Фреймворки Python

Розв’язання задач лінійного програмування

Загальні відомості

- Графічний метод розв'язання завдання лінійного програмування заснований на її геометричній інтерпретації. Подібна інтерпретація, за невеликим винятком, можлива лише для задач з двома змінними.
- Саме для такої задачі легко на площині виконати необхідні побудови. Для більшої кількості змінних така побудова неможлива і для розв'язання складніших задач використовуються інші аналітичні методи.
- Для обгрунтування методів розв'язання задач лінійного програмування сформулюємо ряд найважливіших теорем, опускаючи їх аналітичні докази.

Основні теорема

Теорема 1. Множина всіх допустимих рішень системи обмежень задачі лінійного програмування є опуклою.

- У окремому випадку, коли в систему обмежень входять тільки дві змінні x_1 і x_2, цю множину можна зобразити на площині.
- Оскільки мова йде про допустимі рішення (x_1, $x_2 \geq 0$), то відповідна множина розташовуватиметься в першому квадранті декартової системи координат.
- Ця множина може бути замкнутою (багатокутник), незамкнутою (необмежена багатогранна область), складатися з єдиної точки і, нарешті, система обмежень-нерівностей може бути суперечливою.

Основні теорема

Теорема 2. Якщо задача лінійного програмування має оптимальне рішення, то воно співпадає з однією (двома) з кутових точок множини допустимих рішень.

- З теореми 2 можна зробити висновок про те, що єдиність оптимального рішення може порушуватися, причому, якщо рішення не єдине, то таких оптимальних рішень буде незліченна множина (всі точки відрізку, що сполучає відповідні кутові точки).

Основні теорема

Теорема 3. Кожному допустимому базисному рішенню задачі лінійного програмування відповідає кутова точка області допустимих рішень, і навпаки.

- Слідством з теорем 2 і 3 є твердження про те, що оптимальне рішення (оптимальні рішення) задачі лінійного програмування, заданої (або приведеною) обмеженнями-рівняннями, співпадає з допустимим базисним вирішенням системи обмежень.
- Таким чином, оптимальне вирішення ЗЛП слід шукати серед кінцевого числа допустимих базисних рішень.

Постановка задачі

- Фабрика виробляє два види фарб: перший - для зовнішніх, а другий - для внутрішніх робіт.
- Для виробництва фарб використовуються два інгредієнта: А і В. Максимально можливі добові запаси цих інгредієнтів складають 6 і 8 т відповідно. Відомі витрати А і В на 1 т відповідних фарб (табл. 1). Вивчення ринку збуту показало, що добовий попит на фарбу 2-го виду ніколи не перевищує попиту на фарбу 1-го виду більш, ніж на 1 т.
- Крім того, встановлено, що попит на фарбу 2-го виду ніколи не перевищує 2 т в добу.
- Оптові ціни однієї тонни фарб рівні: 3 тис. Грн. для фарби 1-го виду; 2 тис. Грн. для фарби 2-го виду.
- Необхідно побудувати математичну модель, що дозволяє
- встановити, яка кількість фарби кожного виду треба виробляти, щоб дохід від реалізації продукції був максимальним.

Параметри задачі про виробництво фарби

Інгредієнти	Витрати інгредієнтів, т інгр/ т фарби		Запас, т інгр./добу
	Фарба 1-го виду	Фарба 2-го виду	
A	1	2	6
B	2	1	8
Ціна, тис	3	2	

Математична модель

I. Функція мети

$$F(x) = 3x_1 + 2x_2 \rightarrow max$$

II. Обмеження

$$\begin{cases} x_1 + 2x_2 \le 6 \\ 2x_1 + x_2 \le 8 \\ x_2 - x_1 \le 1 \\ x_2 \le 2 \\ x_1, x_2 \ge 0 \end{cases}$$

Алгоритм розв’язання задачі граф. методом

I. У обмеженнях задачі замінити знаки нерівностей на знаки точної рівності і побудувати відповідні прямі.

1) $x_1 + 2x_2 \le 6 \rightarrow x_1 + 2x_2 = 6$

$$\begin{Bmatrix} x_1 = 0 \\ x_2 = 3 \end{Bmatrix} \begin{Bmatrix} x_1 = 6 \\ x_2 = 0 \end{Bmatrix}$$

