Паралельна реалізація симплекс-методу

Визначення симплекс методу

Симплекс-метод - це алгоритм, що використовується при вирішенні оптимізаційної задачі лінійного програмування. Лінійне програмування - це розділ математики, що займається вирішенням екстремальних задач (знаходження екстремуму функції) на множині простору, заданому системою лінійних рівнянь та нерівностей. Оптимізація - задача виявлення мінімуму або максимуму (екстремума) цільової функції. Цільова функція - це функція декількох змінних, підлягає оптимізації в цілях вирішення будь-якої оптимізації завдання (наприклад, задачі об'ємного планування).

Спочатку вихідну задачу лінійного програмування приводять до канонічного вигляду, потім складають симплекс-таблицю вигляду:

базис	B	x_1	…	x_k	…	x_j	…	x_n
…	…				…			
x_i	b_i			a_{ik}		a_{ij}		
…								
x_r	b_r			a_{rk}	…	a_{rj}		
…	b_m				…			
$f(x)$	0					$-c_j = -\Delta_j$		

де в стовпці «базис» вказуються базисні змінні, а в останньому рядку стовпця «базис» пишеться f(x). У стовпець «B» записуються вільні члени обмежень bi і значення цільової функції (на 1-му етапі воно дорівнює 0, тобто ніякого прибутку).

У стовпчиках xj для НЕ базисних змінних вказуються коефіцієнти при не базисних змінних з обмежень задачі. У стовпчиках базисних змінних міститься тільки 0 або 1 на перетині стовпця з відповідним рядком базисної змінної. В останньому рядку -cj - це коефіцієнти при змінних цільової функції взяті з протилежним знаком.

Симплекс-таблиця складена, тепер опишемо сам симплекс-метод.

Крок 1: Виконується перевірка отриманого базисного плану на оптимальність за умовою: якщо при будь-якому ДБР (допустиме базисне рішення) в симплекс-таблиці всі коефіцієнти рядка f (x) (тобто -cj) не від’ємні, то дане ДБР оптимальне, отже КІНЕЦЬ рішення. В іншому випадку:

Крок 2: Перехід до нового базисного плану. Для цього з числа небазисних змінних з негативними значеннями в останньому рядку (тобто -cj<0) вибирається змінна, що вводиться в базис - xk, це змінна якої відповідає найбільша за модулем негативна оцінка:

$$|\Delta_k| = \max |\Delta_j| , \quad \Delta_j < 0,$$

$$\Delta_k = \min \Delta_j$$

Стовпець, що відповідає змінній x_k, називається головним, або ведучим. Елементи даного стовпця позначаються через a_{ik}. Якщо виявиться кілька однакових найбільших по модулю негативних оцінок, то вибирається будь-яка з відповідних змінних.

Крок 3: Вибираємо змінну r - змінну, яка виводиться з базису. Дана змінна знаходиться зі співвідношення:

$$\frac{b_r}{a_{rk}} = \min_j \frac{b_i}{a_{ik}}, \; a_{ik} \geq 0$$

Рядок таблиці, в якій отримано найменше відношення елемента стовпця «В» до відповідного позитивного елементу ведучого стовпця, є провідною, або головною. Елементи головного рядка позначаються через a_{rj}. Обрана змінна x_r буде виводитися з базису, тобто вона виключається. Якщо виявиться кілька однакових найменших значень відношень, то вибирається будь-яка з відповідних їм змінних. Елемент, який стоїть на перетині головного стовпчика і рядка називається головним, або ведучим, і позначається a_{rk}.

Крок 4: Для визначення нового базисного плану проводиться перерахунок елементів симплекс-таблиці, і результати заносяться в нову таблицю. Вибрані змінні серед базисних і не базисних, що лежать на головному рядку і головному стовпці, міняються місцями.

Процедура перерахунку елементів виконується наступним чином:

а) елементи головного рядка необхідно поділити на провідний елемент:

$$b_k = \frac{b_r}{a_{rk}}; \quad a_{kj} = \frac{a_{rj}}{a_{rk}}$$

б) елементи отриманого рядка множаться на $-a_{ik}$, і результати сумуються з i-тим рядком, причому $i \neq k$:

$$a_{ij}^{'} = a_{ij} - \frac{a_{rj} \cdot a_{ik}}{a_{rk}}$$

$$b_i^{'} = b_i - \frac{b_r \cdot a_{ik}}{a_{rk}}$$

$$f^{'}(x) = f(x) - \frac{b_r \cdot \Delta_k}{a_{rk}}$$

$$\Delta_j^{'} = \Delta_j - \frac{a_{rj} \cdot \Delta_k}{a_{rk}}$$

Після визначення нової симплекс-таблиці переходять до кроку 1.

