Szkice rozwiązań

XVIII OIJ, zawody drugiego stopnia
2 marca 2024

Olimpiada Informatyczna Juniorów

Dziękujemy Wam za udział w zawodach II stopnia XVIII Olimpiady Informatycznej Juniorów. Mamy nadzieję, że dobrze bawiliście się rozwiązując zadania olimpijskie i że udało się Wam zdobyć jak najwięcej punktów. Poniżej znajduje się opis proponowanych rozwiązań.

Kulturalna kolejka

Rozwiązanie wolne

Możliwe jest wykonanie symulacji stanu kolejki, sekunda po sekundzie. Konieczne jest w tym celu napisanie procedury, która ze stanu kolejki w sekundzie t ustala stan kolejki w sekundzie $t+1$, a następnie wielokrotne uruchomienie tej procedury aż do uzyskania stanu, w którym kolejka jest pusta.

Aby ustalić następny stan kolejki, wystarczy każde wystąpienie fragmentu `"X."` zamienić na `".X"`. Można przy tym założyć, że salon sprzedażowy jest kropką, którą co każdą iterację przywracamy znowu na kropkę, jeżeli wszedł tam klient.

Wykonanie przejścia do następnego stanu kosztuje więc czas $O(n)$, a łatwo można pokazać, że takich iteracji w najgorszym przypadku również należy wykonać $O(n)$, stąd dostajemy, że algorytm działa w czasie $O(n^2)$. Pozwalało to na zawodach osiągnąć przyzwoitą częściową punktację (około 44%).

Rozwiązanie wzorcowe

Spójrzmy na zadanie inaczej: rozważając kolejne osoby w kolejce, od prawej do lewej, ustalmy dla każdej z nich, jaki jest czas, w jakim doszliby do salonu, gdyby nie było nikogo innego w kolejce, a następnie dodajmy do tego stratę wynikającą z liczby sytuacji, w której ta osoba nie mogła się przesunąć do przodu. Maksimum z sumy tych wartości po wszystkich osobach stojących w kolejce to rozwiązanie zadania.

Jeżeli pewne dwie osoby kiedykolwiek w całym procesie przejścia do salonu spotkają się w kolejce (będą sąsiednimi znakami `X` z wejścia) to jeżeli pierwsza dojdzie do salonu w czasie t sekund, to druga dojdzie w czasie $t+2$ (jedna sekunda dłużej ze względu na zwiększoną odległość oraz jeszcze jedna sekunda dłużej ze względu na zwiększone opóźnienie ruchu). Rozważając więc osoby od prawej do lewej możliwe jest ustalenie dla każdej kolejnej osoby jej parametrów wynikowych opisane w akapicie powyżej: ponieważ osoby w kolejce nigdy się nie wyprzedzają, jedyną osobą, która potencjalnie może spowolnić daną osobę A zgodnie z opisem powyżej jest osoba B, która bezpośrednio ją poprzedza. A ponieważ wyniki ustalane są od prawej do lewej, wiadomo po jakim czasie osoba B opuści kolejkę, a więc możliwe jest ustalenie czy takie spowolnienie nastąpi. Ostatecznie, wszelkie obliczenia zajmują czas $O(n)$.

Ciąg rosnący

Rozwiązanie wolne

Możliwe jest zgadywanie parametru K i wykonanie symulacji czy uzyskane rozwiązanie byłoby wtedy poprawne, zapamiętując rozwiązanie, które pozostawia najwięcej elementów. Kandydatów na wartość K jest tyle ile wynosi maksymalna wartość w ciągu, a naiwne sprawdzenie każdego z nich kosztuje czas $O(n)$, co nie pozwala myśleć o zaliczeniu największych testów, a jedynie o częściowych punktach (około 36%) za testy z mniejszymi ograniczeniami.

Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2022/2023, 2023/2024, 2024/2025".

Rozwiązanie wzorcowe

Jeżeli pewna liczba x występuje w ciągu więcej niż jeden raz, to możemy ją zignorować, ponieważ gdyby została wybrana do rozwiązania to uzyskany ciąg nie byłby ściśle rosnący. Wyklucza więc to z rozważań wszystkie K będące dzielnikami liczby x.

