Podstawy i Zastosowania Złożoności Obliczeniowej
Zadanie 1

Jakub Grobelny

Podpunkt a

Teza.
$3COL \leq_P Tutorzy$

Dowód.
Pokażemy, że istnieje obliczalna w czasie wielomianowym funkcja f taka, że $\forall_G(G \in 3COL \Leftrightarrow f(G) \in Tutorzy)$.

```
procedura f(G(V, E)):
    przemianuj(G)
    konflikty := []
    studenci := |V| + 1
    dla każdego v z V:
        dodaj {"zrzeda": |V| + 1, "nielubiany": v} do konflikty
        dla każdego w sąsiadującego z v:
            dodaj {"zrzeda": v, "nielubiany": w} do konflikty
    zwróć {"studenci": studenci, "konflikty": konflikty}

procedura przemianuj(G(V, E)):
    dla każdego i z 1...|V|:
        zmień nazwę wierzchołka V[i] na i
```

Funkcja f obliczana przez powyższy program zamienia dany graf G na instancję problemu $Tutorzy$ poprzez zamienienie każdego wierzchołka na studenta. Każdy wierzchołek v, od którego w grafie G wychodzi krawędź incydentna do wierzchołka w, staje się *zrzędą* v, która nie chce mieć tego samego tutora co student w. Dodatkowo tworzymy jednego sztucznego studenta o numerze $|V| + 1$, który nie lubi wszystkich innych studentów. Na początku wywołujemy procedurę `przemianuj`, która sprawia, że wszystkie wierzchołki mają nazwy będące liczbami z odpowiedniego zakresu.

Widać, że program działa w czasie $O(|V| \cdot |E| + T(G))$, gdzie $T(G)$ to czas działania procedury `przemianuj`, gdyż każdy wierzchołek grafu odwiedzamy jeden raz, a dla każdego wierzchołka sprawdzamy jego sąsiadów również tylko raz (w najgorszym przypadku każdy wierzchołek sąsiaduje z każdym). Dodawanie obiektów do listy można zrealizować w czasie stałym a pomocnicza procedura `przemianuj` działa w czasie wielomianowym. Funkcja f jest zatem obliczalna w czasie wielomianowym względem rozmiaru danego grafu.

Pokażemy teraz, że $\forall_G(G \in 3COL \Leftrightarrow f(G) \in Tutorzy)$:

$\Rightarrow$

Weźmy dowolny graf $G \in 3COL$. Skoro G jest 3-kolorowalny, to jeżeli dodamy do niego nowy wierzchołek v, taki że sąsiaduje z każdym innym istniejącym już wierzchołkiem, to graf ten będzie 4-kolorowalny (wystarczy wziąć dowolne 3-kolorowanie G a nowy wierzchołek pomalować na nowy, czwarty kolor). Niech G^+ będzie grafem G z dodanym wyżej opisanym wierzchołkiem v. Wówczas każdy wierzchołek z G ma odpowiadającego mu studenta w $f(G)$, zaś dodatkowy wierzchołek v odpowiada sztucznie dodanemu studentowi, który nie lubi nikogo. Każda krawędź z G^+ jest reprezentowana przez *konflikty*.

Wprowadźmy bijekcję $g : Kolor \to Tutor$, która odwzorowuje kolory na tutorów. Każdemu studentowi s możemy przydzielić tutora $g(c)$, gdzie c jest kolorem wierzchołka w z G^+ odpowiadającego studentowi s. Skoro G^+ jest 4-kolorowalny, a g jest bijekcją, to żaden student z $f(G)$ nie będzie w grupie ze studentem, którego nie lubi, czyli $f(G) \in Tutorzy$.

$\Leftarrow$

Weźmy dowolny graf G taki, że $f(G) \in Tutorzy$. Pokażemy, że $G \in 3COL$.

Niech $\mathtt{T} = f(G)$ i niech H będzie grafem o poniższej konstrukcji:

- Zbiór wierzchołków grafu H to $\{1, 2, ..., \texttt{T.studenci}\}$
- Dla dowolnych v i w, jeżeli do `T.konflikty` należy `{"zrzeda": `v`, "nielubiany": `w`}`, to wówczas w H istnieje krawędź pomiędzy v a w.

H jest w oczywisty sposób 4-kolorowalny, gdyż wybór jednego spośród czterech tutorów odpowiada wyborowi jednego spośród czterech kolorów, a skoro krawędzie w grafie H opisują relację *konflikty* z obiektu `T`, to wówczas mamy gwarancję, że dwa wierzchołki o tym samym kolorze (tutorze) nie sąsiadują ze sobą w grafie H.

Z definicji f wiemy również, że w `T` istnieje student s, który jest *zrzędą* i nie lubi wszystkich innych studentów. W takim razie w H istnieje wierzchołek v, taki że jest połączony z każdym innym wierzchołkiem w tym grafie. Skoro $f(G) \in Tutorzy$, to student s jest jedynym studentem przypisanym do swojego tutora t, bo w przeciwnym razie byłby w grupie razem z kimś, kogo nie lubi. Z tego wynika, że w 4-kolorowaniu grafu H odpowiadającemu przydziałowi tutorów, wierzchołek v jako jedyny ma swój kolor c. Niech H^- będzie grafem powstałym poprzez usunięcie wierzchołka v oraz incydentnych do niego krawędzi z grafu H. W oczywisty sposób H^- jest 3-kolorowalny.

