DOCTORAATSPROEFSCHRIFT

2011 | School voor Informatietechnologie
Kennistechnologie, Informatica, Wiskunde, ICT

Mining Correlated Motif Pairs from Protein-Protein Interaction Networks

Proefschrift voorgelegd tot het behalen van de graad van
Doctor in de Wetenschappen, Informatica, te verdedigen door:

Peter BOYEN

Promotor: prof. dr. Frank Neven
Copromotor: dr. Dries Van Dyck

Maastricht University

universiteit hasselt

D/2011/2451/53

Acknowledgements

First and foremost, I offer my sincerest gratitude to my advisor, Frank Neven. He has guided and supported me, let me work in my own way when appropriate, and pushed me when I needed pushing. He has taught me much about what makes good research, a good paper, a good scientist. Without him I would have never started, let alone finished this thesis.

A special mention goes to Dries Van Dyck, my co-advisor. His passion for the work was inspirational and I have learned a lot from his seemingly endless flow of ideas.

Aalt-Jan van Dijk has been helping me to understand and process the biological side of my thesis since the very beginning. So I would like to thank him as well.

I would like to thank everybody who was important to the successful realization of this thesis, inspirationally, as well as conversationally.

It has been a pleasure working with and among the members of Infolab (past and present). In my time here, many of you have become not only good colleagues, but good friends as well.

I am grateful for the faith my friends have had in me and also for their inquiries to my progress.

Last but not least, I thank my family, in particular my parents, for supporting me throughout *all* my university studies.

Diepenbeek, December 2011

Contents

1

Introduction

1.1 Motivation

Proteins form the basic building blocks of all living organisms. They perform all basic functions to which end they must often physically bind together, forming different complexes. Large-scale biological networks are available for several organisms, incorporating an ever increasing number of interactions within the same organism [STdS$^+$08]. Such data demonstrate how proteins function as part of an interaction network, but provide no insight into how interactions are encoded in protein sequences. These interactions occur via connections between parts of the respective proteins, called *binding sites*. Protein residues (amino acids) involved in such connections are called *binding residues* or *interface residues*. Knowledge of binding sites is critical for the prediction of novel protein-protein interactions (PPI's), understanding of the evolution of protein interactions, or for the creation of drugs to target a specific protein. Unfortunately, the discovery of these sites requires laborious and expensive biological experiments. In fact, it is estimated that at the present rate of protein structure determination, it would take 20 years to determine all interaction types using current experimental techniques [AR04]. Moreover, even if this would be accomplished, one would still have to deal with predicting for a given interacting sequence to what interaction type it adheres.

Therefore, we have developed a novel methodology to help us find potential binding sites computationally, to complement and guide the effort into their discovery. An implementation as a tool is discussed in Appendix A.

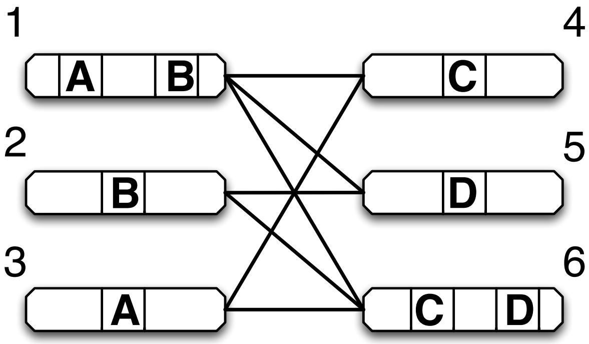


Figure 1.1: Compatible binding sites {A, C} and {B, D} as correlated motifs in sequences.

1.2 Problem statement

For the above reasons, we wish to determine if it is possible to locate protein binding sites using only widely available information, such as the Protein-Protein Interaction (PPI-)network and the sequences of its proteins, as input.

1.2.1 Correlated Motif Mining (CMM)

Correlated Motif Mining (CMM) is an approach to identify binding sites by looking for a consensus pattern, called a *motif*, in one set of proteins (most of) which interact with (almost) all proteins which contain another consensus pattern. If so, both patterns are likely to represent a part of the surface of the molecules which makes interactions possible through a physical binding.

We represent a PPI-network as a graph where the nodes are proteins, which are labelled with their amino acid sequence, and there is an edge between two nodes if their corresponding proteins interact. For example, in Figure 1.1 we see the graph representation of a PPI-network with 6 nodes and 7 interactions. Then, the problem becomes identifying those patterns where the nodes whose labels adhere to the patterns form a large (quasi-)biclique within the graph representation of the network. For instance, in Figure 1.1 the patterns {A, C} and {B, D} represent two such correlated motifs. In particular, there is an undirected edge between two protein sequences when the first one contains motif A, and the second one motif C, and similarly for motif B and D.

Several computational approaches have been proposed to locate binding sites through the mining of overrepresented pairs of motifs in the sequences of interacting proteins [LSY$^+$07, LLW06, LLLW07, LSLW08, THSN06]. Currently, despite the development of these algorithms, it is unclear which fraction of interfaces can be described by such correlated motifs. However, results clearly indicate that correlated motifs do contain information about interfaces [LSY$^+$07, LLW06, LLLW07, LSLW08, THSN06].

Methods for CMM can be subdivided into two classes:

- interaction-driven [LLW06, LLLW07, LSLW08], and
- motif-driven approaches [LSY$^+$07, THSN06].

Interaction-driven methods mine for (quasi-)bicliques, i.e., subsets of vertices for which (almost) every vertex from one set is connected to (almost) all vertices of the other set. Such subgraphs exhibit a type of all-versus-all (or most-versus-most) interaction. A motif pair representing the corresponding interaction sites is then derived from the sequence carried by the vertices. The motif-driven approach, in contrast, starts from candidate motif pairs whose support is then evaluated in the network. Although both approaches have shown to produce biologically meaningful results, we study the second approach as it has several conceptual advantages over the first:

- motif pairs are mined directly, not derived;
- *all* proteins containing one of the motifs, and not a subset, are taken into account; and,
- if the interactions between two sets of proteins are a consequence of multiple compatible binding sites, such as $\{\mathtt{A}, \mathtt{C}\}$ and $\{\mathtt{B}, \mathtt{D}\}$ in Figure 1.1, the interaction-driven method necessarily merges them into one motif pair.

Currently, only two techniques have been introduced and implemented for the motif-driven approach towards CMM. Unfortunately, both methods differ not only in the mining method but also in the used notion of support for correlated motifs. The first method by Tan et al. [THSN06], called D-STAR, uses a χ^2-based scoring function to determine the support, but the underlying mining method does not scale to networks containing more than 250 proteins. As contemporary biological networks contain up to thousands of proteins (see Section 2.7), scalability is an increasingly important issue. The second method, called MotifHeuristics, employs a different, probabilistically motivated notion of support called p-score. This method is developed by Leung et al. [LSY$^+$07] and does scale to larger networks. Although the authors argue in their paper that MotifHeuristics is superior to D-STAR, it remains unclear if the latter is due to the different support measure or the underlying mining method.

Given that biological data is both noisy and incomplete, we cannot simply look for the motif pairs selecting the largest bicliques. We must somehow weigh completeness against size. To do that, we present a thorough, empirical study of the effectiveness of various existing and new notions of support for correlated motifs. We evaluate them in terms of precision and recall on artificial networks with implanted motifs at different noise levels. These experiments clearly show that a χ^2-based support measure is vastly superior in discovering highly interaction-descriptive motif pairs.

We formally prove, under reasonable assumptions concerning the used notion of support, the complexity of CMM is NP-hard and its associated decision problem is NP-complete. In particular, we show our χ^2-based support measure adheres to these assumptions. We therefore approach the problem as a combinatorial optimization problem.

More specifically, we present SLIDER, a generic metaheuristic containing two steepest ascent methods (M-SLIDER and SEQ-SLIDER), the key components of which are their neighborhood functions, based on viewing a motif as a window that slides over the amino acid sequence of one of the proteins. In contrast with more common neighborhood functions, they have a clear biological interpretation: they are based on the philosophy that if a motif overlaps with part of a binding site in a sequence, it should be able to slide towards the binding site in a few steps. So both neighborhood functions want to find neighboring motifs that could be close to each other on actual proteins. The difference is that M-SLIDER considers as neighbors of a motif all motifs which could theoretically be near it on any protein whereas SEQ-SLIDER only takes motifs actually nearby on a single selected protein. Although SLIDER can be used with an arbitrary support measure, we use the χ^2-based support measure, as our empirical study clearly indicates this is the best support measure known so far.

We validate SLIDER by showing its methods outperform, in both speed and accuracy, all existing motif-driven approaches on retrieving implanted motif pairs from artificial networks. When the data follows our model, SLIDER performs near optimal, finding almost all implanted motif pairs in only a short period of time. Unfortunately, as an effect of motifs in biological networks not adhering to our strict model of motifs, there is a large amount of motifs that covers a very similar, if not identical, part of the network. As a result, the best scoring motif pairs often refer to highly similar selected networks and to get a decent coverage of the network, we would need to incorporate an unmanageable amount of motif pairs. So, even though SLIDER is successful in finding the best motif pairs according to a support measure, the resulting binding sites all occur in proteins in the small, densest part of the network. Though knowledge of these binding sites is useful, it is important for biological tests to know as many as possible throughout the entire network.

1.2.2 Correlated Motif Covering (CMC)

To address the low recall of SLIDER, we investigate a radically different approach where instead of scoring the power of a pattern to explain interactions on an individual basis, we score the explicative power of a set of motif pairs as a whole. In essence, we target a minimal set of motif pairs $\mathcal{M}$ which covers a maximal part of the network. To balance the minimality of $\mathcal{M}$ with the 'goodness' of $\mathcal{M}$ representing the network, we employ a minimum description length measure to score sets of motif pairs. We formalize the latter as the Correlated Motif Covering (CMC) problem.