Приклад табличного симплекс методу

Необхідно вирішити задачу лінійного програмування. Цільова функція:

$2x_1+5x_2+3x_3+8x_4 \rightarrow min$

Обмежуючі умови:

$3x_1+6x_2-4x_3+x_4\leq12$
$4x_1-13x_2+10x_3+5x_4\geq6$
$3x_1+7x_2+x_3\geq1$

Приведемо систему обмежень до канонічного виду, для цього необхідно перейти від нерівностей до рівності, з додаванням додаткових змінних. Так як наша задача - завдання мінімізації, то нам необхідно перетворити її до задачі на пошук максимуму. Для цього змінимо знаки коефіцієнтів цільової функції на протилежні. Елементи першої нерівності записуємо без змін, додавши до нею додаткову змінну x5 і змінивши знак "≤" на "=". Оскільки, друга і третя нерівності мають знаки "≥" необхідно поміняти знаки їх коефіцієнтів на протилежні і внести в них додаткові змінні x6 і x7 відповідно. В результаті отримали еквівалентну задачу:

$3x_1+6x_2-4x_3+x_4+x_5=12$
$-4x_1+13x_2-10x_3-5x_4+x_6=-6$
$-3x_1-7x_2-x_3+x_7=-1$

Переходимо до формування вихідної симплекс таблиці. У рядок F таблиці заносяться коефіцієнти цільової функції з протилежним знаком.

	x1	**x2**	**x3**	**x4**	**Вільний член**
F	2	5	3	8	0
X5	3	6	-4	1	12
X6	-4	13	**-10**	-5	-6
X7	-3	-7	-1	0	-1

У складеній нами таблиці є від’ємні елементи в стовпці вільних членів, знаходимо серед них максимальний за модулем - це елемент: -6, він задає провідний рядок - X6. У цьому рядку також знаходимо максимальний по модулю негативний елемент: -10 він знаходиться в стовпці X3 який буде ведучим стовпцем. Змінна в провідній рядку виключається з базису, а змінна відповідна ведучому Стовпцю включається в базис. Перерахуємо симплекс-таблицю:

	X1	X2	X6	X4	Вільний член
F	0.8	8.9	0.3	6.5	-1.8
X5	4.6	0.8	-0.4	3	14.4
X3	0.4	-1.3	-0.1	0.5	0.6
X7	-2.6	-**8.3**	-0.1	0.5	-0.4

У складеній нами таблиці є від’ємні елементи в стовпці вільних членів, знаходимо серед них максимальний за модулем - це елемент: -0.4, він задає провідний рядок - X7. У цьому рядку також знаходимо максимальний по модулю негативний елемент: -8.3 він знаходиться в стовпці X2 який буде ведучим стовпцем. Змінна в провідному рядку виключається з базису, а змінна відповідна ведучому стовпцю включається в базис. Перерахуємо симплекс-таблицю:

	X1	X7	X6	X4	Вільний член
F	-1.988	1.072	0.193	7.036	-2.229
X5	4.349	0.096	-0.41	3.048	14.361
X3	0.807	-0.157	-0.084	0.422	0.663
X2	**0.313**	-0.12	0.012	-0.06	0.048

Так як в стовпці вільних членів немає негативних елементів, то знайдено допустиме рішення. В рядку F є негативні елементи, це означає що отримане рішення не оптимальне. Визначимо провідний стовпець. Для цього знайдемо в рядку F максимальний по модулю від’ємний елемент - це -1.988. Провідним рядком буде той для якого відношення вільного члена до відповідного елементу ведучого стовпця мінімальне. Провідним рядком є X2, а ведучий елемент: 0.313.

	X2	X7	X6	X4	Вільний член
F	6.351	0.31	0.269	6.655	-**1.924**
X5	-13.895	1.763	-0.577	3.882	13.694
X3	-2.578	0.152	-0.115	0.577	0.539
X1	3.195	-0.383	0.038	-0.192	0.153

Так як в рядку F немає від’ємних елементів, то знайдено оптимальне рішення. Так як вихідним завданням був пошук мінімуму, то оптимальним рішенням буде вільний член рядка F, взятий з протилежним знаком. F = 1.924 при значеннях змінних рівних: x3 = 0.539, x1 = 0.153. Змінні x2 і x4 не входять до базис, тому x2 = 0 і x4 = 0.

Програмна реалізація симплекс-методу

Наведемо програмну реалізацію симплекс-методу на мові програмування C#. Вхідні дані: симплекс-таблиця без базисних змінних в стовпцях. Тобто таблиця повинна бути побудована тільки за коефіцієнтами при змінних з обмежень задачі та цільової функції. Формат вхідних даних: двовимірний масив з елементів типу double.

