

# Контрольное домашнее задание № 2

Кобызов Илья

# Содержание

<b>Введение</b>	<b>2</b>
<b>Наивный алгоритм</b>	<b>3</b>
1.1 Алфавит из 2 символов . . . . .	3
1.2 Алфавит из 4 символов . . . . .	3
1.3 Алфавит из 2 символов, использование символов подстановки . . . . .	4
1.4 Алфавит из 4 символов, использование символов подстановки . . . . .	4
1.5 Вывод по наблюдениям . . . . .	4
<b>Алгоритм Кнута-Морриса-Пратта</b>	<b>5</b>
2.1 Алфавит из 2 символов . . . . .	5
2.2 Алфавит из 4 символов . . . . .	5
2.3 Алфавит из 2 символов, использование символов подстановки . . . . .	6
2.4 Алфавит из 4 символов, использование символов подстановки . . . . .	6
2.5 Вывод по наблюдениям . . . . .	6
<b>Алгоритм Кнута-Морриса-Пратта с уточненными границами</b>	<b>7</b>
3.1 Алфавит из 2 символов . . . . .	7
3.2 Алфавит из 4 символов . . . . .	7
3.3 Алфавит из 2 символов, использование символов подстановки . . . . .	8
3.4 Алфавит из 4 символов, использование символов подстановки . . . . .	8
3.5 Вывод по наблюдениям . . . . .	8
<b>Вывод</b>	<b>9</b>

## Введение

В этом отчете приведены три алгоритма поиска с кратким описанием, и экспериментальными результатами времени работы. Для тестирования поиска использовался шаблон переменной длины в диапазоне  $[100; 3000]$  с шагом 100, два варианта алфавита из 2 и 4 символов, а также 2 варианта строки поиска длиной 10000 и 100000 символов. Итого получается  $2 \text{ типа} \times 2 \text{ размера} \times 3 \text{ алгоритма} = 12$  графиков.

Дополнительно в шаблон поиска на случайные места вставлялось от 1 до 4 символов подстановки '?', которые равны любому символу из строки поиска, что добавило ещё  $4 \text{ варианта символов} \times 3 \text{ алгоритма} \times 2 \text{ размера} \times 2 \text{ типа} = 48$  графиков.

Для лучших результатов время замерялось  $k$  раз и усреднялось.

# Наивный алгоритм

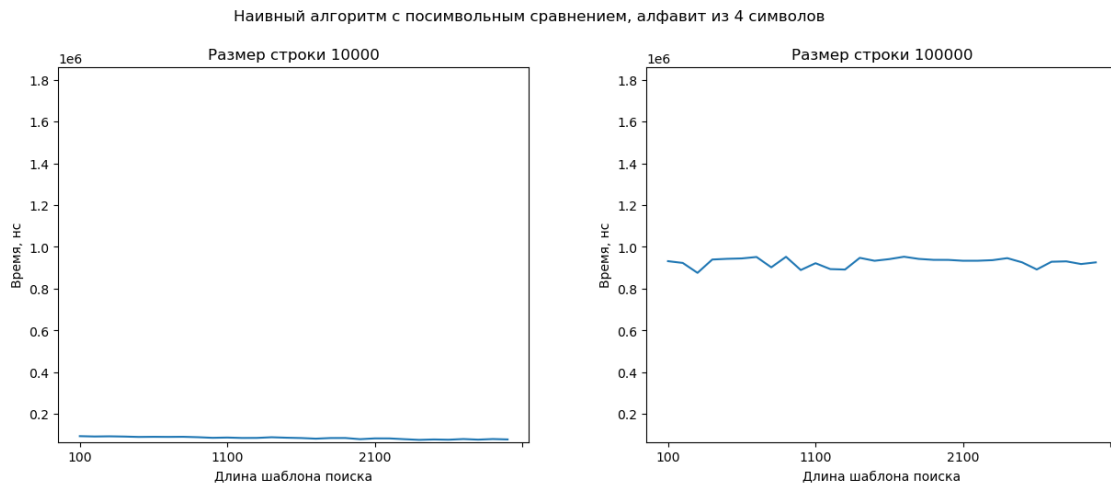
**Наивный алгоритм** сравнивает все возможные подстроки в строке поиска с шаблоном путем простого перебора и посимвольного сравнения.

