УДК 519.85(023)

СПОСОБ ПОИСКА ПО ОБРАЗЦУ

© 2016 А. И. Сухачев[1], В. М. Довгаль[2], В. В. Гордиенко[3]

[1]аспирант кафедры программного обеспечения
и администрирования информационных систем
e-mail: askur46@gmail.com
[2]профессор, докт. техн. наук, кафедра программного обеспечения
и администрирования информационных систем
e-mail: vmdovgal@yandex.ru,
[3]канд. техн. наук, доцент, кафедра программного обеспечения
и администрирования информационных систем
e-mail: vvgord@yandex.ru

Курский государственный университет

В статье рассматривается один из подходов к решению проблемы акселерации поиска подстроки в строке, основанный на особенной структурированной форме представления подстроки (образца), и приводятся результаты экспериментов в виде графиков зависимости числа сопоставлений букв рабочего алфавита в процессе поиска позиций вхождений образца от специфических свойств размещения итераций вначале образца и строке при разных мощностях алфавитов и числа позиций образцов в обрабатываемом слове для сопоставления разработанного алгоритма с некоторыми известными прототипами. Приведены скоростные преимущества разработанного алгоритма по сравнению с реализацией способа поиска по образцу Кнута–Морриса–Пратта (КМП) и Бойера–Мура.

Ключевые слова: поиск, строка, подстрока, образец, мощность, алфавит, позиция образца, итерация, сопоставление.

Развитие алгоритмических, программных и технических средств информатики в значительной степени стимулируется ростом числа задач поиска по образцу непосредственно в потоках данных, распределенных хранилищах, банках данных с монотонным увеличением их объемов, что влечет за собой повышение требований к скорости поиска информации. На каждом из этих направлений приложений методов поиска по образцу цели и средства различаются, но они имеют общность в контексте передачи, хранения и обработки символьной информации. Поиск в целом всегда был одной из ключевых задач, стимулирующих развитие средств информатики. На сегодняшний день существует множество методов, алгоритмов и различных реализаций поиска информации: от простой подстроки в строке до его реализации в объемных массивах данных, а также в поисковых машинах различного назначения.

Рассмотрим простейший вариант поиска данных, сравнив алгоритмы и механизмы поиска подстроки (образца) в строке (обрабатываемое слово). Ниже приведены основные факторы, влияющие на результаты поиска и его качественные показатели.

1. Мощность алфавита. В зависимости от количества символов в используемом алфавите, те или иные алгоритмы поиска будут давать различные результаты. Если при использовании алфавита с малой мощностью (например, бинарный алфавит) алгоритм дает хорошие показатели, то при использовании алфавитов, содержащих в качестве своих элементов высокий показатель их разнообразия, он может проигрывать алгоритмам-аналогам в скорости работы.

2. Грамматика языка. Те или иные особенности грамматики могут быть плохо сочетаемыми с эвристиками, которые ускоряют поиск в общих ситуациях.

3. Тип образца. Длина искомого слова-образца, разнообразие символов, преобладание и/или повторение (итераций) тех или иных фрагментов слов – все эти факторы также существенно влияют на скорость поиска и на выбор алгоритма для эффективного решения конкретных задач.

4. Возможность предварительной индексации исходной строки в виде слова конечной длины. При наличии такой возможности открываются пути минимизации временной сложности алгоритма поиска при увеличении показателя его емкостной сложности.

5. Необходимость поиска сразу нескольких строк или приблизительный поиск. Достаточно специфические ситуации, которые обычно требуют нестандартных решений, например при наличии побочных свойств некоторых алгоритмов.

6. Свойства аппаратной платформы при реализации поиска по образцу. Наличие или отсутствие аппаратных средств, те или иные особенности типа процессора играют важную роль при выборе метода поиска.

В различных сочетаниях эти факторы могут по-разному влиять на область эффективного использования того или иного алгоритма и, как следствие, на результат поиска [Кормен и соавт. 2000].