Вхідні дані передаються як аргумент, при створенні екземпляра класу Simplex. При виклику методу Calculate як аргумент ви повинні передати одновимірний масив з елементів типу double, довжиною в кількість змінних в цільової функції. В цей масив будуть записані знайдені значення невідомих. Вихідні дані: метод Calculate повертає посилання на двовимірний масив, що містить розв'язану симплекс-таблицю. Крім того рішення буде занесене в масив, переданий в якості аргументу в метод Calculate. Формат вихідних даних: двовимірний масив з елементів типу double і одновимірний масив з елементів типу double.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Simplex
{
    public class Simplex
    {
        //source - симплекс таблиця без базисних змінних
        double[,] table; //симплекс таблиця

        int m, n;

        List<int> basis; //список базисних змінних

        public Simplex(double[,] source)
        {
            m =source.GetLength(0);
            n = source.GetLength(1);
            table = new double[m, n + m - 1];
            basis = new List<int>();

            for (int i = 0; i < m; i++)
            {
                for (int j = 0; j < table.GetLength(1); j++)
                {
                    if (j < n)
                        table[i, j] = source[i, j];
                    else
                        table[i, j] = 0;
                }
                //виставляємо коефіцієнт 1 перед базисною змінною в рядку
                if ((n + i) < table.GetLength(1))
                {
                    table[i, n + i] = 1;
                    basis.Add(n + i);
                }
            }
```

```
    n = table.GetLength(1);
}

//result – в цей масив будуть записані отримані значення X
public double[,] Calculate(double[] result)
{
    int mainCol, mainRow; //провідні стовпець і рядок

    while (!IsItEnd())
    {
        mainCol = findMainCol();
        mainRow = findMainRow(mainCol);
        basis[mainRow] = mainCol;

        double[,] new_table = new double[m, n];

        for (int j = 0; j < n; j++)
            new_table[mainRow, j] = table[mainRow, j] / table[mainRow, mainCol];

        for (int i = 0; i < m; i++)
        {
            if (i == mainRow)
                continue;

            for (int j = 0; j < n; j++)
                new_table[i, j] = table[i, j] - table[i, mainCol] * new_table[mainRow, j];
        }
        table = new_table;
    }

    //заносимо в result знайдені значення X
    for (int i = 0; i < result.Length; i++)
    {
        int k = basis.IndexOf(i + 1);
        if (k != -1)
            result[i] = table[k, 0];
        else
            result[i] = 0;
    }

    return table;
}

private bool IsItEnd()
{
    bool flag = true;

    for (int j = 1; j < n; j++)
    {
        if (table[m - 1, j] < 0)
        {
            flag = false;
            break;
        }
    }

    return flag;
}

private int findMainCol()
{
```

```
            int mainCol = 1;

            for (int j = 2; j < n; j++)
                if (table[m - 1, j] < table[m - 1, mainCol])
                    mainCol = j;

            return mainCol;
        }

        private int findMainRow(int mainCol)
        {
            int mainRow = 0;

            for (int i = 0; i < m - 1; i++)
                if (table[i, mainCol] > 0)
                {
                    mainRow = i;
                    break;
                }

            for (int i = mainRow + 1; i < m - 1; i++)
                if ((table[i, mainCol] > 0) && ((table[i, 0] / table[i, mainCol]) < (table[mainRow, 0] / table[mainRow,
mainCol])))
                    mainRow = i;

            return mainRow;
        }


    }
}
```

Наведемо також приклад програми, що використовує реалізований вище симплекс-метод. Вирішимо, для прикладу, завдання з такими обмеженнями і цільовою функцією:

$$f(x_1, x_2) = 6x_1+5x_2 \rightarrow \max$$

Обмежуючі умови:

$$-3x_1+5x_2\le 25$$
$$-2x_1+5x_2\le 30$$
$$x_1\le 10$$
$$3x_1-8x_2\le 6$$
$$x_1,x_2\ge 0$$

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace testSimplex
{
    class Program
    {
        static void Main(string[] args)
        {
```

```
            double[,] table = { {25, -3,  5},
                                {30, -2,  5},
                                {10,  1,  0},
                                { 6,  3, -8},
                                { 0, -6, -5} };

            double[] result = new double[2];
            double[,] table_result;
            Simplex S = new Simplex(table);
            table_result = S.Calculate(result);

            Console.WriteLine("Розв’язок симплекс-таблиці:");
            for (int i = 0; i < table_result.GetLength(0); i++)
            {
                for (int j = 0; j < table_result.GetLength(1); j++)
                    Console.Write(table_result[i, j] + " ");
                Console.WriteLine();
            }

            Console.WriteLine();
            Console.WriteLine("Розв’язок:");
            Console.WriteLine("X[1] = " + result[0]);
            Console.WriteLine("X[2] = " + result[1]);
            Console.ReadLine();
        }
    }
}
```

Завдання

Розпаралелити за даними виконання алгоритму на етапі обчислення значень нової симплекс-таблиці (всіх рядків крім провідного), а саме виконання *кроку 4-б)* алгоритму. Додати примітиви синхронізації для зупинки всіх потоків, що виконують *крок 4-б)*, щоб заповнення симплекс-таблиці завершилось до моменту переходу на новий крок алгоритму. Кожний потік має заповнити nRow/threadCount рядків симплекс-таблиці, де nRow – загальна кількість рядків таблиці, а threadCount – задана кількість потоків. Рекомендовано вибирати threadCount рівним кількості процесорів.

