THE EFFICIENCY OF STRING MATCHING ALGORITHMS ON NATURAL LANGUAGES TEXT

Abdul Kadir ERSİN
abdulkadir.ersin@trakya.edu.tr
Computer Engineering Department
Trakya University – Edirne / TURKEY

Aydın CARUS
aydinc@trakya.edu.tr
Computer Engineering Department
Trakya University – Edirne / TURKEY

Altan MESUT
altanmesut@trakya.edu.tr
Computer Engineering Department
Trakya University – Edirne / TURKEY

Abstract

In this study, the changes in the performance of the string matching algorithms when they are used with different natural languages are searched. For this purpose, 8 different text corpuses which are the same in size are prepared for 8 different natural languages. 6 different string matching algorithms are used with these corpuses and the searching times are found and recorded. Because of the string matching algorithms depend on the alphabet, the natural languages are divided into two groups according to the number of elements in their alphabets; half of them uses 120 while the other half uses 140 individual elements. The aim of this grouping is to show that the difference in the performance of these 6 algorithms is not only depends on the number of elements in alphabets, but it is also depends on the structural changes of these 8 natural languages.

Keywords: String Matching Algorithms, Natural Languages.

INTRODUCTION

String matching is a work that aims to find all occurrences of a searched string in a text that includes finite elements. Nowadays, string matching algorithms are used widely in word processing, genetics, search engines in internet, etc. The performance of a string matching algorithm mainly depends on the used alphabet. A string matching algorithm that is efficient in DNA search may not be efficient in natural language. The main reason is the difference of the number of elements in the alphabets [1]. However, the research on "How the performance of string matching algorithms differs in natural languages that have the same number of elements in their alphabets?" is not performed yet.

In this research, widely used string matching algorithms are tested on different natural languages and the results are given for three different string length.

ALGORITHMS

Six algorithms that have different specifications and structures are used in this research. These algorithms can search in a specific order from left to right or right to left or they can use both of these two directions. Specifications of these algorithms are given in this section.

- **Boyer-Moore Algorithm:** This algorithm has two pre-processing phases; bad-character shift and good-suffix shift. It makes the shifting processes with using the maximum value that is gained in pre-processing phases [2].
- **Turbo Boyer-Moore Algorithm:** As difference from Boyer-Moore, it consists in remembering the factor of the text that matched a suffix of the pattern during the last attempt (and only if a good-suffix shift was performed) [3].
- **Horspool Algorithm:** It uses the bad-character shift pre-processing phase in the Boyer-Moore algorithm. First it compares last character. Then it compares from beginning of the pattern from left to the right [4].
- **Quick Search Algorithm:** It uses the bad-character shift pre-processing phase in the Boyer-Moore algorithm. It can use the character after the last character, which is positioned on text factor for shifting [5].

- **Smith Algorithm:** This algorithm performs Horspool bad-character rule and Quick Search bad-character rule and selects the maximum value between them for performing the shifting process [6].
- **Raita Algorithm:** This algorithm compares last, first and middle characters of the searched string respectively. If all of them are matched, comparing process continues from the second character to the last character. However, the middle character is going to be compared again [7].

TEST ENVIRONMENT AND CORPUSES

Eight languages are selected to evaluate the performances of the string matching algorithms in different languages; Turkish, English, German, French, Italian, Spanish, Dutch and Finnish. These languages are separated into two groups according to the total number of characters, numbers and other symbols used in these languages. The first group which contains 120 symbols includes Turkish, English, Dutch and Finnish, while the second group which contains 140 symbols includes German, French, Italian and Spanish. Although the numbers of symbols are equal for all languages in a group, all of the symbols of these languages in this group are not exactly equal to each other.

The text corpuses that have size of 30 MB are prepared for each language. Turkish Corpus is prepared from ODTU Corpus [8] and some articles, stories and novels. Some symbols in ODTU Corpus are removed, because they are related to XML structure and not used in natural language. The corpuses of the other 7 languages are prepared with stories and novels from Project Gutenberg (www.gutenberg.org).

