Efficient Discovery of Common Patterns in Sequences

Pavel P. Kuksa, Vladimir Pavlovic

Department of Computer Science
Rutgers University
{pkuksa, vladimir}@cs.rutgers.edu

Finding motifs or repeated patterns in data is of wide scientific interest [11, 8, 5, 6]. For example, elucidating motifs in DNA sequences is a critical first step in understanding biological processes as basic as the RNA transcription. There, the motifs can be used to identify promoters, the regions in DNA that facilitate the transcription. Finding motifs can be equally crucial for analyzing interactions between viruses and cells or identification of disease-linked patterns. Discovery of motifs in music sequences, text, or time series data is a fundamental, general means of summarizing, mining and understanding large volumes of data. In this work, we develop algorithms that (1) improve search efficiency compared to existing algorithms, (2) have complexity that does not depend on the alphabet set size, (3) are exact (exhaustive) and deterministic.

For the purpose of this study, motifs are (short) patterns that occur in an exact or *approximate* form in all or most of the strings in a data set. Consider a set of input strings S of size $N = |S|$ constructed from an alphabet Σ. The solution for the (k, m, Σ, N)-motif finding problem (Figure 1) is the set M of k-mers (substrings of length k), $M \subseteq \Sigma^k$, such that each motif $a \in M$, $|a| = k$, is at Hamming distance at most m from all (or almost all) strings $s \in S$.

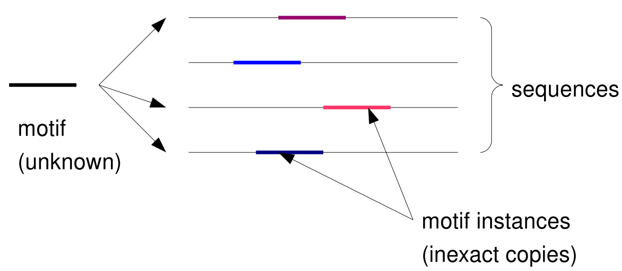

Figure 1: The motif search problem.

A number of approaches have been proposed for efficient motif search, including graph methods (WINNOWER) [8], explicit trie traversal (MITRA) [3], explicit mapping (Voting algorithms) [1], suffix trees [10], sorting and enumeration [2], etc. Existing exhaustive algorithms use *explicit* exploration of the motif space and require time proportional to the size of the *neighborhood* of a k-mer, i.e., the number of sequences at Hamming distance at most m away from it. This size, $V(m,k) = \sum_{i=0}^{m} \binom{k}{i}(|\Sigma| - 1)^i$, depends on the alphabet size, and can lead to high computational complexity and running times, as shown in Table 1.

Table 1: Exact algorithms for motif search

Algorithm	Time Complexity	Space Complexity
SPELLER [10]	$O(nN^2V(m,k))$	$O(nN^2/w)$
MITRA [3]	$O(knNV(m,k))$	$O(nNk)$
CENSUS [4]	$O(knNV(m,k))$	$O(nNk)$
Voting [1]	$O(nNV(m,k))$	$O(nV(m,k))$
RISOTTO [9]	$O(nN^2V(m,k))$	$O(nN^2)$
PMS [2]	$O(n^2NV(m,k))$	$O(n^2N)$

In contrast to existing exact exhaustive algorithms, we approach the problem of motif finding by performing an efficient *implicit* search. This search does not depend on the alphabet size, making the new algorithms applicable to many practical applications that require analysis of large-$|\Sigma|$ sequences. We build this approach upon our recent work [7] where we show how to *count* the number of k-mers shared by two sequences based on a combinatorial argument. Here we extend this argument to efficiently find a fixed set of *stems*, patterns with wildcards, that represent this shared set of k-mers. The number of stems necessary to describe the intersection of neighborhoods for two k-mers at Hamming distance $d(a,b)$ can be shown to be $\sum_{i=d-m}^{m} \binom{d}{i} \sum_{j=1}^{m-i} \binom{d-i}{j}$ and does not depend on the alphabet size.

Topic: data mining, pattern recognition, sequence modeling
Preference: oral/poster

Table 2: Running time as a function of alphabet size $|\Sigma|$. (k, m) instances denote implanted motifs of length k with up to m substitutions.

