

Algorytmy i struktury danych

Kolokwia (3)

algorytmy grafowe, programowanie dynamiczne, algorytmy zachłanne

2016/17

B

1. Dana jest mapa kraju w postaci grafu $G = (V, E)$, gdzie wierzchołki to miasta a krawędzie to drogi łączące miasta. Dla każdej drogi znana jest jej długość (wyrażona w kilometrach jako liczba naturalna). Alicja i Bob prowadzą (na zmianę) autobus z miasta $x \in V$ do miasta $y \in V$, zamieniając się za kierownicą w każdym kolejnym mieście. Alicja wybiera trasę oraz decyduje kto prowadzi pierwszy. Proszę zaproponować algorytm, który wskazuje taką trasę (oraz osobę, która ma prowadzić pierwsza), żeby Alicja przejechała jak najmniej kilometrów. Algorytm powinien być jak najszybszy (ale przede wszystkim poprawny). Proszę oszacować złożoność zaproponowanego algorytmu, zakładając, że graf jest reprezentowany macierzowo.

2. Złodziej widzi na wystawie po kolei n przedmiotów o wartościach $A[0], A[1], \dots, A[n-1]$. Złodziej chce wybrać przedmioty o jak największej wartości, ale resztki przyzwoitości zabraniają mu ukraść dwa przedmioty leżące obok siebie. Proszę zaimplementować funkcję:

```
int goodThief( int A[], int n );
```

która zwraca maksymalną wartość przedmiotów, które złodziej może ukraść zgodnie ze swoim kodeksem moralnym oraz wypisuje numery przedmiotów które powinien wybrać. Proszę uzasadnić poprawność algorytmu oraz oszacować jego złożoność czasową. Algorytm powinien być możliwie jak najszybszy (ale przede wszystkim poprawny).

3. Dany jest zbiór przedziałów otwartych $A = \{(a_1, b_1), \dots, (a_n, b_n)\}$. Proszę zaproponować algorytm (bez implementacji), który znajduje taki zbiór X , $X \subseteq \{1, \dots, n\}$ że (a) $|X| = k$ (gdzie $k \in \mathbb{N}$ to dany parametr wejściowy), (b) dla każdych $i, j \in X$, przedziały (a_i, b_i) oraz (a_j, b_j) nie nachodzą na siebie, oraz (c) wartość $\max_{j \in X} b_j - \min_{i \in X} a_i$ jest minimalna. Jeśli podzbioru spełniającego warunki (a) i (b) nie ma, to algorytm powinien to stwierdzić. Algorytm powinien być możliwie jak najszybszy (ale przede wszystkim poprawny).

2015/16

[z forum]

1. Szachownica $N \times N$, ustawiono pewną ilość skoczków. Opisać algorytm który sprawdzi czy jest możliwa sekwencja ruchów spełniająca:
 - każdy ruch kończy się zbiciem skoczka
 - sekwencja kończy się gdy zostanie jeden skoczek.

2.

```
struct field {  
    int value;  
    int long_j;  
    int short_j;  
};
```

Z każdego pola można skakać tylko o ilość pól zapisaną w `long_j` lub `short_j`. Napisać program który zwróci maksymalną wartość jaką możemy osiągnąć poprzez przejście z pola 0 do $n-1$. Można założyć że z każdego pola da się dojść do pola $n-1$.

3. Zbiór przedziałów $\{[a_1, b_1], \dots, [a_n, b_n]\}$, każdy przedział należy do $[0, 1]$. Opisać algorytm który sprawdzi czy jest możliwy taki wybór przedziałów, aby cały przedział $[0, 1]$ zawierał się w wybranych odcinkach. Przedział ma składać się z jak najmniejszej ilości odcinków.

A

1. W miasteczku są sklepy i domy. Trzeba sprawdzić jak daleko do najbliższego sklepu mają mieszkańcy.

```
struct Vertex {
    bool shop; // true-sklep, false-dom
    int* distances; // tablica odległości do innych wierzchołków
    int* edges; // numery wierzchołków opisanych w distances
    int edge; // rozmiar tablicy distances (i edges)
    int d_store; // odległość do najbliższego sklepu
};
```

Zaimplementować funkcję `distanceToClosestStore (int n, Vertex* village)` uzupełniającą `d_store` dla tablicy Vertexów i oszacować złożoność algorytmu.

