|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** ***ИУК «Информатика и управление»***

**КАФЕДРА** \_\_***ИУК4 «Программное обеспечение ЭВМ, информационные технологии»***

**ЛАБОРАТОРНАЯ РАБОТА №4**

**«Задачи целочисленного линейного программирования»**

**ДИСЦИПЛИНА: «Моделирование»**

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИУК4-72Б | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Карельский М.К. )  (Подпись) |
| Проверил: | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Никитенко У.В. )  (Подпись) |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | | |
|  | - Балльная оценка:  - Оценка: | |

Калуга, 2023

**Цель:** сформировать практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для стандартных вычислений при решении задач целочисленного линейного программирования на основе сравнения результатов.

**Задачи:** применить методы отсечений и комбинаторные методы к задаче целочисленного программирования, указанной в варианте, сравнить результаты, выдвинуть и обосновать гипотезу целесообразности использования того или иного подхода в зависимости от предложенной задачи и ее вариаций, точности результата, трудоемкости, сложности алгоритма, сложности обоснования применимости метода, вычислительной эффективности алгоритма.

**Вариант 7**

Найдите оптимальный план задачи целочисленного линейного программирования, используя

* первый алгоритм Гомори;
* второй алгоритм Гомори (x1 – произвольное, x2 – целое);
* метод ветвей и границ (решение проиллюстрируйте схемой).

**Решение:**

***Первый алгоритм Гомори:***

Канонический вид:

Симплекс-таблица:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | План |
| x1 | 1 | 0 | -5/21 | 2/21 | 32/21 |
| x2 | 0 | 1 | 2/21 | -5/21 | 46/21 |
| z | 0 | 0 | -1/3 | -2/3 | 40/3 |

Решение в произвольных числах:

Целые части:

Дробные части:

Дополнительные ограничения целочисленности:

Добавление строки:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | x5 | План |
| x1 | 1 | 0 | -5/21 | 2/21 | 0 | 32/21 |
| x2 | 0 | 1 | 2/21 | -5/21 | 0 | 46/21 |
| x5 | 0 | 0 | -16/21 | -2/21 | 1 | -11/21 |
| z | 0 | 0 | -1/3 | -2/3 | 0 | 40/3 |

Преобразование симплекс-таблицы:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | x5 | План |
| x1 | 1 | 0 | 0 | 1/8 | -5/16 | 27/16 |
| x2 | 0 | 1 | 0 | -1/4 | 1/8 | 17/8 |
| x3 | 0 | 0 | 1 | 1/8 | -21/16 | 11/16 |
| z | 0 | 0 | 0 | -5/8 | -7/16 | -167/16 |

Дробные части:

Дополнительные ограничения целочисленности:

Добавление строки:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | x5 | x6 | План |
| x1 | 1 | 0 | 0 | 1/8 | -5/16 | 0 | 27/16 |
| x2 | 0 | 1 | 0 | -1/4 | 1/8 | 0 | 17/8 |
| x3 | 0 | 0 | 1 | 1/8 | -21/16 | 0 | 11/16 |
| x6 | 0 | 0 | 0 | -1/8 | -11/16 | 1 | -11/16 |
| z | 0 | 0 | 0 | -5/8 | -7/16 | 0 | -167/16 |

Преобразование симплекс-таблицы:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | x5 | x6 | План |
| x1 | 1 | 0 | 0 | 2/11 | 0 | -5/11 | 2 |
| x2 | 0 | 1 | 0 | -3/11 | 0 | 2/11 | 2 |
| x3 | 0 | 0 | 1 | 4/11 | 0 | -21/11 | 2 |
| x5 | 0 | 0 | 0 | 2/11 | 1 | -16/11 | 1 |
| z | 0 | 0 | 0 | -6/11 | 0 | -7/11 | -14 |

Оптимальный целочисленный план:

***Второй алгоритм Гомори:***

Канонический вид:

Симплекс-таблица:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | План |
| x1 | 1 | 0 | -5/21 | 2/21 | 32/21 |
| x2 | 0 | 1 | 2/21 | -5/21 | 46/21 |
| z | 0 | 0 | -1/3 | -2/3 | 40/3 |

