Comparative Analysis of the Tsarev Combined Algorithm

Mikhail Sadovsky[1] , Dmitry Dokuchaev[2]

[1]Institute of Computational Modelling of the Siberian Branch of the Russian Academy of Sciences, 50/44 Akademgorodok, Krasnoyarsk, 660036, Russia
[2]Siberian Federal University, 79 Svobodny pr., Krasnoyarsk, 660041, Russia

Abstract

This paper presents the combined Tsarev algorithm and its comparison with the Brute Force algorithm in terms of time performance and efficiency for problems of searching for the longest pattern in sequences. In the present study experiments were made with different types of DNA sequences such as: Candida Albicans, Candida Dublinesis and Coronavirus.

Keywords

Pattern recognition, combined Tsarev algorithm, Brute Force algorithm, comparative analysis

1. Introduction

Searching for the longest pattern in sequences is a hard calculation problem with complexity increasing with the recurrence length. The search for the longest recurrent pattern in DNA sequences is an example of such problem.

To solve the problems, various approaches and methods, often in combination, are implemented. Here we use two algorithms: the Brute Force algorithm and the combined string searching algorithm proposed by S.Tsarev. This algorithm is a new non trivial combination of the Knuth-Morris-Pratt and Boyer-Moore algorithms.

We study the time performance of these algorithms in the cases when the number of recurrent patterns is not high. The combined algorithm proposed by Tsarev is rather efficient in finding long common subsequences (both for the exact matching and fuzzy one). The efficiency of this algorithm is provided by exclusion of a high number of subsequences from the comparison. The length of the sought subsequence is a free parameter of the algorithm not known to a researcher in advance. Thus, one has to estimate the expected subsequence length. In some cases, it can be easily estimated from the content of a problem; however, it is not always so.

2. Tsarev algorithm for a common subsequence search

Let us consider the problem of the longest pattern search using the combined Tsarev algorithm. Let N_1 and N_2 be two symbol sequences from a finite alphabet. In our cases, the alphabet comprises four symbols *A*, *C*, *G*, *T*. Find all comparatively long common subsequences occurred in them. "Comparatively long" means here that the length of a common subsequence (if any) is no smaller than $\max(N_1, N_2)\,/20$.

Let k be the length of the window for searching and L be the expected length of the pattern. Following the Tsarev algorithm, let us introduce the constraint $= int(\sqrt{L})$. Using the window let us split the sequence N_1 into the frequency dictionaries with the length k with the step k, and split the sequence N_2 into the dictionaries with the length k with the step *k-1*, respectively. One obtains two

SibDATA 2021: The 2nd Siberian Scientific Workshop on Data Analysis Technologies with Applications 2021, June 25, 2021, Krasnoyarsk, Russia

EMAIL: msad@icm.krasn.ru (M. Sadovsky)
ORCID: 0000–0002–1807–0715 (M. Sadovsky)

CEUR Workshop Proceedings (CEUR-WS.org)

sets of frequency dictionaries for each of the presented subsequences at the input. Assuming that there exists a common subsequence with the length, L then, in the frequency dictionaries with the length *k* and step *k* and the length *k* and step *k-1* there must be at least one coinciding word.

Let us make a comparison of the two sets of the frequency dictionaries following the principle "each to each" until we find a coincidence. Consider the case of the exact coincidence of the words with the length *k*. If the respective coincidence is found, we expand two words to the left and then, to the right, respectively, until the elements of the sequences N_1 and N_2 coincide. The obtained expansion will be the sought longest pattern.

Due to the fact that in many applied problems the value *L* is not known in advance, it becomes necessary to estimate it. This can be done by changing the value of the parameter *k*. Namely, if none of the words of the length *k* matches, one needs to decrease the parameter *k*. Decreasing the parameter will automatically decrease the expected value *L*. By repeating the search procedure, one can find the new longest pattern. The main purpose of this study is to describe a method for the suboptimal choice of the decrease character of the parameter *L*.