Асимптотическая сложность алгоритма составляет  $O(n * t)$ , где  $n$  – длина строки поиска, а  $t$  – длина шаблона

## 1.1 Алфавит из 2 символов

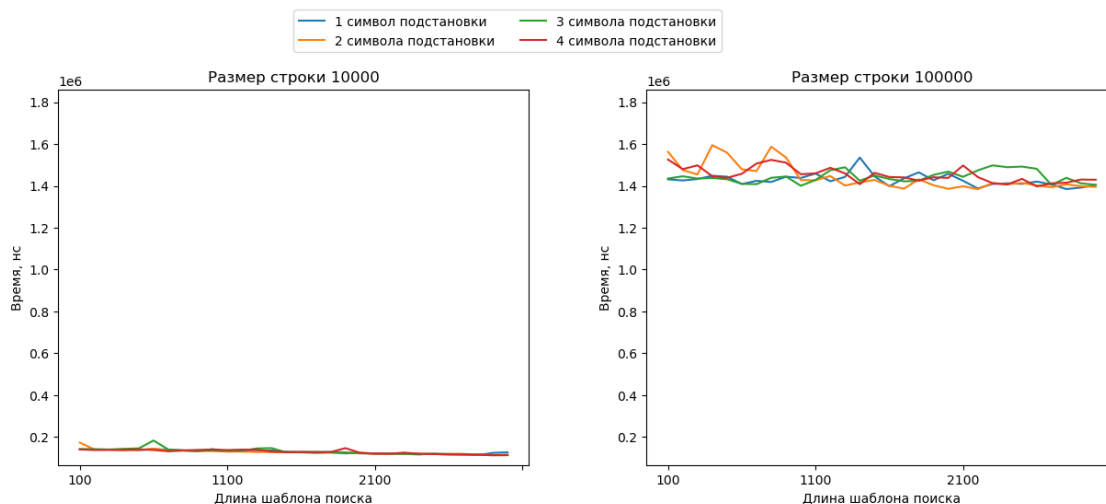


## 1.2 Алфавит из 4 символов



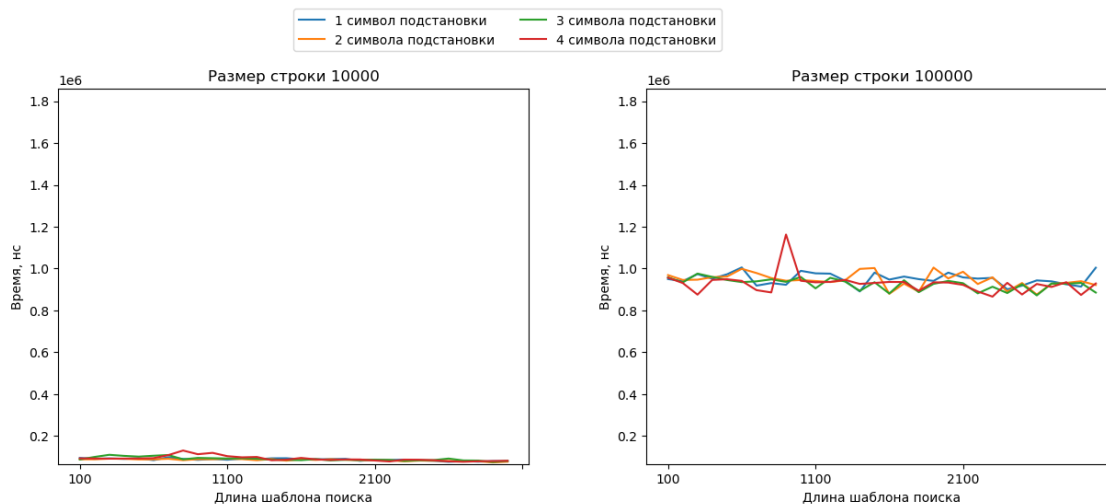
### 1.3 Алфавит из 2 символов, использование символов подстановки

Наивный алгоритм с посимвольным сравнением, 1-4 символов подстановки, алфавит из 2 символов



### 1.4 Алфавит из 4 символов, использование символов подстановки

Наивный алгоритм с посимвольным сравнением, 1-4 символов подстановки, алфавит из 4 символов



### 1.5 Вывод по наблюдениям

"Плохая" сложность подтвердилась, так как можно заметить значительное увеличение времени поиска при увеличении длины строки поиска.

