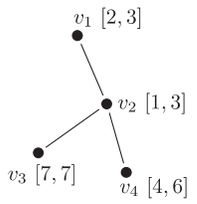
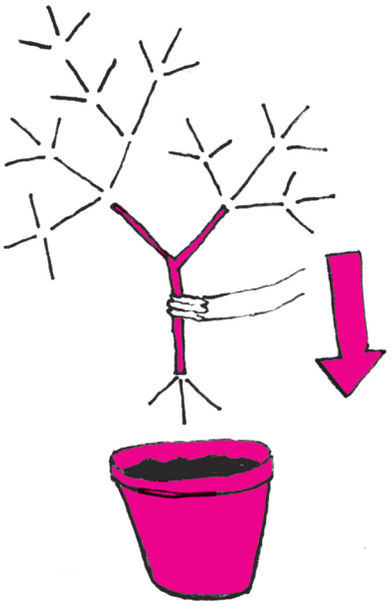
Informatyczny kącik olimpijski (86): Sieć światłowodowa

Tym razem omówimy zadanie *Fiber-optic Network*, które pojawiło się na zeszłorocznych zawodach ACM-ICPC Asia Mudanjiang Regional Contest. W zadaniu tym mamy dane drzewo o n wierzchołkach, a dla każdego wierzchołka v zadane liczby naturalne $1 \leqslant L_v \leqslant R_v \leqslant M$ (ograniczenia to $n \leqslant 50$, $M \leqslant 50\,000$). *Dobrym przyporządkowaniem* będziemy nazywać przyporządkowanie każdemu wierzchołkowi v liczby całkowitej z odpowiadającego mu przedziału $[L_v, R_v]$ w taki sposób, aby liczby przyporządkowane wierzchołkom połączonym krawędzią były względnie pierwsze. Dla przejrzystości oznaczmy, że dla danego przyporządkowania f liczba przypisana wierzchołkowi v to $f(v)$. W zadaniu proszą nas, aby dla każdego wierzchołka obliczyć sumę liczb przyporządkowanych mu we wszystkich dobrych przyporządkowaniach. Wyniki należy podać modulo $10^9 + 7$, zatem wszystkie działania będziemy wykonywać modulo ta liczba.



Dla $f(v_2) = 1$ liczby w pozostałych trzech wierzchołkach możemy wybrać dowolnie, co daje 6 przyporządkowań $(2/3, 1, 7, 4/5/6)$. Dla $f(v_2) = 2$ mamy jedno przyporządkowanie $(3, 2, 7, 5)$, a dla $f(v_2) = 3$ dwa $(2, 3, 7, 4/5)$. Odpowiedzi dla kolejnych wierzchołków to 22, 14, 63 i 44.

Chwila myślenia i odrobina algorytmicznego rozsądku doprowadzają nas do wniosku, że w zadaniu nie ma na tyle „regularności” (cokolwiek by to znaczyło), aby dało się rozwiązać je inaczej niż poprzez wyznaczenie dla każdego wierzchołka v oraz każdej możliwej wartości k, dla ilu dobrych przyporządkowań zachodzi $f(v) = k$. To właśnie będzie naszym celem.



Jak można się spodziewać, zapewne sporo nam pomoże, jeżeli najpierw ukorzenimy nasze drzewo (w dowolnym wierzchołku). Rozwiązanie będzie bazowało na programowaniu dynamicznym. Oznaczymy przez T_v poddrzewo ukorzenione w wierzchołku v, a przez $dp[v][k]$ liczbę dobrych przyporządkowań, dla których zachodzi $f(v) = k$, gdy ograniczymy się jedynie do wierzchołków z poddrzewa T_v. Niewątpliwie dla v będących liśćmi zachodzi $dp[v][k] = 1$, gdy $L_v \leqslant k \leqslant R_v$, i $dp[v][k] = 0$ dla pozostałych wartości k. Załóżmy teraz, że chcemy wyznaczyć odpowiedź dla wierzchołka v, który ma s synów $u_1, \ldots, u_s$ i mamy już wyznaczone tablice $dp[u_1], \ldots, dp[u_s]$. Liczba dobrych przyporządkowań dla poddrzewa T_v spełniających $f(v) = k$ to iloczyn liczb dobrych przyporządkowań dla poddrzew T_{u_i}, dla których $\mathrm{nwd}(f(u_i), k) = 1$. Zatem jeśli wprowadzimy oznaczenie

$$P[u][k] := \sum_{\mathrm{nwd}(k,j)=1} dp[u][j]$$

(zauważmy, że tak zapisana suma jest dobrze określona, gdyż tylko skończona liczba składników jest niezerowa), to dostaniemy wzór

