ASD 2020/2021 – Klasówka 2

(14 stycznia 2021)

Zadanie 1 [6 punktów]

W tym zadaniu drzewa wyszukiwań binarnych są zaimplementowane tak, jak opisano na wykładzie. Dostęp do drzewa jest dany przez dostęp do jego korzenia.

a) [2 punkty] Zaproponuj wydajny algorytm obliczający wysokość danego AVL-drzewa.
b) [1 punkt] Do początkowo pustego drzewa typu „splay" wstawiamy kolejno klucze 1, 2, …, 2021. Ile wynosi wysokość tak otrzymanego drzewa?
c) [3 punkty] Dla h > 0, podaj jaka jest minimalna, a jaka maksymalna liczba węzłów w drzewie czerwono-czarnym o wysokości h, zawierającym dokładnie jeden węzeł czerwony.

Uwaga: wysokość drzewa mierzymy liczbą krawędzi na najdłuższej ścieżce od korzenia do liścia.

Zadanie 2 [7 punkty]

Mamy trzy, początkowo puste stosy S_1, S_2, S_3. Wykonujemy ciąg ruchów, z których każdy polega na dodaniu do wybranego dowolnie stosu jednego żetonu (operacja Push). Jeśli po dodaniu nowego żetonu, wysokości dwóch różnych stosów i oraz j są takie same oraz większe od 0, rozpoczynamy proces czyszczenia stosów realizowany za pomocą procedury Oczyść(i,j).

```
Oczyść(i,j)::
begin
   k := indeks stosu różny od i oraz j;
   repeat
      Push(Sj,Pop(Si));
      if (h(Si) > 0) AND (h(Si) = h(Sk)) then begin a := j; j := k; k := a end
      else
         if (h(Sj) = h(Sk)) then begin a := i; i := k; k := a end
   until h(Si) = 0
end;
```

Uwaga: Pop(S) jest funkcją, której wynikiem jest element usuwany ze stosu S; h(S) oznacza wysokość stosu S. W tym zadaniu operacjami elementarnymi są operacje stosowe Push i Pop.

a) [3 punkty] Udowodnij, że operacja czyszczenia zawsze się kończy.
b) [4 punkty] Ile wynosi koszt zamortyzowany jednego ruchu z uwzględnieniem operacji czyszczenia stosów.

Zadanie 3 [7 punktów]
Zaprojektuj strukturę danych, która umożliwia wydajne wykonywanie następujących operacji na dynamicznym zbiorze S złożonym z liczb całkowitych, z których każda ma przypisaną wagę całkowitoliczbową:

Ini(S):: S := ∅; {*operacja wykonywana tylko raz na samym początku*}

Insert(i,w):: wstaw do zbioru S liczbę i, której waga wynosi w; jeśli i jest już w zbiorze, to zmień wagę i na w

Interval(i):: podaj parę l, p taką, że $l \le i \le p$, dla każdego $k \in \{l, l+1, \ldots, p\}$ mamy $k \in S$, $l-1 \notin S$, $p+1 \notin \$$
{*operacja ta jest wykonywana tylko dla* $i \in S$*; polega na wyznaczeniu maksymalnego przedziału kolejnych liczb całkowitych ze zbioru S, do którego należy i*}

Weight(i):: podaj sumę wag elementów z przedziału Interval(i)
{ *operacja ta jest wykonywana tylko dla* $i \in S$ }