AVL stromy

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

KMI/ZADS - Základní algoritmy a datové struktury

Uzel

```
{
        "id":None,
        "left":None,
        "right":None,
        "parent":None,
        "bf":0
}

avl_tree={"root":None}
```

Rotace

Pomocné funkce

```
def set_left_child(p,c):
    p["left"]=c
    if c != None:
        c["parent"]=p

def set_right_child(p,c):
    p["right"]=c
    if c != None:
        c["parent"]=p
```

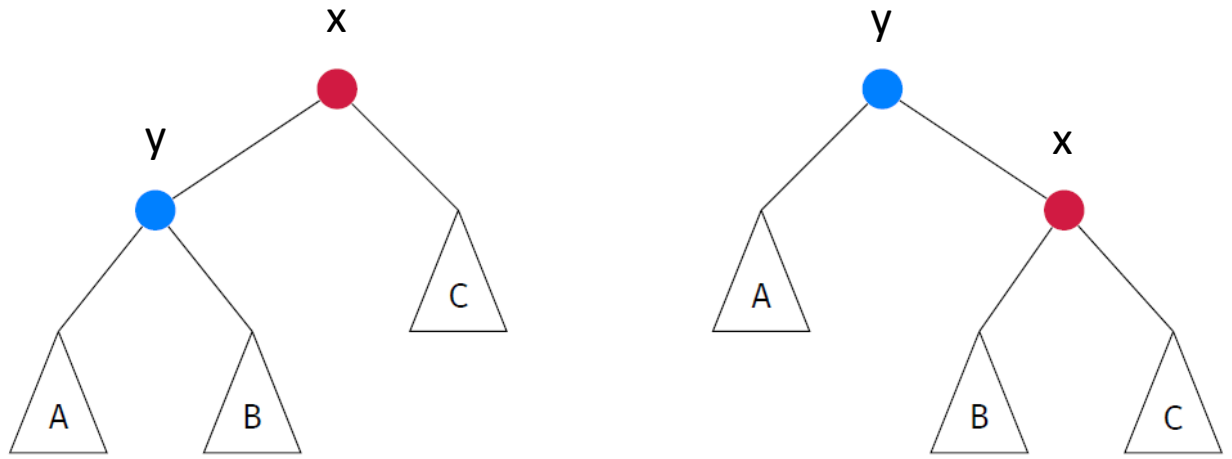
Pomocné funkce

```
def set_root(t, x):
    t["root"] = x
    if x != None:
        x["parent"] = None

def transplant_tree(t, u, v):
    if u["parent"] == None:
        set_root(t, v)
    else:
        x = u["parent"]
        if u == x["left"]:
            set_left_child(x, v)
        else:
            set_right_child(x, v)
```

Jednoduché rotace

```
def rotate_right(t, x):
    y = x["left"]
    set_left_child(x, y["right"])
    transplant_tree(t, x, y)
    set_right_child(y, x)
    if y["bf"]==1:
        x["bf"] = 0
        y["bf"] = 0
        return -1, y # zmena vysky, uzel v horni pozici (po rotaci)
    else: # y.bf==0
        x["bf"] = 1
        y["bf"] = -1
        return 0, y # zmena vysky, uzel v horni pozici (po rotaci)
```

