Informatyczny kącik olimpijski (65): Opóźnione wyszukiwanie

W tym kąciku zajmiemy się zadaniem *Delayed search*, które pojawiło się w 2004 r. w konkursie Internet Problem Solving Contest, organizowanym co roku przez Słowaków. Zadanie jest wariacją na temat znanej zabawy „zgadnij, o jakiej liczbie myślę": organizatorzy wybrali pewną liczbę naturalną m z przedziału $[1, n]$, a my mamy ją wyznaczyć, zadając pytania „jak się ma wybrana liczba do liczby k", na które dostajemy odpowiedź, że jest ona mniejsza, większa lub równa k. Zadanie ma jednak dodatkowe urozmaicenie: odpowiedź na i-te pytanie dostajemy dopiero po zadaniu pytania numer $d + i$ (w szczególności, pierwszą odpowiedź po zadaniu $d + 1$ pytań). Ponieważ za każde zadane pytanie naliczane są nam punkty karne, chcemy znaleźć liczbę m za pomocą jak najmniejszej liczby pytań. Zabawa kończy się, gdy usłyszymy odpowiedź „wybrana liczba jest równa k". Warto dodać, że chcemy zminimalizować liczbę pytań zadanych w pesymistycznym przypadku – organizatorzy nie muszą ustalać m zawczasu, mogą swą decyzję opierać na naszych pytaniach tak, aby wymusić na nas zadanie jak największej ich liczby.

Dla $d = 0$ rozwiązanie jest znane: używamy wyszukiwania binarnego. Każde pytanie dzieli przedział, w którym może znajdować się m, na pół, potrzebujemy zatem $\lfloor \log n \rfloor + 1$ pytań. Pożyteczne będzie przyjrzenie się tej strategii bliżej. Załóżmy, że mamy ustaloną liczbę pytań, k. Zastanówmy się, jaka jest największa długość przedziału n, dla której k pytań wystarczy do wyznaczenia m? Oznaczmy tę liczbę przez $\ell_d(k)$.

Oczywiście, $\ell_0(1) = 1$, bo musimy zadać jedno pytanie o liczbę 1, by dostać odpowiedź, że zgadliśmy poprawnie. Załóżmy, że mamy $k > 1$ oraz przedział $[1, n]$ i że zadajemy pierwsze pytanie o liczbę i. Mamy do rozważenia trzy przypadki: albo mamy szczęście, $m = i$ i gra się kończy, albo $m < i$, zostaje nam $k - 1$ pytań i wiemy, że szukana liczba znajduje się w przedziale $[1, i - 1]$, albo $m > i$, mamy $k - 1$ pytań i przedział $[i + 1, n]$. Ponieważ $k - 1$ pytań wystarczy dla przedziału o długości $\ell_0(k - 1)$, więc zmieścimy się w limicie pytań, o ile $i - 1, n - i \leqslant \ell_0(k - 1)$. Opłaca nam się, żeby przedziały $[1, i - 1]$ i $[i + 1, n]$ były jak najdłuższe, więc możemy przyjąć dowolne $n \leqslant 1 + 2\ell_0(k - 1)$ oraz $i = \lceil n/2 \rceil$. Zatem

$$\ell_0(k) = 1 + 2\ell_0(k - 1).$$

Rozwiązując tę rekurencję, dostajemy, że $\ell_0(k) = 2^k - 1$, co dla „klasycznych" 20 pytań daje nam przedział długości $\ell_0(20) = 1\,048\,575$.

Dla $d \geqslant 1$ możemy zastosować trywialną strategię: po każdym pytaniu czekamy na odpowiedź, zadając d „głupich" pytań. Taka strategia wymaga zadania $(d + 1)(\lfloor \log n \rfloor + 1)$ pytań. Nie jest zbyt dobrze, gdyż w ten sposób 20 pytań pozwoli nam dla $d = 1$ jedynie na przedział długości 1023. Zdradzimy zaś, że optymalna strategia daje $\ell_1(20) = 10\,945$. Musimy więc umieć wyznaczać kolejne pytania, zanim dostaniemy odpowiedź na poprzednie.

Rozważmy przypadek $d = 1$. Załóżmy, że pierwsze dwa pytania są o liczby i oraz j (bez straty ogólności $i < j$). W tym momencie znamy już odpowiedź na pierwsze z tych pytań. Jeśli więc gra się nie skończyła, to albo $m < i$ i wtedy pytanie o j jest stracone (wiemy na pewno, że $m < j$), zatem mamy $k - 2$ pytań i przedział $[1, i - 1]$. Jeśli zaś $m > i$, to zostajemy z przedziałem $[i + 1, n]$, w którym zadaliśmy już jedno pytanie j. Mogliśmy więc przewidzieć tę sytuację i użyć jako j pierwszego pytania z optymalnej strategii dla $d = 1$ i przedziału długości $n - i$. Dzięki temu nie marnujemy tego pytania i w sumie zostaje nam $k - 1$ pytań. Stosując rozumowanie jak poprzednio, mamy

$$\ell_1(k) = 1 + \ell_1(k - 2) + \ell_1(k - 1).$$

Ponieważ $\ell_1(1) = 0$, $\ell_1(2) = 1$, więc rozwiązaniem rekurencji jest $\ell_1(k) = F_k - 1$, gdzie F_k jest k-tą liczbą Fibonacciego. Zatem strategia postępowania dla $d = 1$ jest następująca: zadajemy pytanie, dzieląc najdłuższy przedział w złotej proporcji.

Podobną strategię możemy zastosować dla $d \geqslant 2$. Zadajemy d pytań: pierwsze o liczbę i, a następne o coraz większe liczby zgodnie z optymalną strategią dla przedziału długości $n - i$. Jeśli gra się nie skończyła, to albo $m < i$ i mamy $k - d - 1$ pytań na przedziale długości $i - 1$, albo $m > i$ i mamy $k - 1$ pytań na pozostałej części przedziału. Zatem

$$\ell_d(k) = 1 + \ell_d(k - d - 1) + \ell_d(k - 1).$$

Okazuje się, że powyższa strategia jest optymalna. Dowód tego faktu jest nieco techniczny, zainteresowanych Czytelników odsyłamy do pracy [1]. Jej autorzy podają również „życiowy" przykład zastosowania „opóźnionego" wyszukiwania binarnego. Chcieli oni mianowicie wyznaczyć optymalny poziom trudności prac domowych dla grupy studentów, zadając im prace i analizując ich wyniki (tzn. sprawdzając, czy sobie z nimi poradzili, czy nie). Z uwagi jednak na organizację zajęć studenci dostawali nową pracę domową co tydzień, a mieli ją oddać na następnych zajęciach. Zatem profesorowie musieli przygotować zadania na zajęcia w tygodniu $i + 1$, zanim sprawdzili zadania z tygodnia i – zastosowali zatem powyższy algorytm dla $d = 1$.

Tomasz IDZIASZEK

[1] A. Ambainis, S.A. Bloch, D.L. Schweizer, *Delayed Binary Search, or Playing Twenty Questions with a Procrastinator*, SODA 1999.