



Wybrane elementy informatyki - wykład 4

Przegląd algorytmów istotnych z punktu widzenia informatyki

dr Marcin Ziótkowski

Instytut Matematyki i Informatyki
Akademia im. Jana Długosza w Częstochowie

23 marca 2017 r.

PROBLEM SORTOWANIA

Problem sortowania (porządkowania) danych jest jednym z najważniejszych i najbardziej praktycznych problemów w informatyce. W dzisiejszych czasach, gdy na serwerach są przechowywane bardzo duże ilości różnego rodzaju danych, ich porządkowanie jest bardzo ważnym zadaniem. Z sortowania korzystamy na każdym niemal kroku - przeglądarki internetowe segregują dane w porządku leksykograficznym, aby łatwiej było znaleźć interesującą użytkownika frazę. W uporządkowanym ciągu łatwiej bowiem znaleźć liczbę, literę lub słowo. Niektóre inne algorytmy (np. wyszukiwanie binarne, znajdowanie mediany) wymagają, aby dane wejściowe były już uporządkowane. W ogólnym przypadku sortowanie ciągu liczb polega na znalezieniu takiej permutacji tego ciągu, aby liczby były ułożone w kolejności niemalejącej lub nierosnącej.

SORTOWANIE TRZECH ORAZ CZTERECH LICZB

Zacznijmy od sortowania trójki liczb: a, b, c . Najłatwiejszym sposobem jest znalezienie na początku liczby, która jest nie większa od pozostałych liczb (minimum), a następnie porównanie dwóch pozostałych liczb. Jest to szczególny przypadek **sortowania przez wybieranie**. Innym pomysłem jest porównywanie liczb parami: a porównujemy z b i w razie potrzeby zamieniamy je miejscami. Następnie drugą liczbę w otrzymanym ciągu porównujemy z trzecią i znów w razie potrzeby zamieniamy je miejscami. Wreszcie wracamy do pierwszej pary liczb i w razie potrzeby zamieniamy jej miejscami. Ten sposób jest bardziej efektywny, wykorzystujemy tu tylko 3 porównania. Jest to szczególny przypadek **sortowania bąbelkowego**.

Rozważmy teraz czwórkę liczb: a, b, c, d . Sposób na uporządkowanie tych liczb może być następujący: Porównujemy a z b oraz c z d . Następnie porównujemy mniejsze liczby w tych dwóch parach, otrzymując $\min\{a, b, c, d\}$ oraz większe liczby w tych dwóch parach, otrzymując $\max\{a, b, c, d\}$. W dalszej kolejności porównujemy dwie ostatnie liczby (tą, która w pierwszym porównaniu nie okazała się minimalną oraz tą, która w drugim porównaniu nie okazała się maksymalną).

SORTOWANIE WIĘKSZEJ ILOŚCI LICZB

Już dla pięciu liczb algorytm sortowania jest znacznie bardziej skomplikowany. Jest jasne, że dla dużej ilości danych nie można w taki sposób ich porządkować. Liczba porównań i podstawień byłaby bardzo duża. Poza tym dla dużej liczby danych wygodniej używać tablic (list). Istnieje wiele lepszych i gorszych, łatwiejszych i bardziej skomplikowanych metod sortowania. Na tym wykładzie chcemy zaprezentować najprostsze (ale niestety nieefektywne dla dużej liczby danych) algorytmy, które oprócz prostoty łączy podobna złożoność obliczeniowa rzędu $o(n^2)$. Zaprezentujemy:

- 1 SORTOWANIE PRZEZ WYBIERANIE
- 2 SORTOWANIE PRZEZ WSTAWIANIE
- 3 SORTOWANIE BĄBELKOWE

SORTOWANIE PRZEZ WYBIERANIE

Idea sortowania przez wybieranie jest następująca: Z początkowego ciągu liczb wybieramy element najmniejszy i w razie konieczności zamieniamy go miejscami z pierwszym elementem. Następnie ze wszystkich pozostałych liczb, za wyjątkiem pierwszej znowu wybieramy minimalną i zamieniamy ją miejscami z drugim elementem. W dalszej kolejności ze wszystkich pozostałych liczb poza pierwszymi dwoma wybieramy minimalną i zamieniamy ją z trzecim elementem itd. Algorytm kończymy w momencie, gdy dojdziemy do końca ciągu, a w zasadzie, gdy do szukania minimum pozostanie dwuelementowy podciąg.

