ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра Математическая кибернетика и информационные технологии

САМОСТОЯТЕЛЬНАЯ РАБОТА

по дисциплине

Теория оптимизации

на тему:

Решение транспортной задачи с использованием библиотек Python: pulp, cvxopt and optimize.

Выполнил: студент гр. 3МПП1901

Шемякин А.В.

Принял: доцент, к.т.н.

Городничев М. Г.

Москва 2019 г.

Оглавление

[Постановка задачи 3](#_Toc27690152)

[Решение 3](#_Toc27690153)

[Использование библиотеки pulp. 4](#_Toc27690154)

[С использованием библиотеки cvxopt: 7](#_Toc27690155)

[С использованием библиотеки scipy.optimize: 9](#_Toc27690156)

[Сравнение использованных библиотек 11](#_Toc27690157)

[Список использованных источников 12](#_Toc27690158)

# Постановка задачи

Существует некоторая фирма, оказывающая транспортные услуги. У фирмы есть склады с некоторым товаром в разных городах. Так же есть другие города такого же количества, как и количество городов со складом, которые потребляют поставляемый товар. Стоимость доставки определена заранее. Суть задачи состоит в оптимизации доставки товара в потребительские точки, минимизируя затраты на перевозку. Решить транспортную задачу требуется программно, с помощью языка Python, используя библиотеки pulp, cvxopt и scypy.optimize.

# Решение

Приведем таблицу, в которой укажем города-поставщики (A1-A5) и количество поставляемого товара, города-потребители (B1-B5) и количество требуемых единиц товара и стоимости доставки между городами (С1-С25).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | B1 = 11 | B2 = 25 | B3 = 46 | B4 = 9 | B5 = 34 |
| A1 = 30 | C1 = 4 | C2 = 11 | C3 = 5 | C4 = 17 | C5 = 15 |
| A2 = 15 | C6 = 6 | C7 = 15 | C8 = 16 | C9 = 7 | C10 = 1 |
| A3 = 25 | C11 = 4 | C12 = 9 | C13 = 2 | C14 = 4 | C15 = 5 |
| A4 = 14 | C16 = 11 | C17 = 5 | C18 = 21 | C19 = 14 | C20 = 10 |
| A5 = 49 | C21 = 10 | C22 = 1 | C23 = 7 | C24 = 10 | C25 = 6 |

Пусть Х – количество перевозимого груза из города в город, тогда функция цели будет функция вида:

F(X) = C1\*X1+C2\*X2+ … + C25\*X25

Задача имеет два типа условий:

1. Условия неравенств:  
   X1+X2+X3+X4+X5 <= 30;  
   X6+X7+X8+X9+10 <= 15;

И так далее…

1. Условия равенств:  
   X1+X6+X11+X16+X21 = 11;  
   X2+X7+X12+X17+X22 = 25;

И так далее…

Используя условия задачи и функцию, переведем решение задачи в код языка Python:

# Использование библиотеки pulp.

PuLP – вспомогательная библиотека языка Python, позволяющая работать с математическими моделями. Так выглядит решение задачи с использованием этой библиотеки:

from pulp import \*

import time

start = time.time()

#Обозначение стоимостей перевозок между городами

c = [4, 11, 5, 17, 15, 6, 15, 16, 7, 1, 4, 9, 2, 4, 5, 11, 5, 21, 14, 10, 10, 1 , 7, 10, 6]

#Переменные для подготовления формирования проблемы оптимизации

x1 = pulp.LpVariable("x1", lowBound=0)

x2 = pulp.LpVariable("x2", lowBound=0)

x3 = pulp.LpVariable("x3", lowBound=0)

x4 = pulp.LpVariable("x4", lowBound=0)

x5 = pulp.LpVariable("x5", lowBound=0)

x6 = pulp.LpVariable("x6", lowBound=0)

x7 = pulp.LpVariable("x7", lowBound=0)

x8 = pulp.LpVariable("x8", lowBound=0)

x9 = pulp.LpVariable("x9", lowBound=0)

x10 = pulp.LpVariable("x10", lowBound=0)

x11 = pulp.LpVariable("x11", lowBound=0)

x12 = pulp.LpVariable("x12", lowBound=0)

x13 = pulp.LpVariable("x13", lowBound=0)

x14 = pulp.LpVariable("x14", lowBound=0)

x15 = pulp.LpVariable("x15", lowBound=0)

x16 = pulp.LpVariable("x16", lowBound=0)

x17 = pulp.LpVariable("x17", lowBound=0)

x18 = pulp.LpVariable("x18", lowBound=0)

x19 = pulp.LpVariable("x19", lowBound=0)

x20 = pulp.LpVariable("x20", lowBound=0)

x21 = pulp.LpVariable("x21", lowBound=0)

x22 = pulp.LpVariable("x22", lowBound=0)

x23 = pulp.LpVariable("x23", lowBound=0)

x24 = pulp.LpVariable("x24", lowBound=0)

x25 = pulp.LpVariable("x25", lowBound=0)

