**Міністерство освіти і науки України**

**Національний технічний університет України «КПІ» імені Ігоря Сікорського**

**Кафедра обчислювальної техніки ФІОТ**

**ЗВІТ**

**з лабораторної роботи №3**

**з навчальної дисципліни «Технології Data Science»**

**Тема:**

**МАКЕТ ІНТЕЛЕКТУАЛЬНОЇ ERP СИСТЕМИ**

**ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ**

**Виконав:**

Студент 4 курсу кафедри ІПІ ФІОТ,

Навчальної групи ІП-11

Трикош І. В.

**Перевірив:**

Професор кафедри ОТ ФІОТ

Писарчук О.О.

**Київ 2024**

## **І. Мета:**

Виявити дослідити та узагальнити принципи формалізації задач, синтезу математичних моделей для автоматизації процесів підтримки прийняття рішень в інтелектуальних ERP системах: програмування обмежень – CP-SAT; багатокритеріальні задачі – Multicriteria decision analysis.

## **ІІ. Завдання:**

1. Для визначення можливості автоматизації бізнес процесів, що реалізовані в компанії замовника Вам пропонується розробити макет програмної реалізації мовою Python обчислювального алгоритму ERP системи підтримки прийняття рішень за умов:

**ІІ рівень складності, 8 балів:**

Варіант 10:

Розробити програмний скрипт, що реалізує багатокритеріальне оцінювання ефективності позашляховиків різних виробників. Формування показників та критеріїв ефективності, синтез багатокритеріальної оптимізаційної моделі здійснити самостійно.

2. Провести аналіз отриманих результатів та верифікацію розробленого скрипта.

3. **Додаткове завдання до рівня ІІ + 2 бали.**

Розробити програмний скрипт, що забезпечує розв’язок задачі лінійного програмування для умов, зазначених в Лекції\_6 з використанням інструментів бібліотеки Google OR-Tools. Порівняти отримані результати із графічним методом розв’язку, що наведено в Лекції\_6. Здійснити опис практичної інтерпретації задачі лінійного програмування, що розв’язана Вами для конкретної прикладної галузі (із власного практичного досвіду, або з аналізу інформаційних джерел). Опис подати у протоколі.

## **ІІІ. Результати виконання лабораторної роботи.**

* 1. **Синтезована математична модель**

Відповідно до умов задачі синтезовано математичну модель, що реалізує багатокритеріальне оцінювання ефективності позашляховиків різних виробників за їх показниками.

Розрахунок оцінки ефективності позашляховика відбувається за формулою:

де – підсумкова оцінка ефективності позашляховика, – нормована вага фактору , – значення фактору для позашляховика , – кількість факторів, – кількість позашляховиків. Якщо фактор максимізований, то для нього використовується оберенене значення – . Таким чином, найоптимальнішим позашляховиком буде той, що має найменше .

Також модель реалізує мінімізацію функції за певних обмежень, використовуючи CP SAT з бібліотеки Google OR-Tools – розв’язувач задач оптимізації, що комбінує використання CP (Constraint Programming) та SAT (Satisfiability) для вирішення задач з обмеженнями. Він дозволяє задавати як жорсткі, так і м’які обмеження та підтримує мінімізацію і максимізацію.

* 1. **Результати архітектурного проектування та їх опис**

На рис. 1 зображена блок-схема до моделі до програмного скрипту №1:

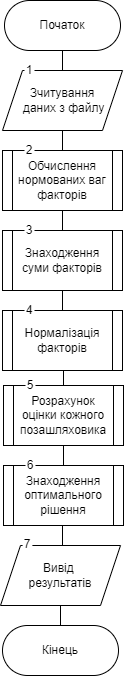


Рисунок 1 – Блок схема алгоритму №1

Робота алгоритму розпочинається зі зчитування даних з файлу у датафрейм. Ці дії виконуються у блоці 1 блок-схеми алгоритму рис.1.

У блоці 2 виконується обчислення нормованих ваг факторів.

У блоці 3 знаходиться сума факторів залежно від мінімізації/максимізації фактора.

У блоці 4 виконується нормалізація факторів залежно від мінімізації/максимізації відповідного фактора.

У блоці 5 відбувається обрахунок оцінки для кожного позашляховика.

У блоці 6 знаходиться оптимальне рішення (позашляховик з найнижчою оцінкою).