2) $2x_1 + x_2 \le 8 \rightarrow 2x_1 + x_2 = 8$

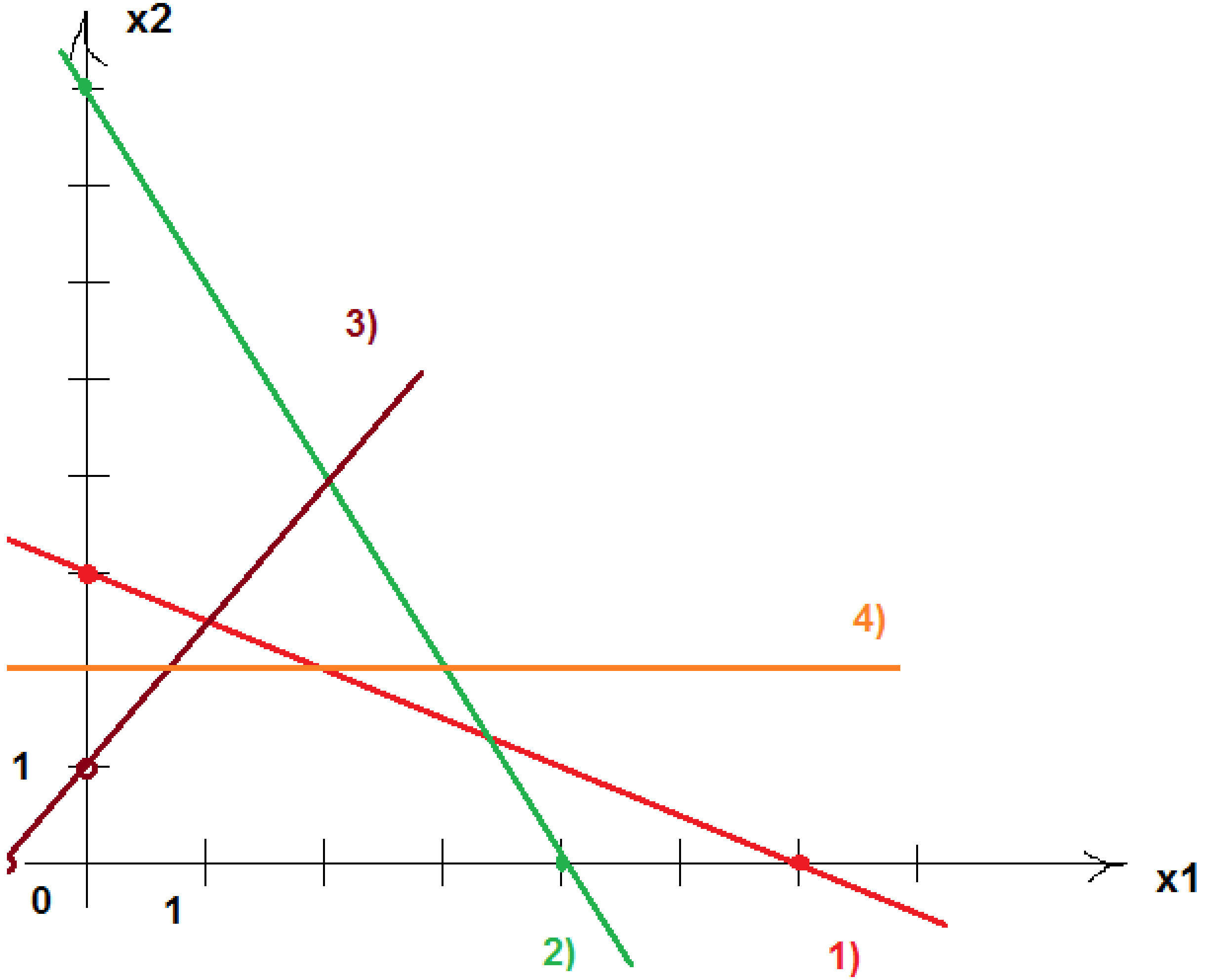
$$\begin{Bmatrix} x_1 = 0 \\ x_2 = 8 \end{Bmatrix} \begin{Bmatrix} x_1 = 4 \\ x_2 = 0 \end{Bmatrix}$$

3) $x_2 - x_1 \le 1 \rightarrow x_2 - x_1 = 1$

$$\begin{Bmatrix} x_1 = 0 \\ x_2 = 1 \end{Bmatrix} \begin{Bmatrix} x_1 = -1 \\ x_2 = 0 \end{Bmatrix}$$