$$dp[v][k] = P[u_1][k] \cdot \ldots \cdot P[u_s][k]. \tag{1}$$

Moglibyśmy po prostu przeiterować po wszystkich możliwych wartościach k przyporządkowanych każdemu wierzchołkowi u_i, ale wtedy otrzymalibyśmy rozwiązanie co najmniej kwadratowe w zależności od M, co byłoby zdecydowanie zbyt wolne. Dlatego stajemy przed kluczowym problemem szybkiego wyznaczenia $P[u][k]$ dla ustalonego syna u. Skoro mowa o względnej pierwszości, to z pewnością warto poznać zbiór (różnych) dzielników pierwszych liczby k; niech będą to liczby $p_1, \ldots, p_m$. Liczby względnie pierwsze z k to wszystkie liczby poza tymi, które są podzielne przez którąkolwiek z liczb $p_1, \ldots, p_m$. Dla $m = 1$ mamy $P[u][k] = \sum_j dp[u][j] - \sum_{p_1|j} dp[u][j]$, co umożliwia szybkie obliczenie żądanej wartości, o ile znalibyśmy wcześniej $\sum_{p_1|j} dp[u][j]$. Życie nie jest jednak takie proste i w ogólnym przypadku nie możemy po prostu odjąć $\sum_{p_i|j} dp[u][j]$ dla $i = 1, \ldots, m$, gdyż wiele ze składników postaci $dp[u][j]$ odjęlibyśmy więcej niż raz (konkretnie tyle razy, ile j ma dzielników wśród liczb $p_1, \ldots, p_m$).

Jeśli wprowadzimy oznaczenia $w(A) := \sum_{a \in A} dp[u][a]$ oraz A_l na zbiór liczb podzielnych przez liczbę l, to dostaniemy zależność

$$P[u][k] = \sum_j dp[u][j] - w(A_{p_1} \cup \ldots \cup A_{p_m}). \tag{2}$$

Na ratunek w obliczeniu (2) przychodzi nam *zasada włączeń i wyłączeń*, a właściwie jej minimalne uogólnienie:

$$w(A_{p_1} \cup \ldots \cup A_{p_m}) = \sum_{1 \leqslant l \leqslant m} \sum_{i_1 < \ldots < i_l} (-1)^{l+1}\, w(A_{p_{i_1}} \cap \ldots \cap A_{p_{i_l}}).$$

Rozwiązanie zadania M 1473.
Odpowiedź. Wystarczą dwa pytania.

Niech wielomian, który wymyśliła Aga, to będzie $P(x) = a_n x^n + \ldots + a_0$. Najpierw Bartek pyta o $P(1)$. Zauważmy, że wówczas $a_i \leqslant P(1)$ dla każdego i. Niech $b = P(1) + 1$. Pytając o $P(b)$ i zapisując tę liczbę w systemie o podstawie b, Bartek otrzyma liczbę

$$a_n b^n + \ldots + a_0 = (a_n \ldots a_0)_b$$

o cyfrach $a_n, \ldots, a_0$, czyli odgadnie P.

Pomimo przerażającego wyglądu wzór ten jest całkiem prosty, ale jeżeli ktoś nie spotkał się z nim wcześniej, warto się zatrzymać i pokontemplować go przez chwilę, np. przekonując się o jego poprawności dla $m = 2, 3$. (Oczywiście, w ogólnej wersji wzoru nie ma znaczenia, czym są konkretnie zbiory A_{p_i} i pozostaje on prawdziwy dla dowolnych zbiorów.)

Zatem kolejną rzeczą, którą musimy zbadać, są zbiory postaci $A_{p_{i_1}} \cap \ldots \cap A_{p_{i_l}}$. Wszystkie liczby $p_{i_1}, \ldots, p_{i_l}$ są pierwsze, więc jeżeli pewna liczba ma być podzielna przez wszystkie z nich, to musi być podzielna przez ich iloczyn (stwierdzenie odwrotne oczywiście też jest prawdziwe), co prowadzi nas do zależności $A_{p_{i_1}} \cap \ldots \cap A_{p_{i_l}} = A_{p_{i_1} \cdots p_{i_l}}$. Dzięki niej wzór (2) możemy wyrazić następująco:

Przykładowo, jeżeli chcemy obliczyć liczby, które nie są względnie pierwsze z $2 \cdot 3 \cdot 5 = 30$, to musimy dodać liczby podzielne przez 2, przez 3, przez 5, odjąć liczby podzielne przez $2 \cdot 3 = 6$, przez $2 \cdot 5 = 10$, przez $3 \cdot 5 = 15$ i dodać liczby podzielne przez $2 \cdot 3 \cdot 5 = 30$. Warto też zauważyć, że potęgi, w których dane liczby pierwsze dzielą k, nie mają żadnego znaczenia – liczby względnie pierwsze z $2 \cdot 3 \cdot 5$ to te same liczby, co liczby względnie pierwsze z $2^2 \cdot 3 \cdot 5$.

$$P[u][k] = \sum_j dp[u][j] + \sum_{1 \leqslant l \leqslant m} \sum_{\substack{i_1 < \ldots < i_l \\ p_{i_1} \cdots p_{i_l} \mid j}} (-1)^l \, dp[u][j]. \tag{3}$$