У блоці 7 відбувається вивід результатів. На цьому виконання алгоритму завершене.

На рис. 2 зображена блок-схема до моделі до програмного скрипту №2:

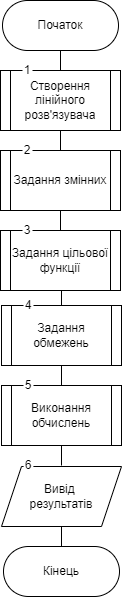


Рисунок 2 – Блок схема алгоритму №2

Робота алгоритму розпочинається зі створення лінійного розв’язувача CP SAT. Ці дії виконуються у блоці 1 блок-схеми алгоритму рис.2.

У блоці 2 виконується задання змінних, які присутні в обмеженнях, та їх проміжків значень.

У блоці 3 задається цільова функція (та, яку потрібно мінімізувати, враховуючи задані обмеження).

У блоці 4 задаються обмеження для змінних.

У блоці 5 відбуваються обчислення (мінімізація функції).

У блоці 6 відбувається вивід результатів. На цьому виконання алгоритму завершене.

* 1. **Опис структури проекту програми**

Для реалізації розробленого алгоритму мовою програмування Python з використанням можливостей інтегрованого середовища PyCharm сформовано проєкт.

Проєкт базується на лінійній бізнес-логіці функціонального програмування та має таку структуру.

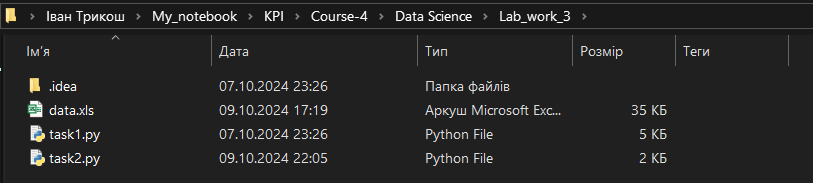


Рисунок 3 – Структура проєкту

Lab\_work\_3 – головний каталог проєкту

task1.py – файл програмного коду алгоритму №1

task2.py – файл програмного коду алгоритму №2

data.xls – дані для алгоритму №1 (критерії, їх ваги і позашляховики)

* 1. **Результати роботи програми відповідно до завдання (допускається у формі скриншотів)**

Результатом роботи програми є сукупність послідовності графічних вікон, що реалізують умови завдання лабораторної роботи.

1. На рисунках 4-5 зображено виконання завдання 1:

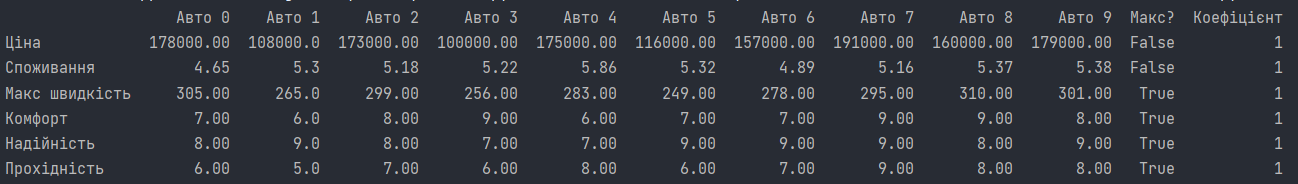


Рисунок 4 – Дані

У рядках знаходяться значення критеріїв для позашляховиків. Стовпець Макс? є булевим і визначає, чи потрібно максимізувати критерій. Стовпець Коефіцієнт містить значення коефіцієнтів кожного критерію.

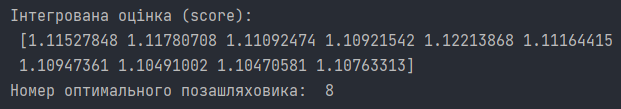


Рисунок 5 – Результати роботи алгоритму 1

Бачимо, що значення з індексом 8 є найоптимальнішим (найменшим), тому Авто 8 є оптимальним вибором.

