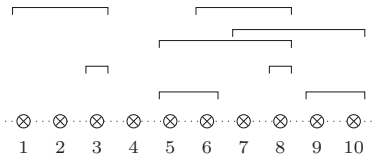
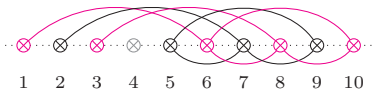
Informatyczny kącik olimpijski (98): Świąteczny łańcuch

Tym razem omówimy zadanie *Świąteczny łańcuch*, które rozwiązywali w tym roku uczestnicy drugiego etapu XXIII Olimpiady Informatycznej. Zadanie jest następujące: należy zaprojektować łańcuch złożony z n różnokolorowych lampek, przy czym dane jest również m wymagań estetycznych, każde w postaci trójki liczb (a_i, b_i, ℓ_i) oznaczającej, że fragmenty łańcucha złożone z lampek o numerach $\{a_i, \ldots, a_i + \ell_i - 1\}$ oraz $\{b_i, \ldots, b_i + \ell_i - 1\}$ muszą być jednakowe (rys. 1). Łańcuch powinien być też jak najbardziej urozmaicony; innymi słowy, powinno być w nim jak najwięcej różnych kolorów lampek. Wartości n i m są rzędu 10^6.



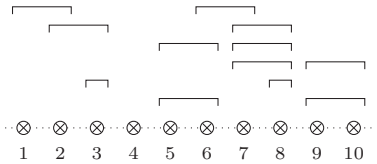
Rys. 1. Przykładowy łańcuch o $n = 10$ lampkach i czterech wymaganiach estetycznych $(1, 6, 3)$, $(5, 7, 4)$, $(3, 8, 1)$ oraz $(5, 9, 2)$. Ostatnie wymaganie mówi, że lampki 5 i 9 muszą mieć ten sam kolor oraz lampki 6 i 10 muszą mieć ten sam kolor.

Rozwiązanie, za które można było zdobyć połowę punktów na konkursie, jest narzucające się. Zbudujmy graf o n wierzchołkach (ponumerowanych liczbami od 1 do n), które odpowiadać będą kolejnym lampkom łańcucha. Teraz dla każdego wymagania estetycznego (a_i, b_i, ℓ_i) łączymy krawędziami wierzchołki dla tych lampek, które muszą mieć ten sam kolor; innymi słowy, dla każdego $j = 0, 1, \ldots, \ell_i - 1$ łączymy krawędzią wierzchołki $a_i + j$ oraz $b_i + j$. Zauważmy, że wszystkie lampki należące do jednej spójnej składowej grafu muszą mieć ten sam kolor. Z kolei lampki z różnych składowych mogą mieć różne kolory, więc aby zmaksymalizować liczbę kolorów w łańcuchu, należy każdej składowej przypisać inny kolor lampek (rys. 2). To rozwiązanie działać będzie w czasie liniowym od wielkości skonstruowanego grafu. Jest zatem zbyt wolne, gdyż liczba krawędzi grafu może być bliska iloczynowi $n \cdot m$, który może wynieść nawet 10^{12}.



Rys. 2. Graf odpowiadający wymaganiom z rysunku 1. Ma trzy spójne składowe $\{1, 3, 6, 8, 10\}$, $\{2, 5, 7, 9\}$ oraz $\{4\}$, więc łańcuch może mieć co najwyżej 3 różne kolory lampek.

Zauważmy, że powyższe rozwiązanie jest wolne, jeśli istnieje dużo wymagań estetycznych o dużych długościach (wartościach ℓ_i). Jednak w takim przypadku jest nieuniknione, że spora część wymagań będzie redundantna (tzn. będzie prowadziła do takich samych wymuszeń kolorów). W naszym przykładzie wymaganie $(5, 7, 4)$ mówi, że fragmenty $\{5, 6, 7, 8\}$ i $\{7, 8, 9, 10\}$ mają takie same kolory lampek, co może być spełnione tylko wtedy, gdy krótsze fragmenty $\{5, 6\}$, $\{7, 8\}$ oraz $\{9, 10\}$ mają te same kolory. Widać zatem, że wymaganie $(5, 9, 2)$ jest w tym przypadku zawsze spełnione. Chcielibyśmy usunąć takie redundantne wymagania. Ponadto, w przypadku wymagań o dużych długościach być może tylko część informacji z wymagania jest redundantna – w takim przypadku dobrym pomysłem mogłoby być podzielenie takiego wymagania na mniejsze kawałki.



Rys. 3. Zastąpienie wymagań dłuższych niż $2^k = 2$ w fazie $k = 1$.

W szybszym rozwiązaniu wykorzystamy te dwa pomysły: będziemy sukcesywnie rozbijać wymagania na mniejsze i na bieżąco usuwać redundancje. Rozwiązanie będzie przebiegać w $\lfloor \log_2 n \rfloor + 1$ fazach dla $k = \lfloor \log_2 n \rfloor, \ldots, 1, 0$. W fazie k-tej zakładamy, że wszystkie wymagania mają długość co najwyżej 2^{k+1}, a następnie każde wymaganie (a_i, b_i, ℓ_i) o długości większej niż 2^k (czyli spełniającej $2^k < \ell_i \leqslant 2^{k+1}$) zastępujemy dwoma wymaganiami o długości 2^k (rys. 3):

$$(a_i, b_i, 2^k) \qquad \text{oraz} \qquad (a_i + \ell_i - 2^k,\ b_i + \ell_i - 2^k,\ 2^k).$$

To spowoduje tymczasowy wzrost liczby wymagań, ale pozwoli nam usunąć redundancje wśród wymagań o długości 2^k. Zbudujmy w tym celu graf G_k, w którym wierzchołki będą znów liczbami od 1 do n, ale będziemy mieli co najwyżej $2m$ krawędzi: dla każdego wymagania $(a_i, b_i, 2^k)$ stworzymy krawędź łączącą wierzchołki a_i oraz b_i. Zauważmy teraz, że jeśli w grafie G_k znajduje się cykl, to znaczy, że część wymagań o długości 2^k daje redundantne informacje. Istotnie: cykl $v_1 \to v_2 \to \ldots \to v_s \to v_1$ wymusza, że dla każdego $0 \leqslant j < 2^k$ kolory lampek ze zbioru $\{v_i + j \mid 1 \leqslant i < s\}$ są takie same. Jednak to samo wymuszenie uzyskujemy, jeśli usuniemy dowolne wymaganie odpowiadające krawędzi z tego cyklu. Powtarzając to rozumowanie dopóty, dopóki w grafie G_k istnieją cykle, możemy zmniejszyć zbiór wymagań długości 2^k do zbioru liczącego $n - 1$ wymagań. Ten zbiór możemy znaleźć, wyznaczając dowolny las rozpinający grafu G_k.

Zatem na koniec fazy k-tej uzyskamy równoważny zbiór wymagań zawierający co najwyżej $m + n$ wymagań długości co najwyżej 2^k. Pojedynczą fazę możemy zaimplementować w czasie $O(n + m)$. Tak więc po fazie $k = 0$ uzyskamy równoważny zbiór $m + n$ wymagań, z których każde jest długości 1. Dla takiego zbioru nasze pierwotne rozwiązanie zadziała w czasie liniowym. Ostatecznie złożoność czasowa algorytmu wyniesie $O((n + m) \log n)$, a pamięciowa $O(n + m)$.

Tomasz IDZIASZEK