имеет дробную часть. Дополнительное ограничение:

Добавление строки:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | x5 | План |
| x1 | 1 | 0 | -5/21 | 2/21 | 0 | 32/21 |
| x2 | 0 | 1 | 2/21 | -5/21 | 0 | 46/21 |
| x5 | 0 | 0 | -2/21 | -20/357 | 1 | -4/21 |
| z | 0 | 0 | -1/3 | -2/3 | 0 | 40/3 |

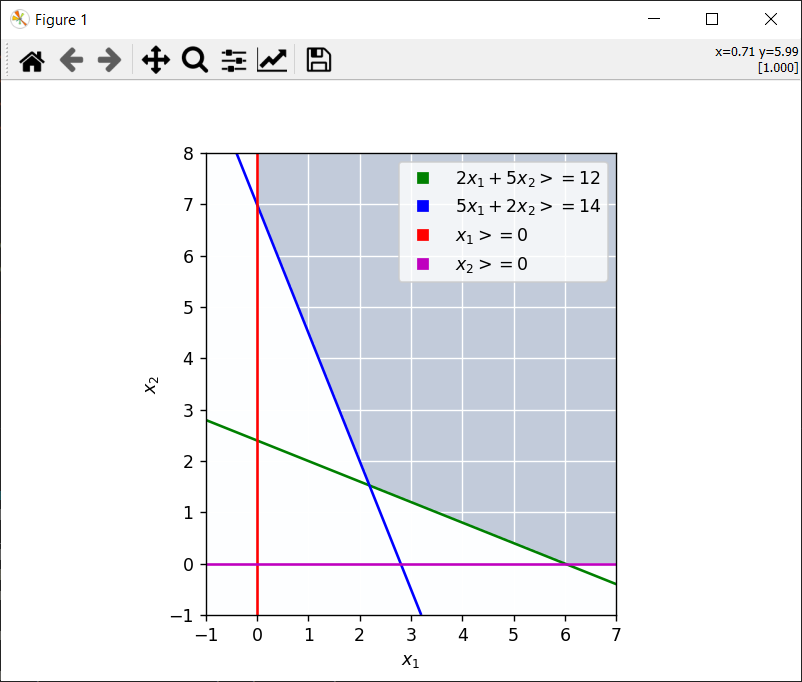
Преобразование симплекс-таблицы:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Базис | x1 | x2 | x3 | x4 | x5 | План |
| x1 | 1 | 0 | 0 | 4/17 | -5/2 | 2 |
| x2 | 0 | 1 | 0 | -5/17 | 1 | 2 |
| x3 | 0 | 0 | 1 | 10/17 | -21/2 | 2 |
| z | 0 | 0 | 0 | -8/17 | -7/2 | -14 |

Оптимальный план:

***Метод ветвей и границ:***

График системы:



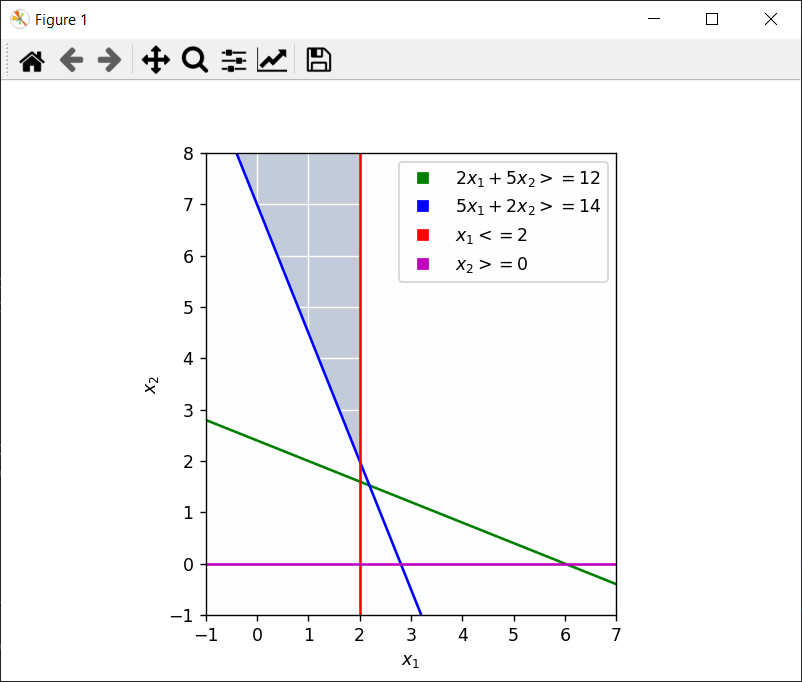
**Рис. 1.** График системы

Минимум достигается при

Разобьем задачу 1 на подзадачи 11 и 12:

* 11:
* 12:

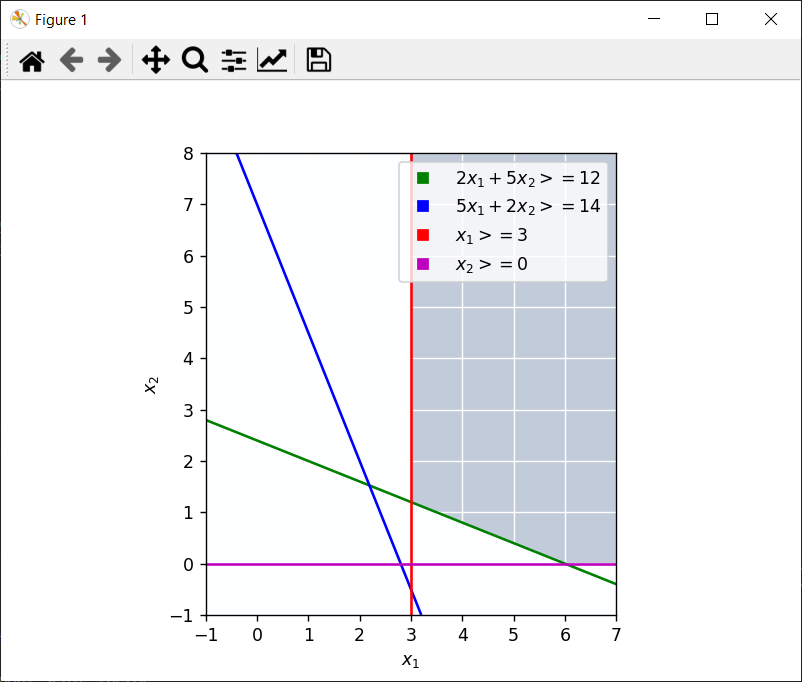
Задача 11:



**Рис. 2.** Задача 11

Решение:

Задача 12:



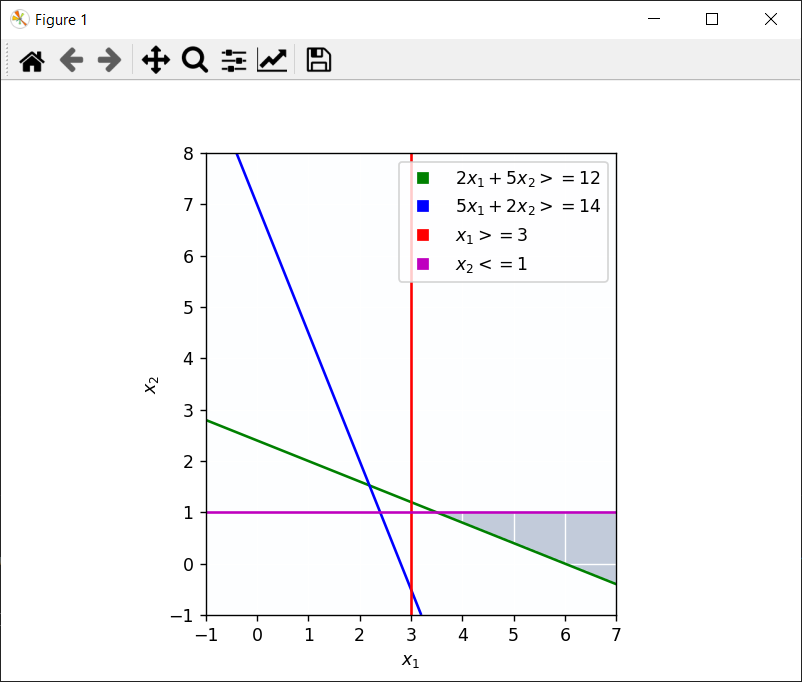
**Рис. 3.** Задача 12

Решение:

Разобьем задачу 12 на подзадачи 121 и 122:

* 121:
* 122:

Задача 121:



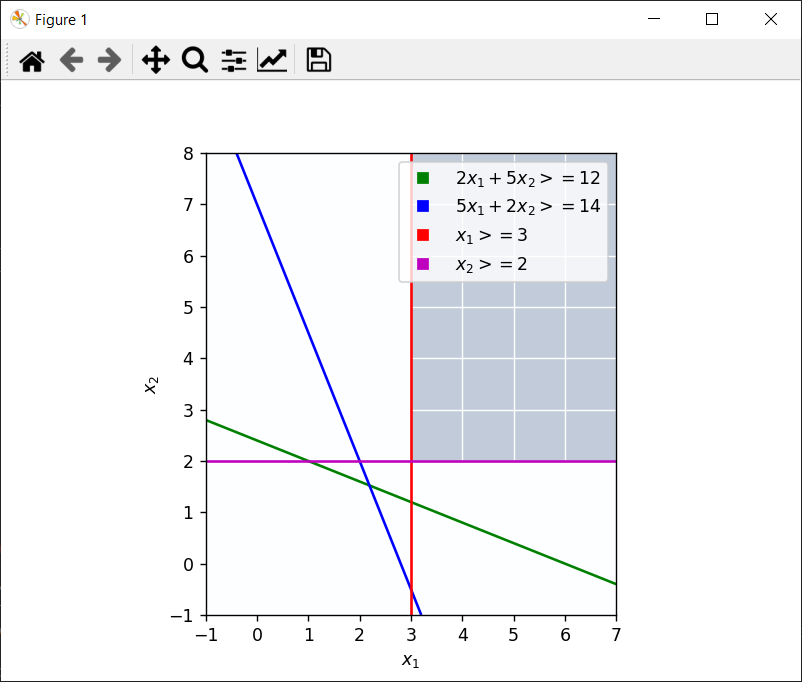
**Рис. 4.** Задача 121

Решение:

Разобьем задачу 121 на подзадачи 1211 и 1212:

* 1211:
* 1212:

Задача 122:



**Рис. 5.** Задача 122

Решение:

Оптимальный план:



**Рис. 6.** Метод ветвей и границ

**Вывод:** в ходе выполнения лабораторной работы были получены практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для стандартных вычислений при решении задач целочисленного линейного программирования на основе сравнения результатов.

**ПРИЛОЖЕНИЯ**

**Листинг:**

***LW4\_1.py:***

from docplex.mp.model import Model

m = Model()

x\_1 = m.integer\_var(name='x\_1', lb=0)

x\_2 = m.integer\_var(name='x\_2', lb=0)

m.add\_constraint(2 \* x\_1 + 5 \* x\_2 >= 12)

m.add\_constraint(5 \* x\_1 + 2 \* x\_2 >= 14)

m.minimize(3 \* x\_1 + 4 \* x\_2)

c = m.get\_cplex()

c.parameters.simplex.limits.iterations.set(100)

c.parameters.lpmethod.set(c.parameters.lpmethod.values.primal)

while c.solution.get\_status() != c.solution.status.optimal:

    c.solve()

    print("=== Симплекс-таблица ===")

    for tableau\_row in c.solution.advanced.binvarow():

        print(tableau\_row)

m.solve()

print("\n=== Решение задачи, где x\_1 - целое, а x\_2 - целое ===")

m.print\_solution()

***LW4\_2.py:***

from docplex.mp.model import Model

m = Model()

x\_1 = m.continuous\_var(name='x\_1', lb=0)

x\_2 = m.integer\_var(name='x\_2', lb=0)

m.add\_constraint(2 \* x\_1 + 5 \* x\_2 >= 12)

m.add\_constraint(5 \* x\_1 + 2 \* x\_2 >= 14)

m.minimize(3 \* x\_1 + 4 \* x\_2)

c = m.get\_cplex()

c.parameters.simplex.limits.iterations.set(100)

c.parameters.lpmethod.set(c.parameters.lpmethod.values.primal)

while c.solution.get\_status() != c.solution.status.optimal:

    c.solve()

    print("=== Симплекс-таблица ===")

    for tableau\_row in c.solution.advanced.binvarow():

        print(tableau\_row)

m.solve()

print("\n=== Решение задачи, где x\_1 - произвольное, а x\_2 - целое ===")

m.print\_solution()

