



Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра автоматики та управління в технічних системах

Лабораторна робота №6

Планування і проведення машинних експериментів з імітаційною моделлю системи

Виконала

студентка групи ІТ-91:

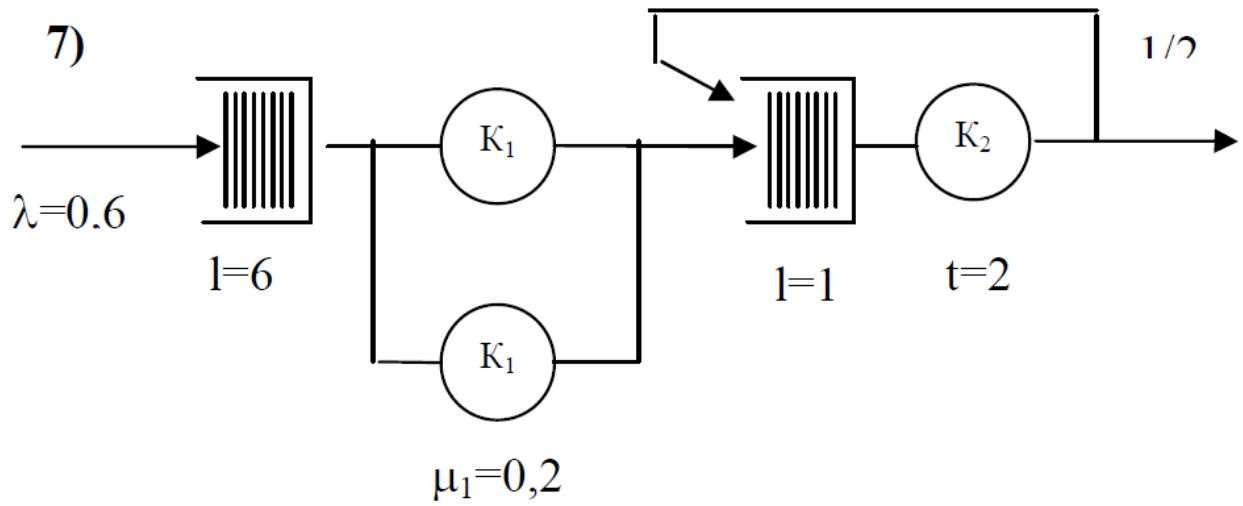
Луцай К. А.

Перевірів:

Іваніщев Б. В.

Київ 2022

Завдання: планувати і провести експеримент з імітаційною моделлю системи, побудованій при виконанні лабораторної роботи 3 або 4.



Варіант 15 (3):

Відгук моделі: середня довжина черги - ПФЕ

Фактори:

X1 – intergeneration time λ – $[10/6, 2]$

X2 – processing time μ - $[0.2, 0.7]$

Y – avg queue length l








Проведено 4 експерименти по 3 вимірювання з різною кількістю згенерованих заявок (500, 1000, 1400) для двох змінних факторів λ , μ

Модел ь реалізовано на мові GPSS, аналіз результатів проведено за допомогою мови Python.

Приклад GPSS репорту:

GPSS World - [model.90.1 - REPORT]

File Edit Search View Command Window Help



GPSS World Simulation Report - model.90.1

Thursday, December 29, 2022 15:20:13

| START TIME | END TIME | BLOCKS | FACILITIES | STORAGES |
|------------|----------|--------|------------|----------|
| 0.000 | 4259.087 | 16 | 0 | 4 |

| NAME | VALUE |
|--------------|-----------|
| LASTREPEAT | 8.000 |
| LOST | 16.000 |
| QUEUE1 | 10000.000 |
| QUEUE2 | 10001.000 |
| SERVER3 | 10003.000 |
| SERVERSSPLIT | 10002.000 |

