Zadanie Programistyczne 2:
Cykle

Maksymilian Neumann

June 20, 2022

1 Implementacja Fisher-Yates shuffles

Funkcja przyjmuje listę w postaci [1,2,. . . ,n](element neutralny) i zwraca jej losową permutacje.

```
import random

def shuffle(given):
    permutation = []
    for i in range(len(given)):
        temp = random.randint(0, (len(given)-1))
        permutation.insert(0, given[temp])
        given.pop(temp)
    return permutation
```

Listing 1: Implementacja Fisher-Yates shuffles

2 Rozkład Permutacji na Cykle

Funkcja przyjmuje permutacje w postaci listy i zwraca rozkład jej na cykle w postaci listy list.

```
def cycler(permutation):
    cycles = []
    for i in range(1, len(permutation) + 1):
        if permutation[i - 1] != 0:
            temp = i
            cycle = []
            cycle.append(i)
            while permutation[temp - 1] != i:
                cycle.append(permutation[temp - 1])
                temp = permutation[temp - 1]
            cycles.append(cycle)
            for j in cycle:
                permutation[permutation.index(j)] = 0
    return cycles
```

Listing 2: Rozkład Permutacji na cykle

3 Test Działania Kodu

Z wykładu dowiedzieliśmy się ,że dla $\pi \in S_n$,gdzie $L(\pi)$ = liczba cykli permutacji π

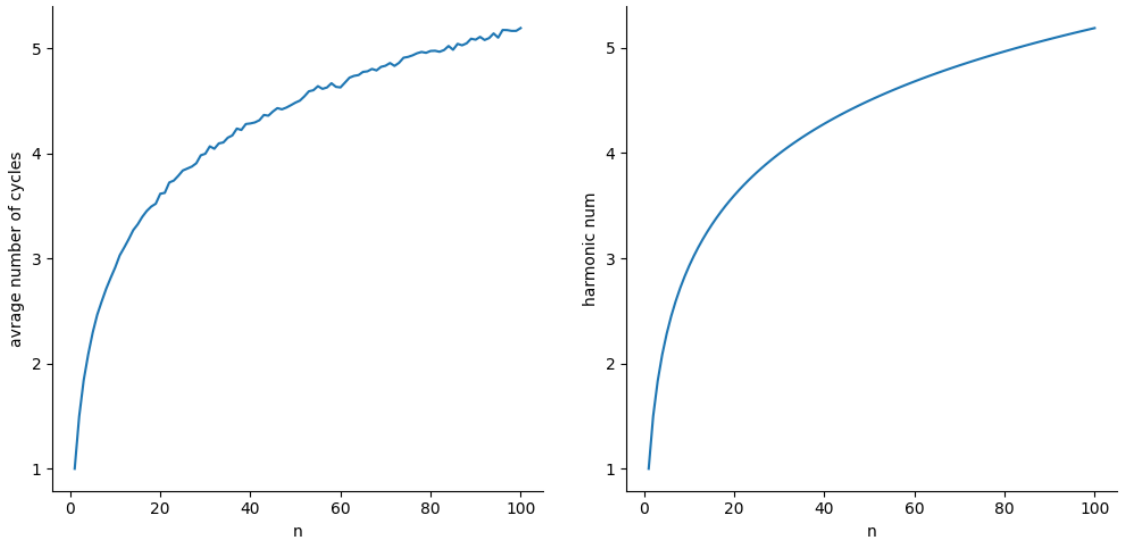
$$E(L) = H_n$$

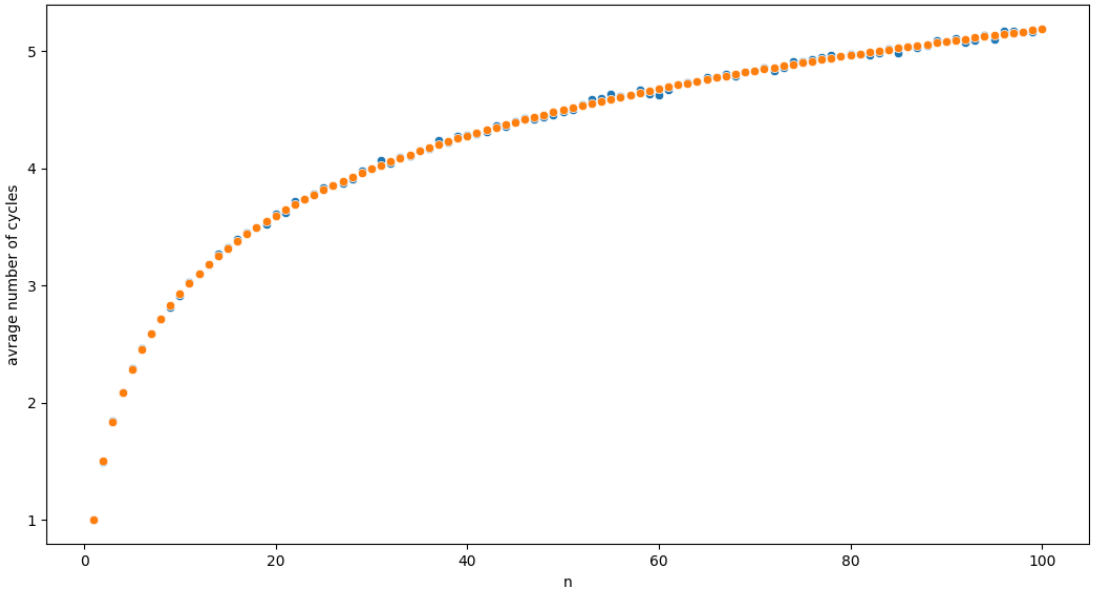
która będzie naszą oczekiwaną wartością

```
from statistics import mean

def test():
    f = open("test.csv", 'w')
    f.write("n;avrage number of cycles cycles\n")
    for n in range(1, 101):
        lengths = []
        for i in range(8000):
            x = list(range(1, n + 1))
            lengths.append(len(cycler(shuffle(x))))
        row = str(n) + ";" + str(mean(lengths)) + "\n"
        f.write(row)
    f.close()
```

Listing 3: Test zapisyjący wyniki do pliku





Wyniki eksperymentu sugerują poprawność kodu.

4 Główne Zadanie

Hipoteza:

Można zauważyć, że dla $\pi \in S_n$,gdzie $\pi = C_1 \circ \ldots \circ C_k$

1. $M_n = \max(\text{len}(C_1), \ldots, \text{len}(C_k)) = \lceil \frac{n}{k} \rceil$

2.Wiemy również że takich maksymalnych długości jest $\begin{bmatrix} n \\ k \end{bmatrix}$

Z 1. i 2. wynika:

$$E(M_n) = \frac{\sum_{k=1}^{n} \lceil \frac{n}{k} \rceil \begin{bmatrix} n \\ k \end{bmatrix}}{n!}$$

Eksperyment:

```
from statistics import mean

def zadanie():
    f = open("zadanie.csv", 'w')
```

```
5     f.write("n;avrage max length of a cycle in a permutation\
      n")
6     for n in range(1, 101):
7         maxLengths = []
8         for i in range(8000):
9             x = list(range(1, n + 1))
10            maxLengths.append(max(len(c) for c in cycler(
      shuffle(x))))
11        row = str(n) + ";" + str(mean(maxLengths)) + "\n"
12        f.write(row)
13    f.close()
```

Listing 4: Eksperyment numeryczny dla głównego zadania