We analyze CMC from a computational viewpoint. We prove that CMC is NP-hard and that it belongs to a class of problems for which it is practically impossible to find "good" approximation algorithms. Here, "good" means approximations with a sub-square root approximation rate. Specifically, we reduce the Red-Blue Set Cover (RBSC) problem, a well-known combinatorial problem, to CMC using polynomial time reductions that maintain constant factors of approximation. On the other hand, we exhibit a reduction from CMC to the *Weighted* Red-Blue Set Cover (WRBSC) problem allowing the transfer of known approximation algorithms for the latter to CMC. Unfortunately, due to the large sizes of biological networks, none of the available algorithms for WRBSC remain feasible when directly adapted for CMC. We therefore introduce the novel heuristic CMC-approx for CMC which is based on an approximation algorithm by Peleg [Pel07] for WRBSC. Although the adaptation no longer guarantees the same approximation rate, we experimentally assess its merits by comparing it to two alternative algorithms for CMC. The first alternative algorithm is SEQ-SLIDER, which is considered as a baseline as it simply mines for the k best motif pairs in a network which are then considered to form a cover. The second alternative is the expected greedy algorithmic solution to CMC. We refer to the latter algorithm as CMC-greedy.

In an experimental validation on biological networks, we confirm that the network coverage of CMC-approx is indeed much larger than that of SEQ-SLIDER and CMC-greedy at the expense of a slightly increased running time. To further evaluate the biological relevance of CMC-approx, we tested the effectiveness of the derived motif pairs in several scenarios. The first scenario is the prediction of protein interactions from 2D-sequences, where CMC-approx is shown to slightly outperform the two other algorithms. In the second scenario, we determine the overlap of our found motif pairs with actual interaction sites in the human and yeast network for which there is 3D-structure information available. We obtain that the coverage (the number of proteins and interactions covered) drastically increases (at the expense of a slightly lower precision) compared to the baseline, thereby greatly improving the utility for experimental biologists who want to predict binding sites to perform further experimental studies and for whom it obviously is very important to obtain predictions for as many proteins as possible.

1.2.3 Applications and future work

Finally, we conclude with some applications and show the potential of several possible avenues of future work:

- The potential of expanding SLIDER with biological knowledge is investigated. First, we use the likelihood that an evolutionary change occurs from one amino acid to another to define a similarity between them. We can use this similarity to make motifs more flexible, and allow them to select areas in the protein sequence that are a biological near-match. We can also employ different types of information that give for each amino acid a likelihood of being part of a binding site, e.g. Relative Solvent Accessibility (RSA) and conservation score. This allows us to make sure that unlikely regions for binding sites are not taken into account in our calculations. Finally, most interactions are known to occur within known subsections of proteins called domains. Thus, by excluding motifs occurring elsewhere, we increase our chances of finding a binding site. We show that by using these various forms of biological data, we can improve the ability of SLIDER to find biologically meaningful results.

- An attempt was made to predict binding sites for higher order complexes, but our simple model, which extends the motif pair into a motif triplet, is unable to capture what is really going on. It would be interesting to conduct further research into this using more complex models.

- We also show that it is possible to adapt our algorithms to take into account not only binary (yes/no) interaction data, but interaction probabilities. Interaction probabilities can be calculated by combining information from different sources and/or by setting probabilities for each method of interaction detection.

- Cross-species motif pair mining can be used as a filtering step to increase accuracy. That is, we provide evidence for the hypothesis that motif positions which are found in more than one species through mining of motif pairs are more likely to be interface residues. This means that a cross-species comparison could be used to filter noise from the predictions. We use our results in an extended test to verify if we can use data from multiple species to predict binding sites with increased certainty.

1.3 Related work

1.3.1 Pattern mining

Many pattern mining applications suffer from pattern explosion. That is, when an interestingness measure is chosen to retrieve novel results, such a large amount of patterns is obtained as to eclipse the data itself by several orders of magnitude. This is mostly due to the locality of the minimum support constraint, which causes patterns to be added to the results set independent of patterns already found, causing great redundancy in the results. This problem is often alleviated, but not solved, by looking for special types of patterns, such as closed [PBTL99] or maximal [Bay98] patterns, that encompass a group of redundant patterns. To provide less redundant patterns, there has been much research toward finding small sets of representative macro-patterns for databases. Bringmann et al. [BZ09] use a post-processing technique to filter item set mining results and remove redundant item sets. They show that this reduction of the result set can improve the quality of classification. Geerts et al. [GGM04] consider the top-k tiling problem. Here, tiles are frequent item sets that cover a large area of the transaction database. Their top-k tiles problem looks for the k (possibly overlapping) tiles that together cover the largest surface in the database. Vreeken et al. [VLS11] introduce KRIMP, which uses MDL to find the item sets that can best be used to compress a database. So far, all these techniques only select item sets that occur exactly within the data. Xiang et al. [XJFD08] adapt tiling to allow for item sets with a few missing items. They focus, however, on covering every cell in the database, while we are more interested in informative patterns which give us insight into the data. None of these techniques focuses on biological data.

1.3.2 Binding site prediction

The prediction of binding sites is mostly used in two fields, the prediction of transcription factor binding sites, or of protein binding sites.

Transcription factor binding sites are the locations on strands of DNA where proteins bind to activate or inhibit its transcription. This search is typically performed by looking for statistically overrepresented motifs among DNA strands of coregulated genes [DD07].

Several tools exist for predicting protein binding sites [THSN06, LSY$^+$07, LLW06, LLLW07, LSLW08, BW05, LZLZ06, LVT07, MJ06, NSO$^+$07, NRS04, OKA$^+$05, PM07, SPNW04, SGG03, TQZ07, ZS01, ZQ07, vDMF$^+$10]. They use several types of information, e.g. amino acid sequence, solvent accessibility, conservation, and many others. This information is not always available, especially for newly sequenced proteins. Many learning methods are used to predict the binding sites, such as SVMs or neural networks. One, in particular, named PRISM [OKA+05] should be mentioned as it also uses motif pairs. Using known pairs of binding sites, it looks for a structural match for the left and right binding sites. If found, it predicts interaction through those sites.

All these works focus on predicting binding sites separately. To the best of our knowledge, this is the first attempt to look for a grand explaining set of binding sites for a protein-protein interaction network.

1.4 Outline

In chapter 2, we formally define the problem of Correlated Motif Mining (CMM). We obtain that CMM is an NP-hard problem and present the generic metaheuristic SLIDER which employs the χ^2-based support measure. We show that SLIDER outperforms existing motif-driven CMM methods and scales to large protein-protein interaction networks.

In chapter 3, we formally define Correlated motif covering (CMC). We show CMC to be NP-hard and prove its hardness of approximation. We create a functional heuristic for CMC, called CMC-approx, and experimentally assess its performance and biological relevance.

In chapter 4, we show some extensions of SLIDER and some applications of our results. We extend SLIDER by providing it with different forms of biological data. We also provide a version that can work with motif triplets instead of motif pairs and is able to use interaction probabilities instead of binary interaction data. Finally, we check that we can use results for various species to reduce noise in the results and increase our certainty of several binding sites.

In Appendix A, we give a short overview of the functionality of the software created as part of this work.

Finally, a Dutch summary is provided in Appendix B.

1.5 Publications

Chapter 2 is based on work presented at ICDM 2009 [BNVD+09], and the extended version published in ACM Transactions on Computational Biology and Bioinformatics [BVDN+11]. Chapter 3 is based on work submitted to ACM Transactions on Computational Biology and Bioinformatics [BNVD+on]. Chapter 4 is based on work yet to be published.

2

Mining the best motif pairs according to a support measure

2.1 Introduction

We consider the problem of finding those pairs of patterns, called *motifs*, where (most of) the proteins containing one motif interact with (almost) all of the proteins containing the other. These motifs are expected to indicate the locations of binding sites. We formalize this problem as the Correlated Motif Mining (CMM) problem in Section 2.2. Next, we discuss two support measures, the p-score and the χ^2-based support measure, in Section 2.3. In Section 2.4, we prove CMM to be NP-hard for a large class of support measures. As a result, we reformulate the search for correlated motifs as a combinatorial optimization problem in Section 2.5 and introduce the generic SLIDER metaheuristic, along with its two methods M-SLIDER and SEQ-SLIDER, based on the idea that we can slide a detector for the optimal motif along the sequence. We discuss SLIDER's space and time complexity in Section 2.6. To validate SLIDER's capability in retrieving the descriptive motif pairs, we create artificial data sets where each interaction is the result of an implanted motif pair. In Section 2.7, we introduce these artificial data sets as well as the biological data sets on which the effectiveness of our methods is then assessed in Section 2.8. Our results show that SLIDER outperforms existing methods in both accuracy and speed.

Notation	Explanation
f	a support measure
X, Y	motifs
V_X	set of proteins containing motif X
$E_{X,Y}$	set of interactions between V_X and V_Y
$G_{X,Y}$	subnetwork of G with vertices $V_X \cup V_Y$ and edges E_{XY}
$E^{max}_{\|V_X\|,[V_Y\|,\|V_X \cap V_Y\|}$	maximum amount of edges in $G_{X,Y}$, given $\|V_X\|, \|V_Y\|, \|V_X \cap V_Y\|$
$\overline{E_{X,Y}}$	expected amount of edges in $G_{X,Y}$
$M = \{X, Y\}$	a motif pair
$M_{k_X, k_Y, k_{X,Y}}$	complete motif pair
$\mathcal{M}$	a set of motif pairs
u, v	proteins
$\{u, v\}$	interaction between protein u and protein v
$\lambda(u)$	protein sequence of u
$pos(X, u)$	set of positions of occurrences of X in $\lambda(u)$
$N(X)$, $N(\{X, Y\})$	neighbor of a motif or motif pair
S	a set of objects
$\mathcal{S}$	a set of sets

Figure 2.1: Table of major notation.

Finally, we conclude in Section 2.9 with a discussion on the choice for steepest ascent and the difference with frequent motif mining.

For the reader's convenience, a table of major notation has been added in Figure 2.1.