Можно заметить, что с увеличением длины шаблона требуемого времени становится меньше. Это объясняется тем, что более длинный шаблон бинарного алфавита имеет меньшую вероятность появления в строке поиска несколько раз, поэтому алгоритм проводит меньше сравнений.

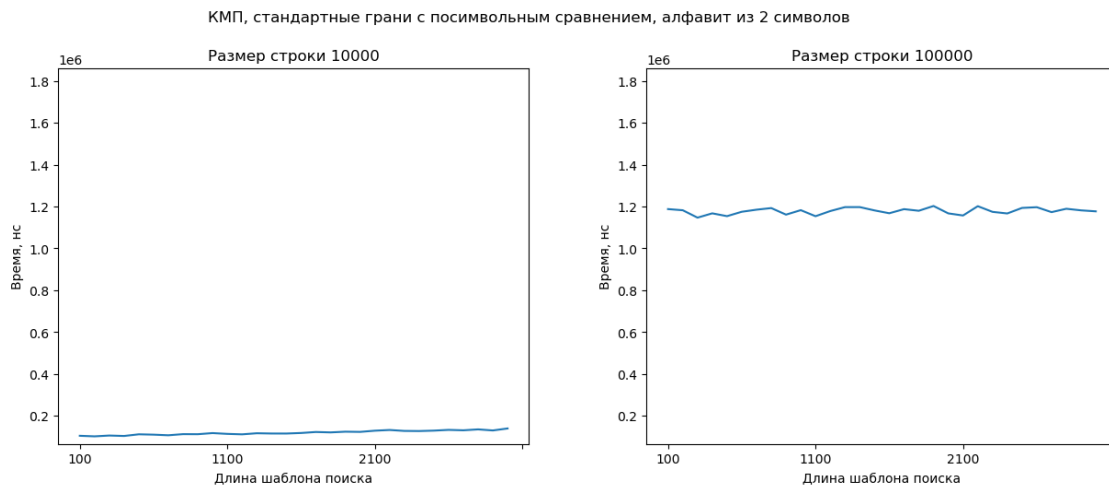
При использовании символов подстановки не замечено каких-либо значительных изменений.

# Алгоритм Кнута-Морриса-Пратта

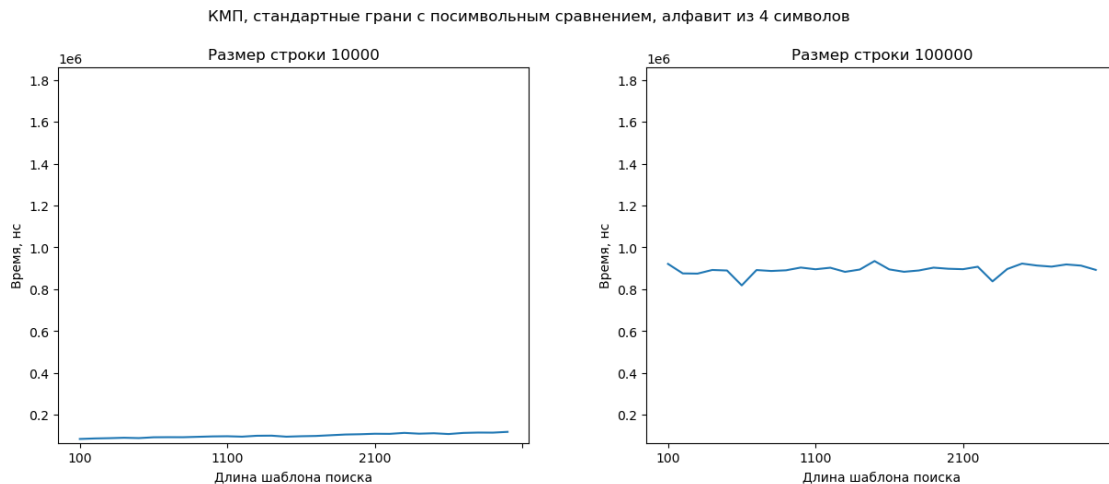
**Алгоритм Кнута-Морриса-Пратта** — алгоритм для поиска всех вхождений заданного шаблона в строку за линейное время относительно длины строки. Основная идея алгоритма - использование префикс-суффиксного массива, позволяющего определить, на какую позицию нужно сдвинуть шаблон в случае несовпадения символа.