4) $x_2 \le 2$

Побудовані прямі

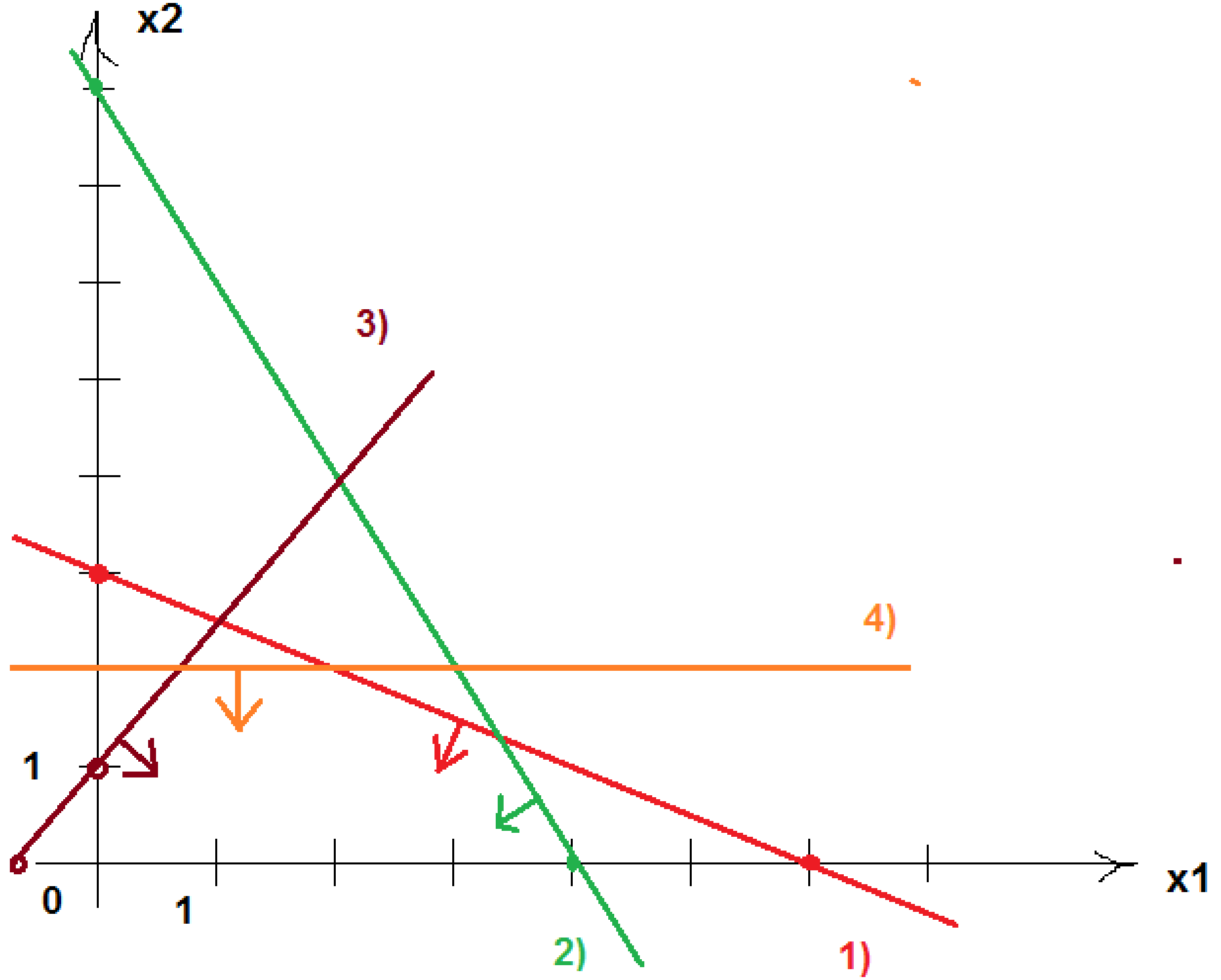


Алгоритм розв’язання задачі граф. методом

II. Знайти і заштрихувати (або позначити стрілками) напівплощини, дозволені кожним з обмежень-нерівностей задачі.

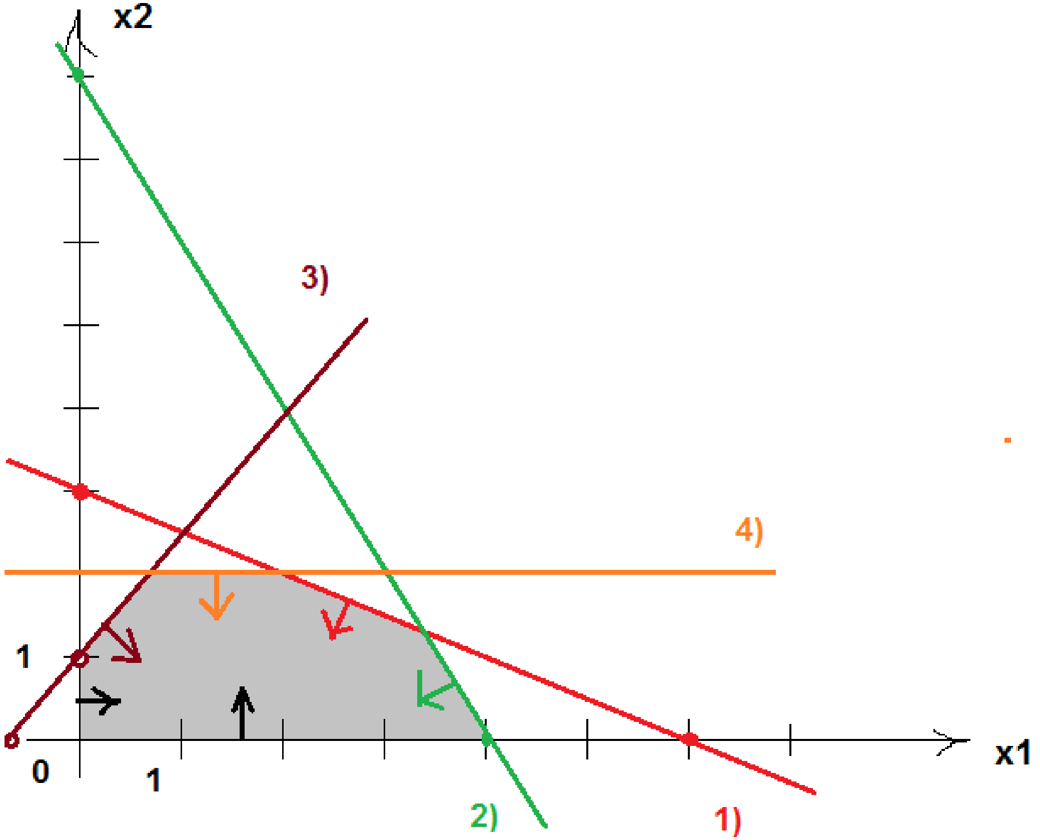
- Для цього підставити в конкретну нерівність координати якої-небудь точки [наприклад (0;0)], і перевірити істинність отриманої нерівності.