SORTOWANIE PRZEZ WYBIERANIE

Uporządkujmy następujący ciąg: 2, 8, 6, 4, 9, 1, 7, 3 Oto kolejne kroki:

1, 8, 6, 4, 9, 2, 7, 3

1, 2, 6, 4, 9, 8, 7, 3

1, 2, 3, 4, 9, 8, 7, 6

1, 2, 3, 4, 9, 8, 7, 6

1, 2, 3, 4, 6, 8, 7, 9

1, 2, 3, 4, 6, 7, 8, 9

1, 2, 3, 4, 6, 7, 8, 9

SORTOWANIE PRZEZ WSTAWIANIE

Sortowanie przez wstawianie przypomina układanie kart w trakcie gry. Przesuwamy wówczas karty na odpowiednie miejsce. Dla liczb będzie to wyglądać następująco: Pierwszą liczbę pozostawiamy bez zmian i traktujemy ją jako uporządkowany już podciąg danego ciągu liczb. Następnie przesuwamy (przez zamianę parami) drugi element ciągu na jego odpowiednie miejsce, uzyskując uporządkowaną dwójkę liczb. Dalej znów przesuwamy kolejny element ciągu (trzeci) i uzyskujemy uporządkowaną trójkę itd. W efekcie uzyskujemy uporządkowany ciąg liczbowy.

SORTOWANIE PRZEZ WSTAWIANIE

Uporządkujmy następujący ciąg: 2, 8, 6, 4, 9, 1, 7, 3. Oto kolejne kroki:

2, 8, 6, 4, 9, 1, 7, 3

2, 6, 8, 4, 9, 1, 7, 3

2, 4, 6, 8, 9, 1, 7, 3

2, 4, 6, 8, 9, 1, 7, 3

1, 2, 4, 6, 8, 9, 7, 3

1, 2, 4, 6, 7, 8, 9, 3

1, 2, 3, 4, 6, 7, 8, 9

Sortowanie bąbelkowe polega na zamianie parami tylko sąsiednich elementów ciągu liczbowego. Najlepiej pokazać tę ideę na porządkowaniu monet o różnych nominałach. Idąc od początku ciągu zamieniamy miejscami te monety dla których nominał monety znajdującej się z lewej strony jest większy od nominału monety znajdującej się z prawej strony.

Zapoznamy się teraz z najważniejszymi algorytmami wykorzystywanymi na lekcjach matematyki oraz informatyki oraz przedstawieniem pewnych możliwości ich udoskonalania w celu uzyskania algorytmów szybszych, które mają mniejszą złożoność obliczeniową. Do takich algorytmów należą m.in. wypisywanie dzielników, znajdowanie NWD oraz NWW dwóch lub większej ilości liczb, rozkład danej liczby naturalnej na czynniki pierwsze, sprawdzanie, czy dana liczba naturalna jest pierwsza, wypisywanie listy liczb pierwszych, sprawdzanie czy dana liczba jest doskonała, wypisywanie listy liczb doskonałych, wypisywanie par liczb zaprzyjaźnionych, a także odczytywanie liczb zapisanych w innym systemie liczenia oraz zapisywanie liczb w systemie o danej podstawie.

Już w szkole podstawowej uczniowie wypisują dzielniki podanej liczby. Problem nie wydaje się trudny dla liczb małych, jednak dla większych liczb staje się skomplikowany nawet dla dorosłych ludzi i w celu jego rozwiązania wykorzystujemy często komputer. Na poprzednim wykładzie przedstawiony został prosty algorytm rozwiązujący ten problem. Dla małych liczb program działa bardzo szybko, dla większych jego działanie jest wolne. Dzieje się tak dlatego, że sprawdzamy po kolei wszystkie liczby począwszy od 1 do liczby n . Okazuje się, że jeśli dzielniki będziemy wypisywać parami (np. dla liczby 40: 1, 40, 2, 20, 4, 10, 5, 8), to możemy sprawdzać dzielniki tylko do liczby całkowitej nie większej od \sqrt{n} , co znacznie poprawia jego szybkość, otrzymujemy złożoność rzędu $O(\sqrt{n})$ zamiast $O(n)$. Na przykład dla liczby 1000000 wystarczy sprawdzić 1000, a nie 1000000 dzielników.

