МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Кафедра информационных систем управления

Отчет по лабораторной работе №11
Вариант 2

Бовта Тимофея Анатольевича
студента 3 курса
специальности «прикладная математика»

Преподаватель:

Д. Ю. Кваша

Минск, 2024 г.

1 Постановка задачи

Сформулируйте заданную проблему как проблему удовлетворения ограничений. Определите переменные, домены и ограничения. Создайте модель. Решите задачу OR-Tools.

[1]:
```
from ortools.sat.python import cp_model
```

2 Общее условие

У вас есть сумка, которая может нести 20 кг. У вас есть набор вещей, которые вы хотите взять с собой, и их вес:

[2]:
```
items = ['book', 'jacket', 'washbag', 'computer', 'boots', 'alarmclock',␣
 ↪'anorak', 'food']
weight = [2, 4, 3, 8, 7, 1, 2, 6]
```

Сформулируем поставленную задачу как задачу целочисленного линейного программирования.

Каждый предмет обозначим отдельной переменной:

- $x_1 = \{\text{book}\}$;
- $x_2 = \{\text{jacket}\}$;
- $x_3 = \{\text{washbag}\}$;
- $x_4 = \{\text{computer}\}$;
- $x_5 = \{\text{boots}\}$;
- $x_6 = \{\text{alarmclock}\}$;
- $x_7 = \{\text{anorak}\}$;
- $x_8 = \{\text{food}\}$.

Также поставим условия

$$x_i \in \{0, 1\},$$

которые означают, что, поскольку у нас по одной единице каждой из вещей, то вещь мы можем либо взять, либо не взять.

Каждая вещь обладает своим весом, причем унести мы можем не больше 20 кг. Это позволяет сформулировать ограничения:

$$2x_1 + 4x_2 + 3x_3 + 8x_4 + 7x_5 + x_6 + 2x_7 + 6x_8 \leq 20.$$

3 Первое подзадание

3.1 Условие

Эти предметы имеют для вас определенную ценность:

[3]:
```
value = [6, 10, 8, 25, 22, 4, 5, 20]
```

Упакуйте предметы, которые вы можете носить в своей сумке, которые принесут вам максимально возможную общую ценность.

Нам нужно максимизировать ценность, соответственно это позволяет нам сформулировать целевую функцию для задачи ЦЛП:

$$6x_1 + 10x_2 + 8x_3 + 25x_4 + 22x_5 + 4x_6 + 5x_7 + 20x_8 \to \max\,.$$

В итоге, собрав все введенные формулы, мы имеем задачу ЦЛП вида:

$$6x_1 + 10x_2 + 8x_3 + 25x_4 + 22x_5 + 4x_6 + 5x_7 + 20x_8 \to \max,$$

$$2x_1 + 4x_2 + 3x_3 + 8x_4 + 7x_5 + x_6 + 2x_7 + 6x_8 \leq 20,$$

$$x_i \in \{0, 1\}.$$

Далее мы можем найти ее решение, применяя библиотеку OR-Tools.

3.2 Решение в OR-Tools

Составим функцию, применяя библиотеку OR-Tools, которая будет находить максимальное значение для задачи целочисленного программирования

[4]:
```
def maximize_value(items, weight, value):
    # Создание модели задачи
    model = cp_model.CpModel()

    # Создаем булевы переменные
    x = [model.NewBoolVar(f'x[{i}]') for i in range(len(items))]

    # Ограничения на вес
    model.Add(sum(weight[i] * x[i] for i in range(len(items))) <= 20)

    # Целевая функция
    objective = sum(value[i] * x[i] for i in range(len(items)))
    model.Maximize(objective)

    # Отыскание решения для модели солвером
    solver = cp_model.CpSolver()
    status = solver.Solve(model)

    if status == cp_model.OPTIMAL:
        selected_items = [items[i] for i in range(len(items)) if solver.
↪Value(x[i]) == 1]
        total_value = solver.ObjectiveValue()

        return selected_items, total_value
```

```
    return [], 0
```

Подставим в данную функцию известные нам значения и получим решение задачи:

[5]:
```
selected_items, total_value = maximize_value(items, weight, value)
print("Selected items:", selected_items)
print("Total value:", total_value)
```

```
Selected items: ['book', 'washbag', 'computer', 'alarmclock', 'food']
Total value: 63.0
```

4 Второе подзадание

Рюкзак также имеет ограниченную вместимость, а общий объем предметов, которые он может поместить внутрь, составляет 2000 см2. Каждый предмет имеет не только вес, но и объем:

[6]:
```
volume = [250, 500, 300, 250, 650, 130, 150, 600]
```

Найдите лучшее решение, как и в первой подзадаче, но общий объем предметов в рюкзаке не может быть больше вместимости рюкзака.