  Якщо нерівність істинна
  то треба заштрихувати напівплощину, що містить дану точку;
  інакше (нерівність хибна) треба заштрихувати напівплощину, що не містить дану точку.
- Оскільки x_1 і x_2 повинні бути невід’ємними, то їх допустимі значення завжди знаходитимуться вище вісь x_1 і правіше за вісь x_2, тобто в I-м квадранті.
- Обмеження-рівність визначає тільки ті точки, що лежать на відповідній прямій, тому необхідно виділити на графіку такі прямі.


x2
3)
4)
1
0
1
x1
2)
1)

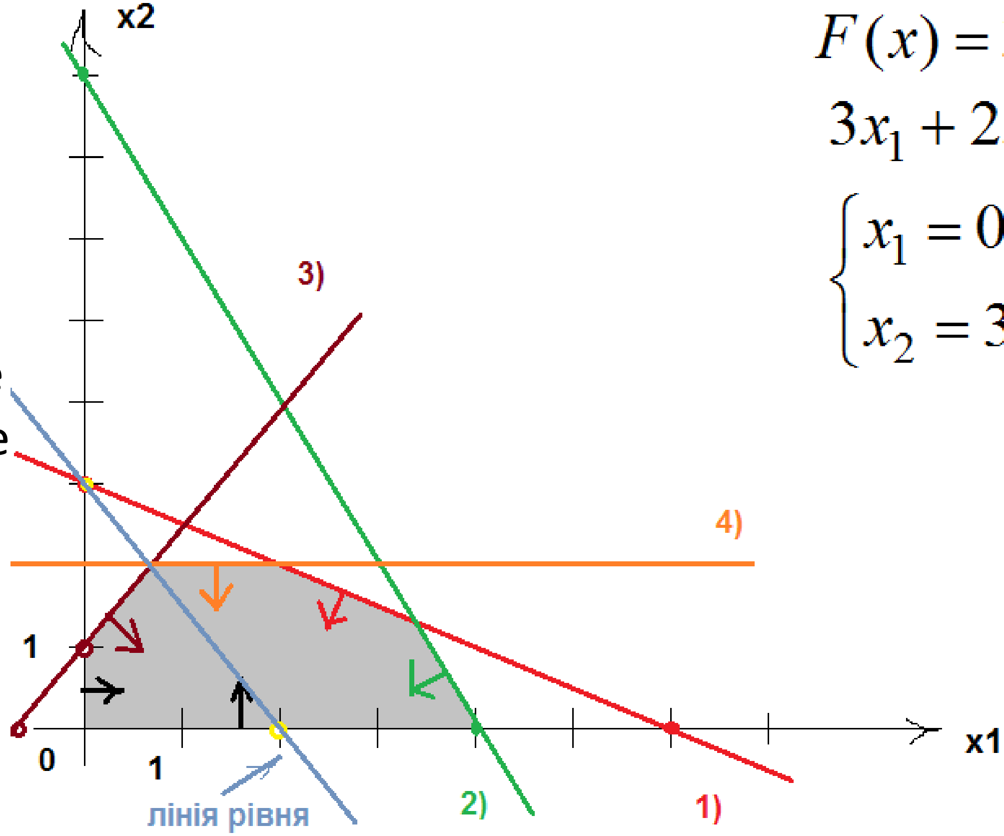
Алгоритм розв’язання задачі граф. методом

III. Визначити ОДР як частину площини, що належить одночасно всім дозволеним областям, і виділити її. За відсутності ОДР задача ***не має рішень***, про що необхідно зробити відповідний висновок.