***LW4\_3.py:***

import itertools

from functools import reduce

import matplotlib

import matplotlib.pyplot as plt

import numpy as np

import sympy

from matplotlib.ticker import MultipleLocator

if \_\_name\_\_ == '\_\_main\_\_':

    conditions = [

        lambda x\_1, x\_2: 2 \* x\_1 + 5 \* x\_2 >= 12,

        lambda x\_1, x\_2: 5 \* x\_1 + 2 \* x\_2 >= 14,

        lambda x\_1, x\_2: x\_1 >= 3,

        lambda x\_1, x\_2: x\_2 >= 2

    ]

    equalities = [

        lambda x\_1, x\_2: 2 \* x\_1 + 5 \* x\_2 - 12,

        lambda x\_1, x\_2: 5 \* x\_1 + 2 \* x\_2 - 14,

        lambda x\_1, x\_2: x\_1 - 3,

        lambda x\_1, x\_2: x\_2 - 2

    ]

    labels = [

        '$2 x\_1 + 5 x\_2 >= 12$',

        '$5 x\_1 + 2 x\_2 >= 14$',

        '$x\_1 >= 3$',

        '$x\_2 >= 2$'

    ]

    colors = ['g', 'b', 'r', 'm']

    x\_1\_bounds = (-1, 7)

    x\_2\_bounds = (-1, 8)

    x\_1\_range = np.linspace(x\_1\_bounds[0], x\_1\_bounds[1], 250)

    x\_2\_range = np.linspace(x\_2\_bounds[0], x\_2\_bounds[1], 250)

    x\_1s, x\_2s = np.meshgrid(x\_1\_range, x\_2\_range)

    axis: plt.Axes

    figure, axis = plt.subplots()

    axis.set\_xlim(\*x\_1\_bounds)

    axis.set\_ylim(\*x\_2\_bounds)

    handles = []

    for equality in equalities:

        axis.contour(

            x\_1s, x\_2s, equality(x\_1s, x\_2s), [0],

            colors=colors[equalities.index(equality)]

        )

        handles.append(

            matplotlib.lines.Line2D(

                [], [], color=colors[equalities.index(equality)],

                marker="s", ls="",

                label=labels[equalities.index(equality)]

            )

        )

    regions = [condition(x\_1s, x\_2s) for condition in conditions]

    intersection = np.array(reduce(lambda \_x, \_y: \_x & \_y, regions))

    extent = (x\_1s.min(), x\_1s.max(), x\_2s.min(), x\_2s.max())

    plt.imshow(

        intersection.astype(int),

        extent=extent,

        origin="lower",

        cmap="Blues",

        alpha=0.25

    )

    plt.xlabel("$x\_1$")

    plt.ylabel("$x\_2$")

    axis.xaxis.set\_major\_locator(MultipleLocator(1))

    axis.yaxis.set\_major\_locator(MultipleLocator(1))

    axis.grid(color='w', linestyle='-')

    plt.legend(handles=handles)

    plt.show()

    sym\_x\_1 = sympy.Symbol('x\_1')

    sym\_x\_2 = sympy.Symbol('x\_2')

    for equality\_1, equality\_2 in list(itertools.combinations(equalities, 2)):

        solution = sympy.solve(

            [

                equality\_1(sym\_x\_1, sym\_x\_2), equality\_2(sym\_x\_1, sym\_x\_2)

            ],

            [sym\_x\_1, sym\_x\_2], particular=True

        )

        x\_1 = solution[sym\_x\_1]

        x\_2 = solution[sym\_x\_2]

        if all(ineq(x\_1, x\_2) for ineq in conditions):

            print('Пересечение графиков ', end='')

            print(labels[equalities.index(equality\_1)], end=' и ')

            print(labels[equalities.index(equality\_2)])

            print(solution)