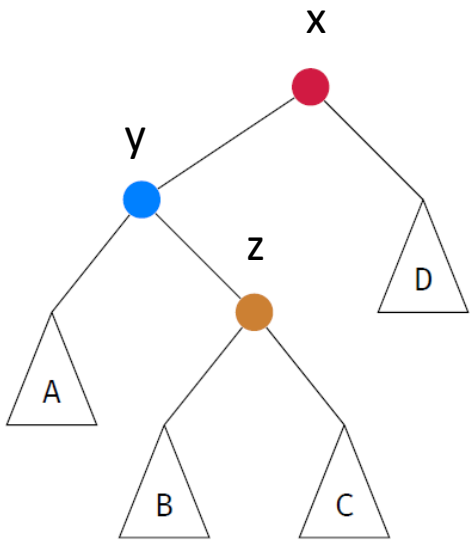


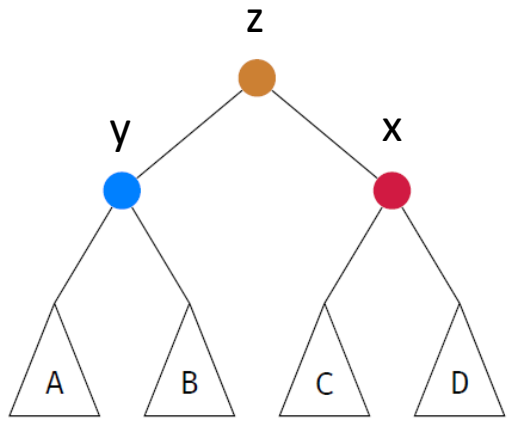
Jednoduché rotace

```
def rotate_left(t, x):
    y = x["right"]
    set_right_child(x, y["left"])
    transplant_tree(t, x, y)
    set_left_child(y, x)
    if y["bf"]==-1:
        x["bf"] = 0
        y["bf"] = 0
        return -1, y # zmena vysky, uzel v horni pozici (po rotaci)
    else: # y.bf==0
        x["bf"] = -1
        y["bf"] = 1
        return 0, y # zmena vysky, uzel v horni pozici (po rotaci)
```

Dvojitá rotace (levo pravá)

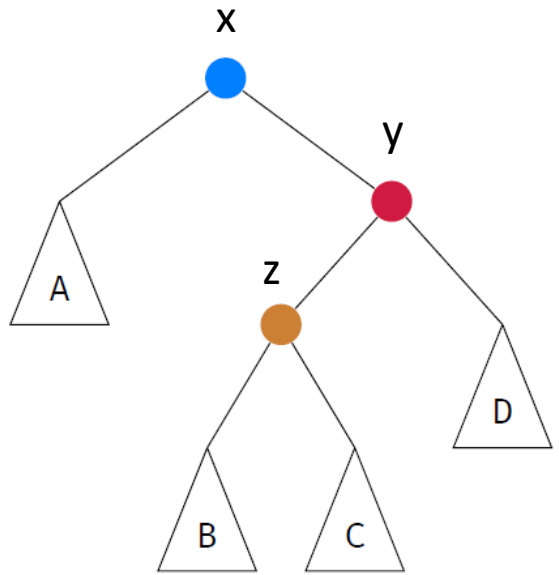
```
def rotate_left_right(t, x):
    y = x["left"]
    z = y["right"]
    set_left_child(x, z["right"])
    set_right_child(y, z["left"])
    transplant_tree(t, x, z)
    set_right_child(z, x)
    set_left_child(z, y)
    x["bf"] = y["bf"] = 0
    if z["bf"]==1:
        x["bf"]=-1
    if z["bf"]==-1:
        y["bf"]=1
    z["bf"]=0
    return -1, z # zmena vysky, uzel v horni pozici (po rotaci)
```

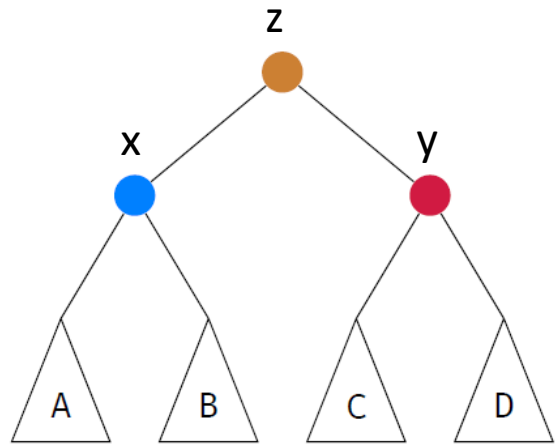




Dvojitá rotace (pravo levá)

```
def rotate_right_left(t, x):
    y = x["right"]
    z = y["left"]
    set_right_child(x, z["left"])
    set_left_child(y, z["right"])
    transplant_tree(t, x, z)
    set_left_child(z, x)
    set_right_child(z, y)
    x["bf"] = y["bf"] = 0
    if z["bf"]==-1:
        x["bf"]=1
    if z["bf"]==1:
        y["bf"]=-1
    z["bf"]=0
    return -1, z # zmena vysky, uzel v horni pozici (po rotaci)
```