Widać, że wielokrotnie spotykamy się tutaj z podobnymi sumami składników $dp[u][j]$, zatem wprowadzając oznaczenie $S[u][i] := \sum_{i \mid j} dp[u][j]$, możemy przepisać wzór (3) w trochę milszej postaci:

$$P[u][k] = S[u][1] + \sum_{1 \leqslant l \leqslant m} \sum_{i_1 < \ldots < i_l} (-1)^l \, S[u][p_{i_1} \cdots p_{i_l}]. \tag{4}$$

Zauważmy, że jeżeli mamy tablicę $dp[u]$, to jesteśmy w stanie relatywnie szybko wyznaczyć tablicę $S[u]$. A konkretnie, aby obliczyć $S[u][i]$, potrzebujemy wysumować $\frac{M}{i}$ składników. Zatem wykonanie tego dla $i = 1, \ldots, M$ będzie miało złożoność czasową $O\big(M\big(1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{M}\big)\big) = O(M \log M)$.

Zastanówmy się teraz, jaką złożoność ma wyznaczanie $P[u][k]$ za pomocą wzoru (4). Zauważmy, że występują w nim jedynie składniki postaci $S[u][i]$, gdzie i jest pewnym dzielnikiem k. Ponadto nie wszystkie dzielniki uwzględniamy (tylko takie niepodzielne przez kwadraty liczb pierwszych), ale ta obserwacja nie jest nam potrzebna, aby uzyskać sensowne oszacowanie. Zatem, aby wyznaczyć wartości $P[u][k]$ dla $k = 1, \ldots, M$, musimy wysumować co najwyżej tyle składników, ile jest łącznie dzielników liczb $1, \ldots, M$, ale ich liczba jest również rzędu $O(M \log M)$. Aby szybko się dostać do tych składników, warto na samym początku programu dla każdej liczby od 1 do M spamiętać jej bezkwadratowe dzielniki.

Podsumowując, mając tablice $dp[u_i]$ dla wszystkich synów wierzchołka v, możemy w czasie $O(sM \log M)$ kolejno wyznaczyć tablice $S[u_i]$ i $P[u_i]$, co pozwoli nam skorzystać ze wzoru (1) do wyznaczenia $dp[v]$. Zatem w sumarycznym czasie $O(nM \log M)$ możemy tą metodą wyznaczyć odpowiedź dla korzenia drzewa. Uruchamiając powyższy algorytm dla każdej możliwości wyboru korzenia, możemy rozwiązać zadanie w czasie $O(n^2 M \log M)$. Jednak autorów zadania taka złożoność nie satysfakcjonuje i wymagane jest rozwiązanie $O(nM \log M)$.

Pomysł pozwalający obliczać nam kompletne informacje dla wszystkich wierzchołków drzewa (a nie tylko te uwzględniające wierzchołki z odpowiadających im poddrzew), polega na spychaniu informacji z korzenia w dół, przesyłając informacje z „naddrzewa”. Niech v będzie pewnym wierzchołkiem, a w jego ojcem w drzewie. Pamiętamy, że tablica dp jest iloczynem odpowiednich wartości tablic P dla sąsiadów w poddrzewie, dlatego jeżeli mamy kompletną tablicę $dp[w]$, to aby obliczyć wkład naddrzewa v o korzeniu w wierzchołku w, powinniśmy podzielić wartości z tablicy $dp[w]$ przez odpowiadające wartości z tablicy $P[v]$, przeliczyć jeszcze raz wartości tablic $S[w]$ i $P[w]$ na podstawie nowych wartości $dp[w]$ (tablice te teraz lekko zmieniają swoje znaczenie, gdyż zmieniamy zbiór sąsiadów, którzy są uwzględniani w ich liczeniu), co pozwala nam wyliczyć kompletną tablicę $dp[v]$ (tzn. taką, w której uwzględniony jest też ojciec). Aby wyznaczyć kompletne $dp[v]$ na podstawie informacji z ojca, potrzebujemy wykonać stałą liczbę przeliczeń tablic, z których każde odbywa się w czasie $O(M \log M)$, co daje algorytm o sumarycznej złożoności $O(nM \log M)$.

Wojciech NADARA

Rozwiązanie zadania F 890.
Początkowe ciśnienie w oponie jest równe p_0, a więc musimy do niej wtłoczyć dodatkowo

$$n = \frac{(p - p_0)V}{\mathrm{R}T}$$

moli powietrza, gdzie V jest objętością opony

$$V = \frac{\pi^2 (D - d/2) d^2}{2}.$$

Te n moli powietrza w temperaturze T pod ciśnieniem p_0 zajmowało objętość

$$V_0 = \frac{n\mathrm{R}T}{p_0}.$$

Sprężając izotermicznie n moli gazu od objętości V_0 do V, wykonujemy pracę

$$W = n\mathrm{R}T \ln\left(\frac{V_0}{V}\right).$$

Ostatecznie otrzymujemy

$$W = (p - p_0)V \ln\left(\frac{p - p_0}{p_0}\right).$$

Po podstawieniu danych liczbowych mamy $W \approx 2450$ J, czyli mniej niż potrzeba do podgrzania 1 litra wody o 1 stopień.