O liczbach autobiograficznych

Piotr ZARZYCKI, Ryszard KUBIAK***

*Instytut Matematyki, Uniwersytet Gdański
**Biuro Informatyki Ubezpieczeniowej Pachocki i Ziajka S.C., Gdańsk

Rzadko się zdarza, by matematyka pojawiała się w gazetach codziennych, bardzo rzadko ludzie niezajmujący się zawodowo matematyką o niej rozmawiają (takie rozmowy zdarzają się najczęściej w okresie egzaminów kończących jakiś etap nauczania). Początek roku 2020 był pod tym względem wyjątkowy, kilkoro znajomych (niematematyków) pytało nas, czy wiemy, jaką szczególną cechę ma liczba 2020 – nie wiedzieliśmy. Sprawdziliśmy... Okazało się, że doniesienia o niezwykłości tej liczby pojawiły się w codziennej prasie. Na przykład w internetowym wydaniu Gazety Wyborczej z 1 stycznia bieżącego roku można znaleźć artykuł Piotra Cieślińskiego o tytule „2020". Ten rok ma w sobie matematyczną zagadkę. Autor pisze tak: *Zauważmy, że jej pierwsza cyfra jest zarazem liczbą zer w jej zapisie dziesiętnym, druga – liczbą jedynek, trzecia – liczbą dwójek, czwarta – liczbą trójek. [...] Podobną własność ma liczba 3 211 000 [...] Liczby o takiej własności, których zapis „opowiada" o nich samych, nazywane są liczbami autobiograficznymi. Jest ich bardzo mało.* Naprawdę bardzo mało. Liczbami autobiograficznymi są na przykład: 1210, 2020, 21 200, 3 211 000... Wielokropek wskazuje tu, że ta lista jest niepełna. Czytelnik Dociekliwy z pewnością spróbuje uzupełnić ją samodzielnie.

Zacznijmy od **zalet dydaktycznych** liczb autobiograficznych. Na powyższej liście nie ma liczb jedno-, dwu- bądź trzycyfrowych. Znalezienie ich to ładne zadanie dla uczniów szkoły podstawowej, nawet w nauczaniu wczesnoszkolnym. Znalezienie liczb autobiograficznych o czterech lub pięciu cyfrach, jak też uzasadnienie nieistnienia liczby autobiograficznej sześciocyfrowej może być ciekawe dla uczniów szkół średnich. Problem znalezienia liczby autobiograficznej siedmiocyfrowej pojawił się w Scottish Mathematical Challenge Examination 1978–1979, a problem dotyczący liczby autobiograficznej dziesięciocyfrowej w Wisconsin Mathematics Science and Engineering Talent Search Examination 1987–1988.

Proponujemy jeszcze jeden przykład dydaktycznego wykorzystania zadania znalezienia liczb autobiograficznych. Polecamy nagranie Tanya Khovanova na YouTube *Can you solve the Leonardo da Vinci riddle?*

Są podane dwie liczby 1210 i 3 211 000. Aby wejść do pomieszczenia, należy podać trzecią, najdłuższą liczbę – szyfr otwierający drzwi.

Znalezienie liczb autobiograficznych cztero- i pięciocyfrowych wymaga rozpatrzenia stosunkowo niedużej liczby przypadków. Dla liczb o sześciu lub więcej cyfrach możliwości jest tak wiele, że warto do poszukiwań zaangażować komputer. Do żmudnej pracy szukania liczb autobiograficznych siedmio- i ośmiocyfrowych wykorzystajmy **program MATHEMATICA**. Najpierw przyjrzyjmy się następującej funkcji:

```
sprawdzenie[n_]:=
If[RotateRight[DigitCount[n]]==PadRight[IntegerDigits[n],10],
Return[True],Return[False]]
```