Асимптотическая сложность алгоритма составляет  $O(n + m)$ .

## 2.1 Алфавит из 2 символов

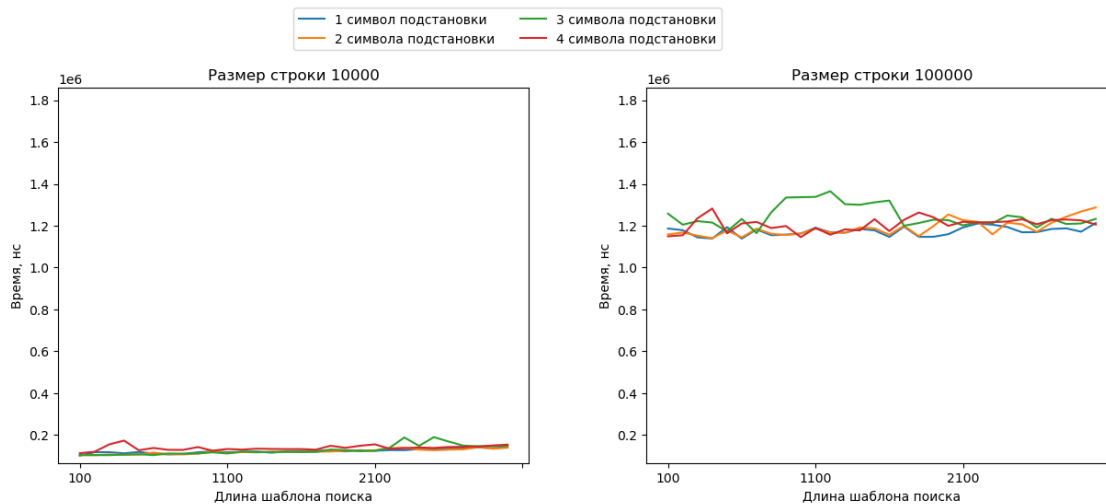


## 2.2 Алфавит из 4 символов



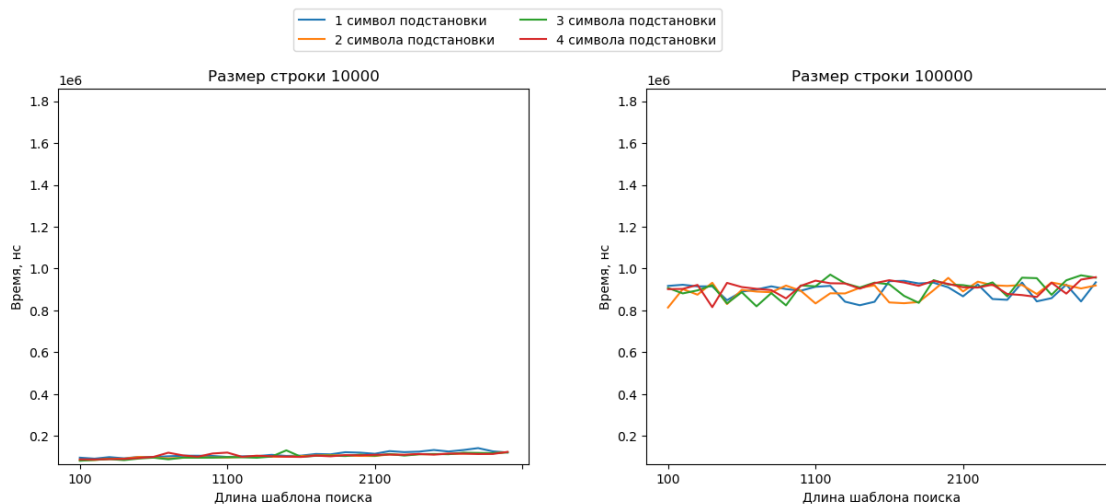
## 2.3 Алфавит из 2 символов, использование символов подстановки

КМП, стандартные грани с посимвольным сравнением, 1-4 символов подстановки, алфавит из 2 символов



## 2.4 Алфавит из 4 символов, использование символов подстановки

КМП, стандартные грани с посимвольным сравнением, 1-4 символов подстановки, алфавит из 4 символов



## 2.5 Вывод по наблюдениям

Алгоритм КМП показывает лучшее время по сравнению с наивным алгоритмом. Также можно заметить, что время возрастает с возрастанием длины шаблона.