В общих случаях для минимизации потерь времени и затрат памяти при осуществлении поиска были разработаны так называемые «классические» алгоритмы. Среди них доминируют способы и алгоритмы Кнута–Морриса–Пратта (КМП) [1977] и Бойера–Мура (БМ) [1977] с различными вариантами модификаций и ускорения в тех или иных случаях. Между тем существуют ситуации, когда применение названных универсальных алгоритмов будет не лучшим вариантом выбора алгоритма при решении задачи поиска по образцу. Это связано с их универсальностью, поскольку при таком условии приходится не учитывать некоторые специфические условия для получения оптимальных результатов в общем наборе исходных данных. Из анализа принципов работы классических алгоритмов на примере алгоритмов КМП и БМ становится очевидным, что одним из главных принципов ускорения являются множественные сдвиги, как на основе префикс-таблицы в алгоритме КМП, так и на основе эвристик стоп-символа и совпавшего суффикса в алгоритме БМ. Как уже отмечалось ранее, в большинстве случаев это обеспечивает преимущество в скорости реализации поиска. Нами было выдвинуто предположение, что при наличии циклично повторяющихся символов множественные сдвиги по слову и/или по образцу могут ухудшить итоговую скорость поиска. В этих условиях предварительный анализ образца может оказать значимое влияние на временную и емкостную составляющие алгоритма поиска, поскольку непосредственный анализ образца во время его выполнения усложняет процесс поиска.

Необходимо отметить, что анализ образца у указанных выше классических алгоритмов подразумевает построение тех или иных таблиц сдвигов. И эта идея, безусловно, давно себя зарекомендовала. Но при наличии в образце или обрабатываемом слове повторяющихся символов или фрагментов (итераций), а также их повторения в структуре слова снижают скорость работы алгоритма. Именно такие примеры образцов и слов часто встречаются при работе с бинарными алфавитами.

Слово представляет собой наборы различных, в том числе и повторяющихся, фрагментов. В бинарном алфавите это будут повторяющиеся фрагменты двух символов в той или иной их последовательности. Любой образец можно представить в виде $a_n b$, где a – фрагмент слова, который повторяется n раз, а b – фрагмент произвольной длины, который повторяется один раз (собственное окончание образца). В двоичном

алфавите в простейшем случае фрагмент a будет иметь длину 1. В других случаях он может представлять собой набор вложенных фрагментов. Например: $(\mathrm{c}_m d)_o t$, где c и $(\mathrm{c}_m d)t$ – фрагменты слов длины m и o соответственно с собственным окончанием слова-образца.

Для выполнения поиска при наличии итераций был разработан алгоритм, в котором отсутствует какой-либо сдвиг по строке, а также присутствует предварительный структурный анализ образца в интересах его вышеприведенного структурного представления.

Эффективность работы алгоритмов условимся определять по количеству операций посимвольного сравнения.

Прежде всего важно определить, какие скоростные результаты поиска будут получены при различных условиях с использованием алфавитов разной мощности. Для тестирования были выбраны три мощности алфавитов: 2, 10 и 204. С учетом того факта, что предугадать расположение позиции образца в обрабатываемом слове невозможно, расположение его при тестировании будет условно произвольное. Длина слова принята равной 5128 символам. На представленных ниже по тексту графиках (рис. 1–18) показаны зависимости количества операций посимвольного сравнения от кратности повторов первого фрагмента образца. Длина первого повторяющегося фрагмента равна 1-му символу. Длина последнего кортежа – «хвоста» равна 5-ти символам. Рассмотрено 45/54/63/72/81/90 вхождений образца. Каждый раз слово условно было разбито на три части, внутри которых хаотично были расположены образцы по три подряд в разном количестве позиций их вхождения в обрабатываемое слово. Для удобства пропорции указывается количество блоков по три образца подряд в каждой из условных частей слова. В последнем случае с 90 повторениями в итерации, все они расположены хаотично по три подряд в строке (обрабатываемом слове).

Вхождений образца: 45. Пропорция: 3/1/1.

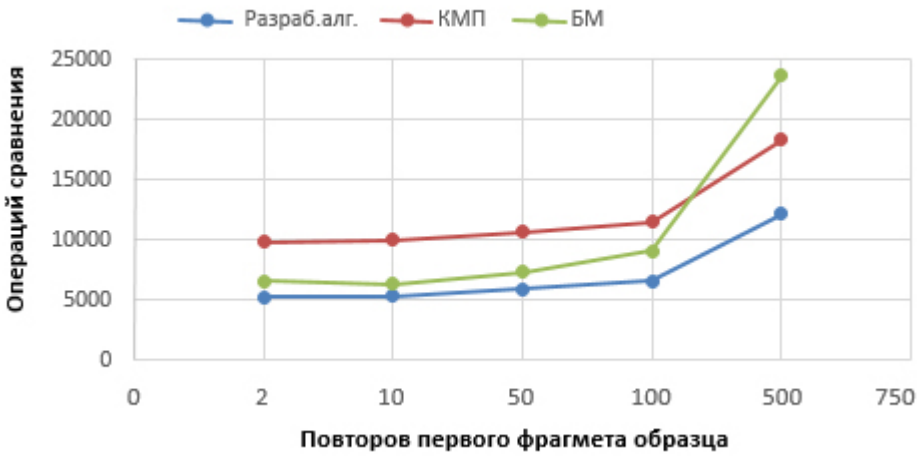

Рис. 1. График зависимостей (алфавит: 2 символа)

