Informatyczny kącik olimpijski (112): Ciąg arytmetyczny i Kamyczki

Tym razem omówimy dwa zadania z jubileuszowej X Olimpiady Informatycznej Gimnazjalistów w roku szkolnym 2015/2016.

Ciąg arytmetyczny: *Dany jest ciąg n liczb naturalnych $a_1, a_2, \ldots, a_n$. Naszym zadaniem jest odpowiedzieć na q zapytań postaci: czy podsłowo $a_i, a_{i+1}, \ldots, a_{j-1}, a_j$ jest ciągiem arytmetycznym?*

Zacznijmy od przypomnienia definicji ciągu arytmetycznego. Otóż, $a_i, a_{i+1}, \ldots, a_{j-1}, a_j$ jest ciągiem arytmetycznym, jeśli różnice między kolejnymi parami wyrazów są równe, czyli:

$$a_{i+1} - a_i = a_{i+2} - a_{i+1} = \ldots = a_j - a_{j-1}.$$

Rozwiązanie $O(q \cdot n)$
Najprostsze rozwiązanie polega na sprawdzeniu dla każdego zapytania, wprost z definicji, czy dany ciąg jest arytmetyczny. W tym celu wyznaczamy różnicę ciągu jako $r = a_{i+1} - a_i$. Następnie sprawdzamy, czy różnica między każdymi dwoma sąsiednimi wyrazami jest równa r. Powyższe rozwiązanie działa w czasie $O(q \cdot n)$.

Rozwiązanie $O(n + q \cdot \log(n))$
Niech r będzie ciągiem różnic między kolejnymi wyrazami:

$$\underbrace{a_2 - a_1}_{r_1}, \underbrace{a_3 - a_2}_{r_2}, \underbrace{a_4 - a_3}_{r_3}, \ldots, \underbrace{a_n - a_{n-1}}_{r_{n-1}}$$

Ciąg $a_i, a_{i+1}, \ldots, a_{j-1}, a_j$ jest arytmetyczny, jeśli odpowiadający mu ciąg różnic $r_i, r_{i+1}, \ldots, r_{j-1}$ jest stały (wszystkie wyrazy są równe). Niech zatem x będzie najmniejszą liczbą w ciągu, zaś y największą. Wówczas wszystkie różnice znajdują się w przedziale $[x; y]$. Stąd, jeśli $x = y$, to ciąg różnic jest stały, czyli pierwotny ciąg jest arytmetyczny. Do wyznaczenia najmniejszego i największego elementu w przedziale można wykorzystać drzewo przedziałowe, które pozwala zrealizować tę operację w czasie $O(\log(n))$. Całe rozwiązanie działa w czasie $O(n + q \cdot \log(n))$.

Rozwiązanie $O(n + q)$
Okazuje się, że istnieje jeszcze szybsze rozwiązanie. Zauważmy, że ciąg różnic możemy podzielić na fragmenty o tych samych wartościach:

$$\overbrace{r_1, r_2}^{1}, \overbrace{r_3, r_4, r_5}^{2}, \ldots, \overbrace{r_{n-2}, r_{n-1}}^{k}$$

Wówczas fragment $r_i, r_{i+1}, \ldots, r_{j-1}$ jest stały, jeśli wszystkie jego wyrazy należą do jednej grupy. Przejdźmy teraz do opisu generowania grup. Ustalmy, że pierwsza różnica tworzy pierwszą grupę. Pozostałe różnice będziemy przeglądali od lewej do prawej. Jeśli rozpatrywana różnica jest równa poprzedniej, to rozszerzamy poprzednią grupę, w przeciwnym przypadku tworzymy nową grupę. Rozwiązanie działa w czasie $O(n + q)$.

Kamyczki: *Danych jest n stosów kamyczków. Wysokości kolejnych stosów to $a_1, a_2, \ldots, a_n$. Celem Małgosi, bohaterki zadania, jest opróżnienie wszystkich stosów. Dziewczyna w jednym ruchu może opróżnić jeden dowolnie wybrany stos (nazwijmy ten ruch A) lub zabrać po jednym kamieniu z wszystkich niepustych stosów (nazwijmy ten ruch B). Ile minimalnie ruchów musi wykonać dziewczynka, aby opróżnić wszystkie stosy?*

Obserwacja: Istnieje optymalne rozwiązanie, w którym najpierw wykonujemy ruchy typu A, a następnie ruchy typu B.

Dowód: Weźmy dowolne optymalne rozwiązanie. Jeśli w tym rozwiązaniu istnieje taki ruch x typu A, który występuje po jakimś ruchu typu B, to ruch x nie ma wpływu na pozostałe stosy oraz wcześniejsze ruchy typu B nie są konieczne do wykonania ruchu x. Stąd ruch x może zostać wykonany przed wszystkimi ruchami typu B.

Na mocy powyższej obserwacji w optymalnym rozwiązaniu najpierw wykonujemy ruchy typu A, a następnie ruchy typu B. Zauważmy, że maksymalna liczba ruchów typu A to n (jest co najwyżej n stosów do opróżnienia). Dla każdej możliwej liczby ruchów A (od 0 do n) obliczmy minimalną liczbę ruchów B, która pozwoli opróżnić wszystkie stosy.

Zastanówmy się teraz, jak dla ustalonej liczby k, oznaczającej liczbę ruchów A, wybrać k stosów do usunięcia tak, aby zminimalizować liczbę ruchów B. Zauważmy wcześniej, że po wykonaniu k ruchów typu A należy wykonać tyle ruchów B, ile wynosi wysokość najwyższego z pozostałych stosów. Stąd, aby zminimalizować liczbę ruchów typu B, należy wybrać k najwyższych stosów, co minimalizuje wysokość najwyższego z pozostałych stosów.

Przejdźmy teraz do opisu sposobu znajdowania k najwyższych stosów. Kolejność stosów nie ma znaczenia, dlatego zacznijmy od posortowania ciągu a rosnąco oraz ustalenia, że $a_0 = 0$. Wówczas k najwyższych stosów to: $a_{n-k+1}, a_{n-k+2}, \ldots, a_n$. Zatem wynik dla ustalonego k to $k + a_{n-k}$ (wykonujemy k ruchów typu A, usuwając k najwyższych stosów, oraz a_{n-k} ruchów typu B, opróżniając wszystkie pozostałe stosy, z których najwyższy ma wysokość a_{n-k}).

Najmniejszą liczbą ruchów, potrzebną do opróżnienia wszystkich stosów, jest minimum z wyników dla każdego k od 0 do n. Rozwiązanie działa w czasie $O(n \cdot \log(n))$.

Bartosz ŁUKASIEWICZ