5 zmniejszyc rysunki

Notatki z AiSD. Nr 12. 16 marca 2005

Drzewa zbalansowane:
Drzewa AVL

IIUWr. II rok informatyki. Przygotował: Krzysztof Loryś

17 Definicja

Definicja 3 *Binarne drzewo przeszukiwań jest* drzewem AVL, *jeśli dla każdego wierzchołka wysokości jego lewego i prawego poddrzewa różnią się o co najwyżej* 1.

Uwaga: Skrót AVL pochodzi od pierwszych liter nazwisk autorów (Adelson-Velskij i Landis).

18 Zasadnicza cecha

Twierdzenie 5 *Wysokość drzewa AVL o n wierzchołkach jest mniejsza niż* $1.4405 \log(n+2)$.

Fakt 19 *Liczba wierzchołków w dowolnym drzewie binarnym jest o* 1 *mniejsza od liczby pustych wska"xników (tj. równych* `NIL`*).*

Dowód (Twierdzenia 5)

Niech $\rho(i) =$ "liczba pustych wska"xników w minimalnym (tj. o najmniejszej 'liczbie wierzchołków) drzewie AVL o wysokości i".

Indukcyjnie po wysokości h drzewa dowodzimy, że $\rho(h) = (h+2)$-a liczba Fibonacciego.

Łatwo sprawdzić, że $\rho(1) = 2$ i $\rho(2) = 3$.

Niech T będzie minimalnym drzewem AVL o wysokości h ($h \geq 3$). Z minimalności T wiemy, że jedno z poddrzew podwieszonych pod jego korzeniem musi być minimalnym drzewem AVL o wysokości $h-1$, a drugie - minimalnym drzewem AVL o wysokości $h-2$. Ponieważ każdy pusty wska"xnik T jest pustym wska"xnikiem w jednym z tych poddrzew, otrzymujemy wzór $\rho(h) = \rho(h-1) + \rho(h-2)$.

Teraz niech N będzie liczbą wierzchołków w T. Z Faktu 19 i powyższych rozważań mamy

$$N + 1 > \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^{h+2} - 1,$$

co po prostych przeksztaceniach daje tezę. □

19 Operacje słownikowe na drzewach AVL

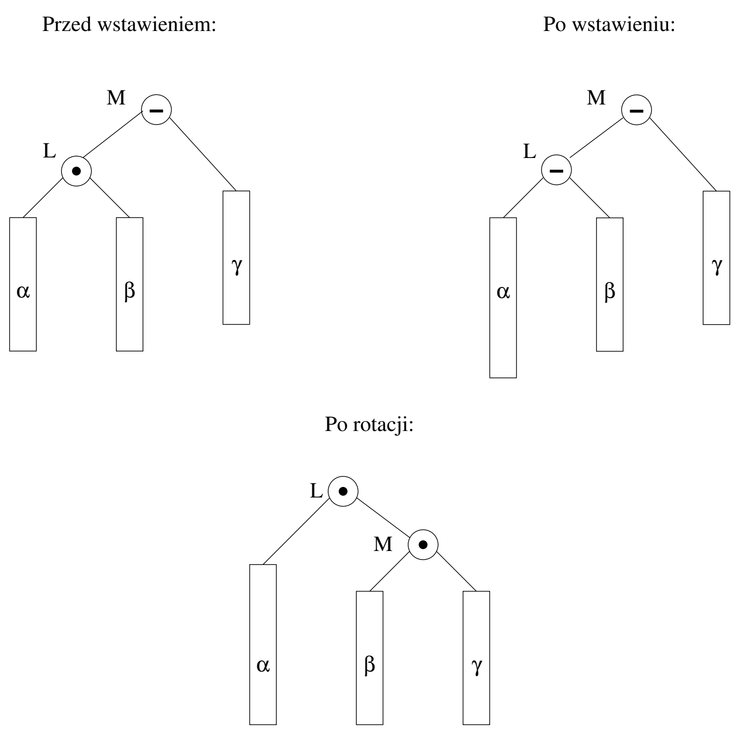
Wyszukiwanie elementu wykonuje się identycznie jak dla zwykłych binarnych drzew przeszukiwań. Pozostałe dwie operacje mogą zaburzyć strukturę drzewa AVL. Przywracanie tej struktury nazywamy *balansowaniem drzewa.*

