

Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики

Кафедра дискретного аналізу та інтелектуальних систем

Лабораторна робота №8
АЛГОРИТМ МАРКОВА
з курсу “Алгоритми та структури даних”

Виконав:
студент групи ПМІ-13
Лук'янчук Денис
Євгенійович

Львів – 2024

Алгоритм Маркова — система послідовних застосувань, підстановок, які реалізують певні процедури отримання нових слів із базових, які побудовані на певному алфавіті. У 1956 році вітчизняним математиком А.А. Марковим було запропоновано нове уточнення поняття алгоритму, яке пізніше було названо його ім'ям.

Алгоритм виконання:

1. Створення класу MarkovAlgorithm: Оголошуємо клас MarkovAlgorithm, який містить приватне поле rules типу vector<pair<string, string>>. Це вектор пар рядків, де перший рядок є лівою частиною правила, а другий - правою.
2. Додавання правил: Визначаємо метод addRule, який дозволяє додавати нові правила до класу MarkovAlgorithm.
3. Виконання алгоритму Маркова: Визначаємо метод run, який застосовує алгоритм Маркова до вхідного рядка. Алгоритм шукає усі можливі ліві частини правил у вхідному рядку і замінює їх правими частинами, поки можливо.
4. Головна функція main: В функції main створюється об'єкт MarkovAlgorithm, додається правило ab -> ba, вхідний рядок встановлюється як abab, і потім викликається метод run, щоб обробити вхідний рядок.

Приклад №1

Дано: input = abba (умова: ab -> ba)

```
Input: abba
Output: bbaa
```

Приклад №2

Дано: input = abba (умова: ab -> ba, ba -> ac)

```
Input: abba
Output: acca
```

Приклад Unit-тесту(без помилок)

```
[=====] Running 3 tests from 1 test suite.
[-----] Global test environment set-up.
[-----] 3 tests from MarkovAlgorithmTest
[ RUN      ] MarkovAlgorithmTest.SimpleRuleApplication
[ OK       ] MarkovAlgorithmTest.SimpleRuleApplication (0 ms)
[ RUN      ] MarkovAlgorithmTest.NoRuleApplication
[ OK       ] MarkovAlgorithmTest.NoRuleApplication (0 ms)
[ RUN      ] MarkovAlgorithmTest.MultipleRuleApplication
[ OK       ] MarkovAlgorithmTest.MultipleRuleApplication (0 ms)
[-----] 3 tests from MarkovAlgorithmTest (1 ms total)

[-----] Global test environment tear-down
[=====] 3 tests from 1 test suite ran. (3 ms total)
[ PASSED ] 3 tests.
```

Приклад Unit-тесту(з помилкою)

```
[=====] Running 3 tests from 1 test suite.
[-----] Global test environment set-up.
[-----] 3 tests from MarkovAlgorithmTest
[ RUN      ] MarkovAlgorithmTest.SimpleRuleApplication
C:\...\Test_8\Test_8.cpp(43): error: Expected equality of these values:
  ma.run("abba")
    Which is: "acca"
  "baca"
[ FAILED   ] MarkovAlgorithmTest.SimpleRuleApplication (0 ms)
[ RUN      ] MarkovAlgorithmTest.NoRuleApplication
C:\...\Test_8\Test_8.cpp(51): error: Expected equality of these values:
  ma.run("abcd")
    Which is: "accd"
  "abcd"
[ FAILED   ] MarkovAlgorithmTest.NoRuleApplication (0 ms)
[ RUN      ] MarkovAlgorithmTest.MultipleRuleApplication
C:\...\Test_8\Test_8.cpp(60): error: Expected equality of these values:
  ma.run("abab")
    Which is: "acbc"
  "bcac"
[ FAILED   ] MarkovAlgorithmTest.MultipleRuleApplication (0 ms)
[-----] 3 tests from MarkovAlgorithmTest (2 ms total)

[-----] Global test environment tear-down
[=====] 3 tests from 1 test suite ran. (3 ms total)
[ PASSED ] 0 tests.
[ FAILED   ] 3 tests, listed below:
[ FAILED   ] MarkovAlgorithmTest.SimpleRuleApplication
[ FAILED   ] MarkovAlgorithmTest.NoRuleApplication
[ FAILED   ] MarkovAlgorithmTest.MultipleRuleApplication
3 FAILED TESTS
```

Висновок: У даному коді реалізовано клас MarkovAlgorithm, який виконує алгоритм Маркова на вхідному рядку. Клас дозволяє додавати правила заміни, а потім застосовує ці правила до вхідного рядка, поки можливо виконувати заміни. Алгоритм працює наступним чином: для кожного правила заміни він шукає у вхідному рядку всі входження лівої частини правила і замінює їх правою частиною. Цей процес повторюється, доки в рядку можливі заміни. Код демонструє основні принципи роботи алгоритму Маркова та використання векторів, пар рядків та інші концепції мови C++.