1. На рисунку 6 зображено виконання завдання 2:

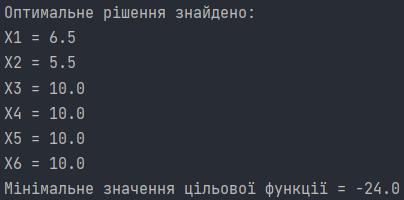


Рисунок 6 – Результати роботи алгоритму 2

Бачимо, що знайдено оптимальні значення для змінних на проміжку від 0 до 10 та заданих обмежень для мінімізації функції Q = -2 \* X1 - 2 \* X2. Знайдене мінімальне значення функції дорівнює -24.

З допомогою графічного методу було знайдено наступні значення:

X1 = 4

X2 = 2

X3 = 0

X4 = 0

X5 = 0

X6 = 0

Мінімальне значення цільової функції при цих значеннях дорівнює -12.

Бачимо, що метод CP SAT з бібліотеки Google OR-Tools краще за графічний метод мінімізував цільову функцію.

1. Опис практичної інтерпретації задачі лінійного програмування:

Потрібно оптимізувати постачання продукції на кілька складів з виробничих підприємств. Кожне підприємство має обмежену кількість ресурсів (сировина та виробничі потужності). Кожен склад має попит на певний тип продукції і витрати між підприємствами і складами можуть варіюватися. Завдання полягає в тому, щоб мінімізувати загальні витрати на виробництво та транспортування, забезпечуючи при цьому задоволення попиту складів на продукцію. Потрібно також врахувати максимальні можливості підприємств та логістичні обмеження.

* 1. **Програмний код, що забезпечує отримання результату (допускається у формі скриншотів)**

Програмний код послідовно реалізує алгоритми на рис.1-2 та спрямований на отримання результатів, поданих на рис.4-6.

Для спрощення програмного коду і раціоналізації обчислень застосовано функціональні механізми створення підпрограм.

При цьому використано можливості Python бібліотек: numpy, pandas, ortools.

Контекстні коментарі пояснюють сутність окремих скриптів наведеного коду програми.

task1.py

*"""  
Завдання:  
 Розробити програмний скрипт, що реалізує багатокритеріальне оцінювання ефективності  
 позашляховиків різних виробників. Формування показників та критеріїв ефективності,  
 синтез багатокритеріальної оптимізаційної моделі здійснити самостійно.  
  
Обрані критерії:  
 \* Ціна (USD) - мінімізований, вага 1  
 \* Споживання пального (літрів на 100 км) - мінімізований, вага 1  
 \* Максимальна швидкість (км/год) - максимізований, вага 1  
 \* Комфорт (оцінка від 1 до 10) - максимізований, вага 1  
 \* Надійність (оцінка від 1 до 10) - максимізований, вага 1  
 \* Прохідність (оцінка від 1 до 10) - максимізований, вага 1  
"""  
  
  
import* pandas *as* pd  
*import* numpy *as* np  
  
*# Для повного виводу даних у консоль*pd.set\_option('display.max\_rows', *None*)  
pd.set\_option('display.max\_columns', *None*)  
pd.set\_option('display.width', *None*)  
  
  
*def* read\_data(file\_name):  
 *""" Зчитування даних з .xls файлу та їх перетворення до типу float """* sample\_data = pd.read\_excel(file\_name, index\_col=0)  
 *for* i *in* range(0, len(sample\_data.columns) - 2):  
 sample\_data[sample\_data.columns[i]] = sample\_data[sample\_data.columns[i]].replace(',', '.', regex=*True*).astype(float)  
 *return* sample\_data  
  
  
*def* model(file\_name):  
 *""" Оцінювання ефективності позашляховиків """  
  
 # Вхідні дані* line\_column\_matrix = read\_data(file\_name)  
 print(line\_column\_matrix)  
 scores = np.zeros(line\_column\_matrix.shape[1] - 2)  
  
 *# Критерії та відповідні коефіцієнти* factors = [factor *for* factor *in* line\_column\_matrix.values]  
  
 *# Нормалізовані фактори* norm\_factors = [factor[:-2] *for* factor *in* factors]  
  
 *# Нормована вага кожного фактору* g\_norm = sum([factor[-1] *for* factor *in* factors])  
 gs = [factor[-1] / g\_norm *for* factor *in* factors]  
  
 *# Знаходження суми факторів (для нормалізації)* sum\_factors = [0 *for* \_ *in* range(line\_column\_matrix.shape[0])]  
 *for* i *in* range(len(factors)):  
 *if* factors[i][-2]: *# Максимізований фактор* sum\_factors[i] = (1 / factors[i][:-2]).sum()  
 *else*: *# Мінімізований фактор* sum\_factors[i] = factors[i][:-2].sum()  
  