Использование символов подстановки немного увеличило время работы алгоритма. При этом вариант с 4 символами работает дольше всего.

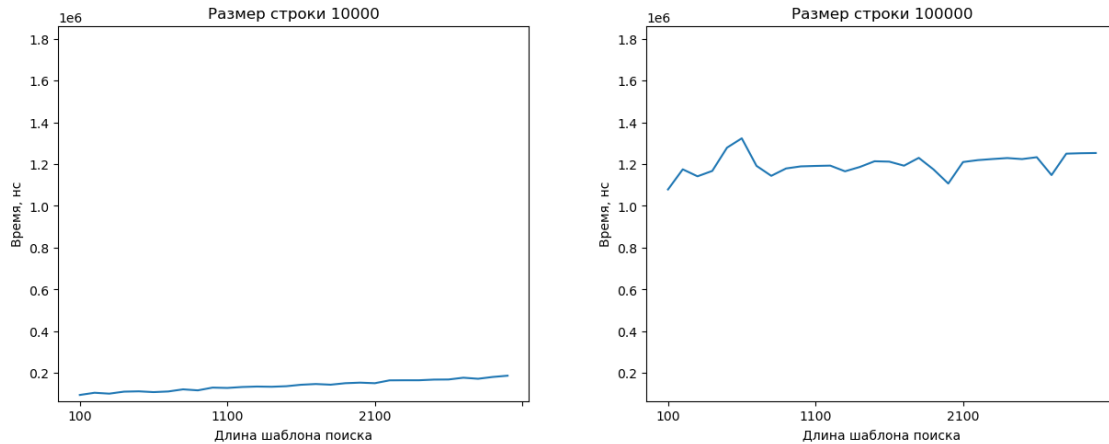
## Алгоритм Кнута-Морриса-Пратта с уточненными границами

**Алгоритм Кнута-Морриса-Пратта с расширенными границами** — алгоритм для поиска всех вхождений заданного шаблона в строку за линейное время относительно длины строки. Алгоритм основан на идее использования расширенных границ, которые являются более точной версией префикс-суффиксных массивов. Расширенная граница определяется как максимальная подстрока, которая является префиксом и суффиксом префикса шаблона.

Асимптотическая сложность алгоритма составляет  $O(n + m)$ .

