Algorytmy arytmetyczne, liczby pierwsze i szybka transformacja Fouriera

wszelkie prawa zastrzeżone
zakaz kopiowania, publikowania i przechowywania
all rights reserved
no copying, publishing or storing

Maciej Hojda

1 Zadanie nr 1 – liczby pierwsze

1.1 Rozkład na czynniki pierwsze

Zaimplementuj funkcję `czp(p)` która zwróci listę czynników pierwszych zadanej liczby naturalnej n. Zrób to rekurencyjnie sprawdzając podzielność liczby przez kolejne liczby naturalne (aż do $\sqrt{n}$) – rekurencja pojawia się, gdy liczba jest podzielna – wtedy uruchamiamy algorytm na jej dzielnikach, o ile nie są pierwsze.

1.2 Generacja liczb pierwszych

Zaimplementuj sito Eratostenesa aby wyznaczyć zbiór liczb pierwszych nie większych od zadanego p.

Algorithm 1 Sito Eratostenesa – `sera(p)`

```
- wejscie: liczba naturalna p > 1.
1 niech x ≜ [x_n]_{n∈{2,3,...,p}} := [1]_{n∈{2,3,...,p}}
2 dla n := 2 do ⌊√p⌋
3   jeśli x_n = 1, to
4     dla j := 2 do ⌊√p/n⌋
5       niech x_{n×j} := 0
6 zwróć x
- wyjście: tablica x dla której, jeśli x_j = 1, to liczba j jest piewsza.
```

2 Zadanie nr 2 – największy wspólny dzielnik

2.1 Wyszukiwanie

Zaimplementuj funkcję szukającą największego wspólnego dzielnika dwóch liczb. Zrób to na dwa sposoby.

- Z wykorzystaniem rozkładu na czynniki pierwsze `aczp(x, y)`.
- Z wykorzystaniem algorytmu Euklidesa `aeuc(x, y)`.

Algorithm 2 Algorytm Euklidesa – `aeuc(x, y)`

```
- wejscie: dwie liczby naturalne x, y.
1 jeśli y = 0, to zwróć x
2 w przeciwnym wypadku zwróć aeuc(y, x mod y)
- wyjście: największy wspólny dzielnik liczb x, y.
```

2.2 Testy wydajności

Przygotuj procedurę testową do sprawdzenia czasu działania obu algorytmów. Niech $x = \prod_{i \in \{1,2,\ldots,6\}} I_1(i) \times I_2(i)$, gdzie I_1 i I_2 to wektory kolejnych cyfr indeksu dwóch członków grupy. Dla x i dla dla kolejnych liczb naturalnych y uruchamiaj oba algorytmy. Zapamiętaj czas działania każdego algorytmu.

Procedurę uruchom ją dla K kolejnych liczb naturalnych, gdzie K jest największe takie, dla którego czas działania całej procedury testowej nie przekracza 5ciu minut. Uzyskane rezultaty wyświetl na wykresach zależności czasu działania (obu) funkcji od testowanej liczby y.

3 Zadanie nr 3 – probabilistyczne testy piewszości

Zaimplementuj dwa algorytmy probabilistycznego testowania pierwszości

- test Fermata,
- test Millera-Rabina.

Test Fermata dla liczby p polega na wielokrotnym losowaniu liczby q takiej, że $q \in [2, 3, \ldots, p)$, oraz q, p są względnie pierwsze. Następnie sprawdzamy, czy $q^{p-1} \mod p = 1$. Jeśli nie, to liczba p nie jest pierwsza. Jeśli natomiast test wyjdzie pozytywnie dla wielu q, to liczba p prawdopodobnie jest pierwsza (im więcej testów, tym większe prawdopodobieństwo).

Test Millera-Rabina dla nieparzystej liczby $p > 1$ zaczyna się od przedstawienia tej liczby w postaci $p = 2^r \times q + 1$ (czyli od znalezienia r, a w konsekwencji q). Następnie wykonujemy:

1 losuj $a \in [2, p-2]$
2 niech $x = a^q mod n$
3 jeśli $x \in \{1, p-1\}$ to wracamy do pkt. 1
4 Powtarzaj $r-1$ razy
5 $x := x^2 \mod p$
6 Jeśli $x = p-1$ to zwróć: prawdopodobnie piewsza
7 zwróć: złożona

Przedstawioną procedurę potwarzamy wielokrotnie. Każde powtórzenie, które nie stwierdza, że liczba p jest złożona zwiększa prawdopodobieństwo, że jest ona piewsza.

4 Zadanie nr 4 – szybka transformacja Fouriera

Zaimplementuj poznany algorytm wyznaczania szybkiej transformacji Fouriera. Zademonstruj działanie algorytmu dla zadanego sygnału (ciągu liczbowego) wyświetlając wykres częstotliwości.

Przetestuj procedurę dla sumy kilku sygnałów sinusoidalnych (po dyskretyzacji) o różnych częstotliwościach, np. dla $y(t) = 5sin(t) + 3sin(2t) + 5sin(5t)$.

5 Zadanie nr 5 – filtracja

Wykorzystaj szybką transformatę Fouriera do filtracji zadanego sygnału – tzn. usunięcia wybranych częstotliwości. Wykonaj następujące czynności:

- wczytaj próbki sygnału,
- zastosuj okno czasowe Hanninga (pomnóż transformowane próbki przez $\frac{1}{2}(1 - cos(\frac{2\pi n}{N-1}))$, gdzie N to liczba próbek, a n to numer próbki),
- wykonaj szybką transformację Fouriera,
- usuń wybrane częstotliwości,
- wykonaj odwrotną szybką transformację Fouriera w celu odtworzenia sygnału w dziedzinie czasu.

6 Zadanie nr 6 (opcjonalne) – filtracja sygnałów dźwiękowych

Zastosuj szybką transformację Fouriera do filtracji (odszumiania) plików dźwiękowych. **Uwaga:** podziel sygnał na (fragmentami zachodzące na siebie) części.