| LABEL | LOC | BLOCK TYPE | ENTRY COUNT | CURRENT COUNT | RETRY |
|------------|-----|------------|-------------|---------------|-------|
| | 1 | GENERATE | 1400 | 0 | 0 |
| | 2 | ENTER | 1400 | 0 | 0 |
| | 3 | TRANSFER | 1400 | 0 | 0 |
| | 4 | ENTER | 1399 | 0 | 0 |
| | 5 | LEAVE | 1399 | 0 | 0 |
| | 6 | ADVANCE | 1399 | 0 | 0 |
| | 7 | LEAVE | 1399 | 0 | 0 |
| LASTREPEAT | 8 | TRANSFER | 2264 | 0 | 0 |
| | 9 | ENTER | 1785 | 0 | 0 |
| | 10 | ENTER | 1785 | 0 | 0 |
| | 11 | LEAVE | 1785 | 0 | 0 |
| | 12 | ADVANCE | 1785 | 0 | 0 |
| | 13 | LEAVE | 1785 | 0 | 0 |
| | 14 | TRANSFER | 1785 | 0 | 0 |
| | 15 | TERMINATE | 920 | 0 | 0 |
| LOST | 16 | TERMINATE | 480 | 0 | 0 |

| STORAGE | CAP. | REM. | MIN. | MAX. | ENTRIES | AVL. | AVE.C. | UTIL. | RETRY | DELAY |
|--------------|------|------|------|------|---------|------|--------|-------|-------|-------|
| QUEUE1 | 6 | 5 | 0 | 2 | 1400 | 1 | 0.722 | 0.120 | 0 | 0 |
| QUEUE2 | 1 | 1 | 0 | 1 | 1785 | 1 | 0.195 | 0.195 | 0 | 0 |
| SERVERSSPLIT | 2 | 2 | 0 | 2 | 1399 | 1 | 0.561 | 0.280 | 0 | 0 |
| SERVER3 | 1 | 1 | 0 | 1 | 1785 | 1 | 0.838 | 0.838 | 0 | 0 |

| FEC XN | PRI | BDT | ASSEM | CURRENT | NEXT | PARAMETER | VALUE |
|--------|-----|----------|-------|---------|------|-----------|-------|
| 1401 | 0 | 4259.174 | 1401 | 0 | 1 | | |

For Help, press F1

Report is Complete.

Clock

Лістинг моделі на мові GPSS:

```

Queue1 STORAGE 6
Queue2 STORAGE 1
ServersSplit STORAGE 2
Server3 STORAGE 1
GENERATE (Exponential(1, 10/5, 1))

ENTER Queue1

```

```

TRANSFER BOTH,,lost
ENTER ServersSplit

LEAVE Queue1
ADVANCE (Exponential(1, 0.7, 1))
LEAVE ServersSplit

LastRepeat TRANSFER BOTH,,lost
ENTER Queue2
ENTER Server3
LEAVE Queue2

ADVANCE 2
LEAVE Server3

TRANSFER 0.5,LastRepeat

TERMINATE 1
lost TERMINATE 1

START 1400

```

Лістинг програми Python:

```

import numpy as np
from sklearn.preprocessing import minmax_scale
import pandas as pd

x1_min, x1_max = 10/6, 2.0
x2_min, x2_max = 0.2, 0.7

n = 4 # кількість точок
c = 4 # кількість коефіцієнтів
m = 3 # кількість випробувань

norm_factors = np.array([[1, 1],
                          [1, -1],
                          [-1, 1],
                          [-1, -1]])

# number of generated entities = [500, 1000, 1400]
# Y - avg count of entities in Queue1
y_values = np.array([[0.222, 0.614, 0.722],
                     [0.0, 0.0, 0.0],
                     [0.536, 2.153, 2.966],
                     [0, 0.371, 0.548]])

factor_cols = ["X0", "X1", "X2", "X1*X2"]
val_cols = ["Y1", "Y2", "Y3"]

# інтеракції між факторами
def interact_factors(factors):
    full_factors = np.zeros((n, c))