2.2 Correlated motif mining

A *Protein-Protein Interaction (PPI-)network* can be represented as a labeled graph $G = (V, E, \lambda)$, with protein set V, interaction set $E \subseteq V \times V$[1], and a labeling $\lambda : V \rightarrow L$, with L a finite alphabet of labels. As we label every protein with its amino acid sequence, the label function λ maps each vertex $v \in V$ to a string $\lambda(v)$ over the alphabet $\Sigma = \{\texttt{A}, \ldots, \texttt{Z}\} \setminus \{\texttt{B}, \texttt{J}, \texttt{O}, \texttt{U}, \texttt{X}, \texttt{Z}\}$.

An *(ℓ, d)-motif* is a string of length ℓ over the alphabet $\Sigma \cup \{\texttt{x}\}$ containing exactly d `x`-characters. The character `x` is interpreted as a wildcard-symbol, i.e., it matches with any character of Σ. For instance, `GAQPRNMY` matches the

[1] Throughout this work, we use $V \times V$ to mean $\{\{u, v\} \mid u, v \in V\}$.

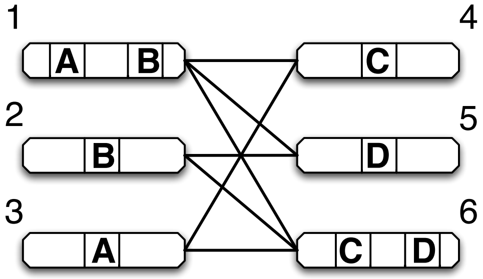


Figure 2.2: An example protein-protein interaction network (repeated).

$(8, 3)$-motif `GxQPxNxY`. We do not allow motifs to start with a wildcard-symbol, eliminating a lot of redundant motifs. Note that motifs starting and ending with a wildcard character are redundant because, in practice, the amino acid sequences are much longer than the motifs.

A protein *contains* an (ℓ, d)-motif X if its amino acid sequence contains a substring of length ℓ that matches X. A protein pair *contains* a motif pair, if one protein contains one motif, and the other protein contains the other. Conversely, we say a motif (pair) *selects* a protein (pair) if the protein (pair) contains it.

A motif pair $M = \{X, Y\}$ now selects a subnetwork of a PPI-network $G = (V, E, \lambda)$ as follows. Let $V_X = \{v \in V \mid v \text{ contains } X\}$, be the set of proteins in the network containing the motif X, and $E_M = E_{X,Y} = \{\{u, v\} \in E \mid u \in V_X \wedge v \in V_Y\}$ be the set of interactions between proteins containing X and proteins containing Y. Similarly, we define the set of anti-edges $A_M = A_{X,Y} = \{\{u, v\} \notin E \mid u \in V_X \wedge v \in V_Y\}$, the set of protein pairs selected by $\{X, Y\}$ without interactions (the false postive interactions for $\{X, Y\}$). The subgraph $G_M = G_{X,Y}$ selected by $\{X, Y\}$ is then

$$G_{X,Y} := (V_X \cup V_Y, E_{X,Y}, \lambda_{|V_X \cup V_Y})$$

with $\lambda_{|V_X \cup V_Y}$ the restriction of λ to $V_X \cup V_Y$. Note that V_X and V_Y can share proteins.

The amount of (ℓ, d)-motif pairs is the amount of ways you can pick 2 motifs out of the amount of (ℓ, d)-motifs, as given by

$$\mathrm{MP}(\ell, d) = \binom{\binom{\ell-1}{d}|\Sigma|^{(\ell-d)}}{2},$$

which, e.g. means there are approximately 6×10^{15} $(8, 3)$-motif pairs.

By $|G|$ we denote the size of G, defined as $|V| + |E| + \sum_{v \in V} |\lambda(v)|$, where $|S|$ denotes the number of elements in a set S and $|s|$ denotes the length of a string s.

In Figure 2.2, we see an example PPI-network $G = (V, E, \lambda)$, with λ partially given in the figure. Proteins 1 and 3 contain the motif A, and the motif A selects those proteins. If you pair either protein 1 or 2, with protein 4, both protein pairs contain the motif pair $\{B, C\}$, though only one has an interaction. The subgraph $G_{B,C}$ consists of proteins 1, 2 (containing B), 4, 6 (containing C) and the three edges between them.

A *support measure* f is a function mapping a motif pair $\{X, Y\}$ and a graph G to a positive real number $f(\{X, Y\}, G)$. We refer to $f(\{X, Y\}, G)$ as the *support* of $\{X, Y\}$ in G. In Section 2.3 and 2.8.2 we discuss and compare several instances of support measures.

The Correlated Motif Mining (CMM) problem is then defined as follows:

The *Correlated Motif Mining* problem (CMM)

- **Input:** A PPI-network G, three numbers $\ell, d, k \in \mathbb{N}^+$, with $d < \ell$ and a support measure f
- **Output:** the k (ℓ, d)-motif pairs $\{X_1, Y_1\}, \ldots, \{X_k, Y_k\}$ with highest support in G with respect to f

2.3 Support measures

Support measures should reflect the power of a motif pair to describe interactions. Several considerations should be taken into account in deciding how to measure the descriptive power of a motif pair for a given PPI-network $G = (V, E, \lambda)$:

- $E_{X,Y}$ should be significantly larger than expected given G, V_X and V_Y; and,
- V_X and V_Y should be large enough to minimize the likelihood that the appearance of the motif X (Y) in the sequences of the proteins in V_X (V_Y) is just by chance.

In other words, we want the motifs X and Y to truly represent an overrepresented consensus pattern in the sequences of the proteins in V_X, respectively V_Y, to increase the likelihood that they correspond to, or at least overlap with,

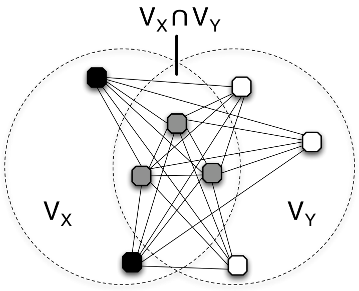


Figure 2.3: An example of a network selected by a complete (5,6,3)-motif pair.

a so called *binding site* — a site on the surface of the molecule that makes interactions between proteins from V_X and V_Y possible through a molecular lock-and-key mechanism.

Before we discuss support measures in detail, we need some more concepts from graph theory. A *bipartite graph* is a graph for which the vertex set can be partitioned into two disjoint sets B and W such that each edge connects a vertex of B with a vertex of W. It is called *balanced* if $|B| = |W|$ and *complete* if each vertex of B is connected to each vertex of W. A complete bipartite subgraph is called a *biclique*. The *edge density* $\text{ed}(G)$ of a graph $G = (V, E)$ is the proportion of edges it has of all its potential edges: $\text{ed}(G) = |E|/\binom{|V|}{2}$.

We call $\{X, Y\}$ a $(k_X, k_Y, k_{X,Y})$-motif pair for a PPI-network $G = (V, E, \lambda)$ if $|V_X| = k_X, |V_Y| = k_Y$ and $|V_X \cap V_Y| = k_{X,Y}$. We call it *complete* if all vertices from V_X are connected with all vertices from V_Y. Clearly, a complete $(k_X, k_Y, k_{X,Y})$-motif pair is an ideal candidate provided that k_X and k_Y are sufficiently large. Figure 2.3 shows an example. As such, the maximal number of edges any $(k_X, k_Y, k_{X,Y})$-motif pair can have in any PPI-network is

$$E^{\max}_{k_X,k_Y,k_{X,Y}} = \left(k_X k_Y - \binom{k_{X,Y}}{2} - k_{X,Y} \right) .$$

2.3.1 A χ^2-based support measure

Tan et al. [THSN06] introduced the χ^2-score for statistical significance as a support measure for CMM:

$$f_{\chi^2}(\{X, Y\}, G) = \begin{cases} \frac{(|E_{X,Y}| - \overline{E_{X,Y}})^2}{\overline{E_{X,Y}}} & \text{if } |E_{X,Y}| > \overline{E_{X,Y}}; \\ 0 & \text{if } |E_{X,Y}| \leq \overline{E_{X,Y}}; \end{cases}$$

with $\overline{E_{X,Y}}$ the expected number of interactions between V_X and V_Y. The value $\overline{E_{X,Y}}$ is calculated by assuming a uniform *density* of edges:

$$\overline{E_{X,Y}} = \mathrm{ed}(G) E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|} \ .$$

If we also use the edge density of the selected subnetwork $\mathrm{ed}(G_{X,Y}) = |E_{X,Y}|/E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|}$ we can rewrite the χ^2-support of $\{X,Y\}$ for which $|E_{X,Y}| > \overline{E_{X,Y}}$ as

$$f_{\chi^2}(\{X,Y\},G) = E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|} \frac{(\mathrm{ed}(G_{X,Y}) - \mathrm{ed}(G))^2}{\mathrm{ed}(G)} \ .$$

As $\mathrm{ed}(G)$ is a constant for a fixed PPI-network, we clearly see in this form that f_{χ^2} uses two criteria to determine the support of a motif pair $\{X,Y\}$:

- the difference in edge density of $G_{X,Y}$ and G, which rewards a larger $E_{X,Y}$ than expected; and
- the (potential) size of $G_{X,Y}$ in terms of the number of edges, which rewards larger V_X and V_Y.

2.3.2 p-score: a probabilistic support measure

The p-score is a measure introduced by Leung et al. [LSY$^+$07] to evaluate the statistical significance of a motif pair $\{X,Y\}$ in a PPI-network $G = (V,E,\lambda)$ by estimating the conditional probability that there are at least $|E_{X,Y}|$ or more interactions between V_X and V_Y given the number of interactions involving V_X and assuming a uniform distribution of interactions over all interaction partners. Motif pairs for which this probability is small are considered to be statistically significant.