1.	$\max F(x) = x_1 + 4x_2$ $x_1 + 2\cdot x_2 \ge 2$ $-x_1 + x_2 \le 3$ $x_1 + x_2 \le 7$ $2\cdot x_1 + x_2 \le 10$ $x_1 \ge 0, x_2 \ge 0$	2.	$\max F(x) = x_1 + x_2$ $x_1 + 3\cdot x_2 \ge 4$ $3x_1 + x_2 \ge 4$ $4\cdot x_1 + x_2 \le 24$ $x_1 + 4\cdot x_2 \le 24$ $x_1 \ge 0, x_2 \ge 0$	3.	$\max F(x) = 2\cdot x_1 - x_2$ $x_1 + 3\cdot x_2 \ge 7$ $2\cdot x_1 - x_2 \ge 0$ $6\cdot x_1 + 10\cdot x_2 \le 78$ $x_1 - 2\cdot x_2 \le 2$ $x_1 \ge 0, x_2 \ge 0$
4.	$\max F(x) = x_1 - 4x_2$ $x_1 + 2\cdot x_2 \ge 4$ $-x_1 + 2\cdot x_2 \le 10$ $x_2 \le 6$ $3\cdot x_1 + 2\cdot x_2 \le 24$ $x_1 \ge 0, x_2 \ge 0$	5.	$\min F(x) = x_1 - 4x_2$ $4\cdot x_{2_1} - x_3 + x_4 = 14$ $-3\cdot x_1 - 2\cdot x_2 + 2\cdot x_3 + x_4 = 2$ $x_2 \le 6$ $3\cdot x_1 + 2\cdot x_2 \le 24$ $x_i \ge 0, i = 1..4$	6.	$\max F(x) = 5\cdot x_1 + x_2$ $x_1 - 3\cdot x_2 \le 1$ $x_1 + x_2 \ge 5$ $x_2 \le 8$ $2\cdot x_1 + x_2 \le 16$ $x_1 \ge 0, x_2 \ge 0$

7. $\max F(x) = 2\cdot x_1 + x_2$ $-x_1 + 2\cdot x_2 \ge 0$ $x_1 + 2\cdot x_2 \ge 4$ $-x_1 + x_2 \le 5$ $x_1 + x_2 \le 9$ $x_1 \ge 0, x_2 \ge 0$	8. $\min F(x) = -x_1 + x_2$ $x_1 + x_2 \ge 2$ $x_1 - 2\cdot x_2 \le 3$ $-2\cdot x_1 + x_2 \le 4$ $2\cdot x_1 + 3\cdot x_2 \le 20$ $x_1 \ge 0, x_2 \ge 0$	9. $\min F(x) = -x_1 + x_2$ $x_1 + x_2 \ge 2$ $x_1 - 2\cdot x_2 \le 3$ $-2\cdot x_1 + x_2 \le 4$ $2\cdot x_1 + 3\cdot x_2 \le 20$ $x_1 \ge 0, x_2 \ge 0$
10. $\min F(x) = 20\cdot x_1 - 13\cdot x_2 - 8\cdot x_3 + 5\cdot x_4$ $4\cdot x_1 + x_2 \ge 8$ $7\cdot x_2 + x_3 + 2\cdot x_4 = 42$ $7\cdot x_1 - 2\cdot x_3 + 3\cdot x_4 = 28$ $2\cdot x_1 - x_2 \le 12$ $x_1 \ge 0, x_2 \ge 0$	11. $\max F(x) = x_2$ $4\cdot x_1 + x_2 \ge 8$ $-2\cdot x_1 + 3\cdot x_2 \le 10$ $x_1 + 2\cdot x_2 \le 16$ $2\cdot x_1 - x_2 \le 12$ $x_1 \ge 0, x_2 \ge 0$	12. $\min F(x) = 5\cdot x_1 - 12\cdot x_2 + 7\cdot x_3 - 4\cdot x_4$ $x_1 + 2\cdot x_2 \ge 4$ $x_1 \le 8$ $x_1 + x_3 + x_4 = 11$ $-x_1 + 2\cdot x_2 - x_3 + x_4 = 3$ $x_i \ge 0, i = 1..4$