ćwiczenia (licencjat dzienny): lista zadań nr 6 *14 listopada 2007 r.*

# ALGORYTMY I STRUKTURY DANYCH

KOSZT ZAMORTYZOWANY

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

1. [∗∗∗] *Licznik binarny* to struktura danych reprezentowana przez $k$–elementową tablicę bitów $C$. W tablicy tej pamiętana jest nieujemna wartość całkowita zwana licznikiem. Wartość licznika jest reprezentowana w systemie binarnym i zapisana w tablicy $C$ jako ciąg bitów ($C_0$ to najmniej znaczący bit a $C_{k-1}$ najbardziej znaczący). Na liczniku można wykonywać jedynie operację *increment*, która zwiększa jego wartość o 1. Operacja ta zaprogramowana jest w oparciu o pisemne dodawanie jedynki do liczby binarnej:

```
procedure INCREMENT (bit C[0...k-1])
{
    for i = 0...k-1 do
        if C_i = 1
            then C_i ← 0;
            else { C_i ← 1; return; }
}
```

   Określ faktyczny koszt wykonania pojedynczej operacji *increment*. Przeanalizuj czas wykonania ciągu $n$ takich operacji na początkowo wyzerowanym liczniku. Dokonaj analizy zamortyzowanej na trzy sposoby: *(i)* metodą kosztu sumarycznego, *(ii)* metodą księgowania i *(iii)* metodą potencjału.

2. [∗] Uzasadnij, że jeśli dopuścimy operację *decrement* na $k$–bitowym liczniku binarnym, która zmniejsza jego wartość o 1, to ciąg $n$ operacji *increment* i *decrement* może wymagać czasu $O(nk)$.

3. [∗∗] Jaki jest całkowity koszt wykonania na *stosie* ciągu $n$ operacji *push*, *pop* i *multipop*, gdy początkowo na stosie znajdowało się $s_0 > 0$ elementów a na końcu pozostało ich $s_n$? Zastosuj *metodę kosztu sumarycznego* do wyznaczenia zamortyzowanego kosztu pojedynczej instrukcji.

4. [∗∗] Rozważny *stos* $S$ z operacjami *pop*, *push* i *multipush*. Instrukcja *multipush(x,k)* umieszcza na szczycie stosu $\min\{k, |S|\}$ wartości $x$ (ilość wkładanych na stos elementów nie może być większa niż początkowy rozmiar stosu). Jaki jest koszt ciągu $n$ operacji na takim stosie? Czy koszt zamortyzowany pojedynczej instukcji będzie stały?

5. [∗∗] Pokaż, jak zaimplementować kolejkę za pomocą dwóch zwykłych stosów, aby zamortyzowany koszt operacji *enqueue* i *dequeue* był rzędu $O(1)$. Zastosuj *metodę księgowania*.

6. [∗∗∗] Rozważmy *tablicę dynamiczną* $T$ ze współczynnikiem powiększania $\kappa = 2$ i operacjami *append* i *cut*. Instrukcja *append(x)* była omówiona na wykładzie; instrukcja *cut()* usuwa ostatni element z tablicy, a gdy współczynnik zajętości stanie się $< \frac{1}{4}$ (czyli $< \frac{1}{\kappa^2}$), to pojemność tablicy jest zmniejszana 2–krotnie ($\kappa$ razy). Zaimplementuj operację *cut*. Przeanalizuj czas wykonania ciągu $n$ operacji *append* i *cut* na początkowo pustej tablicy dynamicznej $T$. Dokonaj analizy zamortyzowanej *metodą potencjału*. Jako funkcję potencjału przyjmij:

$$\phi(T) = \begin{cases} 2 \cdot T.n - T.capacity & : \quad \alpha(T) \geq 1/2 = \frac{1}{\kappa} \\ T.capacity\,/\,2 - T.n & : \quad \alpha(T) < 1/2 = \frac{1}{\kappa} \end{cases}$$

7. [∗∗] Niech dana będzie funkcja potencjału $\phi$ taka, że $\phi(D_i) \geq \phi(D_0)$ dla wszyskich $i > 0$, oraz $\phi(D_0) \neq 0$. Wykaż, że istnieje inna funkcja potencjału $\phi'$ spełniająca warunki $\phi'(D_0) = 0$ i $\phi'(D_i) \geq 0$ dla wszystkich $i > 0$, oraz że koszty zamortyzowane operacji liczone względem $\phi'$ są takie same jak koszty zamortyzowane liczone względem $\phi$.