МИНОБРНАУКИ РОССИИ
ФГБОУ ВО «СГУ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

ПРАКТИЧЕСКИЕ ЗАДАНИЯ ПО КУРСУ «НЕЙРОННЫЕ СЕТИ»

ОТЧЕТ О ПРАКТИКЕ

студента 5 курса 531 группы
направления 10.05.01 — Компьютерная безопасность
факультета КНиИТ
Токарева Никиты Сергеевича

Проверил
доцент _____ И. И. Слеповичев

Саратов 2023

1 Создание ориентированного графа

1.1 Описание задачи №1

На вход: текстовый файл с описанием графа в виде списка дуг:

$$(a_1, b_1, n_1), (a_2, b_2, n_2), \ldots (a_k, b_k, n_k),$$

где a_i – начальная вершина дуги i, b_i – конечная вершина дуги i ($a_i \neq b_i$), n_i – порядковый номер дуги в списке всех заходящих в вершину b_i дуг. Т.е. допустим в ориентированном графе будут заданы дуги, например (a_1, b_1) и (a_2, b_1), тогда в описании данного графа будут заданы дуги (a_1, b_1, n_1) и (a_2, b_1, n_2), где номера этих дуг упорядоченны и различны.

На выходе:

а) Ориентированный граф с именованными вершинами и линейно упорядоченными дугами (в соответствии с порядком из текстового файла). Структура графа должна записываться в файл формата XML или JSON.

б) Сообщение об ошибке в формате файла, если ошибка присутствует.

1.2 Примеры исполнения программы

Пример №1:

Рассмотрим ориентированный граф, который имеет следующую запись:

$$(v1, v2, 1), (v3, v4, 1), (v2, v5, 1), (v5, v6, 1), (v5, v7, 1),$$
$$(v4, v8, 1), (v4, v9, 1), (v9, v6, 2)$$

Этот граф можно представить в следующем виде, как показано на рисунке 1.

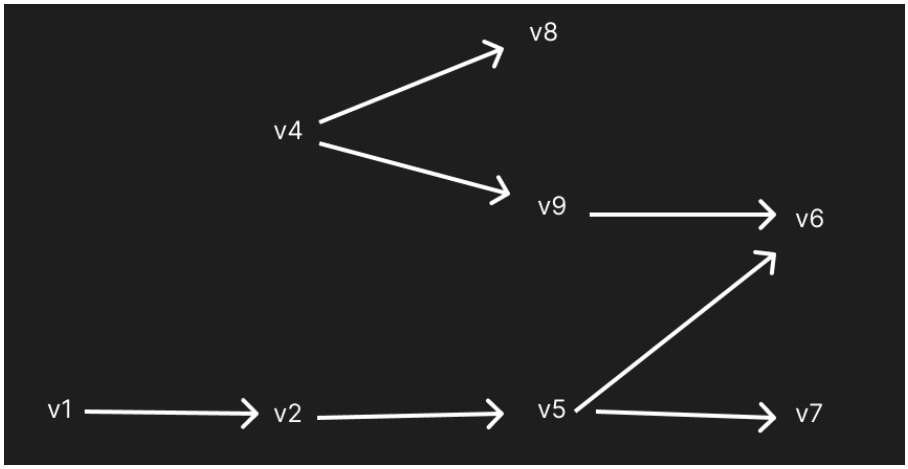

Рисунок 1 – Изображение ориентированного графа

Тогда после выполнении программы результатом будет являться следующая структура, которая будет записана в файл формата JSON.

```
{
    "graph":
    {
        "vertex":
        [
            "v1", "v2", "v3", "v4", "v5",
            "v6", "v7", "v8", "v9"
        ],
        "arc":
        [
            {"from": "v1", "to": "v2", "order": 1},
            {"from": "v3", "to": "v4", "order": 1},
            {"from": "v2", "to": "v5", "order": 1},
            {"from": "v5", "to": "v6", "order": 1},
            {"from": "v5", "to": "v7", "order": 1},
            {"from": "v4", "to": "v8", "order": 1},
            {"from": "v4", "to": "v9", "order": 1},
            {"from": "v9", "to": "v6", "order": 2}
        ]
    }
}
```

Пример 2:

Рассмотрим ориентированный граф, который имеет следующую запись:

$$(a, b, 1), (c, d, 1), (b, e, 1), (d, e, 1), (e, f, 1)$$

На рисунке 2 показано изображение данного графа.

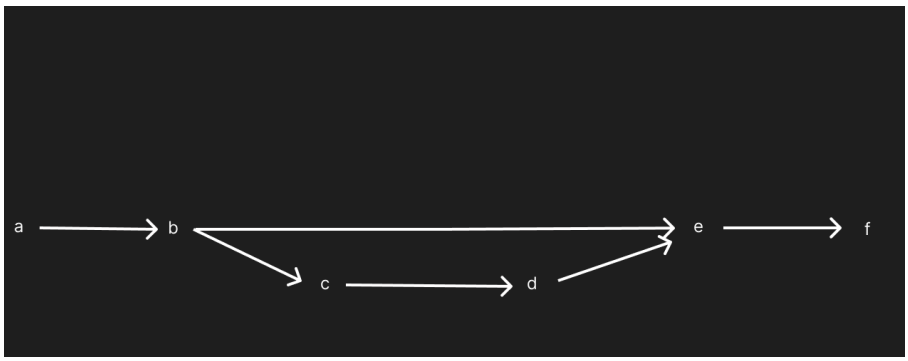

Рисунок 2 – Изображение ориентированного графа

Исходя из рисунка, можно заметить, что такой вариант графа вполне допустим, однако в описании дуг была допущена ошибка: дуги $(b, e, 1)$ и $(d, e, 1)$ имеют одинаковый порядковый номер. В таком случае программа выводит сообщение об ошибке, что в графе некорректно заданы номера.

2 Создание функции по графу

2.1 Описание задачи №2

На входе: ориентированный граф с именованными вершинами как описано в задании 1.

На выходе: линейное представление функции, реализуемой графом в префиксной скобочной записи:

$$A_1(B_1(C_1(\dots), \dots, C_m(\dots)), \dots, B_n(\dots))$$

Способ проверки резульата:

а) выгрузка в текстовый файл результата преобразования графа в имя функции.

б) сообщение о наличии циклов в графе, если они присутствуют.

2.2 Примеры исполнения программы

Пример 1:

Рассмотрим ориентированный граф, который имеет следующую запись:

$$(v1, v2, 1), (v3, v2, 2), (v2, v4, 1), (v4, v5, 1)$$

На рисунке 3 показано изображение данного графа.

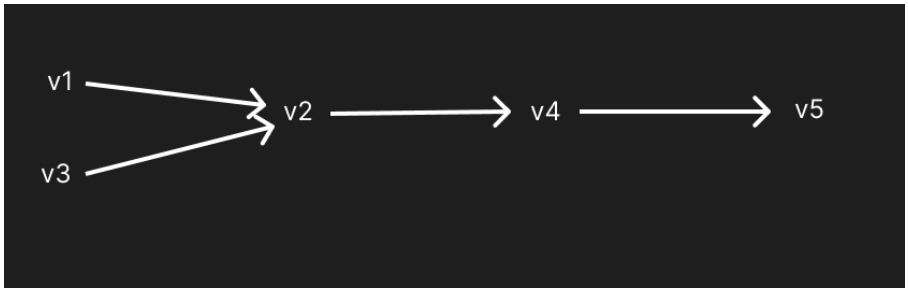

Рисунок 3 – Изображение ориентированного графа

В результате работы программы в файл будет сделана следующая запись:

$$v5(v4(v2(v1(), v3())))$$

Пример 2:

Рассмотрим ориентированный граф, который имеет следующую запись:

$$(v1, v2, 1), (v2, v3, 1), (v3, v1, 1)$$

На рисунке 4 показано изображение данного графа.

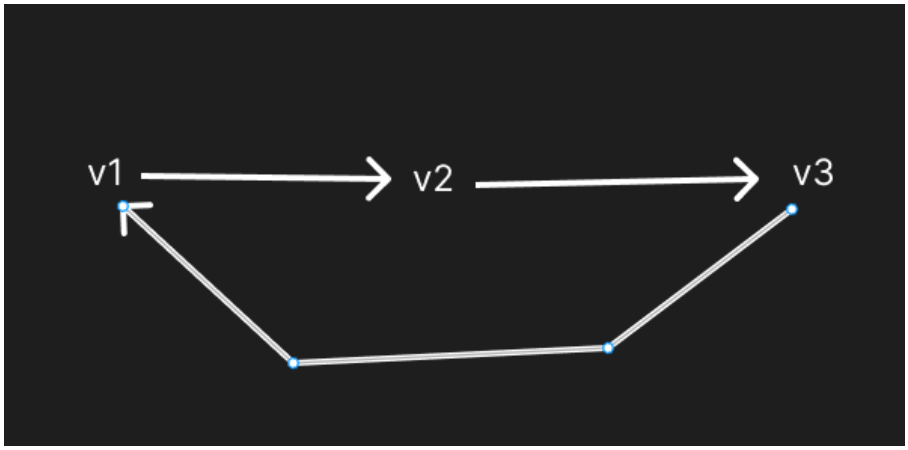

Рисунок 4 – Изображение ориентированного графа

Исходя из рисунка, видно, что в данном графе присутствует цикл. Поэтому результатом программы будет вывод об ошибке в консоль.

3 Вычисление значение функции на графе

3.1 Описание задачи №3

На входе:

а) Текстовый файл с описанием графа в виде списка дуг (смотри задание 1).

б) Текстовый файл соответствий арифметических операций именам вершин:

$$a_1 : 1\text{-я операция}$$
$$a_2 : 2\text{-я операция}$$
$$\dots$$
$$a_n : n\text{-я операция},$$

где a_i – имя i-й вершины, i-я операция – символ операции, соответствующий вершине a_i.

Допустимы следующие символы операций:

$+$ – сумма значений,

$*$ – произведение значений,

exp – экспонирование входного значения,

число – любая числовая константа.

На выходе: значение функции, построенной по графу а) и файлу б).

Способ проверки результата: результат вычисления, выведенный в файл.

3.2 Примеры исполнения программы

Пример 1:

Рассмотрим ориентированный граф, который имеет следующую запись:

$$(v1, v2, 1), (v1, v2, 2), (v2, v6, 1), (v3, v5, 1), (v4, v5, 2), (v6, v7, 1), (v5, v7, 2)$$

Также имеются соответствия арифметических операций именам вершинам, записанных в файле формата JSON:

```
{
    "v1" : 1,
    "v2" : "+",
    "v3" : 5,
    "v4" : 12,
```

```
        "v5" : "*",
        "v6" : "exp",
        "v7" : "+"
    }
```

На рисунке 5 показано изображение данного графа.

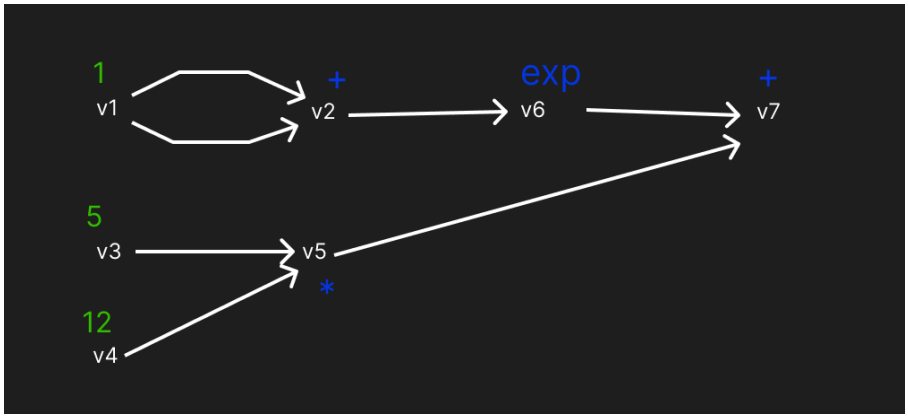

Рисунок 5 – Изображение ориентированного графа

После всех проверок входных данных на корректность, программа представляет ориентированный граф в следующем виде:

$$v7(v6(v2(v1(), v1())), v5(v3(), v4()))$$

Подставив в соответствие именам вершин арифметические операции получается следующая запись:

$$+(exp(+(1, 1)), *(5, 12))$$

Таким образом программе необходимо вычислить выражение, представленное в префиксной записи. В итоге программа записывает в текстовый файл результат данного выражения: 67.38905609893065.

Пример 2:

Теперь рассмотрим ориентированный граф, который имеет достаточную простую запись:

$$(v1, v3, 1), (v2, v3, 2)$$

Также имеются следующие соответствия:

```
{
    "v1" : 1,
    "v2" : "+",
    "v3" : 5
}
```

На рисунке 6 показано изображение данного графа.

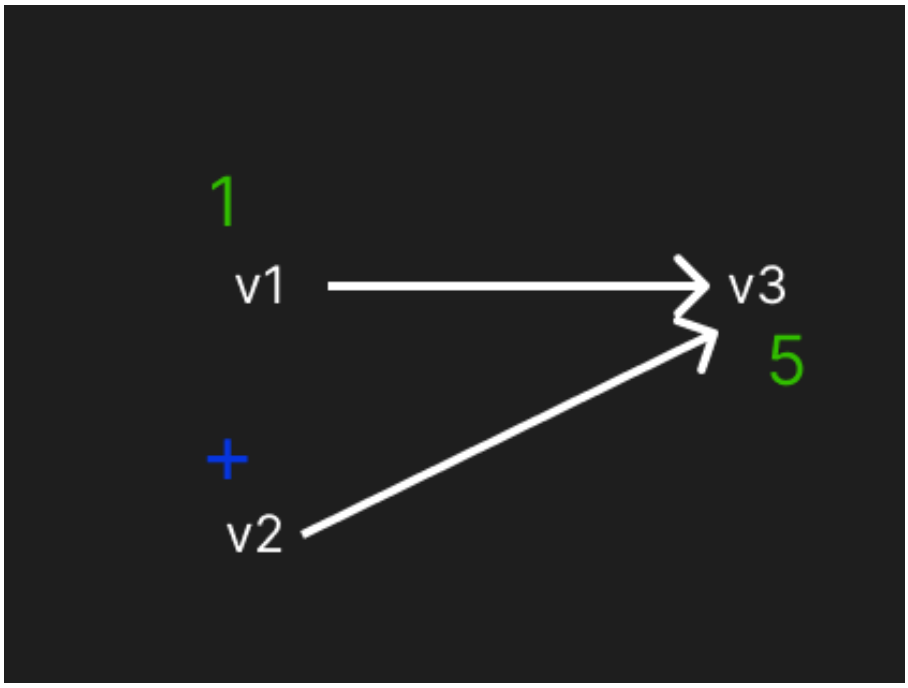

Рисунок 6 – Изображение ориентированного графа

Исходя из рисунка и текущих соответствий, можно заметить, что в данном случае невозможно будет вычислить результат. Поэтому программа выдает в качестве ответа в консоль сообщение об ошибке.

4 Построение многослойной нейронной сети

4.1 Описание задачи №4

На входе:

а) Текстовый файл с набором матриц весов межнейронных связей:

$$M_1 : [a^1_{11}, a^1_{12}, \ldots, a^1_{1n_1}], \ldots, [a^1_{m_1 1}, a^1_{m_1 2}, \ldots, a^1_{m_1 n_1}]$$
$$M_2 : [a^2_{11}, a^2_{12}, \ldots, a^2_{1n_2}], \ldots, [a^2_{m_2 1}, a^2_{m_2 2}, \ldots, a^2_{m_2 n_2}]$$
$$\ldots$$
$$M_p : [a^p_{11}, a^p_{12}, \ldots, a^p_{1n_p}], \ldots, [a^p_{m_p 1}, a^p_{m_p 2}, \ldots, a^p_{m_p n_p}]$$

б) Текстовый файл с входным вектором в формате:

$$x_1, x_2, \ldots, x_k.$$

На выходе:

а) Сериализованная многослойная нейронная сеть (в формате XML или JSON) с полносвязной межслойной структурой. Файл с выходным вектором – результатом вычислений НС в формате:

$$y_1, y_2, \ldots, y_k.$$

б) Сообщение об ошибке, если в формате входного вектора или файла описания НС допущена ошибка.

4.2 Примеры исполнения программы

5 Реализация метода обратного распространения ошибки для многослойной НС

5.1 Описание задачи №5

На входе:

а) Текстовый файл с описанием НС (формат см. в задании 4).

б) Текстовый файл с обучающей выборкой:

$$[x_{11}, x_{12}, \ldots, x_{1n}] \to [y_{11}, y_{12}, \ldots, y_{1m}]$$
$$\ldots$$
$$[x_{k1}, x_{k2}, \ldots, x_{kn}] \to [y_{k1}, y_{k2}, \ldots, y_{km}]$$

Формат описания входного вектора x и выходного вектора y соответствует формату из задания 4.

в) Число итераций обучения (в строке параметров).

На выходе: Текстовый файл с историей N итераций обучения методом обратного распространения ошибки:

1 : 1-я ошибка

2 : 2-я ошибка

...

N : N-я ошибка