Данное условие позволяет нам сформулировать еще одно ограничение на поставленную задачу:

$$250x_1 + 500x_2 + 300x_3 + 250x_4 + 650x_5 + 130x_6 + 150x_7 + 600x_8 \leq 2000.$$

Таким образом, задача ЦЛП переформулируется и будет иметь следующий вид:

$$6x_1 + 10x_2 + 8x_3 + 25x_4 + 22x_5 + 4x_6 + 5x_7 + 20x_8 \to \max,$$

$$2x_1 + 4x_2 + 3x_3 + 8x_4 + 7x_5 + x_6 + 2x_7 + 6x_8 \leq 20,$$

$$250x_1 + 500x_2 + 300x_3 + 250x_4 + 650x_5 + 130x_6 + 150x_7 + 600x_8 \leq 2000,$$

$$x_i \in \{0, 1\}.$$

4.1 Решение в OR-Tools

Скопируем функцию из предыдущего подзадания, но добавим в нее новые ограничения:

[7]:
```
def maximize_value(items, weight, value, volume):
    # Создание модели задачи
    model = cp_model.CpModel()

    # Создаем булевы переменные
    x = [model.NewBoolVar(f'x[{i}]') for i in range(len(items))]

    # Ограничения на вес
    model.Add(sum(weight[i] * x[i] for i in range(len(items))) <= 20)

    # Ограничения на объем
    model.Add(sum(volume[i] * x[i] for i in range(len(items))) <= 2000)
```

```
    # Целевая функция
    objective = sum(value[i] * x[i] for i in range(len(items)))
    model.Maximize(objective)

    # Отыскание решения для модели солвером
    solver = cp_model.CpSolver()
    status = solver.Solve(model)

    if status == cp_model.OPTIMAL:
        selected_items = [items[i] for i in range(len(items)) if solver.
↪Value(x[i]) == 1]
        total_value = solver.ObjectiveValue()

        return selected_items, total_value

    return [], 0
```

[8]:
```
selected_items, total_value = maximize_value(items, weight, value, volume)
print("Selected items:", selected_items)
print("Total value:", total_value)
```

```
Selected items: ['book', 'washbag', 'computer', 'alarmclock', 'food']
Total value: 63.0
```

Таким образом, введение дополнительных условий не повлияло на решение задачи ЦЛП.

5 Третье подзадание

5.1 Условие

Некоторые вещи в сочетании стоят больше, а некоторые меньше. Вот дополнительный (или уменьшенный) балл, который вы получаете за каждую пару:

[9]:
```
extra_value = [    [0, 0, 0, -5, 0, 0, 0, 0],
                   [0, 0, 0, 0, 3, 0, -2, 0],
                   [0, 0, 0, 0, 0, 0, 0, 0],
                   [-5, 0, 0, 0, 0, -2, 0, 0],
                   [0, 3, 0, 0, 0, 0, 0, 0],
                   [0, 0, 0, -2, 0, 0, 0, 0],
                   [0, -2, 0, 0, 0, 0, 0, 0],
                   [0, 0, 0, 0, 0, 0, 0, 0]    ]
```

Если extra_value[i1, i2] = 3 и предметы i1 и i2 находятся в вашей сумке, то общая стоимость вашей сумки увеличивается на 3. Естественно, если extra_value[i1, i2] = -2, то она уменьшается на 2. Расширьте свою модель для первой подзадачи, чтобы максимизировать общую сумму с измененными значениями.

Эти дополнительные условия позволяют переформулировать целевую функцию. Но для этого

введем некоторые обозначения:

- $x = (x_1, \ldots, x_8)^T$ – вектор всех предметов, $x_i \in \{0, 1\}$;
- $\mathcal{V}_{extra}$ – матрица дополнительного балла за каждую пару данная по условию;
- ω – вектор весов данный в общем условии;
- v – вектор ценности данный в первом подзадании;
- ν – вектор ценности данный во втором подзадании.

В итоге задача ЦЛП станет иметь вид

$$v^T x + \mathcal{V}_{extra}(x^T x) \to \max,$$
$$\omega^T x \leq 20,$$
$$\nu^T x \leq 2000,$$
$$x_i \in \{0, 1\}.$$

5.2 Решение в OR-Tools

Возьмем функцию из второго подзадания и добавим к целевой функции новое условие:

[10]:
```python
def maximize_value(items, weight, value, volume, extra_value):
    # Создание модели задачи
    model = cp_model.CpModel()

    # Создаем булевы переменные
    x = [model.NewBoolVar(f'x[{i}]') for i in range(len(items))]

    # Ограничения на вес
    model.Add(sum(weight[i] * x[i] for i in range(len(items))) <= 20)

    # Ограничения на объем
    model.Add(sum(volume[i] * x[i] for i in range(len(items))) <= 2000)

    # Целевая функция
    objective = sum(value[i] * x[i] for i in range(len(items)))
    for i1 in range(len(items)):
        for i2 in range(len(items)):
            var = model.NewBoolVar('')
            model.AddBoolAnd([x[i1], x[i2]]).OnlyEnforceIf(var)
            objective += extra_value[i1][i2] * var
    model.Maximize(objective)

    # Отыскание решения для модели солвером
    solver = cp_model.CpSolver()
    status = solver.Solve(model)

    if status == cp_model.OPTIMAL:
```

```
        selected_items = [items[i] for i in range(len(items)) if solver.
↪Value(x[i]) == 1]
        total_value = solver.ObjectiveValue()

        return selected_items, total_value

    return [], 0
```

[11]:
```
selected_items, total_value = maximize_value(items, weight, value, volume,␣
↪extra_value)
print("Selected items:", selected_items)
print("Total value:", total_value)
```

```
Selected items: ['jacket', 'computer', 'boots', 'alarmclock']
Total value: 67.0
```

Таким образом, результат изменился по сравнению с предыдущем подзаданием.

В итоге, собрав все условия в единую задачу, мы получили решение:

$$\max\left(v^T x + \mathcal{V}_{extra}(x^T x)\right) = 67, \quad x = (0,1,0,1,1,1,0,0)^T.$$