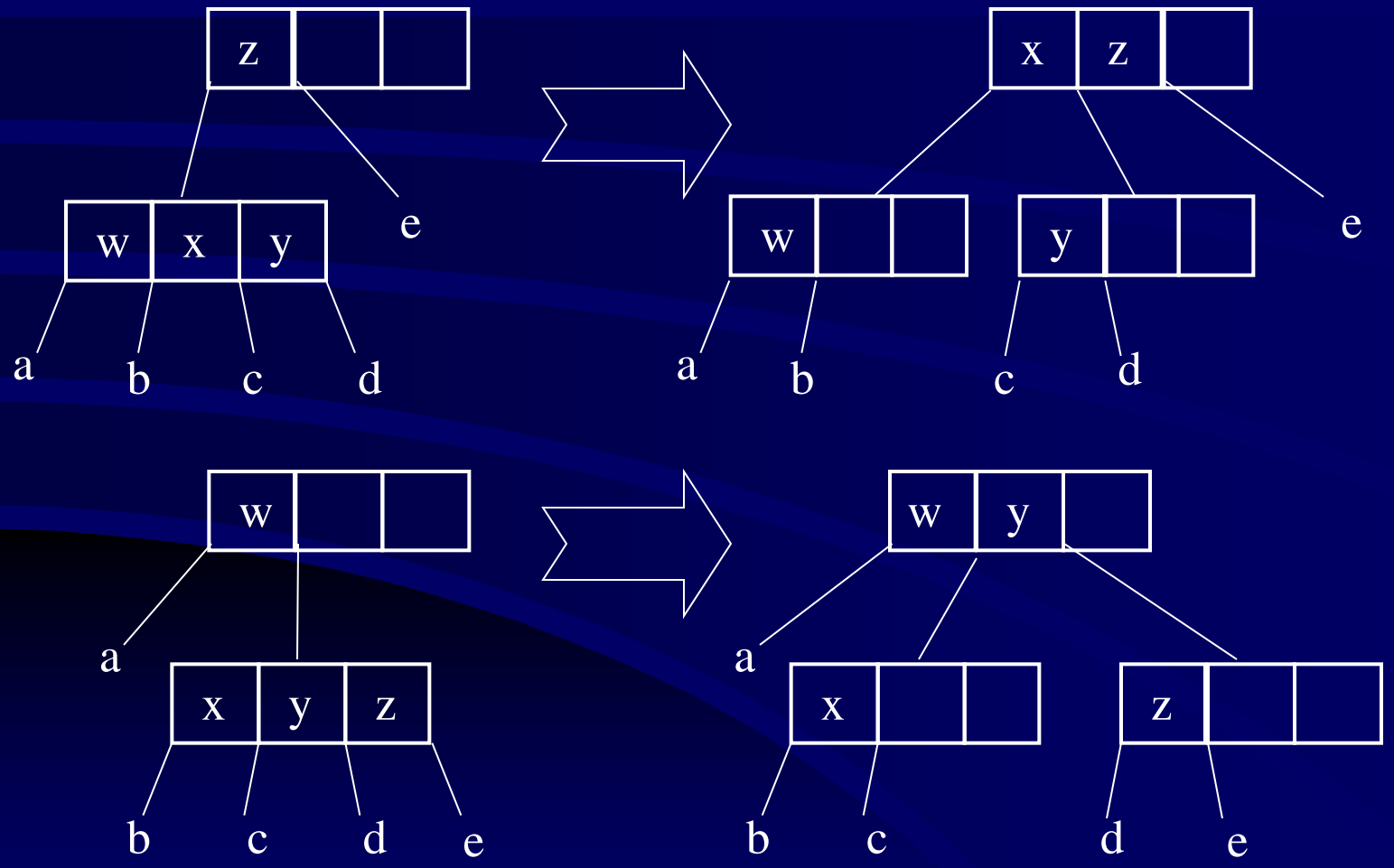
در مسيري كه از ريشه تا برگ محل درج شدن عنصر طي مي شود، اگر به 4-node برخورد شود، تقسيم (split) انجام مي شود
در تقسيم هر 4-node حالتهاي مختلفي ممكن است :

- حالت اول: 4-node، ريشه درخت باشد:



درج عنصر در 2-3-4 Tree

- حالت دوم: پدر 4-node يك 2-node باشد:



درج عنصر در 2-3-4 Tree

- حالت سوم: پدر 4-node يك 3-node باشد: