ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Теорія алгоритмів

ЛАБОРАТОРНА РОБОТА №4

Тема: «Композиції алгоритмів. Алгоритмічні системи. Нормальні алгоритми Маркова»

Виконав:

Ст. *Лук'янчук Денис*

Група ПМІ-23

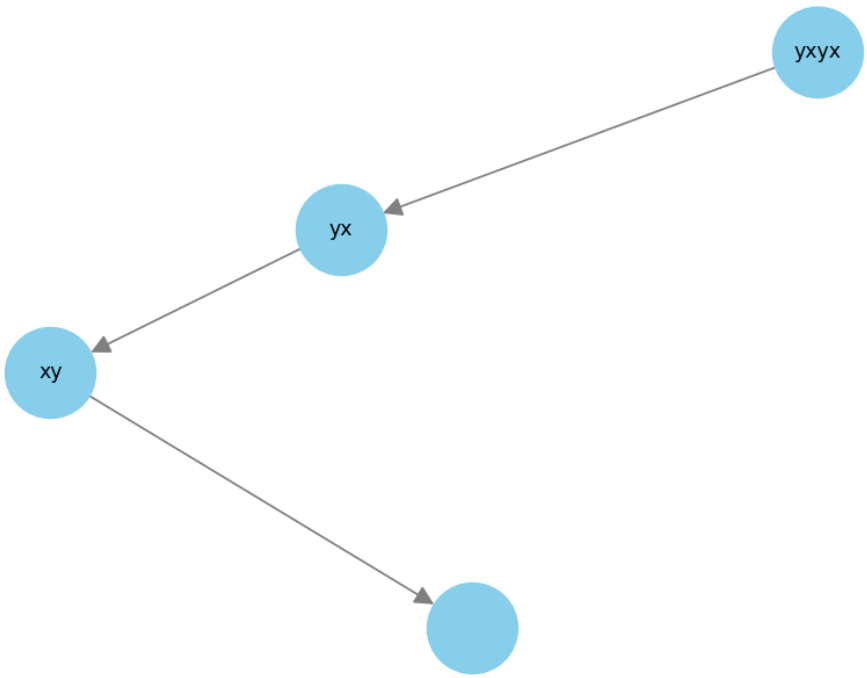
2025

Тема: «Композиції алгоритмів. Алгоритмічні системи. Нормальні алгоритми Маркова»

Мета роботи: вивчмти та практично застосувати композиції алгоритмів, нормальних алгоритмів Маркова, а також алгоритмів для маніпуляцій зі строками та числами, заданими ланцюжками символів. Зокрема, передбачається дослідження способів композиції алгоритмів, таких як суперпозиція, об'єднання, розгалуження та ітерація, а також реалізація операцій віднімання, додавання, множення, видалення повторних символів та пропусків за допомогою нормальних алгоритмів Маркова.

Хід роботи:

1. **Завдання 1-2: Побудова графа та перетворення слів за допомогою підстановок**:
    - Для реалізації цього завдання створено функцію apply_substitution(), яка застосовує підстановки до заданого слова. Зокрема, вона замінює підрядки "xy" на порожній рядок (видаляє їх) та "yx" на "xy".
    - Функція process_queue() виконує переробку всіх слів у черзі, застосовуючи підстановки та додаючи нові слова до черги, якщо вони ще не відвідані.
    - Для відстеження етапів перетворення слів створено функцію generate_word_sequence(), яка генерує послідовність слів після кожного перетворення.
    - Для візуалізації графа змін слів використовуються бібліотеки networkx і matplotlib. Граф показує, як відбувається перетворення слів за допомогою підстановок.



2. **Завдання 3: Віднімання чисел (A - B)**:
    - Для реалізації алгоритму віднімання створено набір правил у вигляді списку кортежів, де кожен кортеж містить шаблон та відповідну підстановку.
    - Функція apply_rules() по черзі застосовує ці підстановки до слова, поки не досягне стабільного стану (коли неможливо застосувати більше правил).
    - Вхідним словом є ланцюжок з символів "1", розділений символом "#", що представляє число AA та BB. Алгоритм по черзі застосовує правила, спрощуючи це слово до остаточного результату.

```
Віднімання 5 - 3:
Крок 1: 11111#111
Крок 2: 1111#11
Крок 3: 111#1
Крок 4: 11#
Крок 5: 11
```

3. **Завдання 4: Додавання чисел (A + B)**:
    - Алгоритм для додавання схожий на алгоритм віднімання, але замість зменшення значення додається символ "1" в результаті кожного кроку, коли застосовується правило підстановки '1#' → '11'.
    - Функція apply_rules() застосовує ці правила до введеного слова, що представляє число AA та BB, і поетапно додає їх, змінюючи значення.

```
Додавання 4 + 5:
Крок 1: 1111#11111
Крок 2: 1111111111
```

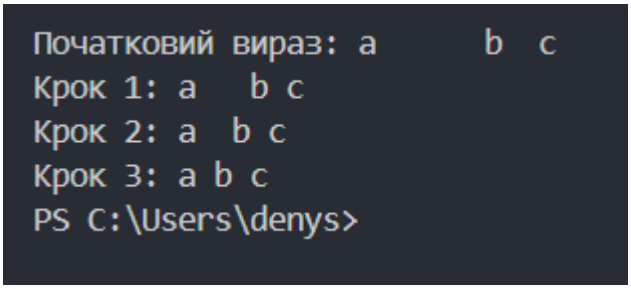
4. **Завдання 5: Множення чисел (A * B)**:
    - Для множення чисел створено набір правил Маркова, які імітують операцію множення в алфавіті, що складається з символів 1,∗,Т,Ф1, *, Т, Ф.
    - Кожне правило має на меті замінити певну комбінацію символів на іншу, поки ланцюжок не буде приведений до фінального результату.
    - Функція markov_multiplication() реалізує множення за допомогою цих правил, і виводить кожен крок, що демонструє перетворення числа 3 на 4 в унарній системі числення.

```
Початковий вираз: 111*1111
Крок 1: 111T*111
Крок 2: 111TT*11
Крок 3: 111TTT*1
Крок 4: 111TTTT
Крок 5: 11T1FTTT
Крок 6: 1T1F1FTTT
Крок 7: T1F1F1FTTT
Крок 8: T1F1F1TFTT
Крок 9: T1F1FT1FFTT
Крок 10: T1F1TF1FFTT
Крок 11: T1FT1FF1FFTT
Крок 12: T1TF1FF1FFTT
Крок 13: TT1FF1FF1FFTT
Крок 14: TT1FF1FF1FTFT
Крок 15: TT1FF1FF1TFFT
Крок 16: TT1FF1FFT1FFFT
Крок 17: TT1FF1FTF1FFFT
Крок 18: TT1FF1TFF1FFFT
Крок 19: TT1FFT1FFF1FFFT
Крок 20: TT1FTF1FFF1FFFT
Крок 21: TT1TFF1FFF1FFFT
Крок 22: TTT1FFF1FFF1FFFT
Крок 23: TTT1FFF1FFF1FFTF
Крок 24: TTT1FFF1FFF1FTFF
Крок 25: TTT1FFF1FFF1TFFF
Крок 26: TTT1FFF1FFFT1FFFF
Крок 27: TTT1FFF1FFTF1FFFF
Крок 28: TTT1FFF1FTFF1FFFF
Крок 29: TTT1FFF1TFFF1FFFF
Крок 30: TTT1FFFT1FFFF1FFFF
Крок 31: TTT1FFTF1FFFF1FFFF
Крок 32: TTT1FTFF1FFFF1FFFF
Крок 33: TTT1TFFF1FFFF1FFFF
Крок 34: TTTT1FFFF1FFFF1FFFF
Крок 35: TTTT1FFF1FFFFF1FFFF
Крок 36: TTTT1FF1FFFFFF1FFFF
Крок 37: TTTT1F1FFFFFFF1FFFF
Крок 38: TTTT11FFFFFFFF1FFFF
Крок 39: TTTT11FFFFFFF1FFFFF
Крок 40: TTTT11FFFFFF1FFFFFF
Крок 41: TTTT11FFFFF1FFFFFFF
Крок 42: TTTT11FFFF1FFFFFFFF
Крок 43: TTTT11FFF1FFFFFFFFF
Крок 44: TTTT11FF1FFFFFFFFFF
Крок 45: TTTT11F1FFFFFFFFFFF
Крок 46: TTTT111FFFFFFFFFFFF
Крок 47: TTTT11FFFFFFFFFFFF
Крок 48: TTTT1FFFFFFFFFFF
Крок 49: TTTTFFFFFFFFFFF
```

5. **Завдання 6: Видалення всіх входжень, крім першого, літери 'a' з рядка**:
    - Створено алгоритм, який видаляє всі входження літери 'a' з рядка, окрім першого.
    - Функція markov_remove_a() по черзі перевіряє, чи є в рядку літера 'a', і якщо вона зустрічається, замінює її на порожній рядок, за винятком першого входження.

```
Початковий вираз: bacaba
Крок 1: bacaba
Крок 2: bcaba
Крок 3: bcba
Крок 4: bcb
```

6. **Завдання 7: Видалення повторних пропусків в рядках**:
    - Алгоритм для видалення повторних пропусків використовує підстановку для заміни подвійних пробілів на один.
    - Функція markov_remove_duplicate_spaces() застосовує це правило до рядка, поки не буде усунуто всі зайві пробіли.



Пояснення кроків виконання:

1. Для кожного завдання спочатку визначено набір правил для заміни символів, що відповідає визначеним операціям (віднімання, додавання, множення, видалення символів тощо).
2. Далі визначено функцію, яка обробляє ці правила та поетапно застосовує їх до вхідного слова.
3. Для демонстрації роботи алгоритмів на конкретних прикладах використовуються вхідні слова (наприклад, ланцюжки символів "1", символи для множення та інші).
4. У результаті кожного кроку виводяться зміни слова, що допомагає зрозуміти, як алгоритм працює на практиці.
5. Для завдань, де застосовується граф, використовується бібліотека networkx для побудови та візуалізації графа, який показує перетворення слів.

Ці кроки дозволяють візуалізувати та зрозуміти принципи роботи алгоритмів Маркова при обробці ланцюжків символів для виконання арифметичних операцій і маніпуляцій з рядками.

Висновок: У ході лабораторного заняття було детально вивчено нормальні алгоритми Маркова та їх застосування для обробки рядків і чисел. Було розглянуто основні принципи композиції алгоритмів через суперпозицію та об’єднання, використовуючи підстановки для трансформації слів. Досліджено арифметичні операції, такі як віднімання, додавання та множення, реалізовані через нормальні алгоритми Маркова, де числа представлені ланцюжками символів.

Завдяки цьому заняттю було наочно продемонстровано, як за допомогою простих правил можна виконувати складні операції на текстових даних. Зокрема, реалізовано алгоритми для видалення повторних символів та пробілів у рядках, що показало важливість застосування таких методів у обробці тексту. Таким чином, виконання лабораторної роботи сприяло глибшому розумінню принципів роботи з алгоритмами Маркова, їх використанню для вирішення конкретних задач, а також значенню графів для візуалізації алгоритмічних процесів.