```

```

full_factors[:, 0] = 1
full_factors[:, 1:-1] = factors
full_factors[:, -1] = factors[:, 0] * factors[:, 1]
return full_factors

# отримання натуральних факторів з нормованих
def natural(norm_factors):
    natural = np.zeros_like(norm_factors, dtype=float)
    natural[:, 0][norm_factors[:, 0] == 1] = x1_max
    natural[:, 1][norm_factors[:, 1] == 1] = x2_max
    natural[:, 0][norm_factors[:, 0] == -1] = x1_min
    natural[:, 1][norm_factors[:, 1] == -1] = x2_min
    return natural

# регресія за факторами та коефіцієнтами
def regression(matrix, coefs, n, c):
    def func(factors, coefs, c):
        y = coefs[0]
        for i in range(c):
            y += coefs[i+1] * factors[i+1]
        return y
    values = np.zeros(n)
    for f in range(n):
        values[f] = func(matrix[f], coefs, c)
    return values

# коефіцієнти з вирішення СЛР
def solve_coef(factors, values):
    A = np.zeros((c, c))
    x_mean = np.mean(factors, axis=0)
    for i in range(1, c):
        for j in range(1, c):
            A[j, i] = np.mean(factors[:, i] * factors[:, j])
    A[0, :] = x_mean
    A[:, 0] = x_mean
    B = np.mean(np.tile(values, (c, 1)).T * factors, axis=0)
    coef = np.linalg.solve(A, B)
    return coef

# коефіцієнти з нормалізованих факторів та значень
def solve_norm_coef(factors, values):
    C = np.zeros(c)
    for i in range(c):
        C[i] = np.sum(values * factors[:, i]) / n
    return C

def print_regression(coefs):
    formula = "Y = "
    for i in range(c):

```

```

        formula += f"{round(coefs[i], 2)}*X{i}"
        if i != c-1:
            formula += " + "
    print(formula)

print(f"X1 min: {x1_min}\tX1 max: {x1_max}")
print(f"X2 min: {x2_min}\tX2 max: {x2_max}")

# формування факторів
factors = natural(norm_factors)
full_factors = interact_factors(factors)

# Перевірка однорідності дисперсії за критерієм Кохрена
# отримання оптимальної кількості випробувань для однієї комбінації факторів
Gt = [6.798, 5.157] # N = 8, m = [2.. 3]
for i in range(len(Gt)):
    y_val = y_values[:, 1-i:]
    std_y = np.std(y_val, axis=1)**2
    print(f"Max Y dispersion: {std_y.max()}")
    Gp = std_y.max()/std_y.mean()
    print(f"m = {i+2} Gp: {Gp}\tGt: {Gt[i]}")
    if Gp < Gt[i]:
        m = i + 2 # кількість випробувань
        val_cols = val_cols[:m]
        f1, f2 = m-1, n
        print(f"Kohren criterion: {m} tries are enough")
        break

print(f"Mean Y dispersion: {std_y.mean()}")

# перевірка за критерієм студента
coefs_value = np.zeros(4)
for i in range(c):
    coefs_value[i] = np.mean(y_val.mean(axis=1) * full_factors[:, i])
    stud_crit = np.abs(coefs_value) / np.sqrt(std_y.mean()/(n*m))
    ts = [2.306, 2.12] # f3 = [8, 16]
    sig_ind = np.argwhere(stud_crit > ts[m-2])
    d = len(sig_ind.flatten())
    print(f"All coefs are significant: {d == c}\t{sig_ind.flatten()}")

# формування планів
plan_cols = factor_cols + val_cols + ["Y_mean", "Y_disp"]
y_mean = y_val.mean(axis=1)

natural_plan = pd.DataFrame(columns=plan_cols)
natural_plan[factor_cols] = full_factors
natural_plan[val_cols] = y_val
natural_plan["Y_mean"] = y_mean.reshape(n, 1)
natural_plan["Y_disp"] = std_y

norm_y = minmax_scale(y_val.reshape(n*m), feature_range=(-1,1)).reshape(n, m)
norm_factors_full = interact_factors(norm_factors)