Алгоритм розв’язання задачі граф. методом

IV. Якщо ОДР – не порожня множина, то побудувати цільову пряму, тобто будь-яку з ліній рівня $c_1x_1 + c_2x_2 = L$, де L – довільне число, наприклад, кратне c_1 і c_2, тобто зручне для проведення розрахунків. Спосіб побудови аналогічний побудові прямих обмежень.



$$F(x) = 3x_1 + 2x_2$$

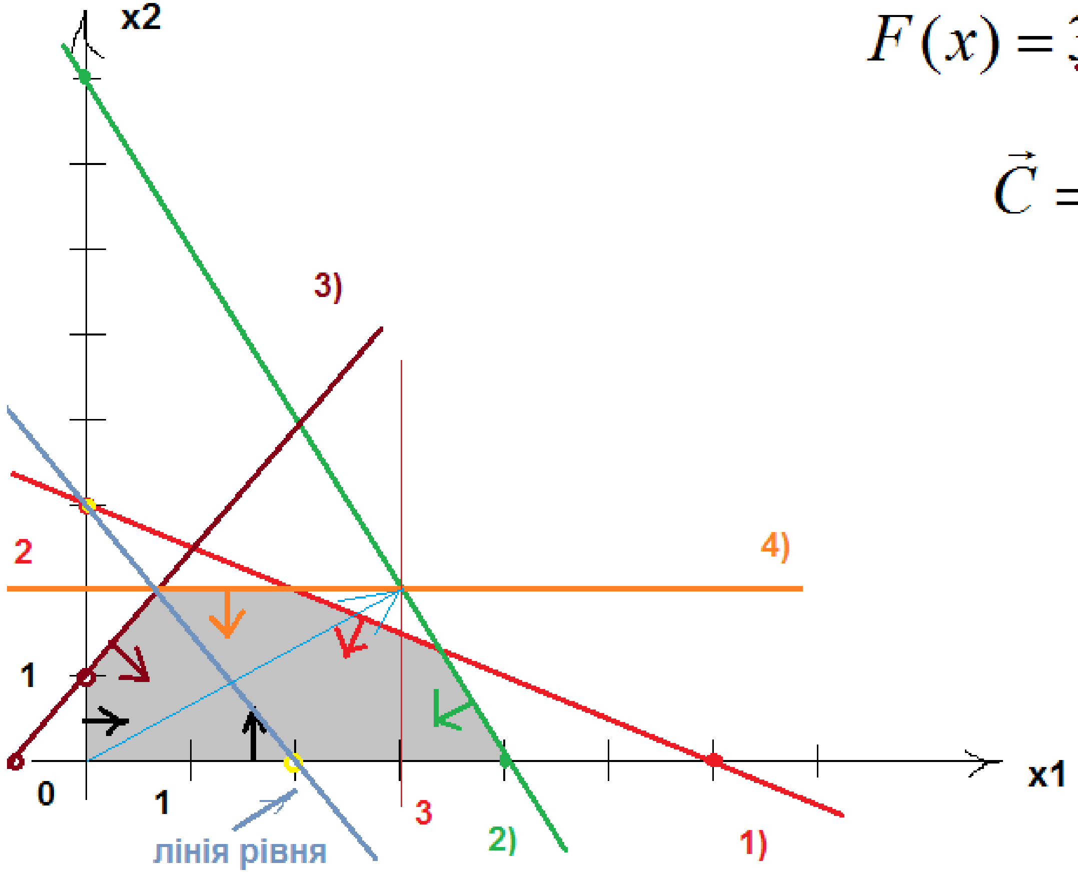
$$3x_1 + 2x_2 = 6$$

$$\begin{cases} x_1 = 0 \\ x_2 = 3 \end{cases} \begin{cases} x_1 = 2 \\ x_2 = 0 \end{cases}$$

Алгоритм розв’язання задачі граф. методом

V. Побудувати вектор, який починається в точці (0;0) і закінчується в точці (c_1,c_2).

Якщо лінія рівня і вектор побудовані вірно, то вони будуть ***перпендикулярні***.