More formally, given a motif pair $\{X,Y\}$ and a PPI-network $G = (V,E,\lambda)$, let $N(V_X) = \{u \mid \exists v \in V_X : \{u,v\} \in E\}$, i.e., the set of all vertices connected with a vertex from V_X, and $E_X = \{\{u,v\} \in E \mid u \in V_X\}$, the set of interactions involving vertices from V_X.

The probability p_X that there are $|E_{X,Y}|$ or more interactions between V_X and V_Y given $V_X, V_Y, N(V_X)$ and E_X is estimated by (see [LSY$^+$07] for details)

$$p_X = \sum_{i=|E_{X,Y}|}^{E^{\max}_{X,Y}} \frac{\binom{i-1}{|N(V_X) \cap V_Y|-1}\binom{|E_X|-i-1}{|N(V_X) \setminus V_Y|-1}}{\binom{|E_X|-1}{|N(V_X)|-1}}$$

where

$$E^{\max}_{X,Y} = \min(|E_X| - |N(V_X) \setminus V_Y|, |V_X||N(V_X) \cap V_Y|)$$

represents the maximal possible size of $E_{X,Y}$. The idea is that p_X is a good estimator for the conditional probability of $|E_{X,Y}|$ or more interactions between V_X and V_Y given $V_X, N(V_X), E_X, V_Y, N(V_Y)$ and E_Y if $|E_{X,Y}|/\overline{E_{Y\to X}}$ is small, with

$$\overline{E_{Y\to X}} = (|E_Y|/|N(V_Y)|)|N(V_Y) \cap V_X|$$

the expected number of interactions between V_Y and $N(V_Y) \cap V_X$ given V_Y, $N(V_Y)$, E_Y and V_X. Of course, similar formulas can be obtained for p_Y and $\overline{E_{X\to Y}}$ and the p-score-based support measure f_p uses the best of both estimators:

$$f_p(\{X,Y\},G) = \begin{cases} 1 - p_X & \text{if } \overline{E_{Y\to X}} \geq \overline{E_{X\to Y}}; \\ 1 - p_Y & \text{if } \overline{E_{Y\to X}} < \overline{E_{X\to Y}}; \end{cases}$$

2.3.3 Comparison of f_{χ^2} and f_p

Comparing f_p with f_{χ^2}, a major difference is that f_{χ^2} bases its support on the whole network G, while f_p-support is based on the statistical significance of a motif pair $\{X,Y\}$ in two subnetworks of the whole PPI-network: $G_X = (V_X \cup N(V_X) \cup V_Y, E_X)$ and $G_Y = (V_Y \cup N(V_Y) \cup V_X, E_Y)$. Moreover, besides the typical edge distribution assumption, f_p implicitly makes the following additional assumptions:

- V_X and V_Y are disjoint;
- every interaction from E_X (E_Y) can be described using X (Y), thus to calculate the support of $\{X,Y\}$ each protein is assumed to have only one binding site.

Finally, we stress a design flaw in the definition of f_p: the approximation p_X becomes less precise when $|E_{X,Y}|/\overline{E_{X\to Y}}$ becomes larger. But the latter happens precisely when the selected subgraph contains more edges than expected, i.e., becomes more interesting. In addition, our experiments in Section 2.8 confirm that f_p is inferior to f_{χ^2} in recovering implanted correlated motifs at different noise levels.

2.4 Complexity of CMM

We will prove that CMM is NP-hard when f_{χ^2} is used as support measure. However, to make the result as broadly applicable as possible, we will prove the NP-hardness of CMM for a whole class of support measures and show at the end of the section that f_{χ^2} is a member of that class.

For technical reasons, we restrict ourselves to support measures which abide by three reasonable conditions. Let $G = (V, E, \lambda)$ be any PPI-network and let $M_{k_X,k_Y,k_{X,Y}}$ be a complete $(k_X, k_Y, k_{X,Y})$-motif pair for G, $k_{X,Y} \leq \min(k_X, k_Y)$. We call a support measure f *compliant* if the following conditions hold for f:

- f is polynomial time computable in the size of G,
- for any two $(k_X, k_Y, k_{X,Y})$-motif pairs $\{X, Y\}$, $\{X', Y'\}$ in G:

$$f(\{X,Y\},G) = 0 \vee \\ \Big(f(\{X,Y\},G) > f(\{X',Y'\},G) \iff |E_{X,Y}| > |E_{X',Y'}|\Big)\,,$$

- $f(M_{k_X+1,k_Y,k_{X,Y}}, G) > f(M_{k_X,k_Y,k_{X,Y}}, G)\,.$

Informally, the first condition says that the support can be computed efficiently, which is crucial for scalability reasons. The second condition states that if the subnetworks selected by two motif pairs differ only in the number of edges, the one which covers more interactions has higher support. Finally, the last condition states that the support of a complete motif pair increases with the size of the selected subnetwork. Hence, the last two conditions formalize the intuition that a good support measure prefers motif pairs which select large, dense subnetworks. On the other hand, the last two conditions also induce some bias as they implicitly assume that the support only depends on V_X, V_Y, $E_{X,Y}$ and/or its relation to the PPI-network as a whole.

We call a support measure f *biclique-maximal* if:

$$f(M_{k,k,0}, G) > f(M_{k,k,k'}, G), \quad 0 < k' \leq k\,.$$

We will now show that CMM is NP-hard by proving that even a simplified version of the associated decision (D) problem is already NP-complete. Let D-CMM be the problem to decide whether for a given PPI-network $G = (V, E, \lambda)$, natural numbers ℓ, d, a real number t and a support measure f, there exists an (ℓ, d)-motif pair $\{X, Y\}$ for which $f(\{X, Y\}, G) \geq t$.

Theorem 2.1. D-CMM *is* NP*-complete for any biclique-maximal compliant support measure f.*

Proof. D-CMM is obviously in NP: since f is compliant and thus polynomial time computable, a motif pair M for which $f(M, G) \geq t$ can serve as polynomial time verifiable certificate.

We will now describe a reduction R which transforms an unlabeled graph $G = (V, E)$, with $V = \{v_1, \ldots, v_n\}$, into a labeled graph $R(G) = G' =$

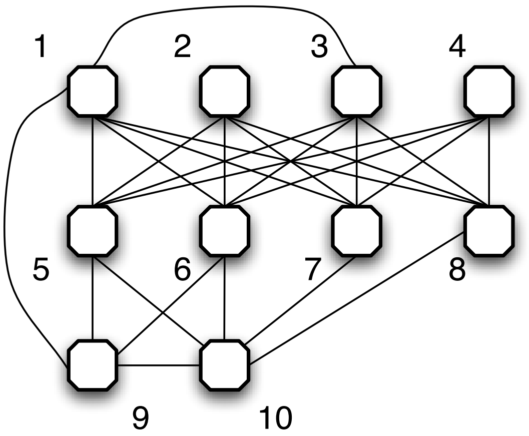


Figure 2.4: Example PPI-network

(V, E, λ). Afterwards, we will show this reduction can be used to prove the NP-completeness of D-CMM for biclique-maximal measures.

For convenience, we will use the alphabet $\Sigma = \{0, 1\}$ and label the vertices of G' as follows: $\lambda(v_i) = w_1^i \dots w_n^i$, with $w_i^i = 1$ and $w_j^i = 0$, for $j \neq i$.

Example 2.2. There are 10 vertices in the graph in Figure 2.4, so the vertices receive the following labels:

$\lambda(1)$	=	1000000000	$\lambda(6)$	=	0000010000
$\lambda(2)$	=	0100000000	$\lambda(7)$	=	0000001000
$\lambda(3)$	=	0010000000	$\lambda(8)$	=	0000000100
$\lambda(4)$	=	0001000000	$\lambda(9)$	=	0000000010
$\lambda(5)$	=	0000100000	$\lambda(10)$	=	0000000001

Each label consists of a string of 0's containing a single 1, with the position of the 1-character indicating its index.

$\triangleleft$

The labels of the vertices are chosen in such a way that for any (n, k)-motif X, $|V_X| \in \{0, 1, k\}$. Indeed, we can discriminate the following cases:

- if X contains at least two 1's then $V_X = \emptyset$;
- if X contains a 1 at position i and all other non-wildcard symbols are 0 then $V_X = \{v_i\}$; and,
- if X contains only wildcard symbols and 0's then $v_i \in V_X$ if the symbol at position i is a wildcard symbol.

Example 2.3. In the example graph $n = 10$. Suppose $k = 4$. Then the (10, 4)-motif `0x100xx01x` selects no vertex, as none of the labels can match this motif; none of them contain more than a single 1-character.

The (10, 4)-motif `0x100xx0x0` only matches $\lambda(3)$, because it is the only one with a `1` on that position.

The motif $X =$ `0x000xx0x0` selects $V_X = \{2, 6, 7, 9\}$, with $|V_X| = 4 = k$.

$\triangleleft$

As such, ignoring the cases with V_X or V_Y empty, and thus $E_{X,Y}$ empty, every motif pair in G' is necessarily a $(1, 1, k')$-, $(1, k, k')$-motif pair, with $k' \in \{0, 1\}$, or a (k, k, k')-motif pair, with $0 \leq k' \leq k$. Moreover, for an (n, k)-motif X containing only 0's and wildcard symbols, v_i will be in V_X if and only if position i of X is a wildcard symbol. In other words, for any subset $W \subseteq V$ of size k, we can choose an X such that $V_X = W$.

Consequently, if $\{X, Y\}$ is a motif pair for which $|V_X| = |V_Y|$, $V_X \cap V_Y = \emptyset$ and $|E_{X,Y}| = E^{\max}_{|V_X|,|V_Y|,0}$, then $(V_X \cup V_Y, E_{X,Y})$ is a balanced complete bipartite graph.

Example 2.4. If we want a motif X to select $V_X = \{3, 4, 6, 8\}$, it suffices to take the $(10, 4)$-motif with only 0-characters and wildcards on the corresponding positions, i.e., $X =$ `00xx0x0x00`. Hence, the motif pair

$$\{X, Y\} = \{\mathtt{xxxx000000}, \mathtt{0000xxxx00}\}$$

selects

$$G'_{X,Y} = \big(\{1, 2, 3, 4\} \cup \{5, 6, 7, 8\}, \{\{u, v\} \mid 1 \leq u \leq 4 < v \leq 8\}, \lambda_{|\{1,\ldots,8\}}\big)$$

which is a balanced $(4, 4)$-biclique.