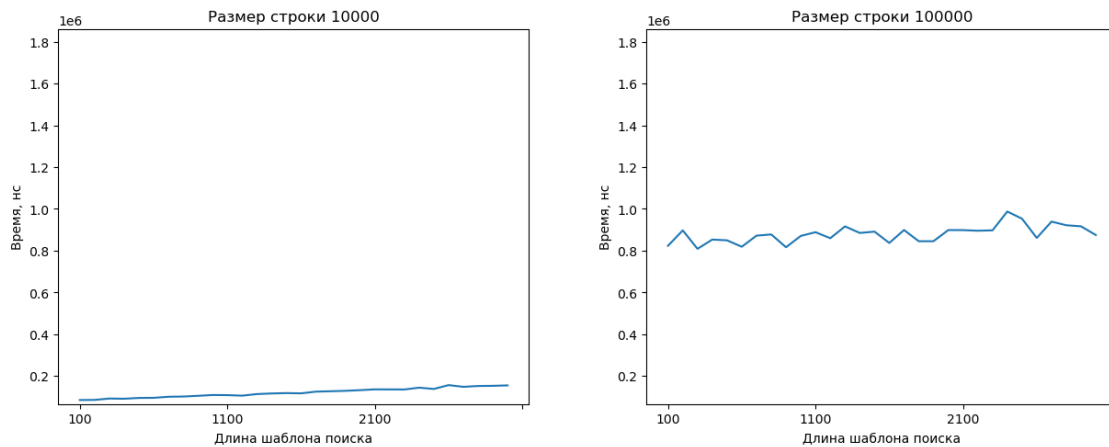
### 3.1 Алфавит из 2 символов

КМП, уточненные грани с посимвольным сравнением, алфавит из 2 символов



### 3.2 Алфавит из 4 символов

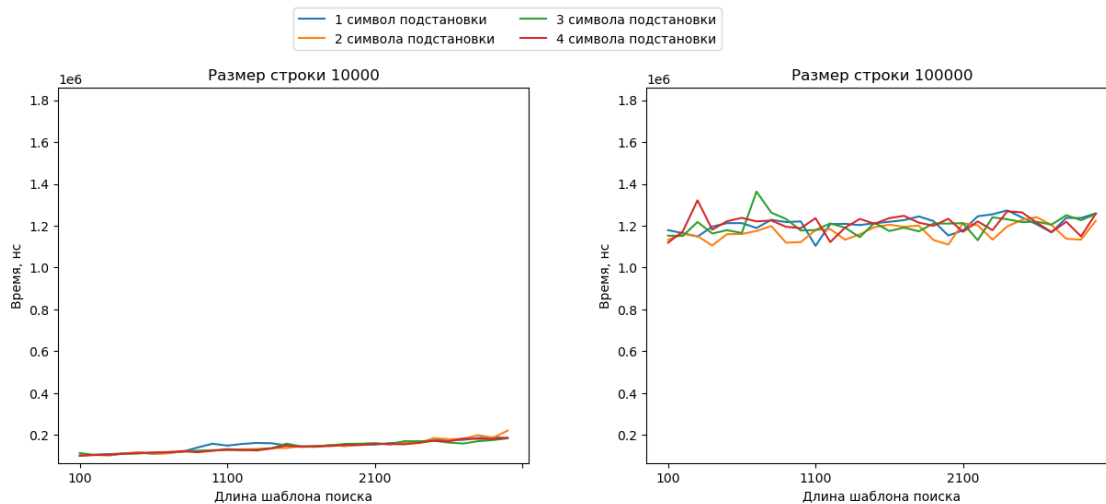
КМП, уточненные грани с посимвольным сравнением, алфавит из 4 символов





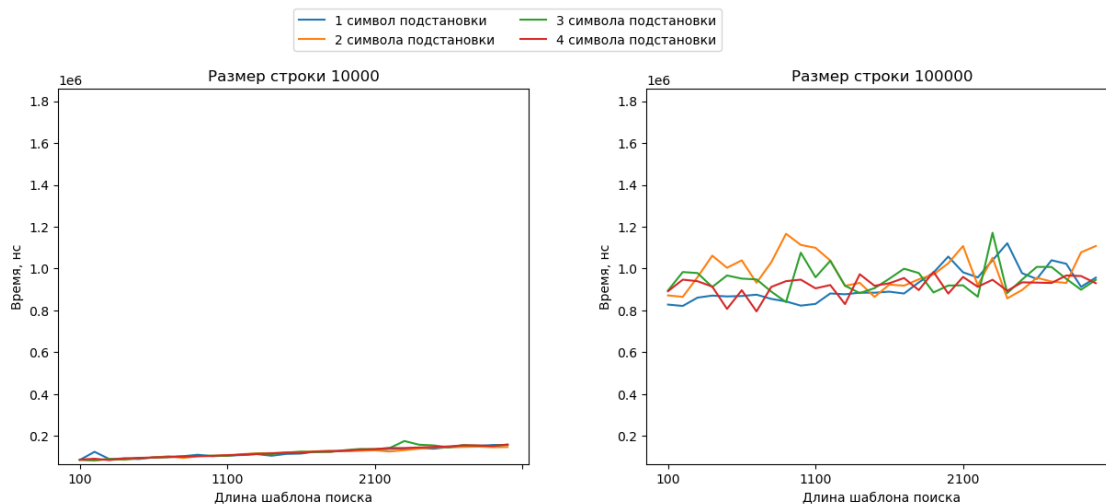
### 3.3 Алфавит из 2 символов, использование символов подстановки

КМП, уточненные грани с посимвольным сравнением, 1-4 символов подстановки, алфавит из 2 символов



### 3.4 Алфавит из 4 символов, использование символов подстановки

КМП, уточненные грани с посимвольным сравнением, 1-4 символов подстановки, алфавит из 4 символов



### 3.5 Вывод по наблюдениям

При сравнении графиков с предыдущими заметно, что применение уточненных граней немного улучшает время.

Использование символов подстановки немного увеличило время работы алгоритма.

## Вывод

По результатам измерений удалось подтвердить то, что наивный алгоритм не является эффективным алгоритмом поиска. Заметна разница между ним и алгоритмом Кнута-Морриса-Пратта, а также между КМП и КМП с уточненными (расширенными гранями).

Можно сделать вывод о КМП с уточненными гранями, как о самом эффективном алгоритме.