 *# Нормалізація факторів  
 for* i *in* range(line\_column\_matrix.shape[1] - 2):  
 *for* j *in* range(len(norm\_factors)):  
 *if* factors[j][-2]: *# Максимізований фактор* norm\_factors[j] = (1 / factors[j][:-2]) / sum\_factors[j]  
 *else*: *# Мінімізований фактор* norm\_factors[j] = factors[j][:-2] / sum\_factors[j]  
  
 *# Розрахунок оцінки (score)  
 for* i *in* range(len(scores)):  
 scores[i] = 0  
 *for* j *in* range(len(gs)):  
 scores[i] += gs[j] \* (1 - norm\_factors[j][i])\*\*(-1)  
  
 *# Знаходження оптимального рішення (позашляховика)* min\_value = float('inf')  
 optimal = 0  
 *for* i *in* range(len(scores)):  
 *if* min\_value > scores[i]:  
 min\_value = scores[i]  
 optimal = i  
 print('\nІнтегрована оцінка (score):\n', scores)  
 print('Номер оптимального позашляховика: ', optimal)  
 *return  
  
  
if* \_\_name\_\_ == '\_\_main\_\_':  
 model(file\_name='data.xls')

task2.py

*"""  
Завдання:  
 Мінімізувати функцію Q = -2\*X1 - 2\*X2 за умов:  
 1.5 \* X1 + 2 \* X2 - X4 <= 12  
 X1 + 2 \* X2 - X3 <=  
 4 \* X1 - X5 <= 16  
 4 \* X2 - X6 <= 12  
 Xj >= 0, (j = 1..6)  
"""  
  
from* ortools.linear\_solver *import* pywraplp  
  
*# Створення лінійного розв'язувача*solver = pywraplp.Solver.CreateSolver('CP\_SAT')  
  
*# Задаємо змінні X1, ..., X6 з умовою, що вони не можуть бути меншими за 0 і  
# не більшими за 10 (для знаходження розв'язку в обмеженій області)*X1 = solver.NumVar(0, 10, 'X1')  
X2 = solver.NumVar(0, 10, 'X2')  
X3 = solver.NumVar(0, 10, 'X3')  
X4 = solver.NumVar(0, 10, 'X4')  
X5 = solver.NumVar(0, 10, 'X5')  
X6 = solver.NumVar(0, 10, 'X6')  
  
*# Задаємо цільову функцію*solver.Minimize(-2 \* X1 - 2 \* X2)  
  
*# Задаємо обмеження*solver.Add(1.5 \* X1 + 2 \* X2 - X4 <= 12)  
solver.Add(X1 + 2 \* X2 - X3 <= 8)  
solver.Add(4 \* X1 - X5 <= 16)  
solver.Add(4 \* X2 - X6 <= 12)  
  
*# Виконуємо обчислення*status = solver.Solve()  
  
*# Виведення результатів  
if* status == pywraplp.Solver.OPTIMAL:  
 print('Оптимальне рішення знайдено:')  
 print(f'X1 = {X1.solution\_value()}')  
 print(f'X2 = {X2.solution\_value()}')  
 print(f'X3 = {X3.solution\_value()}')  
 print(f'X4 = {X4.solution\_value()}')  
 print(f'X5 = {X5.solution\_value()}')  
 print(f'X6 = {X6.solution\_value()}')  
 print(f'Мінімальне значення цільової функції = {-2 \* X1.solution\_value() - 2 \* X2.solution\_value()}')  
*else*:  
 print('Оптимальне рішення не знайдено.')

## **IV. Висновки.**

У ході виконання лабораторної роботи було проведено дослідження принципів формалізації задач, синтезу математичних моделей для автоматизації процесів підтримки прийняття рішень в інтелектуальних ERP системах: програмування обмежень – CP-SAT; багатокритеріальні задачі – Multicriteria decision analysis. Було знайдено оптимальний позашляховик за критеріями та виконано мінімізацію функції із заданими обмеженнями з використанням Google OR-Tools. Також було виконано опис практичної інтерпретації задачі лінійного програмування.

Виконав: студент Трикош І. В.