Ponieważ interesują nas jedynie wartości występujące w ciągu wejściowym dokładnie jeden raz, możemy zapamiętać tablicę pozycji $pozycja[\cdot]$: niech $pozycja[x]$ jest pozycją liczby x w ciągu (możemy przyjąć -1, jeżeli x występuje w ciągu więcej niż raz oraz -2, jeżeli x nie występuje w ciągu w ogóle). Możliwe jest teraz przyspieszenie sprawdzania pojedynczej wartości K: aby dana wartość K stanowiła poprawnego kandydata na ciąg zgodny z warunkami Bajtka, w ciągu $pozycja[K], pozycja[2K], pozycja[3K], \ldots$ nie może być żadnej wartości -1, zaś po pominięciu wszystkich wartości -2, powinniśmy uzyskać ciąg rosnący. Koszt obliczenia tablicy $pozycja[\cdot]$ wynosi $O(n)$, zaś koszt sprawdzenia pojedynczego K wynosi w takim wypadku $O(\frac{\max\{A_i\}}{K})$. Niech $M = \max\{A_i\}$, wtedy $\frac{M}{1} + \frac{M}{2} + \ldots + \frac{M}{M} = \Theta(M \log M)$, stąd nasze rozwiązanie działa w czasie $O(n + M \log M)$, co przy $M \leq 10^6$ prowadzi do uzyskania maksymalnej punktacji.

Scrabble

Na początek zauważmy, że możemy zliczyć ile literek każdego typu jest w słowie P, którym dysponuje Bajtosia. Można stworzyć tablicę $ile[\cdot]$, która dla każdej pozycji w alfabecie zlicza powyższą wartość (w tym celu można skorzystać z konwersji typu `char` na `int` w C++ lub analogicznej funkcji `ord` w Pythonie).

Rozwiązanie wolne

Możliwe jest rozciąganie okienka (przedziału) na słowie S. Wystarczy przetestować każdą możliwą pozycję rozpoczęcia i i zakończenia j obliczając tablicę $ile'[\cdot]$ dla fragmentu $S[i..j]$. Aby sprawdzić czy fragment znajdujący się w okienku od i do j może być skonstruowany z literek Bajtosi, wystarczy sprawdzić czy dla każdego znaku alfabetu x zachodzi $ile'[x] \leq ile[x]$.

Tablicy $ile'[\cdot]$ nie trzeba obliczać za każdym razem od nowa. Jeżeli indeks j przesunie się o jeden do przodu (poszerzenie okienka z prawej strony), wystarczy zwiększyć jedną komórkę tablicy. Każde poprawne okienko (w którym Bajtosia ma dostatecznie dużo literek każdego typu), które spełnia $j - i + 1 \geq \frac{|P|}{2}$ zwiększa wynik, który należy wypisać o 1.

Takie rozwiązanie nie otrzyma maksymalnej punktacji, ponieważ sprawdzanych par (i, j) jest kwadratowo wiele względem długości wejścia, a czas sprawdzania pojedynczej pary jest proporcjonalny do rozmiaru alfabetu. Prowadzi to do uzyskania jedynie częściowej punktacji (około 46%).

Możliwe jest przyspieszenie czasu sprawdzania pojedynczego okienka do czasu stałego, jeżeli sprawdzać tylko przedłużenia dobrych okienek w tym miejscu tablicy ile', które się zmieniło, jest to jednak nadal niewystarczające.

Rozwiązanie wzorcowe

Okazuje się, że niewielka optymalizacja będzie prowadziła do uzyskania rozwiązania liniowego: sprawdzając w powyższym rozwiązaniu okienka zaczynające się na pozycji i, można znaleźć największe j, dla którego okienko $S[i..j]$ jest konstruowalne z dostępnych liter. Skoro więc okienko $S[i..j]$ jest dobre, to również okienko $S[i+1..j]$ będzie dobre: nie ma więc potrzeby sprawdzać każdej pary (i, j), a w każdym kroku możliwe jest jedynie przesuwanie albo i albo j do przodu, łatwo też zaktualizować tablicę $ile'[\cdot]$ zmniejszając jedynie jedną komórkę. W ten sposób łączna liczba sprawdzonych kandydatów będzie $O(|S|)$ (jest liniowo wiele przesunięć indeksu i oraz liniowo wiele przesunięć indeksu j).

Możliwe jest więc nawet osiągnięcie łącznego czasu obliczeń $O(|S| + |P|)$, co prowadziło oczywiście do uzyskania maksymalnej punktacji. Rozwiązania multiplikatywnie gorsze o stałą rozmiaru alfabetu miały szansę również na uzyskanie podobnych wyników, zależnie od jakości implementacji.

Olimpiada finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2022/2023, 2023/2024, 2024/2025".