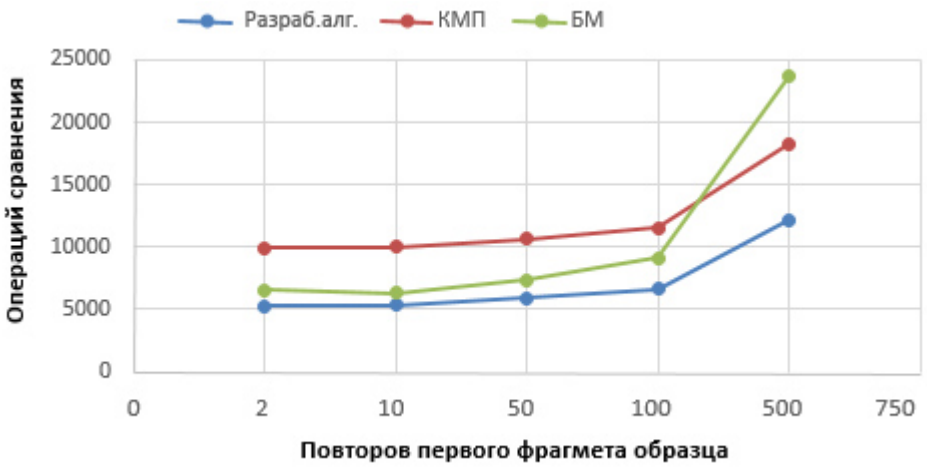

Рис. 2. График зависимостей (алфавит: 10 символов)

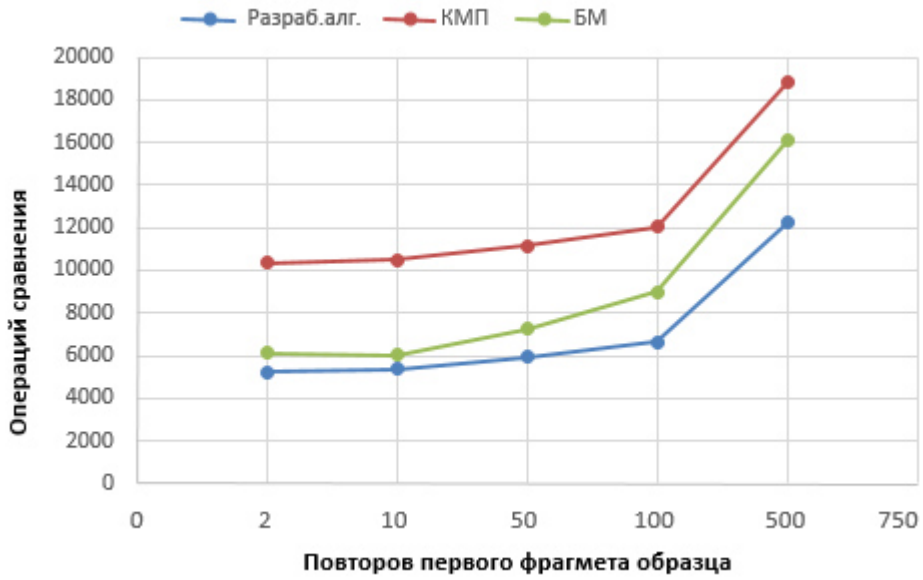

Рис. 3. График зависимостей (алфавит: 204 символа)

Вхождений образца: 54. Пропорция: 0/3/3.

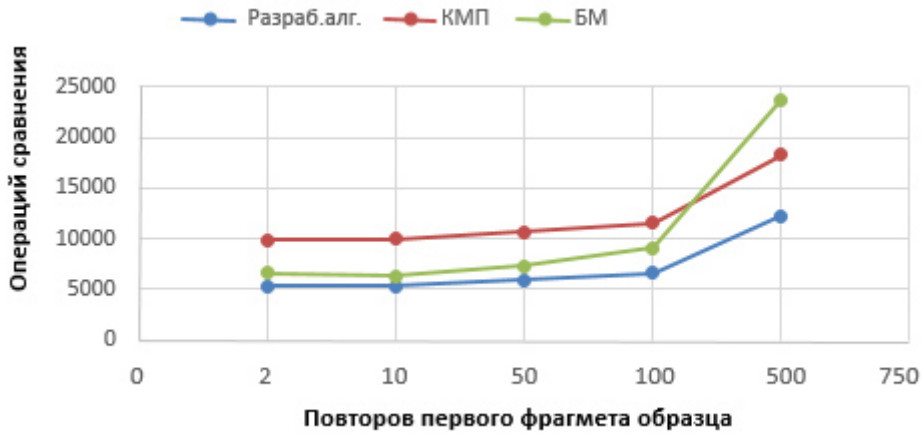

Рис. 4. График зависимостей (алфавит: 2 символа)

