

# Лабораторная работа №4

Лиганкина Анна

Вариант 20

10 января 2026 г.

## Содержание

1	Задание	2
2	Исследование языка	2
3	Наивный парсер	3
4	Оптимизированный парсер	3
5	Оценка эффективности	4

# 1 Задание

Дан язык:

$$L = \{ (a|b)^* c (a|b)^* (ab)^2 ba \}$$

Необходимо:

- проанализировать язык на КС-свойство, в случае его наличия - на регулярность;
- построить «наивный» парсер слов для языка, используя рекурсивный разбор с возвратами (парсер не должен заикливаться);
- построить оптимизированный парсер слов для языка. Оценить сверху его вычислительную сложность;
- посредством фазз-тестирования проверить эквивалентность парсеров и построить сравнительные графики их времени работы на случайных словах, принадлежащих языку и не принадлежащих языку (два тестовых пула).

# 2 Исследование языка

Заметим, что исходный язык можно переписать в виде:

$$L = \{ w \mid w = xsuuba \& x = (a|b)^* \& y = (a|b)^* \& xaby = yba \}$$

так как длина предпросмотра совпадает с длиной конца слова ( $|ab| = |ba|$ ).

## Доказательство, что язык не является КС

Так как контекстно-свободные языки замкнуты относительно пересечения с регулярными языками, то пересечем  $L$  с  $R = a^*ca^*ba^+$  и докажем, что получившийся язык  $L_R = L \cap R$  не КС.

Слово  $x$  однозначно определяется разделителем  $c$  и состоит из букв  $a$ , либо является пустым словом. Пусть  $x = a^n, n \geq 0$ . Так как в словах из  $R$  содержится лишь одна буква  $b$ , то она соответствует букве  $b$  из обязательного блока  $ba$ , а значит,  $y$  тоже состоит из букв  $a$ . Пусть  $y = a^m, m \geq 0$ .

Рассмотрим условие  $xaby = ybax$ :

$$a^n aba^m = a^m baa^n$$

$$a^{n+1}ba^m = a^m ba^{n+1} \Rightarrow m = n + 1$$

Таким образом, язык  $L_R$  выглядит так:

$$L = \{w \mid w = a^n ca^{2n+2}ba^{n+1} \text{ \& } n \geq 0\}$$

По лемме о накачке (длина накачки  $N$ ) рассмотрим слово  $a^N ca^{2N+2}ba^{N+1} \in L$ :

- если  $|vxy| < N$  попадает в один из блоков из букв  $a$ , то нарушается баланс с другими буквами  $a$ , и мы выходим из языка;
- если  $|vxy| < N$  попадает на стык блоков букв  $a$  и накачка задевает букву  $c$  или букву  $b$ , то нарушается структура слова, где лишь одна буква  $b$  и одна буква  $c$ , и мы выходим из языка;
- если  $|vxy| < N$  попадает на стык блоков букв  $a$  и мы качаем одновременно 2 блока букв  $a$  (3 не можем, так как  $|vxy| < N$ ), то все равно рушится баланс с оставшимся блоком  $a$ , и мы выходим из языка.

Поэтому язык  $L_R$  не является контекстно-свободным, а, следовательно, и язык  $L$ .

### 3 Наивный парсер

Наивный парсер считывает и запоминает  $x$  до символа  $c$ . Далее, он пытается рекурсивно «угадать» длину слова  $y$ . Запоминает  $y$ , а затем бежит по хвосту слова и сравнивает его одновременно с  $xaby$  и  $ybax$ . Если  $y$  достиг конца слова, то исходное слово не принадлежит языку. Таким образом, оценка сложности данного парсера -  $O(n^2)$ , где  $n$  - длина входного слова.

### 4 Оптимизированный парсер

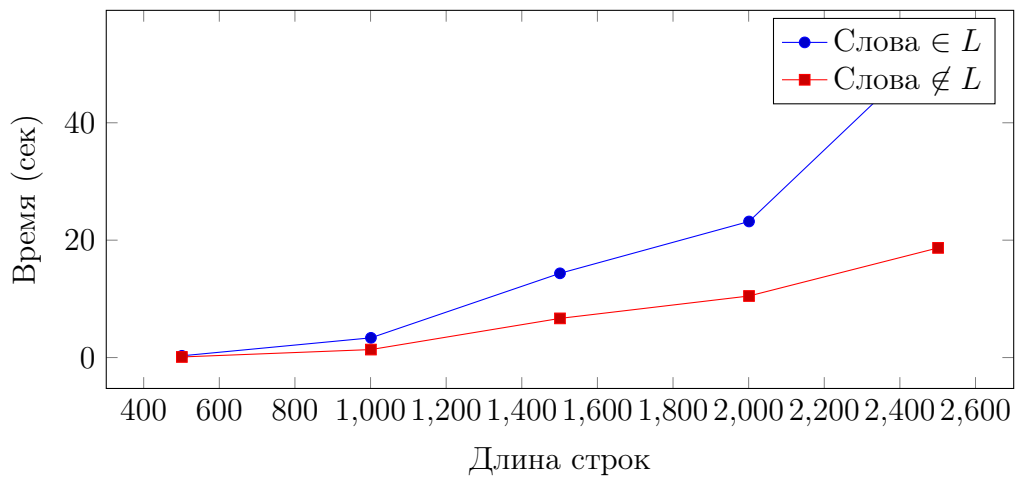
Оптимизация парсера происходит за счет следующего:

- если длина слова четна, то слово автоматически не подходит языку;

- длина  $y$  однозначно вычисляется по формуле «(длина слова - 3)/2 - длина  $x$ », благодаря чему алгоритмическая сложность парсера -  $o(n)$ , где  $n$  - длина входного слова;
- слова  $x$  и  $y$  являются палиндромами, а также  $xaby$  и  $yba x$ , так как  $xaby = yba x$ .

## 5 Оценка эффективности

Наивный парсер



Оптимизированный парсер

