ECAI 2014
T. Schaub et al. (Eds.)

doi:10.3233/978-1-61499-419-0-1057

A Decomposition Approach for Discovering Discriminative Motifs in a Sequence Database[1]

David Lesaint [2] and **Deepak Mehta** [3] and **Barry O'Sullivan**[3] and **Vincent Vigneron**[2]

Abstract. This paper addresses the discovery of discriminative n-ary motifs in databases of labeled sequences. We consider databases made up of positive and negative sequences and define a motif as a set of patterns embedded in all positive sequences and subject to alignment constraints. We formulate constraints to eliminate redundant motifs and present a general constraint optimization framework to compute motifs that are exclusive to the positive sequences. We cast the discovery of closed and replication-free motifs in this framework and propose a two-stage approach whose last stage reduces to a minimum set covering problem. Experiments on protein sequence datasets demonstrate its efficiency.

1 Introduction

Constraint Programming is a recent alternative to ad-hoc algorithms for itemset and pattern mining problems [3, 1, 2]. We follow this approach and introduce a constraint optimization approach for mining discriminative motifs in databases of labeled sequences. We consider a database D of sequences consisting of a "positive class" D^+ and a "negative class" D^-. In this context, we are interested in computing an n-ary motif that is common and exclusive to the positive class. Informally, an n-ary motif is a set of n pattern embeddings subject to alignment constraints which we call c-blocks.

We define a pattern as a sequence of solid characters possibly interspersed with free characters called hashs and denoted #. For instance, sequence ADACDCEC embeds pattern A##C#C at locations 1 and 3. A block is a given embedding of a pattern in a sequence. A pattern may appear in different sequences of the database, the set of which is called the cover of the pattern. We say that a pattern is positive if its cover includes the positive class. A c-block is associated with a positive pattern and represents the choice of one block per sequence in the pattern's cover. By definition, a c-block ensures that the number and relative positioning of hashs in its pattern are preserved in each embedding sequence. This constraint is paramount to tackle multiple alignment problems as found, for instance, in protein sequence alignment where hashes are interpreted as evolutionary point mutations. A motif is a set of c-blocks and the cover of a motif is the intersection of the covers of its c-blocks.

Fig. 1 shows blocks for a database of three sequences. For instance, AA is a positive pattern; $(\texttt{AA}, 6, s_1)$, $(\texttt{AA}, 8, s_2)$ and $(\texttt{AA}, 1, s_3)$ are blocks. $(\texttt{AA}, \{6, 8, 1\})$ and $(\texttt{AAC\#E}, \{1, 1\})$ are c-blocks but $(\texttt{YE}, \{4\})$ is not as YE is not positive. A possible motif is $\{(\texttt{A\#C}, \{1, 1, 1\}), (\texttt{AAC}, \{1, 1, 1\}), (\texttt{AAC\#E}, \{1, 1\}), (\texttt{AA}, \{6, 8, 1\})\}$ whose cover is D^+.

	s_1	s_2	s_3
Sequences	AACYEAA	AACZEZZAA	AAC
Blocks	A#C	A#C	A#C
	AAC	AAC	AAC
	AAC#E	AAC#E	
	AA	AA	AA

Figure 1. Possible blocks for a database $D^+ = \{s_1, s_2\}$ and $D^- = \{s_3\}$.

We are interested in motifs that satisfy every required constraint on the positive class (commonality) but falsify any of them when interpretation extends to negative sequences (discriminativeness). We propose a generic framework for this motif discovery problem (MDP) where each constraint has to be interpreted over the positive class (commonality computation) as well as on any extension of the positive class (exclusivity computation). We say that a motif excludes a negative sequence if it does not cover the sequence or if any motif with the same patterns and embeddings over the positive class violates a constraint when interpretation extends to the sequence. A solution to a MDP is a motif that satisfies all constraints on the positive class and excludes the largest number of negative sequences. The framework allows additional optimization criteria (e.g., minimal motif cardinality) via a lexicographic objective function.