Wynikiem polecenia `DigitCount[n]` jest lista dziesięciocyfrowa. Jej pierwszy element to liczba jedynek w zapisie liczby n, drugi to liczba dwójek, ..., dziesiąty to liczba zer. `RotateRight` przesuwa elementy listy o jedno miejsce w lewo, a pierwszy element na koniec listy. Komenda `IntegerDigits[n]` zapisuje liczbę w postaci listy kolejnych jej cyfr, natomiast `PadRight` dopisuje z prawej strony listy `IntegerDigits[n]` brakującą liczbę zer tak, by otrzymana lista miała 10 elementów. Program do szukania siedmiocyfrowych liczb autobiograficznych jest następujący:

```
auto7:=(L={};Do[If[sprawdzenie[n]==True,AppendTo[L,n],n=n+1],
{n,1000000,9999999}];Return[L])
```

Startując od listy pustej i sprawdzając kolejne liczby siedmiocyfrowe, po czym ewentualnie dołączając (komenda `AppendTo`) do listy znalezione liczby autobiograficzne, otrzymujemy **po około 10 minutach** liczbę 3 211 000. Niewielka modyfikacja programu `auto7` pozwala znaleźć liczbę autobiograficzną ośmiocyfrową, ale przeszukiwanie trwało **ponad godzinę**. Dla liczb ośmiocyfrowych warto zmniejszyć zakres przeszukiwań, od 10 000 000 do 59 999 999. Łatwo uzasadnić, że ośmiocyfrowa liczba autobiograficzna nie

Rozwiązanie zadania M 1658.
Każdego uczestnika turnieju potraktujmy jako wierzchołek, a każdy mecz jako krawędź pewnego grafu skierowanego T, skierowaną od zwycięzcy do przegranego. Usadzenie k zawodników przy okrągłym stole zgodnie z warunkami zadania jest równoważne istnieniu skierowanego cyklu długości k (krótko: *k-cyklu*).

Udowodnimy, że:
- w T istnieje 3-cykl;
- jeśli w T istnieje k-cykl, to istnieje też $(k+1)$-cykl, dla $k = 3, \ldots, n-2$.

Wyniknie stąd, że w T istnieje $(n-1)$-cykl, czyli teza zadania.

Niech $A_1 \to A_2 \to \ldots \to A_n \to A_1$ będzie wyjściowym n-cyklem oraz niech $i \geqslant 3$ będzie najmniejszym indeksem o tej własności, że $A_i \to A_1$ (zbiór takich indeksów jest niepusty, gdyż należy do niego n). Wówczas $A_1 \to A_{i-1} \to A_i \to A_1$, więc w T istnieje 3-cykl.

Niech $B_1 \to B_2 \to \ldots \to B_k \to B_1$ będzie pewnym k-cyklem, gdzie $3 \leqslant k \leqslant n-2$, oraz niech X będzie dowolnym wierzchołkiem spoza tego cyklu. Jeśli istnieje takie i, że $B_i \to X \to B_{i+1}$, to uzyskujemy $(k+1)$-cykl poprzez wstawienie X pomiędzy B_i oraz B_{i+1} (przyjmujemy $B_{k+1} = B_1$).
W przeciwnym przypadku są dwie możliwości: $X \to B_i$ dla każdego $i = 1, 2 \ldots, k$ lub $B_i \to X$ dla każdego $i = 1, 2, \ldots, k$; niech $\mathcal{X}_1$ będzie zbiorem wierzchołków X pierwszego typu, a $\mathcal{X}_2$ – zbiorem wierzchołków X drugiego typu.