Możemy zauważyć, że $\forall_{G',H'}(f(G') = f(H') \Leftrightarrow G'$ jest izomorficzny z $H')$. Własność ta wynika z tego, że pomocnicza procedura `przemianuj` przeetykietowała jedynie wierzchołki (graf otrzymany po zaaplikowaniu `przemianuj` do grafu jest izomorficzny z oryginalnym) a dalsza część przekształcenia f jest w pełni odwracalna. Widać, że $f(G) = f(H^-)$, bo H^- ma dokładnie takie same krawędzie jak G (z dokładnością do nazw wierzchołków) oraz tyle samo wierzchołków. W takim razie grafy H^- oraz G są izomorficzne. Zaaplikowanie izomorfizmu nie zmienia kolorowalności grafu, więc G jest 3-kolorowalny, czyli $G \in 3COL$.

□

Podpunkt b

Teza.
Problem $Tutorzy$ można rozwiązać w czasie wielomianowym, przy założeniu, że będzie co najwyżej 15 zrzęd.

Dowód.

Pokażemy, że dla dowolnej stałej liczby zrzęd K istnieje algorytm działający w czasie wielomianowym, który rozwiązuje problem $Tutorzy$.

```
procedura tutorzy(S):
    S' := S bez studentów, którzy nie są zrzędami
    P  := wszystkie poprawne przydziały tutorów dla S'
    jeżeli P jest puste:
        zwróć NIE
    dla każdego p z P:
        dla każdego studenta s z S, który nie jest zrzędą:
            T := tutorzy przydzieleni zrzędom nielubiącym s
            jeżeli |T| = 4:
                wyjdź z pętli i rozpatrz kolejne p
        zwróć TAK
    zwróć NIE
```

Zasada działania:

Algorytm najpierw przydziela tutorów dla podzbioru studentów, którzy są zrzędami (dowolnym algorytmem rozwiązującym ogólny problem $Tutorzy$). Jeżeli nie da się przydzielić tutorow dla samych zrzęd, to wówczas można od razu zwrócić odpowiedź „`NIE`", gdyż tym bardziej nie można wtedy przydzielić tutorów reszcie studentów. Następnie rozpatrujemy wszystkie poprawne przydziały tutorów dla zrzęd. Żeby dokończyć przydział tutorów, musimy wybrać tutora dla każdego pozostałego studenta. Można zauważyć, że jedyni studenci, jakich trzeba rozważyć przy doborze tutora, to zrzędy. Algorytm zlicza wszystkich unikalnych tutorów jacy zostali przydzieleni zrzędom, które nie lubią aktualnie rozpatrywanego studenta. Jeżeli jest to czterech różnych tutorów, to wówczas nie ma żadnego dostępnego dla rozpatrywanego studenta. W takim przypadku wychodzimy z wewnętrznej pętli i rozpatrujemy kolejny przydział `p`. Jeżeli okaże się, że w wewnętrznej pętli udało się znaleźć tutora dla każdego studenta, to zwracamy odpowiedź „`TAK`". Jeżeli rozpatrzymy bez sukcesu wszystkie przydziały `p`, to wówczas zwracamy odpowiedź „`NIE`".

Analiza złożoności poszczególnych kroków:

1. Przefiltrowanie studentów aby uzyskać instancję `S'` problemu można wykonać w czasie wielomianowym. Wystarczy wziąć listę konfliktów z `S`, usunąć te konflikty, gdzie zrzęda nie lubi studenta, który nie jest zrzędą a następnie wyznaczyć nowe numery dla studentów aby leżały w odpowiednim przedziale.

2. Przydziały `P` możemy wyznaczyć dowolnym algorytmem rozwiązującym problem *Tutorzy*, ponieważ w `S` jest nie więcej niż K studentów (a K jest ustaloną stałą), więc zajmuje to czas $O(1)$.

3. `|P|` jest ograniczne przez stałą 4^K. W związku z tym, zewnętrzna pętla wykona się stałą liczbę razy.

4. Liczba wykonań wewnętrznej pętli jest rzędu $O($`S.studenci`$)$, a więc liniowa.

5. Wyznaczenie `T` zajmuje najwyżej czas $K\cdot$ `S.studenci` (może być w czasie stałym jeżeli przed rozpoczęciem pętli stworzylibyśmy macierz sąsiedztwa studentów, w której zapisana byłaby informacja o tym, przez jakich studentów dany student jest nielubiany).

Sumarycznie czas potrzebny na wykonanie programu `tutorzy(S)` jest wielomianowy.

Pokazaliśmy, że istnieje algorytm rozwiązujący problem *Tutorzy* z ograniczeniem do K zrzęd w czasie wielomianowym. Jeżeli istnieje taki algorytm dla dowolnej stałej K, to w szczególności da się rozwiązać taki problem w czasie wielomianowym przy założeniu, że będzie co najwyżej 15 zrzęd, co należało udowodnić.

□