According to evaluate differences of string matching algorithms depend on string length, three different string lengths are used in tests: 10 character (short string), 100 character (middle length string) and 200 character (long string). 100 different strings for each language and for each string length are obtained from different positions of the related corpus to use in tests.

The C codes of the selected algorithms which are taken from Christian Charras and Thierry Lecroq [9] are compiled with Visual Studio .NET 2005 in *Release* mode. The tests are made in a computer which has Core 2 Duo 1.83 GHz processor, 1024 MB main memory, 80 GB hard disk drive and Windows operating system installed.

TESTS AND RESULTS

100 different strings are searched in each corpus for each string length in tests. Total search times and comparison numbers of these 100 strings are given in different tables for each group.

The search time results of the first group are given in Table 1. Colored cells of this table indicate the best values for languages. Turkish has the best results in short and middle length strings for all algorithms. However, the results of Turkish are not the best in long strings for Raita and Quick Search algorithms. Finnish generally has the second best results after Turkish. For some algorithms English is better than Finnish.

		Turkish			English			Dutch			Finnish		
		Short	Middle	Long	Short	Middle	Long	Short	Middle	Long	Short	Middle	Long
Methods	Horspool	2,40	1,36	1,47	2,68	1,51	1,65	2,81	1,54	1,50	2,67	1,44	1,51
	Raita	2,49	1,35	1,56	2,57	1,42	1,61	2,76	1,45	1,51	2,57	1,52	1,55
	Quick Search	2,75	1,52	1,65	3,13	1,64	1,58	3,09	1,65	1,62	2,86	1,53	1,79
	Boyer-Moore	3,40	1,60	1,62	3,70	1,73	1,70	3,93	1,89	1,79	3,67	1,75	1,81
	Smith	3,83	1,60	1,69	4,54	1,77	1,78	4,27	1,72	1,71	4,11	1,85	1,80
	Turbo Boyer-Moore	4,10	1,72	1,73	4,59	1,92	1,83	4,95	2,03	2,03	4,67	1,94	1,89

Table 1. *Search time results for Group 1 (s)*

Dutch has the worst results in middle length strings except for Smith and Raita algorithms and it has the worst results in short strings except for Quick Search and Smith algorithms. However, in long strings, Dutch has the worst result only for Turbo Boyer-Moore algorithm and it has the best result in Raita algorithm. As seen in Table 1, although all the languages in this group have the same alphabet size, they do not give the same results. This situation shows that the structure of

a language can effect the search process. Generally, Raita algorithm has the best and Turbo Boyer-More algorithm has the worst performance in all natural languages.

The search time results of the second group are given in Table 2. The results of German and French in short strings are generally better than other languages. Spanish is the worst language for 3 algorithms and Italian is the worst language for other 3 algorithms in short strings.

		German			French			Italian			Spanish		
		Short	Middle	Long	Short	Middle	Long	Short	Middle	Long	Short	Middle	Long
Methods	Horspool	2,59	1,40	1,53	2,54	1,45	1,72	2,65	1,59	1,58	2,76	1,48	1,61
	Raita	2,58	1,50	1,51	2,63	1,43	1,52	2,61	1,55	1,56	2,67	1,52	1,62
	Quick Search	2,84	1,49	1,69	3,00	1,54	1,74	3,01	1,52	1,70	2,95	1,56	1,61
	Boyer-Moore	3,74	1,66	1,73	3,64	1,70	1,68	3,75	1,83	1,73	3,80	1,71	1,68
	Smith	4,24	1,62	1,85	4,24	1,67	1,75	4,42	1,88	1,76	4,38	1,75	1,79
	Turbo Boyer-Moore	4,52	1,89	1,92	4,57	1,87	1,90	4,68	1,94	1,87	4,62	1,89	1,94

Table 2. *Search time results for Group 2 (s)*

French has the best results in middle length strings. German gave a good performance in middle length strings like it had in short strings. The performance of Spanish in middle length strings is better than its short length performance. Italian has the second best result in Quick Search algorithm while its performance is worse in other algorithms. The performances of German and French in long strings are worse than their performance in short strings. Spanish and Italian are increased their performance in long strings.