$$F(x) = 3x_1 + 2x_2$$

$$\vec{C} = (3, 2)$$

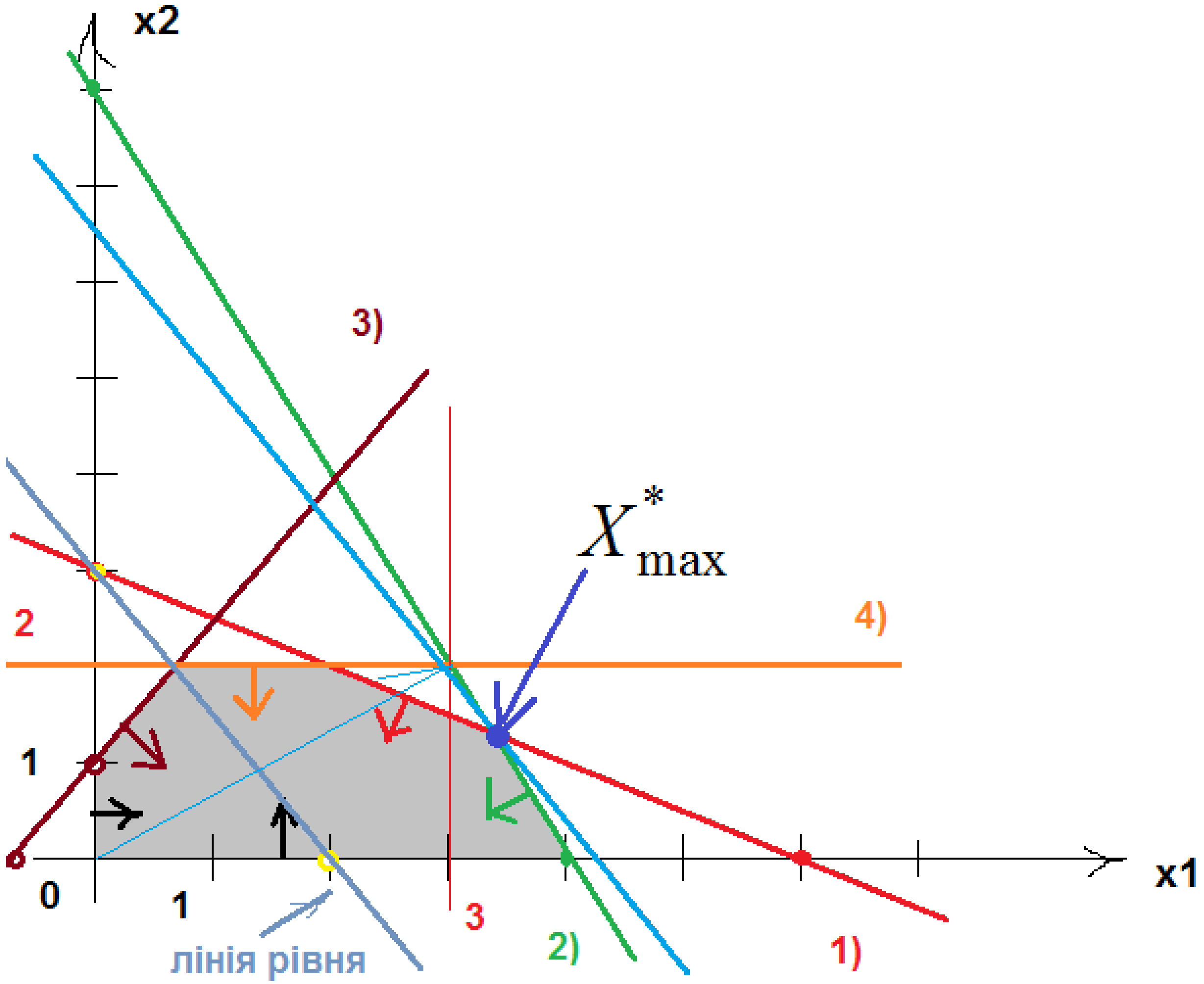
Алгоритм розв’язання задачі граф. методом

VI.

- При пошуку max функції мети пересувайте лінію рівня ***в напрямі*** вектора градієнта,

- при пошуку min функції мети – ***проти напряму*** вектора .
- ***Остання*** за ходом руху вершина ОДР буде точкою max або min функції мети.

- Якщо такої крапки (крапок) не існує, то зробіть висновок про ***необмеженість функції мети на множині планів*** зверху (при пошуку max) або знизу (при пошуку min).


x2
3)
X^*_{max}
2
4)
1
0
1
лінія рівня
3
2)
1)
x1

Алгоритм розв’язання задачі граф. методом

- Визначте координати точки max (min) функції мети
- Для обчислення координат оптимальної точки X* розв’яжіть систему рівнянь прямих, на перетині яких знаходиться X*

$$\begin{cases} x_1 + 2x_2 = 6 \\ 2x_1 + x_2 = 8 \end{cases}$$

$$x_1 = 6 - 2x_2$$

$$2(6 - 2x_2) + x_2 = 8$$

$$12 - 4x_2 + x_2 = 8$$

$$-3x_2 = -4$$

$$x_2 = \frac{4}{3} = 1\frac{1}{3}$$

$$x_1 = 6 - 2 \cdot \frac{4}{3} = \frac{18}{3} - \frac{8}{3} = \frac{10}{3} = 3\frac{1}{3}$$

$$X^* = \left(3\frac{1}{3}, 1\frac{1}{3}\right)$$

Алгоритм розв’язання задачі граф. методом

- Обчисліть значення функції мети L(X*).

$$F(x) = 3 \cdot 3\frac{1}{3} + 2 \cdot 1\frac{1}{3} =$$

$$= \frac{30}{3} + \frac{8}{3} = \frac{38}{3} = 12\frac{2}{3}$$

Розв’язання ЗЛП з використанням scipy.optimize

```
from scipy.optimize import linprog
import time
start = time.time()
c = [-3,-2] #Функция цели
A = [[1,2],[2,1],[-1,1],[0,1]]  #'A'
b = [6,8,1,2] #'b'
x0_bounds = (0, None)
x1_bounds = (0, None)
res = linprog(c, A_ub=A, b_ub=b, bounds=[x0_bounds, x1_bounds])
print(res)
stop = time.time()
print ("Time :")
print(stop - start)
```