Kolejnym problemem jest znajdowanie NWW dwóch lub większej ilości liczb naturalnych. W szkole często wykorzystuje się do tego rozkład liczby naturalnej na czynniki pierwsze. Jednak znajdowanie NWW może być oparte na algorytmie nie wykorzystującym tego rozkładu. Algorytm taki jest szybszy, a jego idea była również przedstawiona na poprzednim wykładzie. Zauważmy, że ta sama idea może mieć zastosowanie przy obliczaniu NWW większej ilości liczb. Jedną z tych liczb (najlepiej największą, ale trzeba tu stosować wówczas algorytmy sortujące) mnożymy przez kolejne dodatnie liczby naturalne do momentu, aż uzyskany iloczyn będzie podzielny przez wszystkie pozostałe liczby. Możemy też napisać program rekurencyjny, ponieważ

$$NWW(a, b, c) = NWW(NWW(a, b), c).$$

Następny problem to znajdowanie NWD dwóch lub większej ilości liczb naturalnych. W szkole również wykorzystuje się do tego rozkład liczby naturalnej na czynniki pierwsze. Znajdowanie NWD może być jednak również dużo łatwiejsze. Możemy na przykład od mniejszej z liczb odejmować liczbę jeden dopóki większa liczba nie podzieli się przez uzyskaną różnicę i jednocześnie różnica ta nie będzie dzielnikiem mniejszej z liczb.

Innym ze znanych sposobów jest na przykład algorytm Euklidesa wykorzystujący jedynie operację odejmowania przedstawiony na poprzednim wykładzie. Ideę tę również możemy rozszerzyć dla większej ilości liczb lub wykorzystać rekurencję, ponieważ dla największego wspólnego dzielnika mamy również zależność:
$$NWD(a, b, c) = NWD(NWD(a, b), c).$$

Problem znajdowania liczb pierwszych jest problemem bardzo ważnym z punktu widzenia matematyki i informatyki. Liczby pierwsze są bowiem wykorzystywane m.in. w kryptografii (szyfrowanie danych) oraz mają wiele innych zastosowań. Test sprawdzania, czy dana liczba jest pierwsza może być oparty na przykład o algorytm omówiony na poprzednim wykładzie. Program ten można przyspieszyć, szukając dzielników tylko do liczby całkowitej nie większej od \sqrt{n} . Daje to złożoność rzędu $O(\sqrt{n})$ zamiast liniowej $O(n)$.

LICZBA DOSKONAŁA - DEFINICJA

Liczbą doskonałą nazywamy liczbę naturalną, która jest równa sumie mniejszych od siebie dzielników.

Liczby doskonałe to wielka zagadka teorii liczb, ponieważ:

- 1 Nie wiadomo, czy liczb doskonałych jest skończenie czy nieskończenie wiele
- 2 Wszystkie znalezione do tej pory liczby doskonałe są parzyste, nie wiadomo czy istnieje nieparzysta liczba doskonała
- 3 Nie ma żadnego efektywnego algorytmu znajdowania liczb doskonałych, istnieje jednak związek pomiędzy liczbami doskonałymi i pewnymi liczbami pierwszymi (dokładniej liczbami pierwszymi Mersene'a, czyli liczbami pierwszymi, które da się przedstawić w postaci $1 + 2 + \dots + 2^n$)

Oto kilka znanych liczb doskonałych:

6, 28, 496, 8128, 33550336, 8589869056, 137438691328.

LICZBY ZAPRZYJAŻNIONE - DEFINICJA

Dwie różne liczby naturalne nazywamy zaprzyjaźnionymi, jeśli jedna z nich jest równa sumie dzielników drugiej liczby mniejszych od niej i na odwrót.

Z liczbami zaprzyjaźnionymi jest podobny problem. Nie wiadomo, czy takich par liczb jest nieskończenie wiele i czy istnieją liczby zaprzyjaźnione, z których jedna byłaby parzysta, a druga nieparzysta.

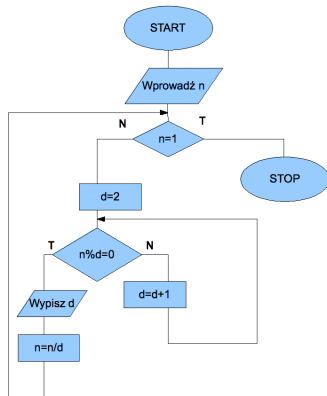
Oto przykłady takich par liczb:

LICZBY ZAPRZYJAŻNIONE

220 i 284 1184 i 1210 2620 i 2924 5020 i 5564 6232 i 6368
10744 i 10856 12285 i 14595