When these two groups and three string lengths are evaluated together, it can be seen that the performance of Turkish is mostly better than other languages. The search times in Table 1 and Table 2 are very similar to each other. Therefore we can say that; the structure of the language is more distinctive feature than alphabet size for search time.

		Turkish			English			Dutch			Finnish		
		Short	Middle	Long	Short	Middle	Long	Short	Middle	Long	Short	Middle	Long
Methods	Horspool	4173	1150	907	4485	1365	1112	4716	1393	1131	4483	1452	1213
	Raita	4172	1149	906	4480	1365	1111	4711	1392	1130	4478	1451	1213
	Quick Search	4090	1140	891	4525	1377	1097	4515	1392	1145	4368	1472	1203
	Boyer-Moore	4137	1086	838	4468	1279	1007	4622	1286	1033	4451	1374	1104
	Smith	3539	727	549	3845	870	672	3777	882	697	3707	952	755
	Turbo Boyer-Moore	4136	1086	837	4468	1279	1007	4620	1286	1033	4450	1374	1104

Table 3. *Comparison number results for Group 1 (x1000)*

		German			French			Italian			Spanish		
		Short	Middle	Long	Short	Middle	Long	Short	Middle	Long	Short	Middle	Long
Methods	Horspool	4474	1277	988	4466	1345	1059	4558	1427	1144	4565	1383	1095
	Raita	4471	1276	988	4462	1345	1058	4556	1426	1143	4561	1382	1094
	Quick Search	4375	1283	981	4416	1362	1056	4474	1416	1138	4447	1341	1066
	Boyer-Moore	4452	1202	918	4463	1261	961	4531	1329	1035	4511	1262	992
	Smith	3685	810	595	3717	855	643	3748	900	702	3741	849	654
	Turbo Boyer-Moore	4452	1201	918	4463	1261	961	4531	1329	1035	4510	1262	992

Table 4. *Comparison number results for Group 2 (x1000)*

Comparison numbers of algorithms are given in Table 3 and Table 4. When all tables are examined together it can be said that comparison numbers are generally related to search times. The algorithm structures and pre-processing phases of some algorithms can effect this relation on a small scale.

The performance distribution is the same for both long and middle length strings. Finnish showed the worst performance in long and middle length strings. This situation shows that the language structure of Finnish is complex. That is to say there are many turn backs while searching. Turkish is the language that has least number of turn backs. German is the second and French is the third in this subject.

CONCLUSION

This work shows that the speed of string matching algorithms can be changed when the language is changed. This main reason is the difference of the structures of languages. The performance of some languages are not good in long and middle length strings while they are good in short strings or vice versa. The results of this work can be used to determine which string matching algorithm is more appropriate for a particular language.

REFERENCES

[1] Lecroq, T., Experimental results on string matching algorithms. *Softw. Pract. Exp.,* 25(7):727-765, 1995

[2] Boyer, R. S., Moore, J. S., A fast string searching algorithm, *Communications of the ACM*. 20:762-772, 1977.

[3] Crochemore, M., Czumaj, A., Gasieniec, L., Jarominek, S., Lecroq, T., Plandowski, W., Rytter, W., Deux méthodes pour accélérer l'algorithme de Boyer-Moore, in *Théorie des Automates et Applications, Actes des 2e Journées Franco-Belges*, D. Krob ed., Rouen, France, 45-63, 1991.

[4] Horspool, R. N., Practical fast searching in strings, *Software - Practice & Experience*, 10(6):501-506, 1980.

[5] Sunday, D. M., A very fast substring search algorithm, *Communications of the ACM*, 33(8):132-142, 1990.

[6] Smith, P. D., Experiments with a very fast substring search algorithm, *Software - Practice & Experience,* 21(10):1065-1074, 1991.

[7] Raita, T., Tuning the Boyer-Moore-Horspool string searching algorithm, *Software - Practice & Experience*, 22(10):879-884, 1992.

[8] Say, B., Zeyrek, D., Oflazer, K., Özge, U., Development of a Corpus and a Treebank for Present-day Written Turkish, *Proceedings of the Eleventh International Conference of Turkish Linguistics*, 183-192, 2002.

[9] Charras, C., Lecroq, T., Handbook of Exact String Matching Algorithms, King's College London Publications, 2004.