Gdyby $\mathcal{X}_1 = \varnothing$, to $B_i \to X$ dla każdego i oraz każdego $X \in \mathcal{X}_2$ – nie mógłby wtedy jednak istnieć n-cykl w T. Podobnie gdyby $\mathcal{X}_2 = \varnothing$, to w T nie mógłby istnieć n-cykl. Również gdyby $X_1 \to X_2$ dla każdej pary $X_1 \in \mathcal{X}_1$, $X_2 \in \mathcal{X}_2$, to w T nie mógłby istnieć n-cykl. Stąd wniosek, że istnieją wierzchołki $X_1 \in \mathcal{X}_1$, $X_2 \in \mathcal{X}_2$ o tej własności, że $X_2 \to X_1$. To oznacza, że istnieje $(k+1)$-cykl

$$X_1 \to B_1 \to B_2 \to \ldots \to B_{k-1} \to X_2 \to X_1.$$

Rozwiązanie zadania F 1014.
Szereg promieniotwórczy ^{238}U to kolejno następujące rozpady α i β^-. Szereg kilkakrotnie się rozgałęzia, ale „wszystkie drogi prowadzą” do końcowego, stabilnego izotopu ^{206}Pb. Na każdej drodze uwalnianych jest 8 cząstek α, a powstające w wyniku kolejnych rozpadów jądra mają czasy połowicznego rozpadu wiele rzędów wielkości krótsze niż ^{238}U – najdłuższy z nich to $2{,}45 \cdot 10^5$ lat dla ^{234}U. Z początkowej liczby n moli ^{238}U w czasie od powstania Ziemi do dziś rozpadnie się:

$$\Delta n = \left(1 - \exp(-\frac{t_Z \ln(2)}{\tau})\right) n,$$

a w wyniku tych rozpadów powstanie $8\Delta n$ moli ^{4}He. W warunkach normalnych (temperatura 0°C, ciśnienie 101 325 Pa) hel jest gazem o objętości molowej $V_m \approx 22{,}414 \cdot 10^{-3}$ m^{3}mol^{-1}. Dla danych zadania: $n = 1$ kg/238 g $\approx 4{,}202$ mola, $8\Delta n \approx 16{,}988$ mola ^{4}He wypełniłby objętość

$V = 16{,}988 \cdot 22{,}414 \cdot 10^{-3}$ m^3 $\approx 0{,}381$ m^3.

może się zaczynać od cyfr 6, 7, 8 ani 9. Na dalsze próby, dla liczb dziewięcio- i dziesięciocyfrowych nie zdecydowaliśmy się, bo czas oczekiwania byłby już nieznośnie długi.

Do poszukiwań liczb autobiograficznych wykorzystaliśmy też **język programowania Python**. Oto kod programu (wersja Pythona co najmniej 3.5):

```
from typing import List

def liczby_kandydatki(dlugosc: int):
    return range(10**(dlugosc-1), 10**dlugosc)

def cyfry_liczby(liczba: int) -> List[int]:
    return [ord(c) - ord('0') for c in str(liczba)]

def ile_razy_w_liscie(liczba: int, lista: List[int]) -> int:
    return len([k for k in lista if k == liczba])

def jest_autobiograficzna(liczba: int) -> bool:
    cyfry = cyfry_liczby(liczba)
    return sum(cyfry) == len(cyfry) and \
        all([ile_razy_w_liscie(n, cyfry) == cyfry[n]
             for n in range(len(cyfry))])

def liczby_autobiograficzne_dlugosci(dlugosc: int) -> List[int]:
    return [n for n in liczby_kandydatki(dlugosc)
            if jest_autobiograficzna(n)]

for ile_cyfr in range(4, 7):
    print('Liczby autobiograficzne długości ' + str(ile_cyfr) + ':')
    print(liczby_autobiograficzne_dlugosci(ile_cyfr))
```

Instrukcja `for ile_cyfr in range(4,7)` na końcu tego programu wyszuka liczby autobiograficzne o długości od 4 do 6. Trwa to kilka sekund. Czytelnik może sprawdzić działanie programu dla liczb o większej liczbie cyfr, zmieniając granice zakresu `range`. Warto tu zwrócić uwagę na warunek `sum(cyfry) == len(cyfry)` w funkcji `jest_autobiograficzna`. Odnosimy się tu do ważnej, a zarazem łatwej do udowodnienia własności liczb autobiograficznych:

(1) *Jeśli $a = a_0 a_1 \ldots a_n$ jest zapisem dziesiętnym liczby autobiograficznej a, to*

$$\sum_{i=0}^{n} a_i = n + 1.$$

Badanie w programie warunku `sum(cyfry) == len(cyfry)` ma na celu przyspieszenie obliczeń. Chodzi o to, że jeśli ten warunek nie jest spełniony, to program nie bada drugiego członu koniunkcji. Ten drugi człon sprawdza spełnienie przez liczbę warunku autobiograficzności. Owo sprawdzenie, czy liczba spełnia definicję autobiograficzności, ma większą złożoność czasową niż porównanie, czy suma cyfr liczby zgadza się z liczbą tych cyfr. Schemat wyszukiwania liczb w przedstawionym powyżej programie pozwolił znaleźć liczby autobiograficzne co najwyżej siedmiocyfrowe w ciągu około minuty, liczbę ośmiocyfrową w ciągu około 5 minut, natomiast do znalezienia liczby dziewięciocyfrowej potrzeba było około 45 minut. Liczba autobiograficzna o dziesięciu cyfrach została znaleziona z użyciem zoptymalizowanego interpretera języka Python o nazwie *pypy*. Trwało to niecałe 2 godziny.

Tak długi czas obliczeń w powyższych programach wynika z użycia w nich „naiwnego” algorytmu. Dla danego n sprawdza on autobiograficzność wszystkich liczb długości n. Liczb takich jest w systemie dziesiątkowym $9 \cdot 10^{n-1}$. Z kolei liczba elementarnych operacji do sprawdzenia autobiograficzności zapisu konkretnej liczby długości n jest rzędu n^2. Z tych obserwacji wynika, że złożoność „naiwnego” algorytmu jest rzędu $O(10^{n-1} \cdot n^2)$. Mamy zatem do czynienia ze złożonością wykładniczą.

Rozwiązanie zadania F 1013.
Siła oporu jest skierowana przeciwnie do prędkości ciała względem powietrza. Podczas ruchu bez wiatru rowerzysta, jadąc z prędkością $\vec{v}$, musi więc stale działać siłą (przeciwną do siły oporu powietrza):

$$\vec{F}_1 = Av^2\frac{\vec{v}}{v},$$

gdzie A jest stałą zależną od kształtu i rozmiarów rowerzysty i roweru. Moc potrzebna do utrzymania stałej prędkości jest wówczas równa $P_1 = \vec{F}_1 \cdot \vec{v} = Av^3$. Gdy pojawia się poprzeczny wiatr o prędkości $\vec{u}$, utrzymanie stałej prędkości (względem drogi) wymaga działania siłą:

$$\vec{F}_2 = A|\vec{v}-\vec{u}|^2\frac{\vec{v}-\vec{u}}{|\vec{v}-\vec{u}|} =$$

$$= A(v^2+u^2)\frac{\vec{v}-\vec{u}}{|\vec{v}-\vec{u}|}.$$

Dla uproszczenia przyjęliśmy, że stała A jest w obu przypadkach taka sama. Moc potrzebna do utrzymania stałej prędkości względem drogi będzie teraz równa:

$$P_2 = \vec{F}_2 \cdot \vec{v} = A\frac{v^2+u^2}{|\vec{v}-\vec{u}|}v^2.$$

Skorzystaliśmy z faktu, że wektory prędkości $\vec{v}$ i $\vec{u}$ są prostopadłe. Ostatecznie otrzymujemy:

$$\frac{P_2}{P_1} = \sqrt{1+\frac{u^2}{v^2}}.$$

Dla podanych w treści zadania prędkości $\frac{P_2}{P_1} = 1,25$.