#Решение проблемы оптимизации

problem = pulp.LpProblem('0',pulp.LpMaximize)

problem += -C1\*x1 - C2\*x2 - C3\* x3 - C4\*x4 - C5\*x5 -C6\* x6-C7\*x7- C8\*x8 -C9\* x9-C10\*x10 - C11\*x11 - C12\* x12 - C13\*x13 - C14\*x14 -C15\* x15-C16\*x16- C17\*x17-C18\* x18-C19\*x19 - C20\*x20 - C21\* x21 - C22\*x22 - C23\*x23 -C24\* x24-C25\*x25, "Функция цели"

problem +=x1 + x2 +x3+x4+x5<= 74,"1"

problem +=x6 + x7 +x8+x9+x10 <= 40, "2"

problem +=x11 + x12+ x13+x14+x15 <= 36, "3"

problem +=x16 + x17+ x18+x19+x20 <= 9, "4"

problem +=x21 + x22+ x23 +x24+x25<= 34, "5"

problem +=x1+ x6+ x11+x16+x21 == 20, "6"

problem +=x2+x7+ x12+x17+x22 == 45, "7"

problem +=x3 + x8+x13+x18+x23 == 30, "8"

problem +=x4 + x9+x14+x19+x24 == 14, "9"

problem +=x5 + x10+x15+x20+x25 == 49, "10"

#Метод решения проблемы

problem.solve()

#Вывод результатов

print ("Результат:")

for variable in problem.variables():

print (variable.name, "=", variable.varValue)

print ("Стоимость доставки:")

print (abs(value(problem.objective))) #По модулю

stop = time.time()

print ("Время :")

print(stop - start)

Запустив код, консоль выведет следующий результат:

Результат:

x1 = 20.0

x10 = 40.0

x11 = 0.0

x12 = 0.0

x13 = 13.0

x14 = 14.0

x15 = 9.0

x16 = 0.0

x17 = 9.0

x18 = 0.0

x19 = 0.0

x2 = 2.0

x20 = 0.0

x21 = 0.0

x22 = 34.0

x23 = 0.0

x24 = 0.0

x25 = 0.0

x3 = 17.0

x4 = 0.0

x5 = 0.0

x6 = 0.0

x7 = 0.0

x8 = 0.0

x9 = 0.0

Стоимость доставки:

433.0

Время :

0.054937124252319336

# С использованием библиотеки cvxopt:

from cvxopt.modeling import variable, op

import time

start = time.time()

#Обозначение стоимостей перевозок между городами

c = [4, 11, 5, 17, 15, 6, 15, 16, 7, 1, 4, 9, 2, 4, 5, 11, 5, 21, 14, 10, 10, 1 , 7, 10, 6]

#Переменные для подготовления формирования проблемы оптимизации

x = variable(25, 'x')

z=(c[0]\*x[0] + c[1]\*x[1] +c[2]\* x[2] +c[3]\*x[3] + c[4]\*x[4] +c[5]\* x[5]

+c[6]\*x[6] + c[7]\*x[7] +c[8]\* x[8]+c[9]\*x[9] + c[10] \* x[10]+c[11]\* x[11]

+c[12]\*x[12] + c[13]\*x[13] +c[14]\*x[14]+c[15]\*x[15] + c[16]\*x[16]

+c[17]\* x[17]+c[18]\*x[18] + c[19]\*x[19]+c[20]\* x[20] +c[21]\*x[21]

+ c[22]\*x[22] +c[23]\* x[23]+c[24]\*x[24])

mass1 = (x[0] + x[1] +x[2] +x[3] + x[4]<= 74)

mass2 = (x[5] + x[6] +x[7] +x[8] + x[9] <= 40)

mass3 = (x[10] + x[11] +x[12] +x[13] + x[14] <= 36)

mass4 = (x[15] + x[16] +x[17] +x[18] + x[19] <= 9)

mass5 = (x[20] + x[21] +x[22] +x[23] + x[24] <= 34)

mass6 = (x[0] + x[5] + x[10] + x[15]+x[20] == 20)

mass7 = (x[1] + x[6] + x[11] + x[16]+x[21] == 45)

mass8 = (x[2] + x[7] + x[12] + x[17]+x[22] == 30)

mass9 = (x[3] + x[8] + x[13] + x[18]+x[23] == 14)

mass10 = (x[4] + x[9] + x[14] + x[19]+x[24] == 49)

x\_non\_negative = (x >= 0)