Rotace

```
def rotate(t, x):
    if x["bf"]==-2:
        y = x["right"]
        if y["bf"]==1:
            return rotate_right_left(t, x)
        else:
            return rotate_left(t, x)
    else: # x.bf==2
        y = x["left"]
        if y["bf"]==-1:
            return rotate_left_right(t, x)
        else:
            return rotate_right(t, x)
```

Test rotace

```
def check_rotate(t, x, change, subtree = None, left = None):
    # zastaveni rekurze
    if x==None or change==0: return
    # dopocitani left podle subtree
    if subtree!=None:
        left = child_is_left(x, subtree)
```

Test rotace

```
if left: # zmena je vlevo
        if change == 1: # zvysil se podstrom
            if x["bf"]==-1:
                x["bf"]=0
            elif x["bf"]==0:
                x["bf"]=1
                check_rotate(t, x["parent"], 1, x)
            else: # x.bf==1
                x["bf"]=2
                new_change, top = rotate(t, x)
                new_change += 1
                check_rotate(t, top["parent"], new_change, x) # neni potreba
```

Test rotace

```
        else: # snizil se podstrom
            if x["bf"]==-1:
                x["bf"]=-2
                new_change, top = rotate(t, x)
                check_rotate(t, top["parent"], new_change, x)
            elif x["bf"]==0:
                x["bf"]=-1
            else: # x.bf==1
                x["bf"]=0
                check_rotate(t, x["parent"], -1, x)
```

Test rotace

```
    else: # zmena je vpravo
        if change == 1: # zvysil se podstrom
            if x["bf"]==1:
                x["bf"]=0
            elif x["bf"]==0:
                x["bf"]=-1
                check_rotate(t, x["parent"], 1, x)
            else: # x.bf==-1
                x["bf"]=-2
                new_change, top = rotate(t, x)
                new_change += 1
                check_rotate(t, top["parent"], new_change, x) # neni potreba
```

Test rotace

```
        else: # snizil se podstrom
            if x["bf"]==1:
                x["bf"]=2
                new_change, top = rotate(t, x)
                check_rotate(t, top["parent"], new_change, x)
            elif x["bf"]==0:
                x["bf"]=1
            else: # x.bf==-1
                x["bf"]=0
                check_rotate(t, x["parent"], -1, x)
```

Vkládání do stromu

Vložení uzlu do stromu

```
def tree_insert(t, z):
    y = None
    x = t["root"]
    while x != None:
        y = x
        if z["key"] < x["key"]:
            x = x["left"]
        else:
            x = x["right"]
    z["parent"] = y
    if y == None:
        t["root"] = z
    elif z["key"] < y["key"]:
        y["left"] = z
        check_rotate(t, y, 1, left=True)
    else:
        y["right"] = z
        check_rotate(t, y, 1, left=False)
```

Úkol

1. Použijte AVL strom pro ukládání údajů o vývoji HDP v jednotlivých čtvrtletích.
2. Napište funkci:
   - `najdi_hdp(s,rok,ctvrtleti)` – která vrátí údaj o vývoji HDP v daném roce a čtvrtletí nebo vrátí informaci, že tento údaj není k dispozici.

- Příklad:
  - Pro data vložená do stromu:

```
data = [{"rok":2022,"ctvrtleti":1,"hdp":1.1},
{"rok":2021,"ctvrtleti":2,"hdp":-2.1},
{"rok":2022,"ctvrtleti":4,"hdp":0.1},
{"rok":2021,"ctvrtleti":3,"hdp":2.3},
{"rok":2021,"ctvrtleti":4,"hdp":-0.5},
{"rok":2022,"ctvrtleti":2,"hdp":-1.3},
{"rok":2022,"ctvrtleti":3,"hdp":3.2},
{"rok":2021,"ctvrtleti":1,"hdp":4.1},
]
```

  - Příkazy:

```
najdi_hdp(t,2021,1)
najdi_hdp(t,2022,3)
najdi_hdp(t,2023,1)
```

  - Vypíší:

```
HDP v {ctvrtleti}. čtvrtletí roku {rok} bylo: 4.1
HDP v {ctvrtleti}. čtvrtletí roku {rok} bylo: 3.2
Tento údaj není k dispozici.
```