$\triangleleft$

Given a graph G and a natural number k, deciding whether G contains a biclique such that both parts are of size k, is called the *balanced complete bipartite subgraph* (BCBS) problem . BCBS is known to be NP-complete [GJ79]. We will now show that we can decide BCBS on G by deciding D-CMM on $R(G) = G'$ for a compliant, biclique-maximal support measure.

Since the support measure is compliant, a complete $(k_X, k_Y, k_{X,Y})$-motif pair will always have higher support than any other $(k_X, k_Y, k_{X,Y})$-motif pair. Let $M_{k_X,k_Y,k_{X,Y}}$ be a complete $(k_X, k_Y, k_{X,Y})$-(n, k)-motif pair for G', $k_{X,Y} \leq \min(k_X, k_Y)$ and $k \geq 2$. We know that, by construction of G', $k_X, k_Y \in \{1, k\}$. As f is compliant and biclique-maximal it holds that:

$$f(M_{k,k,0}, G') > f(M_{1,k,0}, G') > f(M_{1,1,0}, G') \\ \wedge f(M_{k,k,0}, G') > f(M_{k,k,1}, G') > f(M_{1,k,1}, G') > f(M_{1,1,1}, G') \ .$$

Thus, G contains a balanced complete bipartite subgraph with both parts of size k if and only if there exists an (n, k)-motif pair $\{X, Y\}$ for which

$$f(\{X, Y\}, G') \geq t = f(M_{k,k,0}, G') \ .$$

Example 2.5. Deciding whether G contains a balanced bipartite complete subgraph with partitions of size 4 is thus equivalent to deciding whether there exists a motif pair $\{X, Y\}$ for which $f(\{X, Y\}, R(G)) \geq t = f(M_{4,4,0}, R(G))$, for any biclique-maximal support measure f. Remark that the vertex and edge set and thus also the edge density are invariant under R:

$$\mathrm{ed}(R(G)) = \mathrm{ed}(G') = \mathrm{ed}(G) = \frac{25}{45} = \frac{5}{9} \, .$$

Hence, for $f = f_{\chi^2}$ and $k = 4$ we get for the support of $\{X, Y\}$:

$$\begin{aligned} f_{\chi^2}(\{X, Y\}, G') \geq f_{\chi^2}(M_{4,4,0}, G') &= E^{\max}_{4,4,0} \frac{(1 - \mathrm{ed}(G'))^2}{\mathrm{ed}(G')} \\ &= 16 \frac{(1 - \frac{5}{9})^2}{\frac{5}{9}} = \frac{256}{45} \approx 5.68889. \end{aligned}$$

Thus, if there exists an $\{X, Y\}$ with a score at least 256/45 then there exists $(4, 4)$-biclique in G (and in G' as R leaves the topology invariant), because it is the only subgraph *that can be selected with a* $(10, 4)$-*motif pair* having a score this high.

◁

The proof is complete by noting that the transformation of G into G' and the calculation of t can be done in polynomial time.

□

It is easy to see that f_{χ^2} is compliant and biclique-maximal. Indeed, for fixed k, the support for a complete $(k, k, k_{X,Y})$-motif pair $\{X, Y\}$ in PPI-network G is

$$E^{\max}_{k,k,k_{X,Y}} \frac{(1 - \mathrm{ed}(G))^2}{\mathrm{ed}(G)} \ ,$$

which is maximal for $k_{X,Y} = 0$. On the other hand, remark that f_p is not compliant because $f_p(\{X, Y\}, G)$ depends on the neighborhood of the selected subnetwork $G_{X,Y}$ in G (G_X and G_Y).

2.5 Algorithms

Since the decision problem associated with CMM is in NP, we can efficiently check if a motif pair has higher support than another which makes it possible to tackle CMM as a search problem in the space of all possible (ℓ, d)-motif pairs. If we add the assumption that similar motifs can be expected to get similar support, it has the typical form of a *combinatorial optimization problem.* In combinatorial optimization, the objective is to find a point in a discrete search space which maximizes a user-provided function f. A number of heuristic algorithms called *metaheuristics* are known to yield good solutions to a wide variety of combinatorial optimization problems.

One such metaheuristic is *steepest ascent* (a form of hill climbing) [AL97]. Steepest ascent algorithms move from the current point to the best neighboring point in the space of candidate solutions until a locally optimal solution is found, i.e., a solution that maximizes f in its neighborhood. Hence, to apply steepest ascent one needs to define a neighborhood function which returns the neighbor points of each point in the search space. The neighborhood function is a key component of the steepest ascent method and has to be chosen carefully and fine-tuned for the problem at hand. The initial points from where steepest ascent is started are randomly chosen. In Section 2.9, we discuss other metaheuristics and explain the choice for steepest ascent.

The main idea behind our steepest ascent algorithm for CMM is illustrated by the pseudo-code in Algorithm 1. To be able to specify the difference between our two methods, we use an abstract neighborhood function N. For reasons of clarity, we use an abstract support measure f and focus on the case in which only the best pair is returned ($k = 1$). In practice, we accumulate the best results found over as many runs as can be completed in a given time frame, and store the results sorted by support.

The method randomMotifPair() picks:

1. a random interaction $\{u, v\}$;

2. a random position p_u in $\lambda(u)$ and p_v in $\lambda(v)$;

3. a random motif X by first picking d random positions in $[p_u+1, p_u+\ell-1]$ as the wildcard positions and taking the remaining positions as the non-wildcard positions; and,

4. a random motif Y from $\lambda(v)$ at position p_v in the same way.

Algorithm 1 The general steepest ascent algorithm with abstract neighbor function applied to CMM (SA-CMM).

```
Input: PPI-network $G = (V, E, \lambda)$, $\ell, d \in \mathbb{N}$, $d < \ell$
Output: $\{X^*, Y^*\}$ best correlated motif pair found in $G$
$\{X^*, Y^*\} \leftarrow$ randomMotifPair()
$maxsup \leftarrow f(\{X^*, Y^*\}, G)$
$sup \leftarrow -\infty$
while $maxsup > sup$ do
    $\{X, Y\} \leftarrow \{X^*, Y^*\}$
    $sup \leftarrow maxsup$
    for all $\{X', Y'\} \in N(\{X, Y\})$ do
        if $f(\{X', Y'\}, G) > maxsup$ then
            $\{X^*, Y^*\} \leftarrow \{X', Y'\}$
            $maxsup \leftarrow f(\{X', Y'\}, G)$
```

To apply steepest ascent to CMM, we need to define a neighborhood function which maps a motif pair $\{X, Y\}$ to its neighbors $N(\{X, Y\})$ in the space of all motif pairs. Consider a motif pair $\{X, Y\}$ and the selected subnetwork $G_{X,Y}$. The main idea behind a steepest ascent algorithm is to gradually improve a candidate solution until it becomes (locally) optimal. Consequently, it is desirable that the subnetwork $G_{X',Y'}$ selected by a neighbor $\{X', Y'\} \in N(\{X, Y\})$ is also "close" to $G_{X,Y}$ in the sense that at least some proteins and interactions are shared between $G_{X,Y}$ and $G_{X',Y'}$. That is, we would like that the candidate solution in the *dual* search space of selected subnetworks also improves gradually to avoid that the algorithm jumps around in the network selecting completely different networks in each step. Suppose for instance that $\{X, Y\}$ is a motif pair which describes (a part of) compatible binding sites in most proteins in V_X and V_Y. If at some point in the algorithm we reach a motif pair $\{X', Y'\}$ for which a significant fraction of the motif hits of X' and Y' overlap with the desired motifs X and Y, it would be undesirable that X' is changed into a motif which has almost no motif hits in V_X.

A straightforward way to ensure that some proteins are kept, is by only considering motifs of the form $\{X, Y'\}$ or $\{X', Y\}$ as candidate neighbors, such that either V_X or V_Y remains in $G_{X',Y'}$. The neighbor functions we will define in the next sections share the principle that one motif remains fixed and that the neighborhood of the pair is defined in terms of a neighbor function N on the motifs, more formally: $\{X', Y'\} \in N(\{X, Y\})$ if $X' \in N(X) \wedge Y' = Y$ or $Y' \in N(Y) \wedge X' = X$.

Hence, to ensure that $G_{X',Y'}$ is also likely to share interactions with $G_{X,Y}$, it suffices to define the neighborhood function N on motifs in such a way that

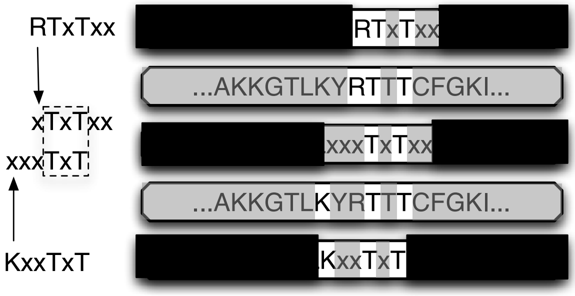


Figure 2.5: Two neighboring (by N^{mot}) $(6, 3)$-motifs seen as sliding windows on a sequence.

V_X shares proteins with $V_{X'}$ for most of the motifs $X' \in N(X)$.

On the other hand, it is also desirable that N is powerful enough to move from any $\{X, Y\}$ to any other $\{X', Y'\}$ in a reasonable number of steps, while keeping $N(\{X, Y\})$ small enough to keep evaluating all neighbors of $\{X, Y\}$ tractable.

