

KOPCE FIBONACCIEGO

IIUWr. II rok informatyki.

Opracował: Krzysztof Loryś

1 Wstęp

Operacją kopcową, której do tej pory nie rozważaliśmy, a która jest ważna w wielu zastosowaniach jest operacja *decrement*(h, p, Δ), polegająca na zmniejszeniu o Δ klucza w elemencie wskazywanym przez p . Wykonywana na kopcach dwumianowych może wymagać czasu $\log n$ (np. zmniejszenie wartości klucza znajdującego się w liściu może spowodować konieczność przesunięcia go aż do korzenia). Taki czas jest nieakceptowalny, gdy liczba operacji *decrement* jest duża.

Pokażemy jak w prosty sposób zmodyfikować kopce dwumianowe, by operacja *deletemin* wykonywała się w stałym czasie amortyzowanym. Otrzymana struktura danych nosi nazwę *kopców Fibonacciego*.

2 Przykład zastosowania - algorytm Dijkstry

Algorytm Dijkstry oblicza najkrótsze odległości wszystkich wierzchołków grafu $G = (V, E)$ od ustalonego wierzchołka s (źródła). Algorytm jest zachłanny. Buduje zbiór X , wierzchołków, których najkrótsza odległość od s jest już ustalona: rozpoczyna od jednoelementowego zbioru $\{s\}$ i na każdym kroku dokłada wierzchołek spoza X leżący najbliżej s . Do wyznaczenia takiego wierzchołka służą wartości $D(u)$, które w każdej fazie algorytmu równe są długości najkrótszej ścieżki od u do s prowadzącej jedynie przez wierzchołki z X . Do pamiętania tych wartości możemy używać kopca, ponieważ na każdym kroku szukamy wierzchołka o minimalnej wartości D . Zwykle kopce nie są tu jednak odpowiednie, ponieważ dołączenie nowego wierzchołka u do X może powodować konieczność uaktualnienia (zmniejszenia) wartości pozostających w kopcu dla wszystkich wierzchołków incydentnych z u . W rezultacie na elementach kopca wykonujemy $|E|$ operacji *decrement* i $|V|$ operacji *deletemin*. Zaimplementowanie algorytmu Dijkstry przy zastosowaniu kopców Fibonacciego da w efekcie jego złożoność $O(m + n \log n)$.

```

procedure Dijkstra
   $X \leftarrow \{s\}$ 
   $D(s) \leftarrow 0$ 
  for each  $u \in V \setminus \{s\}$  do  $D(u) \leftarrow l(s, u)$ 
  while  $X \neq V$  do
    Niech  $u \in V \setminus X$  o minimalnej wartości  $D(u)$ 
     $X \leftarrow X \cup \{u\}$ 
    for each  $\langle u, v \rangle \in E$  takiej, że  $v \in V \setminus X$  do
       $D(v) \leftarrow \min(D(v), D(u) + l(u, v))$ 

```

3 Struktura kopców Fibonacciego

Podobnie jak kopce dwumianowe, kopce Fibonacciego są zbiorami drzew, których wierzchołki pamiętają elementy zgodnie z porządkiem kopcowym. Teraz jednak drzewa nie muszą być drzewami dwumianowymi.

Przyjmujemy taki sam sposób pamiętania drzew i elementu minimalnego, jak w przypadku kopców dwumianowych (wersja lazy). Ponadto w każdym wewnętrznym wierzchołku kopca pamiętamy war-

tość logiczną, mówiącą czy wierzchołek ten utracił jednego ze swoich synów w wyniku operacji *cut* - patrz niżej.

4 Operacje

Operacje *makeheap*, *insert*, *findmin* i *meld* wykonujemy w taki sam sposób jak na kopcach dwumianowych.

4.1 Operacja $cut(h, p)$

Operacja ta zastosowana do wierzchołka wewnętrznego (tj. takiego, który nie jest korzeniem) wskazywanego przez p , odcina go od swojego ojca p' i dołącza (operacją *meld* poddrzewo zakorzenione w p do listy drzew kopca. Jeśli p jest pierwszym synem jakiegoś ojca p' , to fakt ten jest zapamiętywany w p' . Jeśli p' wcześniej utracił już jakiegoś syna, to wykonujemy operację $cut(h, p')$. W ten sposób będziemy wędrować w górę drzewa odcinając odpowiednie poddrzewa tak długo, aż napotkamy korzeń lub wierzchołek, który dotąd nie utracił żadnego syna.