#Метод решения проблемы

problem =op(z,[mass1,mass2,mass3,mass4 ,mass5,mass6,mass7,mass8,mass9,mass10, x\_non\_negative])

problem.solve(solver='glpk')

#Вывод результатов

print("Результат Xopt:")

for i in x.value:

print(i)

print("Стоимость доставки:")

print(problem.objective.value()[0])

stop = time.time()

print ("Время :")

print(stop - start)

Запуск кода приведет к следующему результату:

Результат Xopt:

20.0

2.0

17.0

0.0

0.0

0.0

0.0

0.0

0.0

40.0

0.0

0.0

13.0

14.0

9.0

0.0

9.0

0.0

0.0

0.0

0.0

34.0

0.0

0.0

0.0

Стоимость доставки:

433.0

Время :

0.019977807998657227

# С использованием библиотеки scipy.optimize:

from scipy.optimize import linprog

import time

start = time.time()

#Обозначение стоимостей перевозок между городами

c = [4, 11, 5,17,15,6,15,16,7,1,4,8,2,4,5,11,5,21,14,10,10,1,7,10,6]

#Переменные для подготовления формирования проблемы оптимизации

A\_ub = [[1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0],

[0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0],

[0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0, 0,0,0,0,0],

[0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0],

[0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 1,1,1,1,1]]

b\_ub = [74,40,36,9,34]

A\_eq = [[1,0,0,0,0, 1,0,0,0,0, 1,0,0,0,0, 1,0,0,0,0, 1,0,0,0,0],

[0,1,0,0,0, 0,1,0,0,0, 0,1,0,0,0, 0,1,0,0,0, 0,1,0,0,0],

[0,0,1,0,0, 0,0,1,0,0, 0,0,1,0,0, 0,0,1,0,0, 0,0,1,0,0],

[0,0,0,1,0, 0,0,0,1,0, 0,0,0,1,0, 0,0,0,1,0, 0,0,0,1,0],

[0,0,0,0,1, 0,0,0,0,1, 0,0,0,0,1, 0,0,0,0,1, 0,0,0,0,1]]

b\_eq = [20,45,30,14,49]

#Решение проблемы оптимизации и вывод результатов

print(linprog(c, A\_ub, b\_ub, A\_eq, b\_eq))

stop = time.time()

print ("Время :")

print(stop - start)

При запуске программа выведет следующий результат:

con: array([5.61001592e-08, 1.49600311e-07, 9.35002546e-08, 3.36600987e-08,

1.64561072e-07])

fun: 432.9999994420423

message: 'Optimization terminated successfully.'

nit: 6

slack: array([3.50000001e+01, 1.31050207e-07, 1.28932754e-07, 1.34667300e-08,

1.08260522e-07])

status: 0

success: True

x: array([1.99999999e+01, 9.30352344e-01, 1.80696476e+01, 4.58622536e-09,

7.31806601e-09, 4.68392016e-09, 1.16447922e-08, 2.56914147e-09,

9.37942568e-09, 3.99999998e+01, 1.61509570e-08, 1.06964765e+00,

1.19303523e+01, 1.39999999e+01, 8.99999998e+00, 4.16105738e-09,

8.99999997e+00, 3.45183172e-09, 5.07957230e-09, 6.89612208e-09,

3.01482756e-09, 3.39999999e+01, 3.32656863e-09, 4.91069582e-09,

4.14734956e-09])

Время :

0.5489616394042969

# Сравнение использованных библиотек

Исходя из полученных результатов, с точки зрения представления выходной информации, scipy.optimize можно считать наиболее удобным и информативным. Так же следует обратить внимание на более точный вывод стоимости доставки – остальные две библиотеки используют округленное значение. Но с точки зрения быстродействия, pulp или cvxopt подходят больше, если речь идет о более объемных испытаниях и более трудоемких задачах. Наиболее быстро с задачей справилась библиотека cvxopt. Таким образом, можно сделать вывод, что использование определенной библиотеки должно производиться исходя из поставленных задач.

# Список использованных источников

1. Кочегурова Е.А. Теория и методы оптимизации. – 171 с.
2. Ашманов С.А. Линейное программирование. – 171 с. Москва «Наука». – 303 с.
3. Решение задач линейного программирования с использованием Python [Электронный ресурс]. URL: <https://habr.com/ru/post/330648> .