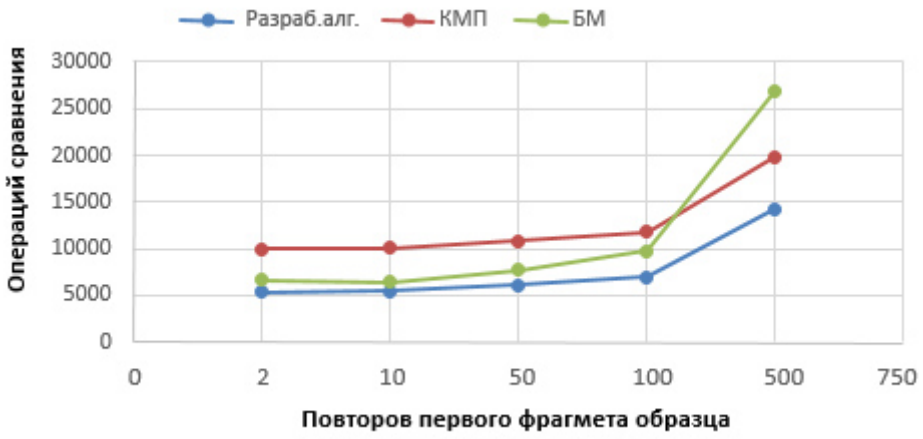

Рис. 5. График зависимостей (алфавит: 10 символов)

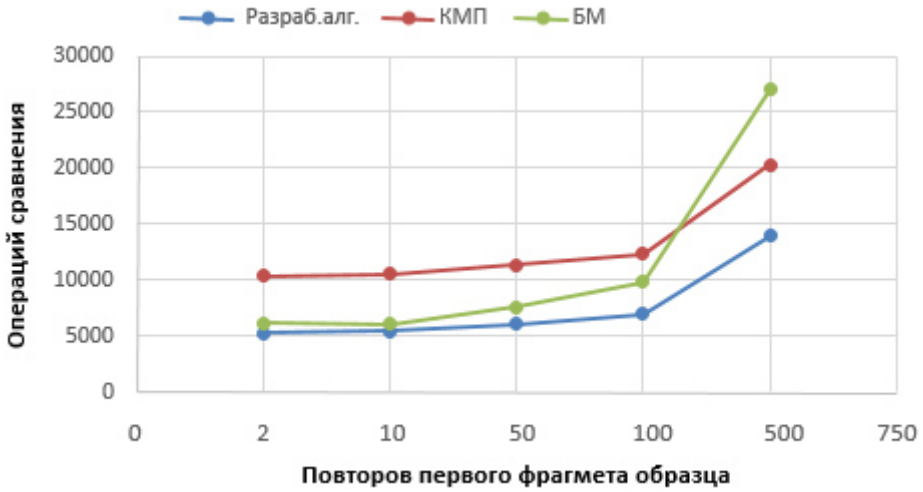

Рис. 6. График зависимостей (алфавит: 204 символа)

Вхождений образца: 63. Пропорция: 5/0/2.

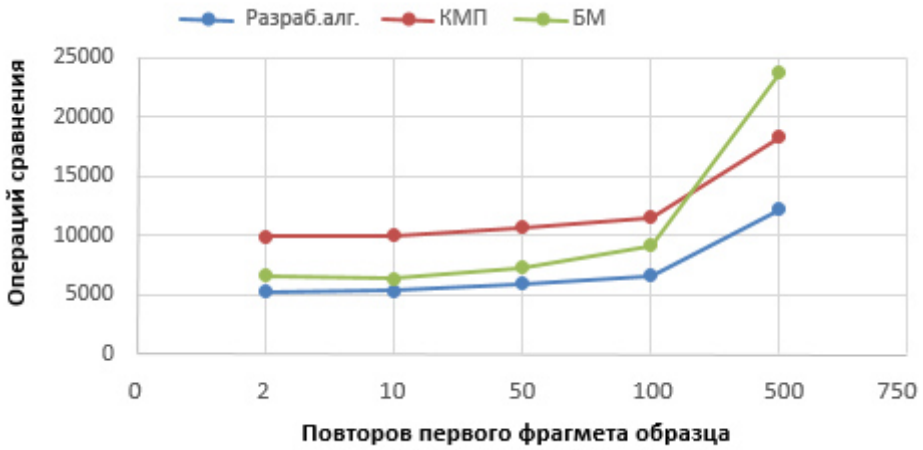

Рис. 7. График зависимостей (алфавит: 2 символа)