Żeby skrócić czas obliczeń, użyjemy lepszego filtrowania, to znaczy ograniczymy zbiór liczb, które w ogóle warto sprawdzać. Kolejny filtr opiera się na następującym twierdzeniu:

(2) *Jeśli $a = a_0a_1\ldots a_n$ jest zapisem dziesiętnym liczby autobiograficznej a, to dokładnie jeden element ciągu $a_1,\ldots,a_n$ jest równy 2, a pozostałe są zerami bądź jedynkami.*

Dowód twierdzenia jest prosty: niech r oznacza liczbę niezerowych elementów ciągu $a_1,\ldots,a_n$. Oznaczmy te niezerowe elementy symbolami $a_{i_1},\ldots,a_{i_r}$. Ponieważ $a_0 > 0$ (a_0 jest pierwszą cyfrą dodatniej liczby całkowitej), więc z własności (1) wynika, że

$$a_{i_1}+\cdots+a_{i_r} = n+1-a_0 = n+1-(n-r) = r+1.$$

Zatem mamy r dodatnich liczb całkowitych, których suma wynosi $r+1$, z czego oczywiście wynika teza twierdzenia.

Wersje programów używające lepszych filtrów

Ponieważ szukanie liczb autobiograficznych dziewięcio- i dziesięciocyfrowych okazało się bardzo czasochłonne, zajmijmy się uważniej tymi najdłuższymi liczbami. W obu przypadkach łatwo zauważyć, że pierwsza cyfra (liczba zer w jej zapisie dziesiętnym) liczby autobiograficznej nie może być równa 7, 8 ani 9. Skorzystamy z własności (2) i stworzymy dużo mniejszy zbiór do przeszukania:

- dla liczb dziewięciocyfrowych pierwsze cyfry mogą być równe 1, 2, 3, 4, 5 lub 6, a na ośmiu pozostałych miejscach mogą pojawić się jedynie cyfry 0, 1 lub 2; łącznie daje to $6\cdot 3^8 = 39\,366$ przypadków;
- dla liczb dziesięciocyfrowych pierwsze cyfry mogą być równe 1, 2, 3, 4, 5 lub 6, a na dziewięciu pozostałych miejscach mogą pojawić się jedynie cyfry 0, 1 lub 2; łącznie daje to $6\cdot 3^9 = 118\,098$ przypadków.

Oto kod programu do przeszukiwania napisany w języku MATHEMATICA:

```
czyauto[lista_]
:=If[RotateRight[DigitCount[FromDigits[PadRight[lista],10]]]
==PadRight[lista,10],Return[True],Return[False]]
```

Programik ten sprawdza, czy liczba r zapisana w postaci listy jej cyfr uzupełnionych ewentualnie zerami jest autobiograficzna. Wyposażeni w powyższe narzędzia możemy napisać kod do poszukiwania liczb autobiograficznych dziewięcio- i dziesięciocyfrowych.

```
auto9[j_]:=(L={};Do[If[czyauto[Prepend[Tuples[{0,1,2},8][[i]],j]]
==True,AppendTo[L,Prepend[Tuples[{0,1,2},8][[i]],j]],i++],
                                                {i,1,3^8}];Return[L])
```

Wyjaśnienia wymaga tutaj komenda `Tuples[0,1,2,8][[i]]`, która tworzy z cyfr 0, 1 oraz 2 wszystkie ciągi ośmiocyfrowe, końcówka `[[i]]` oznacza wybranie z otrzymanej, złożonej z 3^8 elementów listy, jej i-tego elementu. Komenda `Prepend[lista,k]` dopisuje z przodu do listy element k.

Program `auto9` znalazł dziewięciocyfrową liczbę autobiograficzną po około 8 sekundach. Dla liczb dziesięciocyfrowych czas poszukiwań też był bardzo dobry; po upływie około 60 sekund pojawiła się ostatnia poszukiwana liczba.

