

ALGORYTMY I STRUKTURY DANYCH

KOLEJKI PRIORYTETOWE

Instytut Informatyki Uniwersytetu Wrocławskiego

Paweł Rzechonek

1. [*] Wykaż, że odwrotnie posortowana tablica jest kopcem.
2. [*] Wykaż, że *heap-sort* nie jest stabilny.
3. **[*] Na wykładzie zdefiniowany został kopiec binarny oraz przedstawiona została jego implementacja tablicowa. Uogólnij pojęcie kopca na kopce d -arne, dla dowolnego $d \geq 2$. Jak taki kopiec włożyć w tablicę i jak potem wyliczać indeksy synów i ojca dla danego elementu? Zilustruj działanie twoich procedur na przykładzie kopca 3-arnego i 4-arnego.
Wskazówka: Użyj tablicy indeksowanej od 0.
4. [***] *Drzewem lewicowym* nazywamy drzewo binarne, w którym dla każdego wierzchołka x prawa skrajna ścieżka od x do liścia jest najkrótszą ścieżką od x do wskaźnika pustego w poddrzewie zakorzenionym w x . Opracuj implementację kolejki priorytetowej, korzystając z drzew lewicowych. Złożoność operacji kolejkowych **insert** i **extract-max** powinna być logarytmiczna względem liczby wszystkich elementów w kolejce.
Wskazówka: Zastanów się, jak połączyć dwa kopce lewicowe wzdłuż prawych skrajnych ścieżek.
5. [*] Zaimplementuj operację łączenia dwóch drzew dwumianowych z porządkiem kopcowym o takim samym stopniu, która zachowuje ten porządek w nowo powstałym drzewie i działa w stałym czasie.
6. [**] Wykaż, że w drzewie dwumianowym stopnia k , które zawiera 2^k węzłów, długość ścieżki od korzenia do dowolnego węzła w tym drzewie jest nie większa niż k .
7. [**] Jak zaimplementować drzewo dwumianowe k -tego stopnia w tablicy 2^k elementowej? Twoje rozwiązanie powinno pozwalać na łatwe łączenie dwóch drzew dwumianowych. Napisz wzory pozwalające ustalić na podstawie indeksu w tablicy numer poziomu, na którym węzeł się znajduje, jego stopień oraz indeksy ojca, brata i kolejnych synów.
8. [***] Dane jest drzewo dwumianowe stopnia k , które zawiera 2^k węzłów. W drzewie tym jest zachowany porządek kopcowy. Następnie zmieniamy w dowolnym wybranym węźle pamiętaną w nim wartość. Porządek kopcowy może więc zostać zaburzony. Jeśli zmienimy wartość na większą, to zwykle *sianie w górę* przywróci ten porządek w $O(\log n)$ krokach, a więc w akceptowalnym czasie. Jeśli natomiast zmienimy wartość na mniejszą, to *sianie w dół* przywróci porządek kopcowy w $O(\log^2 n)$ krokach, a to jest zdecydowanie za długo. Opracuj metodę, która po zmianie wartości na mniejszą we wskazanym elemencie przywróci porządek kopcowy w czasie $O(\log n)$.