We present non-replication, non-inclusion, coverage and closure constraints on motifs to eliminate redundant motifs or address domain-specific requirements. Replication is a pattern subsumption constraint. Informally, a pattern p replicates a pattern p' if p is obtained from p' by replacing hashs with solid characters or adding new characters and possibly hashs left or right. For instance, pattern AAC#E replicates AAC which itself replicates A#C. Replication holds between two c-blocks if it does between the patterns. Non-replication simply requires that a motif be replication-free. Inclusion is a block subsumption constraint. Informally, a block b includes a block b' if b is obtained from b' by substituting hashs or app-/pre-pending new characters with possibly hashs. For instance, block $(\texttt{AAC\#E}, 1, s_1)$ includes $(\texttt{AAC}, 1, s_1)$ which itself includes $(\texttt{A\#C}, 1, s_1)$. Inclusion holds between two c-blocks over $X \subseteq D$ if block inclusion holds on one of their common sequences in X, if any. Non-inclusion over X requires that a motif be inclusion-free over X.

Coverage over $X \subseteq D$ simply requires that a motif covers X. Note that every motif satisfies this constraint on the positive class but may violate it on any extension. Closure is the dual of coverage and ensures no c-block can be extended in a motif. Informally, a c-block is closed over $X \subseteq D$ if there is no other c-block including it in each sequence common to X and its cover. For instance, c-block $(\texttt{AAC}, \{1, 1, 1\})$ in Fig.1

[1] This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289 and in part from Ulysses 2013 research award.
[2] LERIA, Université d'Angers, France, {lesaint,vigneron}@info.univ-angers.fr
[3] Insight Centre for Data Analytics, University College Cork, Ireland, {d.mehta,b.osullivan}@4c.ucc.ie

is not closed over D^+ as $(\texttt{AAC\#E}, \{1,1,1\})$ extends it but it is closed over D. Motif $\{(\texttt{AAC}, \{1,1,1\}), (\texttt{AAC\#E}, \{1,1\})\}$ covers D^+ but not D and it is neither closed nor inclusion-free over its cover. $\{(\texttt{AAC}, \{1,1,1\}), (\texttt{AA}, \{6,8,1\})\}$ covers D and is inclusion-free but is not closed, neither replication-free over D^+. $\{(\texttt{AAC\#E}, \{1,1\}), (\texttt{AA}, \{6,8,1\})\}$ covers D^+ and is both inclusion-free and closed over its cover but it is not replication-free.

We cast one particular problem in this framework called Replication-free MDP (RMDP). The RMDP requires motif closure and non-replication over the positive class and minimum coverage over the negative class. The RMDP also prioritizes maximum exclusivity over minimum slack (slack is the maximum number of consecutive hashs in the c-blocks of a motif) and minimum cardinality. A solution to a RMDP is a motif that is replication-free, closed over the positive class and that covers the minimum number of negative sequences. This definition motivates a two-stage approach where closed c-blocks are computed first before addressing minimum negative coverage as a minimum set covering problem.

2 A Decomposition Approach for RMDP

This section describes an approach to solve the RMDP. From a computational viewpoint, the main bottleneck is that a solution motif may contain an exponential number of c-blocks. This contrasts with itemset or single pattern computation problems as in that case the size of the solution is bounded by the maximum number of items in any transaction. In our case however, the number of patterns may be exponential in number and modelling the whole problem as a monolithic constraint optimization problem is space-wise challenging. Therefore, we propose a two-step approach where first we compute an inclusion-free set of closed c-blocks over D^+ and then extract an optimal motif.

As memory requirements may remain prohibitive, we implement a lazy approach where the value of slack is incremented only if required. More precisely, we vary the value of slack starting from 0 to a maximum allowed value, and compute the c-blocks and an optimal motif at each step. We exit the loop as soon as all negative sequences are excluded. We describe below the method for computing an optimal motif for a given value of slack.