Oto kluczowa funkcja `liczby_kandydatki` z programu w języku Python, w nowej wersji, która używa lepszych filtrów (pełny kod można znaleźć na `www.deltami.edu.pl`).

```
def liczby_kandydatki(dlugosc: int) -> List[int]:
    zbior_cyfr_poczatkowych = [str(c)
                               for c in range(1, dlugosc - 1) if c < 7]
    zbior_cyfr_dalszych = ['0', '1', '2']
    war = wariacje(zbior_cyfr_dalszych, dlugosc - 1)
    return [int(c + ''.join(w))
            for c in zbior_cyfr_poczatkowych for w in war]
```

```
def wariacje(zb: List[str], k: int) -> List[List[str]]:
    if k == 0:
        return []
    elif k == 1:
        return [[e] for e in zb]
    else:
        war = wariacje(zb, k - 1)
        return [[e] + w for e in zb for w in war]
```

Ta wersja funkcji nie bada już wszystkich liczb zadanej długości, lecz jedynie te, które istotnie warto sprawdzać na podstawie obserwacji matematycznych. Warto zwrócić uwagę na użycie rekursji w sposobie zaprogramowania funkcji `wariacje`. Program z nową wersją funkcji `liczby_kandydatki` znajduje wszystkie liczby autobiograficzne w ciągu zaledwie paru sekund. Oto, jak wspaniałe wyniki przynosi współdziałanie matematyki i informatyki!

Uwagi końcowe

W podobny sposób można rozwiązać problem liczb autobiograficznych w innych niż dziesiątkowy systemach liczenia. Herb R. Bailey i Roger G. Lautzenheiser w artykule *A Curious Sequence* (Mathematics Magazine, vol. 66, nr 1, 1993) podają twierdzenie:

Dla każdej liczby naturalnej $n \geqslant 6$ liczba $a_0a_1 \ldots a_n$ jest autobiograficzna wtedy i tylko wtedy, gdy $a_0 = n-3$, $a_1 = 2$, $a_2 = 1$, $a_{n-3} = 1$ oraz $a_i = 0$ dla pozostałych i.

Twierdzenie to w pełni charakteryzuje „długie” liczby autobiograficzne.

Przygotował Łukasz BOŻYK

Zadania

M 1657. Przy dwóch n-osobowych stołach usiadło $2n$ osób. Co minutę pewne dwie osoby, które nie siedzą przy tym samym stole, zamieniają się miejscami. Przypuśćmy, że *każda para* osób zamieniła się miejscami *dokładnie raz* (czyli upłynęło $n(2n-1)$ minut).

(a) Udowodnić, że składy osób przy stołach są takie same jak na początku, tzn. jeśli na początku pewne dwie osoby siedziały przy tym samym stole, to po upływie $n(2n-1)$ minut również siedzą przy tym samym stole.
(b) Wyznaczyć wszystkie liczby całkowite $n \geqslant 1$, dla których taka sytuacja jest możliwa.

Rozwiązanie na str. 21

M 1658. W turnieju wzięło udział n osób ($n \geqslant 4$). *Każda para* osób rozegrała *dokładnie jeden* mecz, który zakończył się zwycięstwem jednej z nich. Po turnieju wszyscy usiedli przy okrągłym stole w taki sposób, że każdy wygrał z osobą siedzącą bezpośrednio po swojej prawej stronie. Wykazać, że można tak pewną osobę wyprosić, a pozostałe przesadzić, aby ta własność pozostała zachowana, tzn. każdy miał na prawo osobę, z którą wygrał.

Rozwiązanie na str. 5

M 1659. Dany jest basen w kształcie pierścienia kołowego, podzielony na $2n$ małych zbiorników poprzedzielanych zawieszonymi nad wodą obręczami ($n \geqslant 2$). W każdym małym zbiorniku (znajdującym się pomiędzy dwiema obręczami) pływa jeden delfin. Co jakiś czas dwa delfiny z sąsiednich zbiorników wykonują akrobację polegającą na jednoczesnym skoku przez obręcz (znajdującą się między zbiornikami) i w rezultacie – zamianie miejscami. Przypuśćmy, że po pewnym czasie *każde dwa* delfiny zamieniły się miejscami *dokładnie raz*. Wykazać, że pewna obręcz nie została użyta do wykonania żadnej akrobacji.

Rozwiązanie na str. 21