19.1 Wstawianie elementu

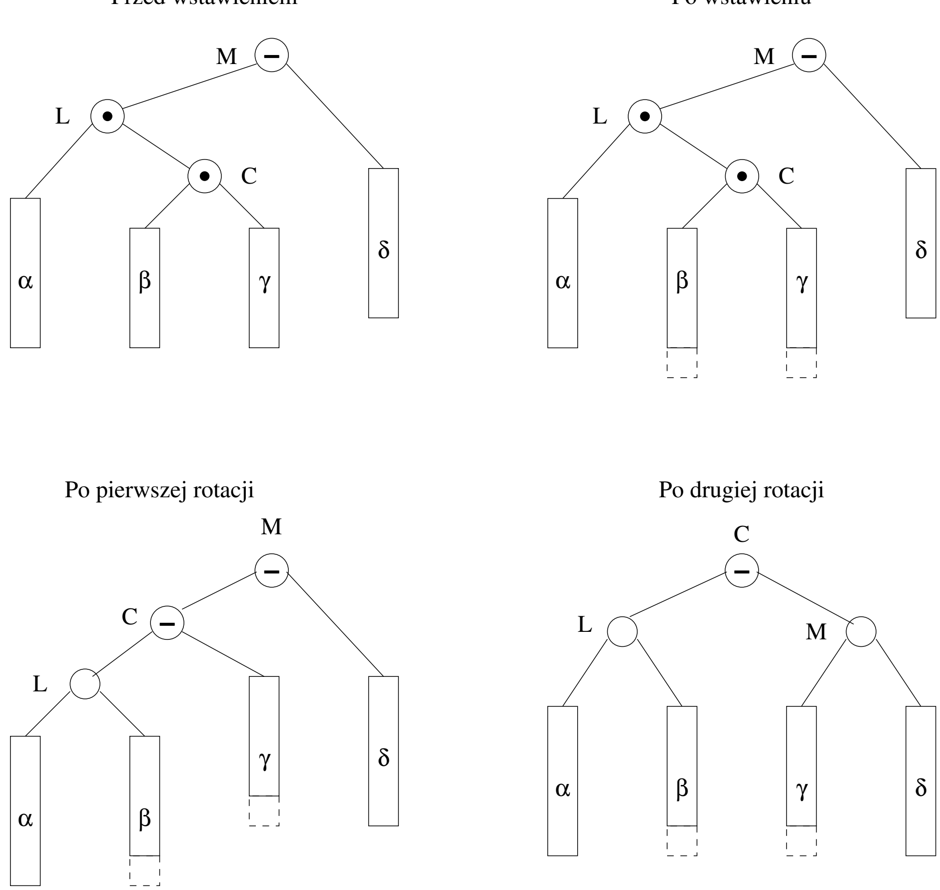
Niech M będzie pierwszym węzłem na drodze od wstawionego elementu do korzenia, w którym nastąpiło naruszenie równowagi drzewa AVL. Oznacza to, że przed operacją wstawienia poddrzewa zakorzenione w M były nierównej wysokości i wstawienie zwiększyło wysokość wyższego poddrzewa. Załóżmy, że tym poddrzewem jest lewe podrzewo i oznaczmy jego korzeń przez L (sytuacja, w której wyższym poddrzewem jest prawe poddrzewo jest symetryczna).

Procedura balansowania musi oddzielnie rozpatrywać dwa przypadki:

(A) W drzewie o korzeniu L zwiększyła się wysokość lewego poddrzewa.



(B) W drzewie o korzeniu L zwiększyła się wysokość prawego poddrzewa.



Uwaga: Po zbalansowaniu wysokość drzewa zakorzenionego w C jest równa wysokości drzewa zakorzenionego w M przed operacją wstawienia. Dlatego nie ma potrzeby przywracania zrównoważenia w innych węzłach poza M.

19.2 Usuwanie elementu

Operacja ta jest znacznie bardziej skomplikowana.
Idea:

Algorytm *DeleteAVLnode*

1. Znale"xć wierzchołek zawierający element g, który chcemy usunąć.
2. Jeśli jest to wierzchołek wewnętrzny to wstawić do niego element g' z drzewa bezpośrednio następny (bąd"x bezpośrednio poprzedni) po g.
3. Powtarzać rekurencyjnie krok 2 dla g', tak długo, aż g' będzie elementem z liścia.
4. Usunąć ten liść. Przejść drogę od tego liścia do korzenia przywracając zrównoważenie wierzchołków na tej drodze przy pomocy rotacji.

UWAGI:

1. Tym razem może się zdarzyć, że trzeba będzie dokonywać rotacji dla wszystkich wierzchołków na tej drodze.
2. Szczegółowy opis drzew AVL można znale"xć w ksiażce [1].

19.3 Koszt

Wszystkie operacje słownikowe na drzewach AVL można wykonać w czasie ograniczonym funkcją liniową od wysokości drzewa, a więc w czasie $O(\log n)$.

20 Zastosowanie drzew AVL do implementacji list

Typowymi operacjami na listach są m.in.:

1. wstawianie elementu na wskazaną pozycję,
2. usuwanie elementu ze wskazanej pozycji,
3. konkatenacja list,
4. podział listy na dwie podlisty wg zadanej pozycji.

Przy tradycyjnych implementacjach list (tj. w tablicach lub przy pomocy zmiennych wska-"xnikowych) niektóre z tych operacji wymagają czasu liniowego. Drzewa AVL pozwalają na implementację list, która umożliwia wykonanie powyższych operacji w czasie $O(\log n)$. Wystarczy w każdym wierzchołku pamiętać liczbę elementów w jego lewym poddrzewie (liczba ta wyznacza pozycję elementu w liście przechowywanej w drzewie zakorzenionym w tym wierzchołku).

Szczegóły pozostawiamy jako temat do samodzielnych studiów.

Literatura

[1] N.Wirth, *Algorytmy + Struktury Danych = Programy.*