Poster H-19

Regular Expression Pattern Matching Using Suffix Array amd Its Application to Motif Search

Authors:
Aki Hasegawa (*Informatics Infrastructure Team, Genome Core Technology Facilities, Genomic Sciences Center, RIKEN*)
Akihiko Konagaya (*Informatics Infrastructure Team, Genome Core Technology Facilities, Genomic Sciences Center, RIKEN*)

Short Abstract: We present a fast regular expression pattern matching system using distributed suffix arrays. Our system enables to find out all occurrences of a pattern in seconds from massive sequences including human genome, nucleic acid, and amino acid. Performance of PROSITE patterns search from SwissProt and nr is measured and evaluated.

Long Abstract:

It is one of the most fundamental bioinformatics issues to grasp all occurrences of the biologically meaningful segments from massive biological sequences. However, today's rapid expansion of genome sequencing projects and increase of sequencing performance keep pushing the in silico sequence analysis to the next stage of higher performance processing. We propose a new high performance regular expression pattern matching system. The system enables to find all matches from massive biological sequences at a time using suffix arrays [3]. It showed that the search time is about 10 times faster than the classical finite automata approach in average.

Biologically meaningful segments; transcription factor binding sites in the genomic sequences, motifs in the primary structures of protein, etc.; can be expressed in several manners such as strings, patterns, stochastic profiles, etc. One of the most useful forms is the pattern written in regular expressions. PROSITE [1] is one of the databases compiling motifs specialized for protein families and domains. One of the motif-describing forms of PROSITE, “pattern” is a similar notation of the regular expression. Currently, more than 1,300 patterns have been registered and keep growing. Several locally executable programs; ps_scan (ScanProsite [2]), patmatdb [4], etc.; have been designed to search motifs using PROSITE patterns. In the classical approach, a given regular expression pattern is converted to an equivalent automaton to scan through the text of biological sequences to find the occurrences. This approach takes search time proportional to the length of pattern and text. We adopted another approach that use suffix array data structure. The suffix array is known as a space economical version of suffix tree. The suffix array is an array of the indexes representing the every suffix of a given text, where the indexes are sorted by their representing suffix in the lexicographic order. Once the suffix array is constructed before the search, we can use the suffix array any number of time. There exist several linear time construction algorithms and the implementation shows enough performance for practical use (Our implementation takes 12 minutes to construct a suffix array for the 975 MB of amino acid sequence text with our facility). It is known that if a suffix array SA of a text T of length m, and a string s of length n was given, all indexes of the occurrences of string s in the text T can be found in O(n log m) time in an interval of the suffix array SA (when an auxiliary array

is also given, O(n + log m) time can be achieved). We applied the suffix array data structure to the regular expression pattern matching. At first, a given regular expression pattern is translated to an internal byte code representation of a parse tree, and interval set of the suffix array is initialized to a set of one whole interval. Then, the interval set is refined according to the traversal of the parse tree. Finally, every index lie in each interval of the set gives occurrences of the match.

We implemented our algorithm on two GNU/Linux servers equipped with four-way Opteron 875 2.2 GHz CPUs and 64 GB memory. They are connected by Gigabit Ethernet. The core programs were written in C language. Our system takes a regular expression as input, and output information of the occurrences such as the location, abundance, neighboring substring. We evaluated our system with other two programs, ps_scan and patmatdb, on every elapsed time to find all the occurrences of the actual PROSITE patterns. The ps_scan is a Perl script specialized in PROSITE motif scanning. The patmatdb is a program in EMBOSS software package and is written in C language. Both of them are available and locally testable. Total 1,331 patterns were taken out of PROSITE database (r19.12), and were searched from SwissProt (SP) and NCBI nr (NR). The average times of five trials to complete the matching to each pattern were investigated (time for detailed output was excluded). The ps_scan took almost constant time for every pattern. The time for the SP and NR is almost proportional to the size of the text and the average times are 17.6 sec and 313 sec, respectively. Similarly, the patmatdb also took about constant time for every pattern. In this case, 13.8 sec (SP) and 214.3 sec (NR) are the average. While our system took varying times according to the complexity of the pattern, the averages are far short. They are 1.9 sec (SP) and 12.1 sec (NR). In the case of SP, 1,282 (96.3%) patterns out of total 1,331 patterns were finished within five seconds. Remarkably, 840 (63.1%) patterns only took less than or equal to one second. Similarly, in the case of NR, 1,119 (84.1%) patterns were finished within five seconds, and 508 (38.2%) were within one second. For our system, the most time consuming pattern was PS000613, that took 100.8 (1428.4) seconds for SP (NR), respectively.

For the most part of the PROSITE patterns, our method succeeded to find all the occurrences in the amino acid databases in seconds, definitely outperformed other methods to the best of our knowledge. Random access to the suffix array often causes high frequency of disk input/output operations, and considerably affects to search performance. Providing large memory capacity, large disk cache, and/or memory disks is a practical way to improve the search performance. Our system can deal with not only protein sequences but also nucleic acid sequences in multiple-genomic scale. We will develop various higher-order sequence analysis applications using our system as a core search engine in near future.

[1] Hulo,N., Bairoch,A., Bulliard,V., Cerutti,L., De Castro,E.,Langendijk-Genevaux,P.S., Pagni,M. and Sigrist,C.J.A. (2006) The PROSITE database, Nucleic Acids Res., 34, D227-D230.

[2] Gattiker,A., Gasteiger,E. and Bairoch,A. (2002) ScanProsite: a reference implementation of a PROSITE scanning tool, Applied Bioinformatics, 1, 107-108.

[3] Manber,U. and Myers,G. (1993) Suffix arrays: A new method for on-line string searches, SIAM Journal on Computing, 22, 935–948.

[4] Rice,P., Longden,I. and Bleasby,A. (2000) EMBOSS: The European Molecular Biology Open Software Suite, Trends Genet., 16, 276–277.