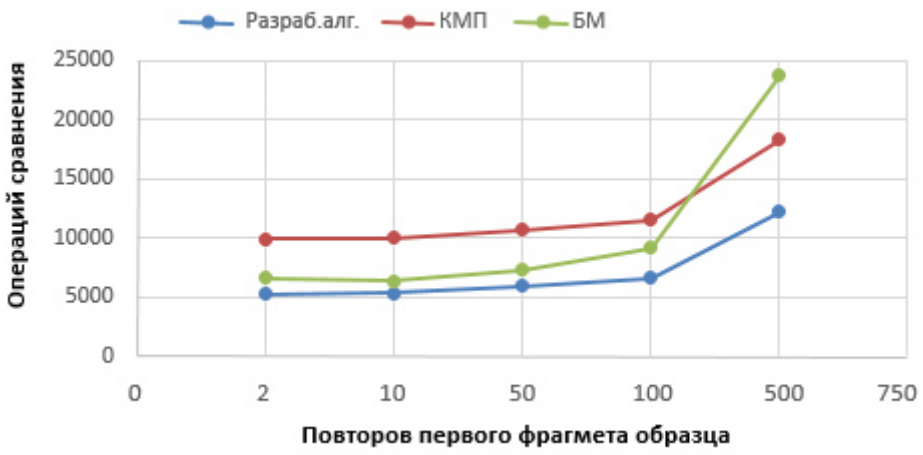

Рис. 8. График зависимостей (алфавит: 10 символов)

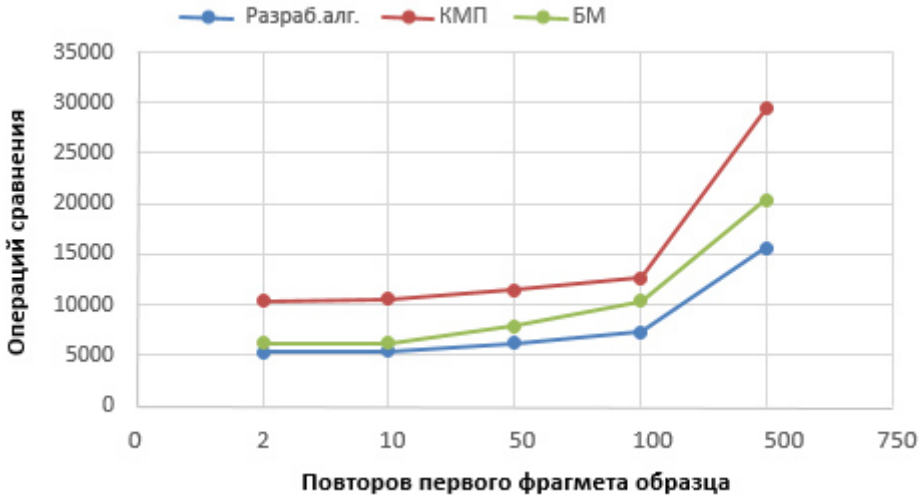

Рис. 9. График зависимостей (алфавит: 204 символа)

Вхождений образца: 72. Пропорция: 1/7/0.

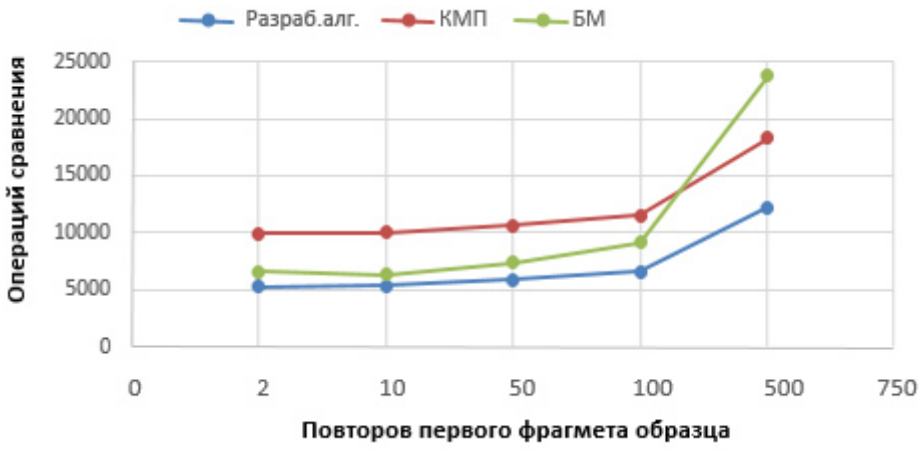

Рис. 10. График зависимостей (алфавит: 2 символа)

