Lab1

Bohdan Tkachenko 256630

October 27, 2023

1 Zadanie9

1.1 Wstep

Celem tego sprawozdania jest analiza algorytmu Fisher-Yates używanego do generowania losowych permutacji. Zbadane zostana trzy aspekty permutacji: średnia liczba permutacji bez stałych punktów, średnia liczba permutacji z jednym punktem stałym oraz średnia liczba cykli w permutacjach.

1.2 Metodologia

Algorytm Fisher-Yates został zaimplementowany w jezyku Python. Do analizy permutacji użyto trzech różnych metryk. Eksperymenty przeprowadzono na tablicach o różnych rozmiarach: 5, 10, 20 i 50. Dla każdego rozmiaru tablicy wykonano 1000 prób.

1.3 Wyniki

- Średnia liczba permutacji bez stałych punktów:
  - Rozmiar 5: 38.8%
  - Rozmiar 10: 34.6%
  - Rozmiar 20: 36.1%
  - Rozmiar 50: 36.3%
- Średnia liczba permutacji z jednym punktem stałym:

- Rozmiar 5: 36.8%
- Rozmiar 10: 36.8%
- Rozmiar 20: 37.1%
- Rozmiar 50: 37.6%

- Średnia liczba cykli:
  - Rozmiar 5: 2.252
  - Rozmiar 10: 2.955
  - Rozmiar 20: 3.659
  - Rozmiar 50: 4.547

1.4 Wnioski

- Obserwujemy podobny odsetek permutacji bez stałych punktów dla różnych rozmiarów tablic, oscylujacy wokół 35-38%.
- Średnia liczba permutacji z jednym punktem stałym również jest dość stabilna i oscyluje wokół 36-37%.
- Średnia liczba cykli rośnie w miare zwiekszania rozmiaru tablicy, co jest zgodne z intuicja.

2 Zadanie 32

2.1 Opis Problemu

Celem tego projektu jest obliczenie formalnej odwrotności szeregów formalnych dla różnych funkcji $f(n)$. Formalna odwrotność szeregu formalnego jest używana w różnych dziedzinach matematyki i inżynierii.

2.2 Definicje i Implementacja

Formalna odwrotność szeregu formalnego $\sum_{n=0}^{\infty} f(n)x^n$ jest szeregiem formalnym $g(x)$, który spełnia warunek $f(x) \cdot g(x) = 1$ w sensie mnożenia szeregów formalnych.

Dla $f(x) = \sum_{n=0}^{\infty} a_n x^n$ i $g(x) = \sum_{n=0}^{\infty} b_n x^n$, gdzie $a_0 \neq 0$, współczynniki b_n sa obliczane według wzoru:

$$b_n = -\frac{1}{a_0} \left(a_1 b_{n-1} + a_2 b_{n-2} + \ldots + a_n b_0\right)$$

z $b_0 = \frac{1}{a_0}$.

Algorytm został zaimplementowany w jezykach Haskell i Java. Implementacja korzysta z powyższej teorii do obliczania współczynników szeregów formalnych.

2.3 Wyniki

Wyniki dla różnych funkcji $f(n)$ sa nastepujace:

- Dla $f(n) = 1$, wyniki sa $[1.0, -1.0, -0.0, \ldots]$
- Dla $f(n) = 2^n$, wyniki sa $[1.0, -2.0, -0.0, \ldots]$
- Dla $f(n) = n!$, wyniki sa $[1.0, -1.0, -1.0, -3.0, \ldots]$
- Dla $f(n) = 1/n!$, wyniki sa $[1.0, -1.0, 0.5, -0.1667, \ldots]$

2.4 Wnioski

Algorytm skutecznie oblicza formalna odwrotność dla różnych funkcji $f(n)$. Wyniki sa zgodne z teoria i potwierdzaja poprawność implementacji. Formalne odwrotności moga być użyteczne w różnych aplikacjach i dalszych badaniach.

3 Zadanie 10

3.1 Wprowadzenie

Celem tego sprawozdania jest analiza kombinatoryczna ciagów binarnych długości n zawierajacych wzory 'aaa' i 'abb'. Zastosowano algorytm programowania dynamicznego oparty na automatach skończonych do rozwiazania problemu.

3.2 Teoria

3.3 Automaty Skończone

Automat dla wzoru 'aaa' ma cztery stany:

1. Stan 0: Brak wystapienia wzoru.
2. Stan 1: Jedno 'a' wystapiło.
3. Stan 2: Dwa 'a' wystapiły.
4. Stan 3: Wzór 'aaa' został znaleziony.

Automat przechodzi miedzy stanami w zależności od kolejnej litery w ciagu. Na przykład, jeśli jesteśmy w stanie 2 i kolejna litera to 'a', przechodzimy do stanu 3.

3.4 Programowanie Dynamiczne

Używamy tablicy dwuwymiarowej dp o wymiarach $(n+1) \times 4$, gdzie $dp[i][j]$ przechowuje liczbe ciagów długości i kończacych sie na stanie j.

3.5 Algorytm

Dla każdego ciagu długości i i stanu j, obliczamy nastepny stan po dodaniu 'a' lub 'b'. To pozwala na zliczenie wszystkich ciagów zawierajacych dana sekwencje.

4 Implementacja

Kod jest napisany w Pythonie i używa tablic dwuwymiarowych dla przechowywania stanów DP.

4.1 Wyniki

4.2 Liczba Ciagów

n	'aaa'	'abb'	Średnia 'aaa'
5	8	12	0.125
10	520	792	0.414
20	825259	1019920	0.746
30	974791728	1070217247	0.890

4.3 Średnia liczba wystapień 'aaa'

Dla $n = 30$, średnia liczba wystapień 'aaa' wynosi około 0.890. Dla $n = 50$, średnia liczba wystapień 'aaa' daży do 1 i wynosi 0.979.

4.4 Złożoność Czasowa

Złożoność czasowa algorytmu wynosi $O(n)$.

4.5 Wnioski z Wyników

Z obliczeń wynika, że liczba ciagów zawierajacych wzory 'aaa' i 'abb' rośnie wykładniczo w stosunku do n. Na przykład, dla $n = 5$ liczba ciagów zawierajacych 'aaa' wynosi 8, a dla $n = 30$ ta liczba wynosi 974791728. To sugeruje, że nawet niewielki wzrost n może znaczaco zwiekszyć liczbe ciagów zawierajacych te wzory.

Dodatkowo, średnia liczba wystapień wzoru 'aaa' w ciagu również rośnie. Dla $n = 5$, średnia liczba wystapień wynosi 0.125, a dla $n = 30$ wynosi około 0.890. To wskazuje, że im dłuższy ciag, tym wiecej razy można spodziewać sie wystapienia wzoru 'aaa'.