ROZKŁAD LICZBY NATURALNEJ NA CZYNNIKI PIERWSZE

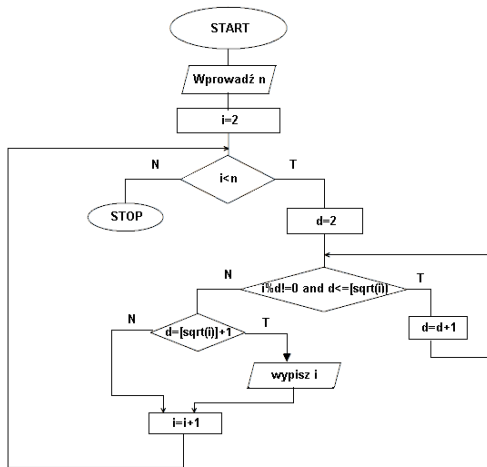
Omówimy algorytm rozkładu liczby na czynniki pierwsze. Przedstawiony poniżej algorytm opiera się o intuicyjne rozumienie rozkładu i rozkład ten jest wykonywany w oparciu o dzielenie przez kolejne liczby naturalne. Na przykład $60 = 2 \cdot 30 = 2 \cdot 2 \cdot 15 = 2 \cdot 2 \cdot 3 \cdot 5$.



Zajmiemy się teraz algorytmami, za pomocą których wypisujemy liczby pierwsze. Na początek oprzemy się na znanym już nam algorytmie sprawdzania, czy liczba jest pierwsza i na jego bazie zbudujemy algorytm, który wypisuje liczby pierwsze mniejsze od wprowadzonej przez użytkownika liczby n . Algorytm wygląda następująco:

WYPISYWANIE LISTY LICZB PIERWSZYCH

WYPISYWANIE LICZB PIERWSZYCH - ALGORYTM

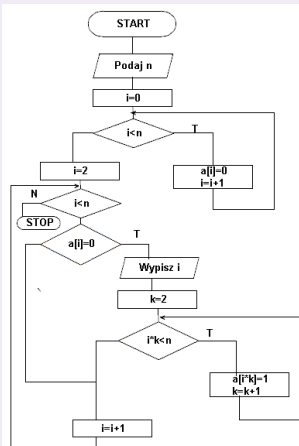


WYPISYWANIE LISTY LICZB PIERWSZYCH - SITO ERATOSTENESA

Sito Eratostenesa jest jednym z najprostszych algorytmów, za pomocą których znajdujemy liczby pierwsze. Polega on na przechodzeniu przez tablicę liczb naturalnych (od dwójki) i wykreślaniu wielokrotności liczb, które w poprzednich krokach nie były wykreślone. Liczby które pozostaną nieskreślone to liczby pierwsze. Do konstrukcji tego algorytmu najlepiej wykorzystać tablice (listy). Algorytm wygląda następująco:

WYPISYWANIE LISTY LICZB PIERWSZYCH - SITO ERATOSTENESA

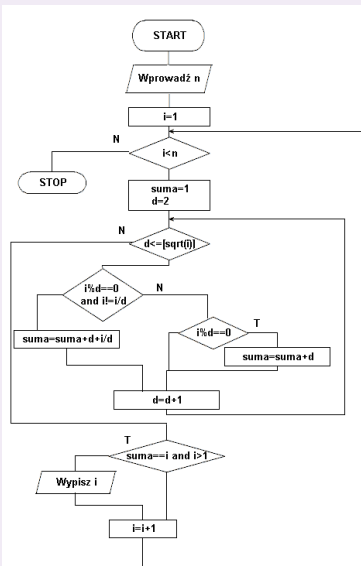
WYPISYWANIE LICZB PIERWSZYCH - ALGORYTM ERATOSTENESA



Omówiony został wcześniej algorytm sprawdzania, czy dana liczba jest doskonała. Na bazie tego algorytmu można skonstruować algorytm wypisywania listy liczb doskonałych mniejszych od wprowadzonej przez użytkownika liczby n . Algorytm wygląda następująco:

WYPISYWANIE LISTY LICZB DOSKONAŁYCH

WYPISYWANIE LICZB DOSKONAŁYCH - ALGORYTM



W matematyce i innych naukach przyrodniczych mamy często do czynienia z obliczaniem wartości wielomianu w danym punkcie. Na przykład, gdy dany jest wielomian $W(x) = 4x^4 + 3x^3 + 2x^2 + 5x + 6$ i chcemy obliczyć jego wartość w punkcie $x = 2$ obliczamy:

$$W(2) = 4 \cdot 2^4 + 3 \cdot 2^3 + 2 \cdot 2^2 + 5 \cdot 2 + 6 =$$
$$4 \cdot 16 + 3 \cdot 8 + 2 \cdot 4 + 10 + 6 = 64 + 24 + 8 + 16 = 112.$$