2.5.1 M-SLIDER: Sliding over motifs

In this subsection, we formally introduce a first neighborhood function N^{mot} on motifs which will be the basis for M-SLIDER (short for motif-SLIDER). N^{mot} is based on the observation that looking for a match of an (ℓ, d)-motif X in a protein can be seen as sliding a window of length ℓ with $\ell - d$ holes over the sequence until the characters in the holes match the non-wildcard characters of X. Hence, any motif X' obtained by closing one hole and creating a new one (not too far from the other ones so as to respect the window size ℓ) will select the same protein we are sliding the window over. In this way, the motif window can slide to the left or right if the new hole is punched before the first or after the last original character. We will call any motif X' a neighbor of X, if it can be obtained from it, by replacing one non-wildcard character with a wildcard and then adding a new non-wildcard character making it an (ℓ, d)-motif again. We can see in Figure 2.5 that moving from `RTxTxx` to `KxxTxT`, by closing the hole over the `R` and opening a new one over the `K`, shifts the window to the left. The motif `RTxxxA` is also a neighbor, but does not select the same protein.

Next, we formally define N^{mot}. For a motif X, denote by $\text{trim}(X)$, the motif obtained from X by removing leading and trailing wildcards. That is, $\text{trim}(\texttt{xTxTxx}) = \texttt{TxT}$. A motif $X' \in N^{\text{mot}}(X)$ if X and X' have the same length and $\mathrm{trim}(Y) = \mathrm{trim}(Y')$ where Y is obtained from X by changing one non-wildcard character into a wildcard, and similarly for Y' and X'. In Figure 2.5, X equals `RTxTxx` while X' equals `KxxTxT`. Now, $X' \in N^{\mathrm{mot}}(X)$ as X (X'), can be transformed into $Y =$ `xTxTxx` ($Y' =$ `xxxTxT`) by changing one non-wildcard character into a wildcard and Y equals Y' after stripping leading and trailing wildcards.

Remember that when applying N^{mot} to pairs of motifs, one of the motifs remains fixed. From our experiments we observed that fixing one motif at each step greatly improves the effectiveness.

It is fairly easy to show that N^{mot} allows to reach any $\{X', Y'\}$ in at most $2(\ell - d)$ steps and $|N^{\mathrm{mot}}(\{X, Y\})| = \Theta(\ell^2)$ which keeps evaluating all neighbors tractable for the typical values for ℓ and d. Moreover, at least $2d(\ell - d)$ neighbors will select a subnetwork that shares at least one interaction with $G_{X,Y}$ (see Section 2.6.3).

Definition 2.6. We define the method M-SLIDER as steepest ascent with
(i) neighborhood function N^{mot}; and,
(ii) support measure f_{χ^2}.

It can be formally shown that, if we assume that the number of steps can be bound by a small constant as observed in our experiments (for instance, in our experiments the number of steps never exceeded 15), M-SLIDER runs in time $O\left(\ell^2 \left(|V|^2 + \ell|V|\lambda_{\max}\right)\right)$, with $\lambda_{\max} = \max_{v \in V} |\lambda(v)|$. Remark that the former is almost linear in the size of G, when $|G| = |V|^2$. However, using a theoretical maximum number of steps $|V|^5$, we obtain the bound $O\left(|V|^5 \; \ell^2 \left(|V|^2 + \ell|V|\lambda_{\max}\right)\right)$ (proof in Section 2.6.5).

2.5.2 SEQ-SLIDER: Sliding over sequences

Although a significant number of the neighbors of a motif pair $\{X, Y\}$ under N^{mot} are expected to select a subnetwork $G_{X',Y'}$ that is also "close" in the network in the sense that $G_{X,Y}$ and $G_{X',Y'}$ share interactions, this property is not guaranteed for any neighbor. For that reason, we also designed a second neighborhood function N^{seq} which focusses on this aspect, but does not guarantee that all other motif pairs can be reached by moving from one neighbor to the other. The N^{seq} neighborhood function forms the basis of our second SLIDER variant SEQ-SLIDER.

N_u^{seq} defines the neighborhood of a motif X on the sequence level by considering all (ℓ, d)-motifs that match a region around the motif hits of X in the sequence of one particular protein $u \in V_X$. The idea is that, in each run, after picking a random pair $\{X, Y\}$ that describes some interaction $\{u, v\}$, we only consider motif pairs based on the region around the motifs hits of X in $\lambda(u)$

and of Y in $\lambda(v)$, i.e.,

$$N^{\text{seq}}_{u,v}(\{X,Y\}) = \{\{X',Y\} \mid X' \in N^{\text{seq}}_u(X)\} \cup \{\{X,Y'\} \mid Y' \in N^{\text{seq}}_v(Y)\}.$$

In that way, $N^{\text{seq}}_{u,v}$ guarantees that the subnetwork $G_{X',Y'}$ selected by any neighbor $\{X',Y'\}$ of a motif pair $\{X,Y\}$ will always contain $\{u,v\}$.

More formally, for an (ℓ,d)-motif X and a protein u, denote by $pos(X,u)$ the set of positions of substrings in $\lambda(u)$ that match X. An (ℓ,d)-motif $X' \in N^{\text{seq}}_u(X)$ if there exist positions $p \in pos(X,u)$ and $p' \in pos(X',u)$ such that $|p-p'| \leq \delta$, where δ is some small distance bound (we use $\delta = \lceil \ell/3 \rceil$). Hence, $N^{\text{seq}}_{u,v}(\{X,Y\})$ defines the neighborhood of $\{X,Y\}$ relative to $u \in V_X$ and $v \in V_Y$.

For instance, the two motifs in Figure 2.5 are also neighbors under N^{seq} as they both have matches in the sequence within the distance bound. The motif `KYxTxx` is an example of a motif that would be a neighbor under N^{seq} but not under N^{mot} as it differs more than one non-wildcard character from the original. The motif `RTxxxA` on the other hand is a neighbor using N^{mot} but not using N^{seq} as it does not have any matches within a δ-region of a match of the original motif.

Thus, for a sufficiently high number of runs, we are likely to have considered a local optimum under $N^{\text{seq}}_{u,v}$ for each $\{u,v\}$ in E, which gives SEQ-SLIDER a bias towards a set of complementary best motif pairs in the sense that all of them together are likely to cover more interactions than the set of best motif pairs returned by M-SLIDER.

From a theoretical point of view however, SEQ-SLIDER has some disadvantages compared to M-SLIDER: it cannot reach every motif pair from an arbitrary motif pair and evaluating all neighbors of a motif pair can be expensive as the number of neighbors $|N^{\text{seq}}(\{X,Y\})|$ can become as large as $\binom{\ell-1}{d}(2\delta+1)(|pos(X,u)|+|pos(Y,v)|)$ (see Section 2.6.4), which can become prohibitive for larger values of ℓ and d. Nevertheless, as we will see in the experimental section, SEQ-SLIDER obtains significantly better results than M-SLIDER in the same time frame for the typically small values of ℓ and d.

Definition 2.7. We define the method SEQ-SLIDER as steepest ascent with
(i) neighborhood function $N^{\text{seq}}_{u,v}$ with $\delta = \lceil \ell/3 \rceil$; and,
(ii) support measure f_{χ^2}.

Let $\lambda_{\max} = \max_{v \in V} |\lambda(v)|$. We formally proved that SEQ-SLIDER runs in time $O\left(\delta\binom{\ell}{d}\lambda_{\max}\ (|V|^2 + \ell|V|\lambda_{\max})\right)$, if we again assume the number of steps is constant, and $O\left(|V|^5\ \delta\binom{\ell}{d}\lambda_{\max}\ (|V|^2 + \ell|V|\lambda_{\max})\right)$ otherwise (proof in Section 2.6.5).

2.6 Time complexity of algorithms

In this section, we will examine the time complexity of SLIDER in greater detail.

2.6.1 Preprocessing step

For the typically small values of ℓ and d, we create an array which contains for each motif X the set of proteins V_X in which the motif appears. In that way, we can obtain V_X for a motif X in constant time during the execution of the algorithm. The preprocessing step is only performed for those values of ℓ and d for which we effectively win time. We can estimate this performance gain by performing a single run with and without preprocessing.

The preprocessing step takes $O\left(\binom{\ell}{d}\left(20^{l-d}+|V|\lambda_{\max}\right)\right)$ time with $\lambda_{\max} = \max_{v\in V}|\lambda(v)|$ and $O\left(20^{\ell-d}\binom{\ell}{d}|V|\right)$ space to store the array.

As there are $20^{\ell-d}\binom{\ell-1}{d}$ motifs, the size of the array is $O(20^{\ell-d}\binom{\ell}{d}|V|)$.

Initially, all lists are empty. Subsequently, we consider each position p in the sequence of each protein u, look at the window W of length ℓ starting at p and enumerate all motifs that match with W by considering all $\binom{\ell-1}{d}$ ways to place wildcards in W. For each such motif X, we add u to the list of proteins containing X unless u is already in the list.

Hence, the constructive part of the preprocessing step takes

$$O\left(\binom{\ell}{d}\sum_{u\in V}|\lambda(u)|\right) \leq O\left(\binom{\ell}{d}|V|\lambda_{\max}\right)$$

time. Remark that we can check if u is already in the list in constant time, because if it is, it must be the last element in the list.

In practice, the bottleneck is the memory required to store all the lists.

2.6.2 Evaluation cost and value range f_{χ^2}

Lemma 2.8. *f_{χ^2} can be evaluated in time $O(|V|^2)$ if the preprocessing step is performed and in time $O(|V|^2+\ell|V|\lambda_{\max})$ otherwise.*

Proof. The dominant operation to evaluate f_{χ^2} is to obtain $|E_{X,Y}|$. If the preprocessing step is performed, we can obtain V_X and V_Y in constant time. Otherwise, we have to construct V_X and V_Y by scanning the sequences of each protein for a match of X or Y. This can be done in time $O(\ell\sum_{u\in V}|\lambda(u)|) \leq O(\ell|V|\lambda_{\max})$.

Using an adjacency matrix, we can determine $|E_{X,Y}|$ from V_X and V_Y in time $O(|V_X||V_Y|) = O(|V|^2)$.