$\lvert\Sigma\rvert$	(9,2) instance		(11,3) instance		(13,4) instance		(15,5) instance	
	MITRA, s	Stemming, s	MITRA, s	Stemming, s	MITRA, s	Stemming, s	MITRA, s	Stemming, s
4	0.89	1.926	17.99	7.468	202.47	64.338	1835	423
20	8.39	0.637	1032.17	1.07	28905	5.247	-	45.66
50	89.82	0.633	12295.73	0.963	685015	2.244	-	27.26
100	265.94	0.645	-	0.967	>1 month	2.227	-	27.1

The main idea of our approach is to first construct a candidate set C which will include all motifs M but also some non-motifs, i.e. $M \subseteq C$, and then efficiently select true motifs from the candidate set. Given C, the complexity of motif finding is then proportional to its size: the motifs can be extracted from C by checking each candidate against the motif property, a task accomplished using $\binom{k}{m}$ rounds of counting sort. To generate C we collect the sets of stems which characterize the common neighbors of all observed pairs of k-mers (a, b). We call these sets the *seed sets*, $H(a, b)$. As alluded to before, finding each $H(a, b)$ is independent of the alphabet size. We construct seed sets for all pairs of k-mers a, b at Hamming distance of at most $2m$ from every string (i.e. potential motif instances).

We evaluated our algorithms on the planted motif problem, a task where synthetic motifs are injected in otherwise motif-less strings. For this problem, we follow the standard setting used in previous studies and synthesize $N = 20$ random strings of length $n = 600$ using iid, uniformly distributed symbols from an alphabet of size $|\Sigma|$. We then embed a copy (with up to m substitutions at random positions) of a motif at a random location in every string. Results in Table 2 show significant reduction in running times compared to state-of-the-art methods, especially for large-$|\Sigma|$ inputs. We are currently running experiments to evaluate our approach on variety of other data and tasks including music data for genre and artist identification. The output of our algorithm can be used as an input to other algorithms for more detailed analysis, for example, filtering based on significance, discriminative motif search using negative input sets, functional annotation, etc.

References

[1] Francis Y. L. Chin and Henry C. M. Leung. Voting algorithms for discovering long motifs. In *APBC*, pages 261–271, 2005.

[2] Jaime Davila, Sudha Balla, and Sanguthevar Rajasekaran. Fast and practical algorithms for planted (l, d) motif search. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(4):544–552, 2007.

[3] Eleazar Eskin and Pavel A. Pevzner. Finding composite regulatory patterns in DNA sequences. *Bioinformatics*, 18(suppl1):S354–363, 2002. http://www.ccls.columbia.edu/compbio/mitra/.

[4] Patricia A. Evans and Andrew D. Smith. Toward optimal motif enumeration. In *WADS*, pages 47–58, 2003.

[5] Jean-Marc Fellous, Paul H. E. Tiesinga, Peter J. Thomas, and Terrence J. Sejnowski. Discovering Spike Patterns in Neuronal Responses. *J. Neurosci.*, 24(12):2989–3001, 2004.

[6] Nebojsa Jojic, Vladimir Jojic, Brendan Frey, Christopher Meek, and David Heckerman. Using "epitomes" to model genetic diversity: Rational design of HIV vaccine cocktails. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 587–594. MIT Press, Cambridge, MA, 2006.

[7] Pavel Kuksa, Pai-Hsi Huang, and Vladimir Pavlovic. Scalable algorithms for string kernels with inexact matching. In *NIPS*, 2008.

[8] Pavel A. Pevzner and Sing-Hoi Sze. Combinatorial approaches to finding subtle signals in dna sequences. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 269–278. AAAI Press, 2000.

[9] Nadia Pisanti, Alexandra M. Carvalho, Laurent Marsan, and Marie-France Sagot. RISOTTO: Fast extraction of motifs with mismatches. In *LATIN*, pages 757–768, 2006.

[10] Marie-France Sagot. Spelling approximate repeated or common motifs using a suffix tree. In *LATIN '98: Proceedings of the Third Latin American Symposium on Theoretical Informatics*, pages 374–390, London, UK, 1998. Springer-Verlag.

[11] Eric P. Xing, Michael I. Jordan, Richard M. Karp, and Stuart Russell. A hierarchical Bayesian Markovian model for motifs in biopolymer sequences. In *In Proc. of Advances in Neural Information Processing Systems*, pages 200–3. MIT Press, 2003.