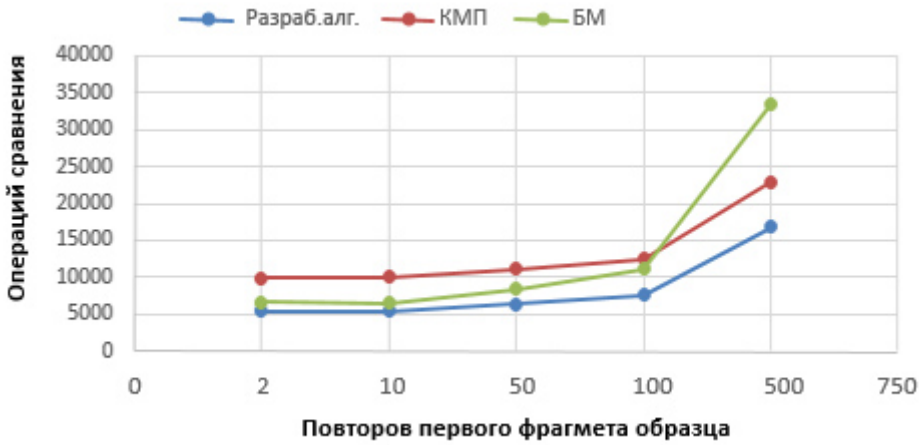

Рис. 11. График зависимостей (алфавит: 10 символов)

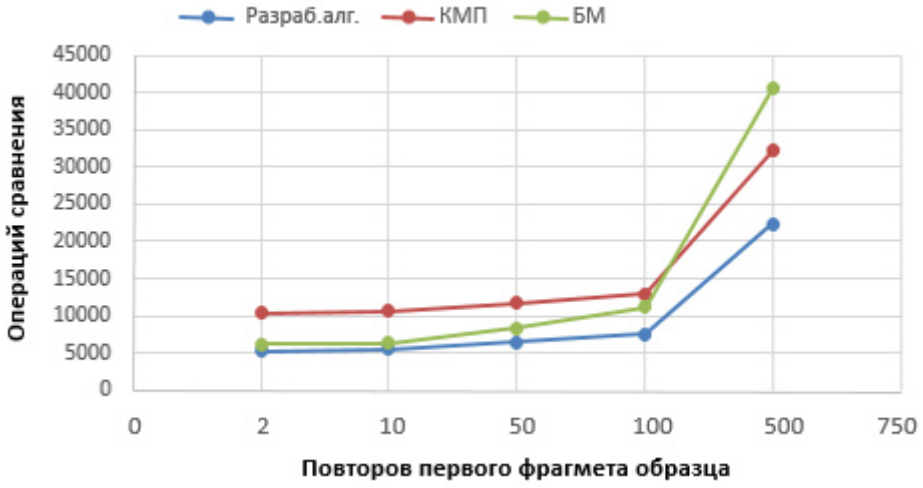

Рис. 12. График зависимостей (алфавит: 204 символа)

Вхождений образца: 81. Пропорция: 4/1/4.

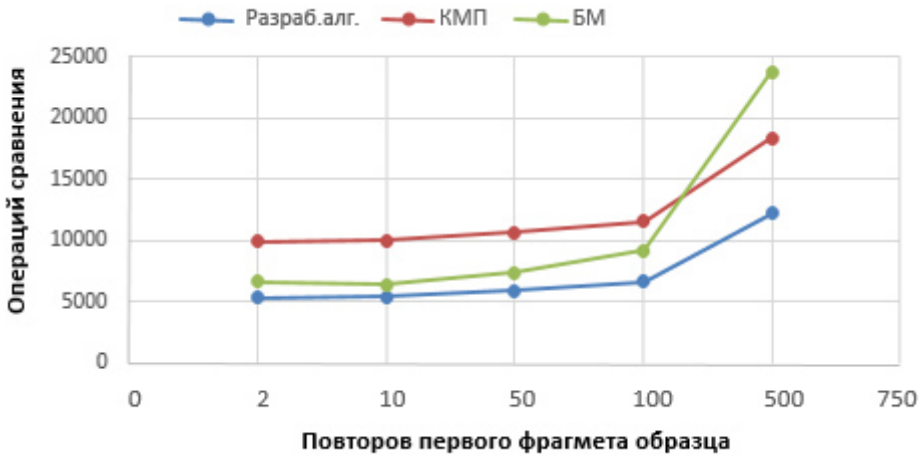

Рис. 12. График зависимостей (алфавит: 2 символа)