2. Problem plecakowy - dwuwymiarowa wersja problemu - oprócz ciężaru jest wysokość towarów.

$\{p_1, \dots, p_n\}$ - przedmioty $v(p_i)$ - wartość przedmiotu p_i
 $r(p_i)$ - ciężar przedmiotu p_i $h(p_i)$ - wysokość przedmiotu p_i

Jaka jest największa możliwa sumaryczna wartość przedmiotów, których ciężar nie przekracza M a wysokość S ? Oszacować złożoność i udowodnić poprawność algorytmu. (Nie trzeba implementować).

3. Forma to koniunkcja klauzul. Klauzula to alternatywa zmiennych lub ich negacji. Przykład: $(\neg a \text{ or } b)$ and $(\neg b \text{ or } c \text{ or } d)$ and $(a \text{ or } d)$ and $(\neg a \text{ or } \neg c \text{ or } d)$. Forma składa się z m klauzul (C_1, \dots, C_m) . Jest n zmiennych (x_1, \dots, x_n) . Jak ustawić zmienne, aby co najmniej połowa klauzul była spełniona? Oszacować złożoność i udowodnić poprawność algorytmu. (Nie trzeba implementować).

A

UWAGA: Wyłącznie zadanie pierwsze wymaga implementacji!

1. Niech $G = (V, E)$ będzie pewnym grafem nieskierowanym a $U \subseteq V$ pewnym podzbiorem jego wierzchołków. Grafem indukowanym $G|_U$ nazywamy graf powstały z G przez usunięcie wszystkich wierzchołków spoza U . Proszę podać i zaimplementować wielomianowy algorytm, który mając na wejściu graf $G = (V, E)$ (reprezentacja przez listy sąsiedztwa) oraz liczbę naturalną k , znajduje maksymalny co do rozmiaru zbiór $U \subseteq V$ taki, że wszystkie wierzchołki w $G|_U$ mają stopień większy lub równy k . Proszę oszacować czas działania algorytmu.
2. Na wejściu dana jest formuła logiczna F postaci CNF (conjunctive normal form), czyli $F = C_1 \wedge C_2 \wedge \dots \wedge C_m$, gdzie każde C_i to tak zwana klauzula, czyli alternatywa zmiennych logicznych lub ich negacji. Formuła w postaci CNF jest Hornowska jeśli każda klauzula zawiera najwyżej jedną niezanegowaną zmienną. Przykłady Hornowskich formuł CNF to: $(x) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y} \vee z)$, $(\bar{x} \vee y) \wedge (x \vee \bar{y})$, czy $(x) \wedge (\bar{x})$. Proszę opisać wielomianowy algorytm, który sprawdza czy Hornowska formuła w postaci CNF jest spełnialna (to znaczy, czy da się przypisać wartości logiczne zmiennym tak, żeby formuła miała wartość *prawda*).
3. Mamy dany ciąg napisów $S = (s_1, \dots, s_n)$ oraz pewien napis t . Wiadomo, że t można zapisać jako złączenie pewnej ilości napisów z S (z powtórzeniami). Na przykład dla $S = (s_1, s_2, s_3, s_4, s_5)$ gdzie $s_1 = ab$, $s_2 = abab$, $s_3 = ba$ oraz $s_4 = bab$, $s_5 = b$, napis $t = ababbab$ można zapisać, między innymi, jako s_2s_4 lub jako $s_1s_1s_3s_5$. Pierwsza reprezentacja ma "szerokość" 3 (przez szerokość rozumiemy długość najkrótszego s_i użytego w reprezentacji) a druga 1. Proszę opisać algorytm, który mając na wejściu S oraz t znajdzie maksymalną szerokość reprezentacji t . Proszę oszacować czas działania algorytmu.