Hence, evaluating f_{χ^2} can be performed in time $O(|V|^2)$ if the preprocessing step is performed and $O(|V|^2 + \ell|V|\lambda_{\max})$ otherwise. □

The following Lemma will be used to bound the total time complexity of both methods.

Lemma 2.9. *Given a PPI-network G, the maximum number of different values f_{χ^2} can take is $O(|V|^5)$.*

Proof. For a fixed G it is clear from the definition of f_{χ^2} that $f_{\chi^2}(\{X,Y\},G)$ only depends on $E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|}$ and $\mathrm{ed}(G_{X,Y})$, which in turn only depend on $|V_X|$, $|V_Y|$, $|V_X \cap V_Y|$ and $|E_{X,Y}|$. As the sizes of the vertex sets are all bounded by $|V|$ and $|E_{X,Y}|$ by $|V|^2$, it follows that the number of different values that $f_{\chi^2}(\{X,Y\},G)$ can take is $O(|V|^5)$. □

2.6.3 $N^{\mathbf{mot}}$ (m-slider)

Lemma 2.10. *The maximum number of neighbors of any (ℓ,d)-motif pair under N^{mot} is $\Theta(\ell^2)$.*

Proof. We will use the notation introduced in Section 2.5.1. First, note that it is possible that Y has no wildcards at the beginning and end. In that case, trim(Y) $= Y$ and the number of possible positions for the new character is equal to $d+1$: the original wildcard characters plus the new one. The other extreme occurs when all wildcard characters of Y are at the beginning or end. In that case trim(Y) is composed of $\ell-d-1$ non-wildcard characters and the number of possible positions for the new character equals $2d+2$: $d+1$ wildcards in front and $d+1$ wildcards at the end. In general, the number of possible positions equals $d+1$ plus the number of wildcards removed by trimming Y which equals $\ell - |\mathrm{trim}(Y)|$. Thus, the number of possible positions ℓ' for the new character is $\ell' = d+1+\ell-|\mathrm{trim}(Y)| \in \mathbb{N}[d+1, 2d+2]$. Hence, the number of neighbors for X equals the number of possible non-wildcard characters to be removed to obtain Y times the number of possible positions ℓ' for the new character times the number of possible new characters: $(l-d)\ell'20 = \Theta(\ell^2)$ as both d and ℓ' are at most linear in ℓ.

The neighborhood of a pair of motifs under N^{mot} is composed of the motif pairs which can be obtained by replacing one of the motifs of the pair by a neighbor under N^{mot}. Because N^{mot} always keeps one of the motifs of a pair fixed, the number of neighbors for a pair of motifs is twice the number of neighbors of an individual motif and thus also $\Theta(\ell^2)$. □

Theorem 2.11. *The maximum number of steps required to change an (ℓ,d)-motif X into any other (ℓ,d)-motif X' by following neighbors under N^{mot}*

is at most $(\ell - d)$. That is, there exists a sequence of (ℓ, d)-motifs $X = X_0, X_1, \ldots, X_k = X'$, such that $X_i \in N^{mot}(X_{i-1})$, $1 \leq i \leq k$, with $k \leq (\ell - d)$.

Proof. Informally, it is easy to see that $(l - d)$ steps suffice as, in each step, we can choose any non-wildcard character to be replaced by a wildcard and afterwards change an arbitrary wildcard into an arbitrary non-wildcard character. Hence, if we ignore to order of the operations for a moment, we can replace all $(l - d)$ non-wildcard character by wildcards to obtain a string of only wildcards and then pick $(l - d)$ of them to be replaced by non-wildcard characters of our choice. Of course, any (ℓ, d)-motif can be created in this way.

For a more formal proof, we give a procedure which keeps the operations in the order as defined by N^{mot}, i.e., it produces a sequence of (ℓ, d)-motifs $X = X_0, X_1, \ldots, X_k = X'$, such that $X_i \in N^{\text{mot}}(X_{i-1})$, $1 \leq i \leq k$ and show that $k \leq (\ell - d)$. Let i be the first position containing a non-wildcard character of X' that differs from the character of X in the same position. We distinguish two cases: the character at position i in X is

- a non-wildcard: in that case we change position i of X into a wildcard.
- a wildcard: in that case we choose the first position j for which there is a wildcard in X' but not in X and change the non-wildcard character at position j in X into a wildcard. Note that j always exists because otherwise it would be impossible that position i is a wildcard in X but not in X' as both X and X' are (ℓ, d)-motifs.

Next, we change position i, which is now necessarily a wildcard, into the character at position i in X' to obtain an (l, d)-motif X_1. The first non-wildcard character of X' that differs from the character in the same position in X_1 must now be at position $j > i$. Hence, by repeating this procedure, we must end up with X' in at most $(l - d)$ steps as the position with the first non-wildcard character in X' that differs from the (l, d)-motif obtained after each iteration increases after each iteration.

For instance, to change $X = \texttt{RxxTxT}$ into $X' = \texttt{KxLxxR}$, the procedure above yields the following sequence:

$$\begin{aligned} X &= \texttt{RxxTxT} \\ &\to (\texttt{xxxTxT}) \to \texttt{KxxTxT} \to (\texttt{KxxxxT}) \to \texttt{KxLxxT} \to (\texttt{KxLxxx}) \\ &\to \texttt{KxLxxR} = X' \ . \end{aligned}$$

□

Theorem 2.12. *Given a motif pair $\{X, Y\}$, and assuming that $G_{X,Y}$ contains at least one interaction, there will be at least $2d(\ell - d)$ neighbors under N^{mot} which select a subnetwork that shares at least one interaction with $G_{X,Y}$.*

Proof. Let's assume $G_{X,Y}$ contains the interaction $\{u,v\}$, with u matching motif X and v matching motif Y. We can take the subsequence of length ℓ matching motif X from the sequence of protein u in $G_{X,Y}$. We can create a neighbor that also matches this subsequence by removing a single non-wildcard and replacing a wildcard with the letter occurring at that position in the subsequence. Since for each of the $(\ell - d)$ non-wildcards, we can do this for each of the d wildcards, we get $d(\ell - d)$ neighbors that match the same protein by changing the motif X.

We can also do this for protein v and motif Y, leading us to a minimum of $2d(\ell - d)$ neighbors that share the interaction. □

2.6.4 $N^{\mathbf{seq}}$ (SEQ-SLIDER)

Lemma 2.13. *The number of neighbors of a motif pair $\{X,Y\}$ under $N^{seq}_{u,v}$ can become as large as: $\binom{\ell-1}{d}(2\delta+1)(|pos(X,u)| + |pos(Y,v)|)$.*

Proof. Consider a motif hit of X starting at position i in $\lambda(u)$. By definition, any (ℓ,d)-motif X' appearing in a δ-region around the motif hit is a neighbor of X under N^{seq}_u. Thus, any string of length ℓ starting at position j, $i-\delta \leq j \leq i+\delta$ can be the basis for X' by replacing d chars by wildcards. Thus, we have $2\delta+1$ possible positions for the motif hit of X' to start within a δ-region around i. For one such position, there are exactly $\binom{\ell-1}{d}$ ways to choose d positions for the wildcards. Thus, assuming all subsequences starting within this δ-region around i are different, we can create up to $\binom{\ell-1}{d}(2\delta+1)$ neighbors based on one motif hit. In total, we have $|pos(X,u)|$ motif hits of X in u, which can yield up to $\binom{\ell-1}{d}(2\delta+1)|pos(X,u)|$ neighbors for X under N^{seq}_u. Applying the same reasoning for Y under N^{seq}_v yields the formula above for the potential size of $N^{\mathrm{seq}}_{u,v}(\{X,Y\})$. □

2.6.5 Time complexity SLIDER

We are now ready to address the time complexity of SLIDER.

Theorem 2.14. *The total time complexity of* M-SLIDER *without preprocessing is*

$$O\left(|V|^5\ \ell^2\ (|V|^2 + \ell|V|\lambda_{\max})\right)$$

and the total time complexity of SEQ-SLIDER *without preprocessing is*

$$O\left(|V|^5\ \delta\binom{\ell}{d}\lambda_{\max}\ (|V|^2 + \ell|V|\lambda_{\max})\right),$$

with $\lambda_{\max} = \max_{v\in V}|\lambda(v)|$. However, if we assume that the total number of steps is constant, as observed in practice, then the bound for M-SLIDER *is*

$$O\left(\ell^2\ (|V|^2 + \ell|V|\lambda_{\max})\right)$$

and for SEQ-SLIDER

$$O\left(\delta\binom{\ell}{d}\lambda_{\max}\left(|V|^2+\ell|V|\lambda_{\max}\right)\right),$$

Proof. The total time complexity of both SLIDER variants is at most the cost of the preprocessing step, if performed, plus

$\{$number of steps$\}$ times $\{$number of neighbors$\}$ times $\{$evaluation cost $f_{\chi^2}\}$.

We know the total cost of the preprocessing step is $O(\binom{\ell}{d}(20^{\ell-d}+|V|\lambda_{\max}))$, but it is only performed for the small values of ℓ and d for which this effectively saves time. Hence, we can ignore this cost and bound the theoretical time complexity of our methods for the case in which no preprocessing is done. If we use the preprocessing step, we can remove the term $O(\ell|V|\lambda_{\max})$ from the evaluation cost of f_{χ^2}.

We know that in practice, the number of steps in one run of both slider variants can be bounded by a small constant. Hence, using Lemma 2.10 and 2.8 the time complexity of M-SLIDER in practice is $O\left(\ell^2\left(|V|^2+\ell|V|\lambda_{\max}\right)\right)$.

To obtain the time complexity of SEQ-SLIDER we need a bound on the number of neighbors which is independent of a particular edge $\{u,v\}$. Clearly, the number of motif hits $|pos(X,u)|$ of a motif X in the sequence of a protein u is at most $|\lambda(u)|\leq\max_{v\in V}|\lambda(v)|=\lambda_{\max}$. Substituting this into the bound on the number of neighbors of N^{seq} from Lemma 2.13 and using Lemma 2.8, we can bound the time complexity of SEQ-SLIDER with $O(\binom{\ell}{d}\delta\lambda_{\max}(|V|^2+\ell|V|\lambda_{\max}))$.