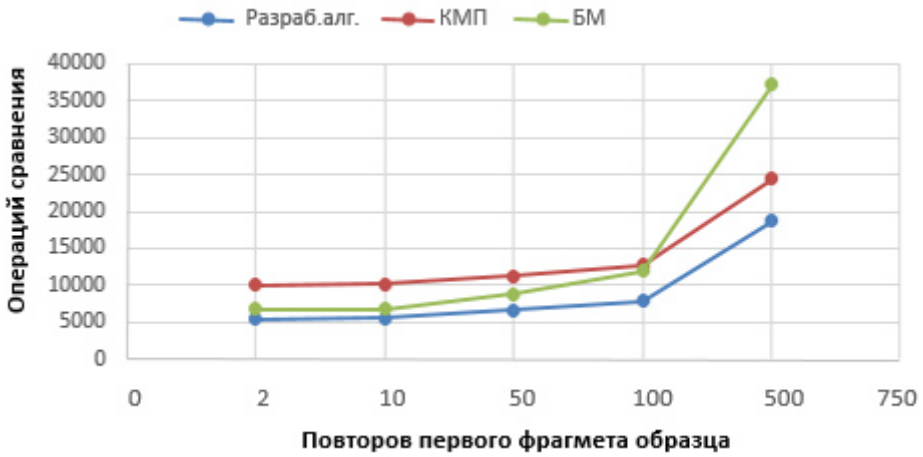

Рис. 14. График зависимостей (алфавит: 10 символов)

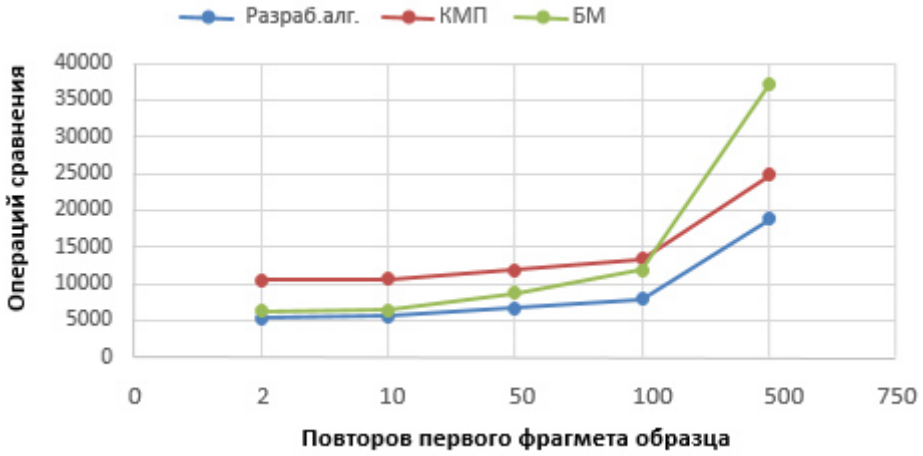

Рис. 15. График зависимостей (алфавит: 204 символа)

Вхождений образца: 90. Пропорции нет.

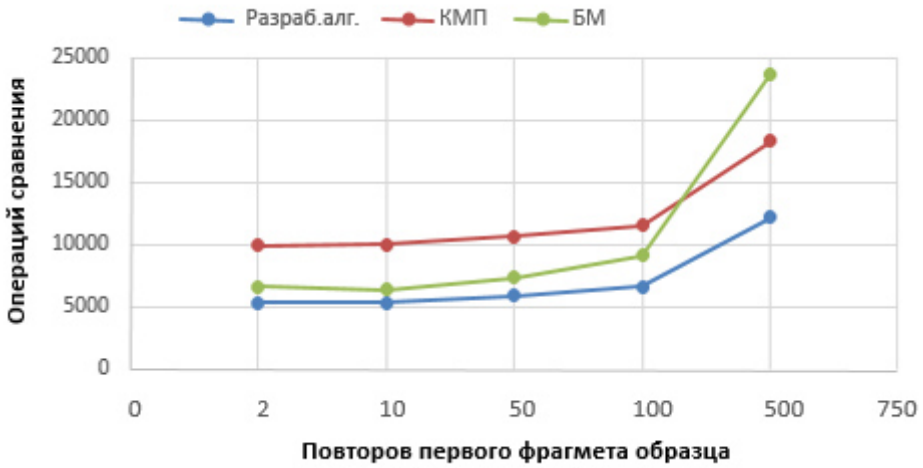

Рис. 16. График зависимостей (алфавит: 2 символа)

