**随机算法第一次实验**

2020283250002 计科一班 王雨航

1、实验目的：

本次实验为设计型实验

通过编程实现，掌握优惠券收集问题

通过仿真实验，加深优惠券收集问题的性能分析结果

2、实验内容：

编写优惠券收集问题仿真代码（编程语言不限，推荐Python）

输入：优惠券类型个数𝑛

输出：实验次数（假设每一次实验收集一张优惠券）

进行仿真实验，观察并记录输入规模与运行时间的关系，并与理论结果进行对比分析（40分）

进行仿真实验：

输入规模为𝑛=30时，计算实验次数在[𝑛𝑙𝑛𝑛−𝑐𝑛, 𝑛𝑙𝑛𝑛+𝑐𝑛]之间的概率𝑃（𝑐值可以任意取）（30 分）

改变输入规模𝑛，观察并记录输入规模𝑛与𝑃的关系，并与理论结果进行对比分析（30 分）

3、实验内容

1. 导入python所需库

import cmath

import math

import numpy as np

import matplotlib.pyplot as plt

1. 编写模拟收集优惠卷的过程

def work(n:int):

dt = set()

time = 0

while check(dt, n) != 1:

time += 1

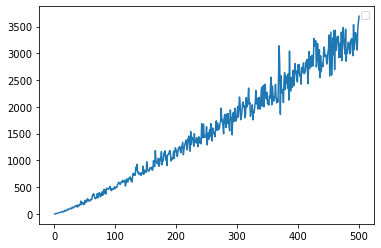
number = np.random.randint(0,n)

dt.add(number)

return time

1. 进行仿真实验，观察并记录输入规模与运行时间的关系：

对数据1~500分别进行模拟50次并取中位数来代替结果：



xs = []

ys = []

def check(dt,n):

return len(dt) == n

def work(n:int):

dt = set()

time = 0

while check(dt, n) != 1:

time += 1

number = np.random.randint(0,n)

dt.add(number)

return time

n = int(input())

for i in range(1,n+1):

tmp = []

for j in range(50):

new\_time=work(i)

tmp.append(new\_time)

tmp.sort()

xs.append(i)

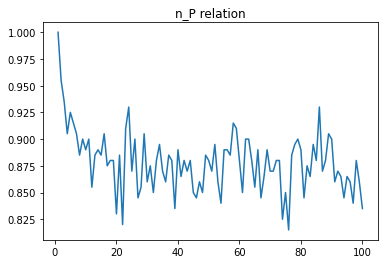
ys.append(tmp[len(tmp)//2])

plt.plot(xs, ys)

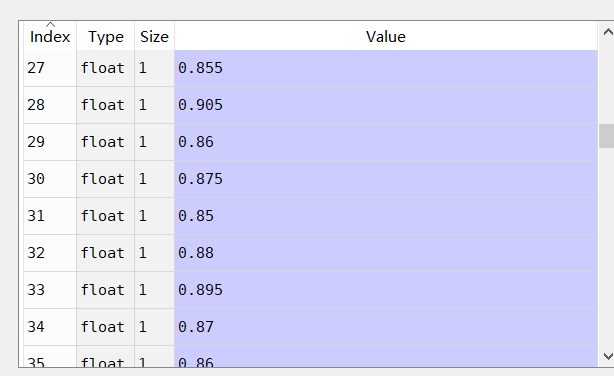
plt.title("n\_time relation")

plt.show()

1. 进行仿真实验改变输入规模，观察并记录输入规模与的关系，并与理论结果进行对比分析。遍历n=1~100的所有情况，每种情况模拟200次统计出概率与输入规模的图像：



同时可以发现n=30的时候的概率为：0.86



# 绘制概率图

xs = []

ys = []

for i in range(1,n+1,1):

tot = 0

son = 0

c = 2

n = i

for j in range(200):

tot += 1

E = int(n\*float(math.log(n)))

res = work(n)

if res >= E - c\*n and res <= E + c\*n:

son += 1

xs.append(i)

ys.append(son/tot)

plt.plot(xs, ys)

plt.title("n\_P relation")

plt.show()

完整代码：

# -\*- coding: utf-8 -\*-

"""

Created on Mon Nov 21 10:55:48 2022

@author: Henry

"""

import cmath

import math

import numpy as np

import matplotlib.pyplot as plt

xs = []

ys = []

def check(dt,n):

return len(dt) == n

def work(n:int):

dt = set()

time = 0

while check(dt, n) != 1:

time += 1

number = np.random.randint(0,n)

dt.add(number)

return time

n = int(input())

for i in range(1,n+1):

tmp = []

for j in range(50):

new\_time=work(i)

tmp.append(new\_time)

tmp.sort()

xs.append(i)

ys.append(tmp[len(tmp)//2])

plt.plot(xs, ys)

plt.title("n\_time relation")

plt.show()

# 绘制概率图

xs = []

ys = []

for i in range(1,n+1,1):

tot = 0

son = 0

c = 2

n = i

for j in range(200):

tot += 1

E = int(n\*float(math.log(n)))

res = work(n)

if res >= E - c\*n and res <= E + c\*n:

son += 1

xs.append(i)

ys.append(son/tot)

plt.plot(xs, ys)

plt.title("n\_P relation")

plt.show()