3. Choice and change of the parameters *k* and *L*

In the present study experiments were made with different types of DNA sequences: Candida Albicans, Candida Dublinesis, Coronavirus. These sets have DNA sequences of different length, which is manifested in the above introduced parameters of the Tsarev algorithm. The Coronavirus set was chosen for comparative analysis. This is due to the length of DNA sequences in this set being approximately equal to 30,000 symbols, which allows applying various algorithms to this set, including ineffective ones, such as Brute Force. This sequence set consists of 67 sequence copies.

A peculiarity of the Brute Force algorithm is that it does not require the introduction of additional parameters. The algorithm is based on the complete brute force search of all possible combinations of subsequences and their direct comparison. Obviously, this approach is ineffective and very costly in terms of the resources of the machine on which the calculations are performed. This algorithm can be applied to a set of Coronavirus sequences without huge time delays and one can obtain an accurate result in finding the longest subsequence. By applying this algorithm to a set of sequences, it was possible to obtain the longest accurate recurrent patterns, as well as to estimate the operating time indicators of the Brute Force algorithm.

The main problem here is the problem of choosing an efficient way to change the parameter k for implementing the Tsarev combined algorithm. In the case of Coronavirus genome analysis the value *L* was unknown. Therefore, at the first step the values $L_1 = 10^3$were chosen, which was approximately equal to $\max(N_1, N_2)/3$. Then, the window length was $k_1 = 100$. The chosen value *L* turned out to be too high: in fact, there are no identical subsequences of such a length in these genomes. A number of experiments was made using different ways of changing the parameter *k*, for example, $k_{i+1} = int(k_i/\sqrt{2})$, $k_{i+1} = int(0.8 * k_i)$, $k_{i+1} = int(0.9 * k_i)$. Having made the choice, we performed calculations according to the Tsarev algorithm described above.

During the experiments, the choice of the initial value L_1 and method of changing the parameter *k* was found to directly affect the search quality and the running time of the Tsarev algorithm. An incorrect choice of the parameter k (a too large value) can give a false effect of the Tsarev algorithm: if k turns out to be comparable with the real length of the longest common subsequence, then the coincidence of such words (of the k length) can occur at the very first step of the Tsarev algorithm, and there is no further expansion (contrary to the theoretical prediction). False triggering of the Tsarev algorithm can be misleading. To test the performance of the combined algorithm, a sufficiently long subsequence was extrapolated from N_1 into N_2. The latter evidences that the software implementation of the algorithm operates correctly.

4. Conclusion

The paper analyzes the combined Tsarev algorithm in comparison with the Brute Force algorithm. Experiments were carried out to evaluate the performance and efficiency of this algorithm, and its dependence on the input data. Particular attention is paid to the selection of parameters for the combined Tsarev algorithm in order to improve the performance of the algorithm taking into account the above criteria.

5. References

[1] A. Apostolico, R. Giancarlo, The Boyer-Moore-Galil string searching strategies revisited, SIAM Journal on Computing 15(1) (1986)95–105.

[2] R. S. Boyer, J. S. Moore, A fast string searching algorithm. Communications of the ACM, 20(10) (1977) 762–772.

[3] L. J. Guibas, A. M. Odlyzko, A new proof of the linearity of the Biter-Moore string searching algorithm, SIAM Journal on Computing. 9(4) (1980) 672–682. doi:10.1109/SFCS.1977.3.

[4] P. D. Smith, Experiments with a very fast substring search algorithm, Software: Practice and Experience 21(10) (1991) 1065–1074. doi:10.1002/spe.4380211006.

[5] S. P. Tsarev, M. G. Sadovsky, New error tolerant method for search of long repeats in DNA sequences. in: Proceedings of International Conference on Algorithm for Computational Biology, AlCoB' 2016, Springer, 2016, pp. 171–182. doi:10.1007/978-3-319-38827-4_14.