```

```

y_norm_mean = norm_y.mean(axis=1)
std_norm_y = np.std(norm_y, axis=1)**2

norm_plan = pd.DataFrame(columns=plan_cols)
norm_plan[factor_cols] = norm_factors_full
norm_plan[val_cols] = norm_y
norm_plan["Y_mean"] = y_norm_mean.reshape(n, 1)
norm_plan["Y_disp"] = std_norm_y

# перевірка критерію Фішера
d = d-1 if d == n else d
Ft = 5.3

# отримання коефіцієнтів
coefs = solve_coef(full_factors, y_mean)
print("Natural coefs", coefs)
reg_val = regression(full_factors, coefs, n, m)
se = np.square(np.subtract(y_mean, reg_val))
print(f"MSE: {np.mean(se)}")

natural_plan["Y_reg"] = reg_val
print(natural_plan)
print_regression(coefs)

disp = m/(n - d) * np.sum(se)
Fp = disp / std_y.mean()
print(f"Natural Fisher crit: {Fp < Ft}\t{Fp}\t{Ft}")

# отримання нормалізованих коефіцієнтів
norm_coefs = solve_norm_coef(norm_factors_full, y_norm_mean)
print("Normalized coefs", norm_coefs)
reg_norm_val = regression(norm_factors_full, norm_coefs, n, m)
se_norm = np.square(np.subtract(y_norm_mean, reg_norm_val))
print(f"MSE: {np.mean(se_norm)}")

norm_plan["Y_reg"] = reg_norm_val
print(norm_plan)
print_regression(norm_coefs)

disp_norm = m/(n - d) * np.sum(se_norm)
Fp_norm = disp_norm / std_norm_y.mean()
print(f"Norm Fisher crit: {Fp_norm < Ft}\t{Fp_norm}\t{Ft}")

```

Результат виконання:

X1 min: 1.6666666666666667 X1 max: 2.0

X2 min: 0.2 X2 max: 0.7

Max Y dispersion: 0.16524225000000012

m = 2 Gp: 3.755708404715027 Gt: 6.798

Kohren criterion: 2 tries are enough

Mean Y dispersion: 0.04399762500000003

All coefs are significant: True [0 1 2 3]

Natural coefs [-0.947 0.3399 18.52 -8.592]

MSE: 66.29667391997464

| | X0 | X1 | X2 | X1*X2 | Y1 | Y2 | Y_mean | Y_disp | Y_reg |
|---|-----|----------|-----|----------|-------|-------|--------|----------|---------|
| 0 | 1.0 | 2.000000 | 0.7 | 1.400000 | 0.614 | 0.722 | 0.6680 | 0.002916 | 12.6968 |
| 1 | 1.0 | 2.000000 | 0.2 | 0.400000 | 0.000 | 0.000 | 0.0000 | 0.000000 | 3.4368 |
| 2 | 1.0 | 1.666667 | 0.7 | 1.166667 | 2.153 | 2.966 | 2.5595 | 0.165242 | 12.5835 |
| 3 | 1.0 | 1.666667 | 0.2 | 0.333333 | 0.371 | 0.548 | 0.4595 | 0.007832 | 3.3235 |

$Y = -0.95 * X0 + 0.34 * X1 + 18.52 * X2 + -8.59 * X3$

Natural Fisher crit: False 12054.591386689548 5.3

Normalized coefs [-0.37845583 -0.39632502 0.46662171 -0.24140256]

MSE: 0.05827519712052395

| | X0 | X1 | X2 | X1*X2 | Y1 | Y2 | Y_mean | Y_disp | Y_reg |
|---|-----|------|------|-------|-----------|-----------|-----------|----------|-----------|
| 0 | 1.0 | 1.0 | 1.0 | 1.0 | -0.585974 | -0.513149 | -0.549562 | 0.001326 | -0.308159 |
| 1 | 1.0 | 1.0 | -1.0 | -1.0 | -1.000000 | -1.000000 | -1.000000 | 0.000000 | -1.241403 |
| 2 | 1.0 | -1.0 | 1.0 | -1.0 | 0.451787 | 1.000000 | 0.725893 | 0.075134 | 0.484491 |
| 3 | 1.0 | -1.0 | -1.0 | 1.0 | -0.749831 | -0.630479 | -0.690155 | 0.003561 | -0.448753 |

$Y = -0.38 * X0 + -0.4 * X1 + 0.47 * X2 + -0.24 * X3$

Norm Fisher crit: False 23.303803330293402 5.3

Висновки: було сплановано і проведено експеримент з імітаційною моделлю системи МО