4.2 Operacja $decrement(h, p, \Delta)$

Zmniejszamy wartość klucza w wierzchołku wskazywanym przez p . Jeśli nowa wartość klucza zakłóca porządek kopcowy (tzn. jest mniejsza od klucza ojca wierzchołka p), wykonujemy $cut(h, p)$.

4.2.1 Zamortyzowany koszt

Teraz każdy wierzchołek ma swoje konto. Będzie ono niepuste tylko u wierzchołków, które utraciły jednego syna.

Operacji $decrement(h, p, \Delta)$ przydzielamy 4 jednostki kredytu. Jedną jednostką opłacamy koszt instrukcji niskiego poziomu i operację *meld* przyłączenia drzewa o korzeniu w p do kopca. Drugą umieszczamy na koncie tego drzewa (obowiązuje nas w dalszym ciągu niezmiennik kredytowy, mówiący, iż na koncie każdego drzewa kopca znajduje się jedna jednostka). Dwie pozostałe jednostki wykorzystujemy tylko wtedy, gdy wykonujemy $cut(h, p)$ i p jest pierwszym synem odciętym od swojego ojca. Umieszczamy je wówczas na koncie ojca p . Jednostki te są wykorzystywane do opłacenia operacji *cut* wykonanej wskutek tego, że ojciec p straci drugiego syna.

4.3 Operacja $deletemin(h)$

Deletemin wykonujemy w sposób analogiczny jak w przypadku kopców dwumianowych. W szczególności podczas redukcji łączymy drzewa o jednakowym rzędzie (zdefiniowanym jako liczba synów korzenia), otrzymując drzewo o stopniu o jeden wyższym. Jedyna różnica wynika z tego, że teraz drzewa nie są dwumianowe i nie można oczekiwać, że łączone drzewa będą identyczne.

Aby wykazać, że $O(\log n)$ nadal ogranicza czas wykonywania tej operacji musimy dowieść, że stopień wierzchołków drzew występujących w kopcach Fibonacciego jest ograniczony przez $O(\log n)$. Oczywiście będzie to także ograniczeniem na liczbę różnych rzędów drzew.

Lemat 1 *Dla każdego wierzchołka x kopca Fibonacciego o rzędzie k , drzewo zakorzenione w x ma rozmiar wykładniczy względem k .*

DOWÓD: Niech x będzie dowolnym wierzchołkiem kopca i niech y_1, \dots, y_k będą jego synami uporządkowanymi w kolejności przyłączania ich do x . W momencie przyłączania y_i do x -a, x miał co najmniej $i - 1$ synów. Stąd y_i też miał wówczas co najmniej $i - 1$ synów, ponieważ przyłączane są tylko drzewa o jednakowym rzędzie. Od tego momentu y_i mógł stracić co najwyżej jednego syna, ponieważ w przeciwnym razie zostałby odcięty od x -a. Tak więc w każdym momencie i -ty syn każdego wierzchołka ma rząd co najmniej $i - 2$.

Oznaczmy przez F_i najmniejsze drzewo o rzędzie i , spełniające powyższą zależność. Łatwo sprawdzić,

że F_0 jest drzewem jednowierzchołkowym, a F_i składa się z korzenia oraz i poddrzew: $F_0, F_0, F_1, F_2, \dots, F_{i-2}$. Tak więc liczba $|F_i|$ wierzchołków takiego drzewa jest nie mniejsza niż $1 + \sum_{j=0}^{i-2} |F_j|$, co, jak łatwo pokazać indukcyjnie, jest równe i -tej liczbie Fibonacciego. Stąd liczba wierzchołków w drzewie o rzędzie k jest nie mniejsza niż ϕ^k , gdzie $\phi = (1 + \sqrt{5})/2$. \square

Wniosek 1 *Każdy wierzchołek w n -elementowym kopcu Fibonacciego ma stopień ograniczony przez $O(\log n)$.*

4.3.1 Operacja *delete*(h, p)

Operację *delete*(h, p) można wykonać najpierw ustanawiając w p minimum kopca (poprzez operację *decrement*($h, p, -\infty$)) a następnie usuwając minimum. Zamortyzowany koszt wynosi $O(\log n)$.

UWAGA: W ten sam sposób możemy wykonywać *delete* na kopcach dwumianowych. Oczywiście wówczas *decrement* musi polegać na przesunięciu zmniejszonego elementu do korzenia drzewa.