We first compute closed c-blocks in three successive steps:

1. We start by computing all "positive blocks" of length 2, i.e., blocks whose patterns are positive and only have 2 solid characters. This is done by selecting the shortest sequence of the positive class and then verifying for each valid block of length 2 whether its pattern is positive. Once done, we know the minimal length positive blocks and their locations in this sequence. This step is $\mathcal{O}(n\,\underline{d}^2\,\overline{d})$ where $n = |D^+|$ and $\underline{d}$ and $\overline{d}$ are the minimum and maximum lengths of the sequences in D^+, respectively.
2. We then compute all inclusion-maximal positive blocks for the shortest sequence, i.e., positive blocks that are not included in any other positive block. The idea is to build a lattice of blocks bottom-up based on the inclusion relation. The procedure is iterative and starts with the minimal length positive blocks. Each iteration merges every possible pair of blocks verifying that the resulting block is positive and within the allowed slack before eliminating any block that is not inclusion-maximal in the set computed so far. This step is $\mathcal{O}(\underline{d}\,m^2(m + n\,\overline{d}))$ where m is the maximum number of blocks occuring in any level of the lattice.
3. We finally compute a set of inclusion-maximal closed c-blocks by extending each inclusion-maximal positive block and removing any included c-block along the way. This step is $\mathcal{O}(mn\,(\overline{d} + m))$.

Table 1. Results of LEAP proteins obtained using RMPD

cid	#proteins	$\underline{d}$	$\overline{d}$	#nonexp	card	slack	length	time1	time2
1	177	117	507	35	13	14	29	66775	237
2	96	122	338	3	9	27	25	276043	402
3	29	86	186	0	1	0	6	92	124
4	83	81	625	690	1	3	2	6745	8
5	60	83	217	0	3	2	8	311	121
6	202	66	843	258	3	6	6	4079	18
7	53	95	341	0	1	4	4	127	23
8	184	136	411	84	2	1	4	7016	17
9	67	78	144	0	1	2	4	45	15
10	76	88	173	2	7	46	18	49962	674
11	24	159	278	0	5	1	13	358	292
12	15	71	117	0	2	0	6	55	57

We then have to extract a replication-free and optimal motif from this set of c-blocks. We address this subproblem as a constraint optimization problem. Let A be the set of the patterns associated with the set of inclusion-maximal closed c-blocks for D^+. Notice that A now contains replication-free patterns. For each sequence $s \in D^-$, we compute the subset E_s of A that does not cover s. The exclusion count is computed by checking whether E_s is empty or not. The objective is to select a minimum number of patterns from A such that at least one from each non-empty set E_s is selected.

3 Empirical Results

We present results to demonstrate the effectiveness of our approach. We investigated with two databases: Late Embryogenesis Abundant Proteins (LEAP) and Small Heat Shock Proteins (SHSP). The LEAP database [4] contains 1066 proteins partitioned into 12 classes. We only present results for LEAP in Table 1 due to lack of space. In the table, `cid` denotes the id of the class, `#proteins` the number of proteins in each class, $\underline{d}$ the minimum size of the protein while $\overline{d}$ denotes the maximum size of the protein of a given class. `nonexp` denotes the number of foreign proteins (i.e., negative sequences) that an optimal motif for a given class was not able to exclude. We found that there were no motifs that could exclude all the foreign proteins for 6 out of 12 classes of LEAP. The slack, cardinality and length measures of the motifs are also depicted in the columns labeled as `slack`, `card`, and `length`. Computation time is given in milliseconds. The times for computing the inclusion-maximal closed c-blocks and an optimal motif for a given class are shown in the columns `time1` and `time2`, respectively. Overall, the results suggest that the presented approach is scalable for handling large instances.

4 Conclusion

We have introduced a constraint optimization framework to compute discriminative n-ary motifs in databases of labelled sequences. Future work involves casting new types of constraints on motifs, improving algorithmic efficiency and scalability, carrying out comparisons with ad-hoc algorithms in Bioinformatics (e.g., multiple sequence alignment), and addressing other data mining tasks.

REFERENCES

[1] E. Coquery, S. Jabbour, L. Sais, and Y. Salhi, 'A SAT-Based approach for discovering frequent, closed and maximal patterns in a sequence.', in *ECAI*, pp. 258–263, (2012).
[2] E. Coquery, S. Jabbour, and L. Sas, 'A constraint programming approach for enumerating motifs in a sequence', in *ICDMW*, pp. 1091–1097, (2011).
[3] T. Guns, S. Nijssen, and L. De Raedt, 'Itemset mining: A constraint programming perspective', *Artificial Intelligence*, **175**, (2011).
[4] G. Hunault and E. Jaspard. The late embryogenesis abundant proteins DataBase, 2013.