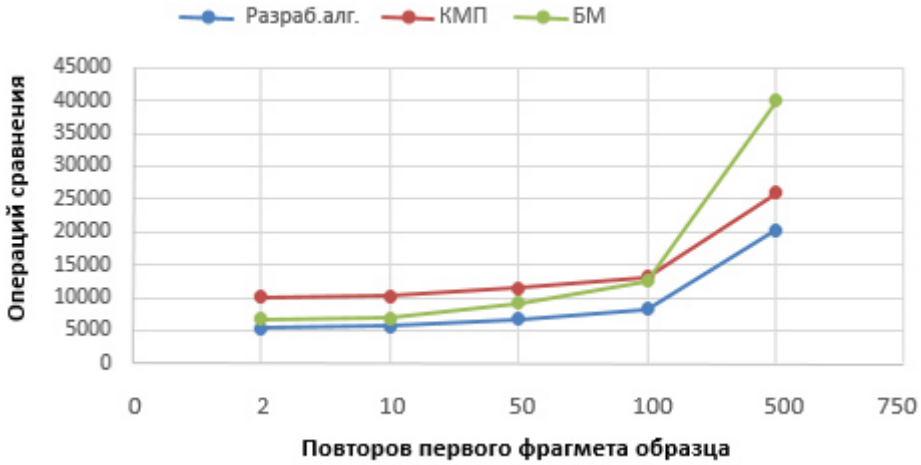

Рис. 17. График зависимостей (алфавит: 10 символов)

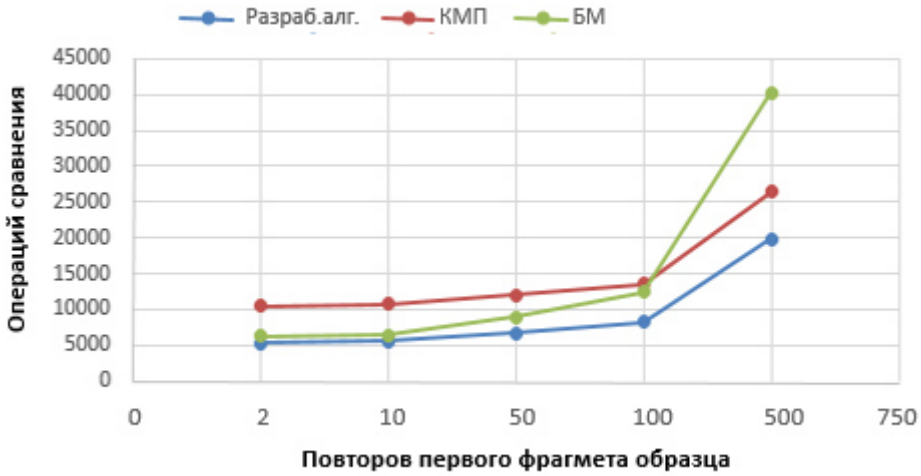

Рис. 18. График зависимостей (алфавит: 204 символа)

Графики наглядно показывают, что при заданных вариантах расположения образца при всех используемых мощностях алфавита обеспечивается преимущество по числу посимвольных сравнений (ось ординат на графиках) разработанных нами способа поиска и алгоритма его реализации с модификацией формы представления образца по отношению к аналогам. С увеличением количества повторений первого символа выигрыш в количестве посимвольных сравнений увеличивается. Это эмпирически подтверждает выдвинутое предположение, что при наличии циклично повторяющихся символов множественные сдвиги по слову и/или по образцу могут ухудшить итоговую скорость поиска. Безусловно, что и предварительный структурный анализ образца несущественно влияет на квалиметрические показатели поиска по образцу, но он выполняется однократно до осуществления поиска позиции образца в обрабатываемых словах.

Библиографический список

Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. М.: МЦНМО, 2000. 960 с.

Boyer R. S., Moore J. S. A fast string searching algorithm. Comm. ACM 20: 1977. 772 p.

Donald Knuth; James H. Morris, Jr, Vaughan Pratt. Fast pattern matching in strings // SIAM Journal on Computing. 1977. 6 (2). 402 p.

Crochemore M., Rytter W. Jewels of Stringology: Text Algorithms. World Scientific Publishing, 2002. 320 p.

Сухачев А.И., Довгаль В.М. Об одном подходе к решению задачи быстрого поиска подстроки в строке и результатах эксперимента // Сб. науч. тр. I Междунар. науч.-техн. конф. (26–27 мая 2015 г.) / Курск. гос. ун-т. Курск, 2015. С. 70–71.