Drzewa poszukiwań binarnych

Definicja Drzewo poszukiwań binarnych to drzewo binarne, w którego węzłach znajdują się pewne wartości i które spełnia następujące warunki:

- dla każdego węzła x, wartości znajdujące się we wszystkich węzłach poddrzewa, którego ojcem jest lewy syn węzła x są mniejsze lub równe od wartości znajdującej się w węźle x
- dla każdego węzła x, wartości znajdujące się we wszystkich węzłach poddrzewa, którego ojcem jest prawy syn węzła x są większe lub równe od wartości znajdującej się w węźle x

Szukanie

```
Tree-search(x,k)
// szuka klucza k w  poddrzewie o korzeniu x
  while x!=NIL and k!=x.key
    if k<x.key
         x=x.left
    else
         x=x.right
  return x  // zwraca NIL gdy nie ma klucza k
```

Wstawianie

Nowy węzeł wstawiamy jako liść. Miejsca, do którego doczepimy nowy węzeł szukamy wędrując ścieżką od korzenia drzewa w dól,zgodnie z porządkiem drzewa.

```
Tree-insert(T,z)
// wstawia węzeł z do drzewa T
  x=T.root
  y=NIL   // y jest ojcem x
  while x!=NIL
    y=x
    if z.key<x.key
         x=x.left
    else x=x.right
  z.p=y
  if y==NIL
    T.root=z
  else if z.key<y.key
            y.left=z
       else y.right=z
```

Usuwanie

Są 3 przypadki: usuwany węzeł nie ma synów, ma jednego syna, ma dwóch synów.

Poniżej są dwie wersje algorytmu.

wersja "naturalna"

Ten pseudokod odzwierciedla w najoczywistszy sposób trzy możliwe przypadki

```
Tree-delete(T,z)
// usuwa węzeł "z" z drzewa T
// wersja "naturalna"
  if z.left==NIL and z.right==NIL
    if z==T.root
        T.root=NIL
    else if z==z.p.left
              z.p.left=NIL
         else
              z.p.right=NIL
  else if z.left != NIL and z.right != NIL
              y=Tree-Minimum(z.right)
              przepisz zawartość wezla y do z
              TreeDelete(T,y) // przypadek (1) lub (2)
  else if z.left != NIL   // z.right==NIL
          z.left.p=z.p
          if z==T.root
             T.root=z.left
          else if z==z.p.left
                  z.p.left=z.left
                else
                   z.p.right=z.left
        else  // z.left==NIL
          z.right.p=z.p
          if z==T.root
             T.root=z.right
          else if z==z.p.left
                  z.p.left=z.left
                else
                   z.p.right=z.left
```

```
Tree-Minimum(x)
// zwraca skrajny lewy węzeł w poddrzewie o korzeniu x
// czyli węzeł o najmniejszym kluczu w tym poddrzewie
    while x.left != NIL
        x = x.left
    return x
```

wersja według podręcznika Cormena (nowsze wydania)

```
Tree-delete(T,z)
// wersja wg nowszych wydań Cormena
  if z.left == NIL
     Transplant(T,z,z.right)
  else if z.right == NIL
          Transplant(T,z,z.left)
  else y=Tree-Minimum(z.right)
       if y.p != z
          Transplant(T,y,y.right)
          y.right=z.right
          y.right.p=y
       Transplant(T,z,y)
       y.left=z.left
       y.left.p=y


Transplant(T,u,v)
// podczepia poddrzewo o korzeniu "v" w miejsce węzła u
// w drzewie T
  if u.p==NIL  // u jest korzeniem
    T.root=v
  else if u==u.p.left  // u jest lewym synem
          u.p.left=v
       else
          u.p.right = v   // u jest prawym synem
       if v != NIL
          v.p=u.p
```

Stwierdzenie Pesymistyczna złożoność wstawienia, szukania i usuwania w drzewie poszukiwań binarnych jest

$\Theta(h)$, gdzie h to wysokość drzewa

$\Theta(n)$, gdzie n to ilość węzłów w drzewie

Definicja *Losowo skonstruowane drzewo poszukiwań binarnych* to drzewo poszukiwań binarnych otrzymane w wyniku wstawienia pewnej ilości parami różnych kluczy do początkowo pustego drzewa, przy czym zakładamy, że każda kolejność wstawianych kluczy jest jednakowo prawdopodobna.

Stwierdzenie Oczekiwana wysokość losowo skonstruowanego drzewa poszukiwań binarnych jest $\Theta(\lg n)$, gdzie n to ilość węzłów w drzewie.

Wniosek Oczekiwana złożoność czasowa wstawienia, szukania i usuwania w losowo skonstruowanym drzewie poszukiwań binarnych jest $\Theta(\lg n)$, gdzie n to ilość węzłów w drzewie.

Uwaga Po wykonaniu operacji usunięcia węzła, losowo skonstruowane drzewo poszukiwań binarnych przestaje być losowo skonstruowane.