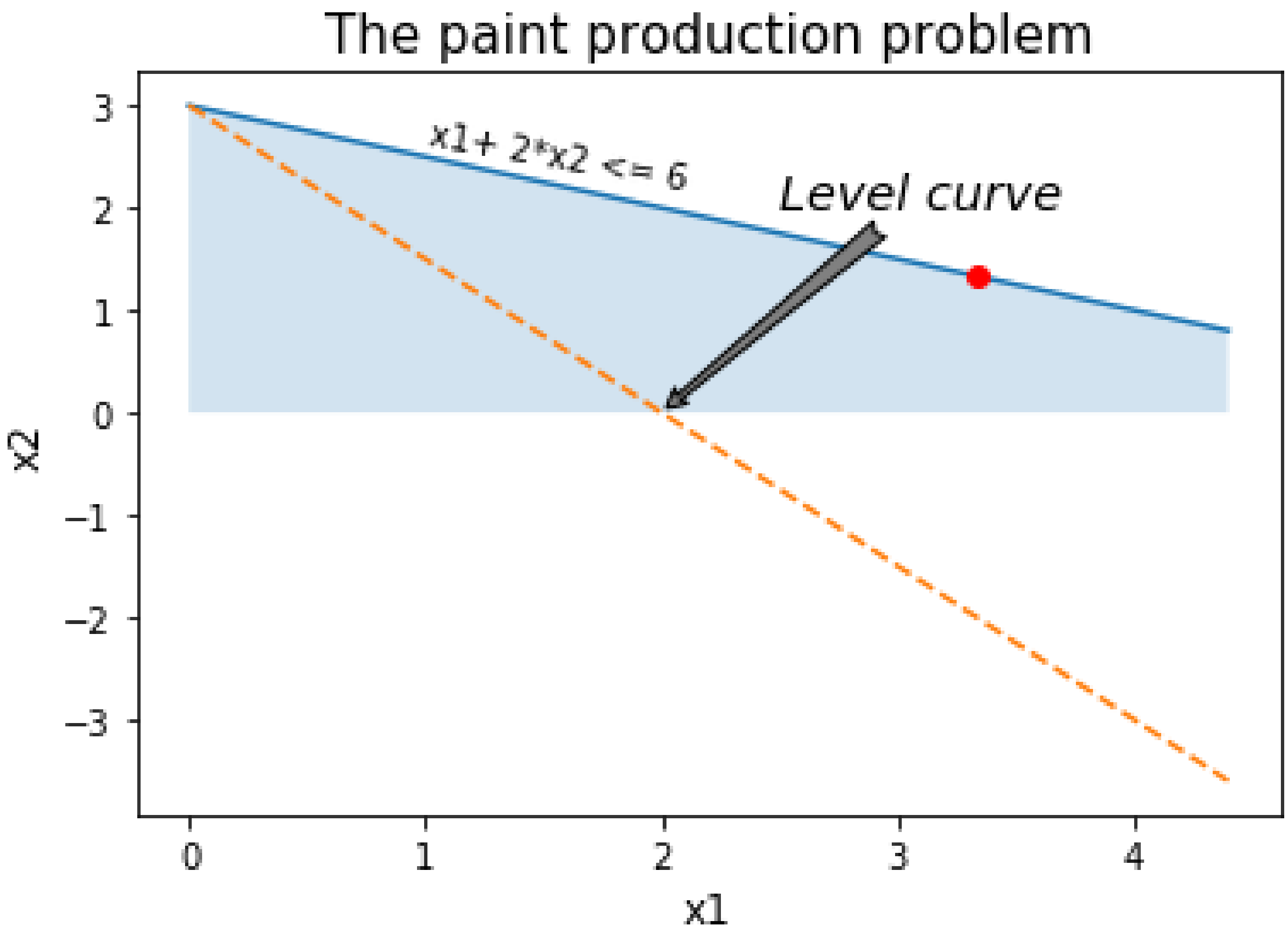
Розв’язання ЗЛП з використанням scipy.optimize

```
     con: array([], dtype=float64)
     fun: -12.666666666663677
 message: 'Optimization terminated successfully.'
     nit: 5
   slack: array([9.94582194e-12, 1.99360528e-11, 3.00000000e+00,
6.66666667e-01])
  status: 0
 success: True
       x: array([3.33333333, 1.33333333])
Time :
0.018000841140747
```