Jednak taki sposób obliczania wartości wielomianu nie jest najbardziej szybki i efektywny, gdyż musimy wykonać wiele mnożeń (w naszym przykładzie aż 10!, ponieważ obliczanie potęg również jest obliczaniem odpowiednich iloczynów). Dla wielomianów o dużych stopniach prowadzi to do długich i żmudnych obliczeń zawierających setki mnożeń, co nawet w przypadku obliczeń komputerowych może prowadzić do długiego działania programów, wykorzystujących obliczanie wartości wielomianów. Istnieje sposób bardziej efektywny, wykorzystujący algorytm Hornera, który omówimy teraz bardziej szczegółowo.

Przedstawmy nasz wielomian w trochę inny sposób (zaczynając od najniższych potęg):

$$\begin{aligned}W(x) &= 6 + 5x + 2x^2 + 3x^3 + 4x^4 = 6 + x(5 + 2x + 3x^2 + 4x^3) = \\ &= 6 + x(5 + x(2 + 3x + 4x^2)) = 6 + x(5 + x(2 + x(3 + 4x)))\end{aligned}$$

Taki sposób przedstawienia wielomianu powoduje, że mamy do czynienia tylko z 4 mnożeniami (dodawanie jest tyle samo). Każdy wielomian można doprowadzić do takiej postaci, a sposób obliczania wartości wielomianu w oparciu o taką jego postać nazywa się algorytmem Hornera.

ALGORYTM HORNERA

Pokażemy teraz jak łatwo w oparciu o ten algorytm obliczyć wartość wielomianu $W(x) = 4x^4 + 3x^3 + 2x^2 + 5x + 6$ np. dla $x = 2$. Obliczenia przedstawimy w tabeli:

krok	1	2	3	4
wartość	$4 \cdot 2 + 3 = 11$	$11 \cdot 2 + 2 = 24$	$24 \cdot 2 + 5 = 53$	$53 \cdot 2 + 6 = 112$

Mało tego, za pomocą tego algorytmu możemy wykonywać szybkie dzielenie wielomianu przez dwumian $x - a$. Tutaj mamy:

$$(4x^4 + 3x^3 + 2x^2 + 5x + 6) : (x - 2) = 4x^3 + 11x^2 + 24x + 53 \text{ R } 112$$

Inaczej można zapisać to tak:

$$4x^4 + 3x^3 + 2x^2 + 5x + 6 = (4x^3 + 11x^2 + 24x + 53)(x - 2) + 112$$

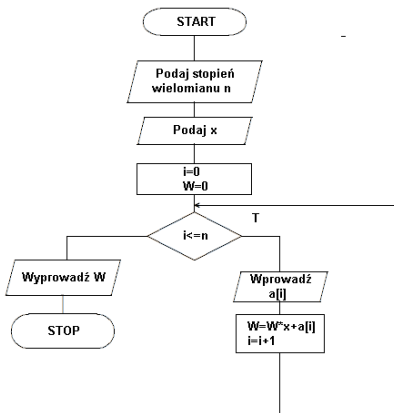
Albo bardziej formalnie:

$$\frac{4x^4 + 3x^3 + 2x^2 + 5x + 6}{x - 2} = 4x^3 + 11x^2 + 24x + 53 + \frac{112}{x - 2}$$

ALGORYTM HORNERA

Przejdziemy teraz do samego algorytmu Hornera i jego ilustracji z użyciem schematu blokowego. Użyjemy algorytmu iteracyjnego wykorzystującego tablice.

ALGORYTM HORNERA



ZASTOSOWANIE ALGORYTMU HORNERA - ZAMIANA LICZBY ZAPISANEJ W INNYM SYSTEMIE LICZENIA NA SYSTEM DZIESIĘTNY

Algorytm Hornera można wykorzystać do odczytywania liczby zapisanej w innym systemie liczenia. Rozpatrzmy przykład:

$$1234_{(5)} = 1 \cdot 5^3 + 2 \cdot 5^2 + 3 \cdot 5 + 4 = 125 + 50 + 15 + 4 = 194$$

Zauważmy, że jest to również wartość wielomianu $1x^3 + 2x^2 + 3x + 4$ w punkcie $x = 5$.