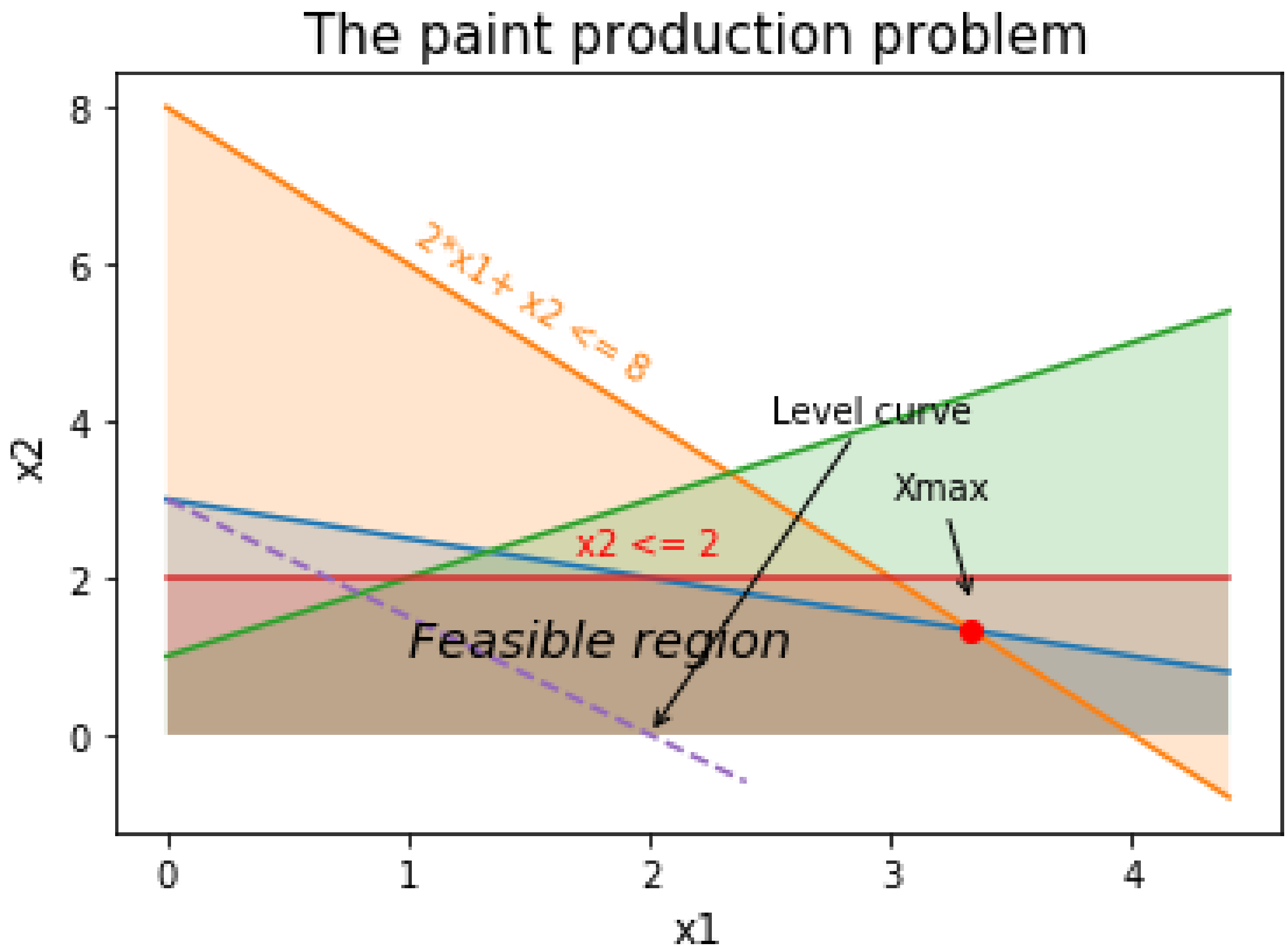
Створення графічного представлення

```
import numpy as np
import matplotlib.pyplot as plt
x = np.arange(0,4.5,0.1)
y = f(b=b[0],a1=A[0][0],x=x,a2=A[0][1])

plt.plot(x,y)
plt.fill_between(x,y, alpha=0.2)
plt.text(1, 2.2, 'x1+ 2*x2 <= 6', rotation=-10 % 360)
plt.plot(x,(c[0]*c[1]-np.abs(c[0])*x)/np.abs(c[1]),'--')

plt.annotate('Level curve', xy=(2, 0),
             xytext=(2.5, 2), size='x-large', style="italic",
             arrowprops=dict(facecolor='gray',arrowstyle="fancy"))

plt.plot(res.x[0], res.x[1], 'ro')

plt.title('The paint production problem', fontsize=15)
plt.xlabel('x1', fontsize=12)
plt.ylabel('x2', fontsize=12, color='black')
plt.savefig("paint.png", dpi=300)
plt.show;
```

Створення графічного представлення



Приблизний вигляд графічного рішення



Використання PuLP

```
from pulp import *
import time
start = time.time()
x1 = pulp.LpVariable("x1", lowBound=0)
x2 = pulp.LpVariable("x2", lowBound=0)
problem = pulp.LpProblem('0',LpMaximize)
problem += 3*x1 +2*x2, "Objective function"
problem += x1+ 2*x2 <= 6, "1-st constrain"
problem += 2*x1+ x2 <= 8, "2-d constrain "
problem += x2 - x1 <= 1, "3-d constrain "
problem += x2 <=2, "4th constrain "
problem.solve()
print ("Result:")
for variable in problem.variables():
    print (variable.name, "=", variable.varValue)
print ("Income:")
print (value(problem.objective))
stop = time.time()
print ("Time :")
print(stop - start)
```

```
Result:
x1 = 3.3333333
x2 = 1.3333333
Income:
12.666666500000002
Time :
0.152008771896362З
```

Дякую за увагу!