As mentioned earlier, these bounds assume that the number of steps can be bounded by a constant. Because both SLIDER-variants are steepest ascent methods which only move to a neighboring point if the support is strictly larger, we can use the number of different values f_{χ^2} can take from Lemma 2.9 as a bound on the number of steps, which results in an extra $O(|V|^5)$ factor in the time complexity bound of both methods. □

2.7 Data

2.7.1 Artificial data

To evaluate the biological relevance of the different notions of support and the power of heuristic methods to retrieve the best motif pairs in terms of describing interactions, we created a number of artificial networks as follows. Each network is composed of 100 proteins which are randomly chosen out of the well-known yeast network [CKX+07]. We then generate 50 random $(8, 3)$-motifs[2] and implant k instances of each motif in the sequences of randomly chosen proteins, with k chosen uniformly from $\mathbb{N}[3, 10]$. Then, we implant motif pairs by randomly selecting two implanted motifs X and Y and connecting each protein containing X with each protein containing Y and repeat this procedure until a chosen minimal edge density e is obtained — we used 0.1, 0.2 and 0.3. Consequently, the network obtained is perfect in the sense that there is an interaction $\{u, v\}$ if and only if a motif pair is present in $\lambda(u)$ and $\lambda(v)$. Because noise and missing data are an important factor in biological networks, we also evaluate the resistance to noise of both the support measures and heuristic methods. To that end, we also created versions of each network with added noise, by choosing a certain noise level a (from 0.05 to 0.3 in steps of 0.05) and switch the edge relation of each pair of vertices with probability a (remove the edge if they are connected and add one if not). We used 105 networks in total — 5 networks for each (e, a)-combination.

We restrict ourselves to networks of 100 proteins because this is more or less the maximum size for which we are still able to mine the motif pairs with highest support for each support measure by a brute force computation within a reasonable time frame, which is necessary to evaluate the results.

As a sanity check, we also constructed networks where only a small portion of interactions can be explained by a motif pair (see Section 2.8.7).

2.7.2 Biological data

To assess the effectiveness on larger networks, we ran our method and MotifHeuristics on the high-confidence PPI-network of yeast consisting of 1 620 proteins and 9 060 interactions [CKX+07] and on the human PPI-network which has 8 872 proteins and 14 230 interactions [KPGK+09], two of the largest and most complete interaction datasets available. The interactions in the human network are curated from the literature [KPGK+09] and the interactions in the yeast dataset are determined using Tandem-Affinity Purification followed by Mass-Spectrometry (TAP/MS), a technique which is used to determine the proteins in a complex [CKX+07]. As a consequence, the interactions determined by TAP/MS contain both direct and indirect interactions. For that reason, it is expected that the human dataset contains less false positive but more false negative interactions in comparison with yeast. Hence, these

[2] Using entropy analysis, research has shown that the highest amount of structural information per sequence length can be found in subsequences of length 7 to 9 (see Figure 1 in [vTV09]).

two interaction datasets are ideal to assess our methods as they are

- large, which allows us to test the scalability of our methods;
- as complete as available at the moment, which allows to assess if the best scoring (ℓ, d)-motif pairs found by our methods and by a brute force method can describe the interactions given enough data;
- complementary in terms of noise, which allows to assess how the descriptive power of the best scoring (ℓ, d)-motif pairs of our methods and a brute force method are affected by different kinds of noise (false positives vs. false negatives).

2.8 Experiments

The brute force runs on yeast and human (which calculate support for each possible motif pair) were run on a computer cluster. All other experiments were run on a 3GHz Mac Pro using 2GB of RAM and 8 cores. In the following, whenever a timing is mentioned and unless explicitly mentioned otherwise, the experiment was run using only 1 core. Nevertheless, we stress that our SLIDER-prototype, implemented in Java, can use as many processors as are available. In this section, we experimentally assess the effectiveness of

1. support measures to assign a support to a motif pair reflecting its power to describe interactions; and,
2. neighborhood functions to find the motif pairs with highest support by exploring the space of all motif pairs.

Furthermore, we compare both SLIDER variants with other motif-driven CMM-methods. To this end, we need a notion of precision[3] that compares an ordered set of motif pairs versus a set of motif pairs which is considered to be the "ground truth". Finally, we assess the effectiveness of the SLIDER variants on the yeast and human PPI-networks.

2.8.1 Precision for motif pairs

Before we define our notion of precision, we need a similarity measure to compare the found motif pairs against the implanted pairs. We define the

[3] The notion of precision we will use is similar to the notion of sensitivity of a binary classifier. Specificity, however, cannot be defined for a ranking problem such as CMM because there is no meaningful notion of true negative.

similarity between an (ℓ, d)-motif pair $\{X, Y\}$ and $\{X', Y'\}$ in a PPI-network $G = (V, E, \lambda)$ as

$$s(\{X, Y\}, \{X', Y'\}, G) = \frac{|E_{X,Y} \cap_{pos} E_{X',Y'}|}{|E_{X,Y} \cup E_{X',Y'}|}$$

where $\{v, w\} \in E_{X,Y} \cap_{pos} E_{X',Y'}$ if there exists substrings s_v and s'_v in $\lambda(v)$ and s_w and s'_w in $\lambda(w)$ such that

- s_v matches with X and s_w with Y;
- s'_v matches with X' and s'_w with Y'; and,
- s_v and s'_v as well as s_w and s'_w are at the same position in $\lambda(v)$, respectively $\lambda(w)$.

Let $S = (M_1, \ldots, M_n)$ be a list of motif pairs, then we reduce S by deleting for every j from 1 to n, every M_i for $i > j$ such that $s(M_i, M_j) = 1$. We denote the reduced version of S by S^*.

Let T be a set of known "ground truth" (ℓ, d)-motif pairs and let $S = (M_1, \ldots, M_n)$ be a list of (ℓ, d)-motif pairs to be compared against T. We define the precision of S against T at rank k as the fraction of motif pairs M_i in S^*, $1 \leq i \leq k$ for which there exists a motif pair M_T in T such that $s(M_i, M_T) = 1$. We note that, when $k = |T|$, the precision as defined above also corresponds to the usual notion of recall.

2.8.2 Evaluation of support measures

We start by assessing the effectiveness of support measures in assigning a support to a motif pair reflecting its power to describe interactions. Since the most descriptive motif pairs in real PPI-networks are unknown, we measure the ability of a support measure to assign the highest support to motif pairs on artificial networks with implanted motifs, as described in Section 2.7.1. We used a collection of networks G^a_e with edge density e and noise level a. We compare the support measures by looking at the precision of the best motif pairs obtained by a brute force method at rank m against the implanted motif pairs on G^a_e, where m equals the number of implanted motif pairs.

To make sure that the f_{χ^2} and f_p assign a meaningful support, we also implemented two naive support measures, called f_c and f_v. The f_c-support in a PPI-network $G = (V, E)$ is simply the number of interactions covered: $f_c(\{X, Y\}, G) = |E_{X,Y}|$ and $f_v(\{X, Y\}, G) =$

$$\frac{|E_{X,Y}|}{E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|} + |V_X \cup V_Y|}\ .$$

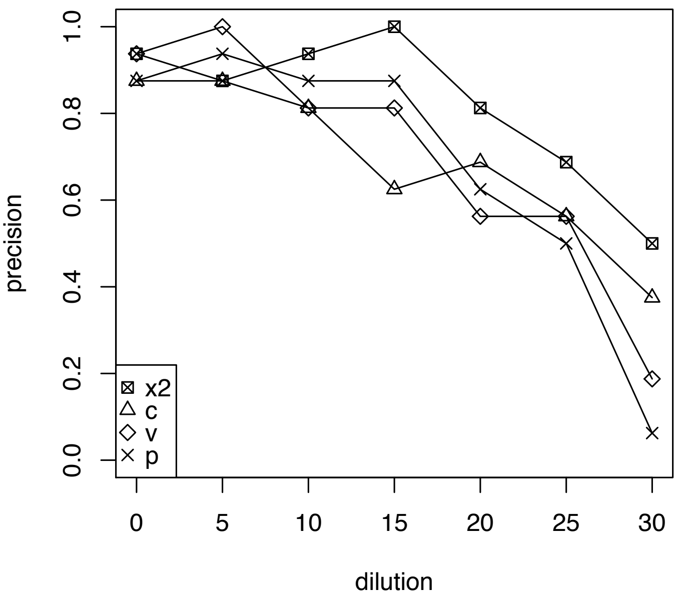


Figure 2.6: Precision of support measures on artificial networks with implanted motif pairs and edge density of 10%.

f_v is the edge density corrected with an extra term in the denominator to prefer larger subnetworks ($E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|}$ grows quadratically in $|V_X \cup V_Y|$). Both measures are naive in the sense that they are independent of the interaction distribution in G. It is straightforward to show that both measures are compliant, thus meeting the basic requirements of a support measure. Moreover, they are biclique-maximal.

A visual inspection of the graphs in Figure 2.6, 2.7, and 2.8 already indicates that f_{χ^2} globally outperforms the other support measures in selecting motif pairs describing actual interactions. Indeed, at every data point, the precision of f_{χ^2} is the best value or very close to the best value of the four support measures considered. Moreover, comparing precision on noisy networks shows that f_{χ^2} is vastly more robust to noise — a crucial aspect since contemporary PPI-networks contain large amounts of both noise and missing data [vMKS$^+$02].

When we compare the results of the brute force runs on yeast for f_{χ^2} and f_p, we also notice that the 1 000 best scoring subnetworks for f_{χ^2}, have an average edge density of 97.2% and a *minimum* edge density of 64%, while those for f_p have an average edge density of 14.5% and a *maximum* edge density of 16.7%. The edge density for the latter is obviously much lower than desired.

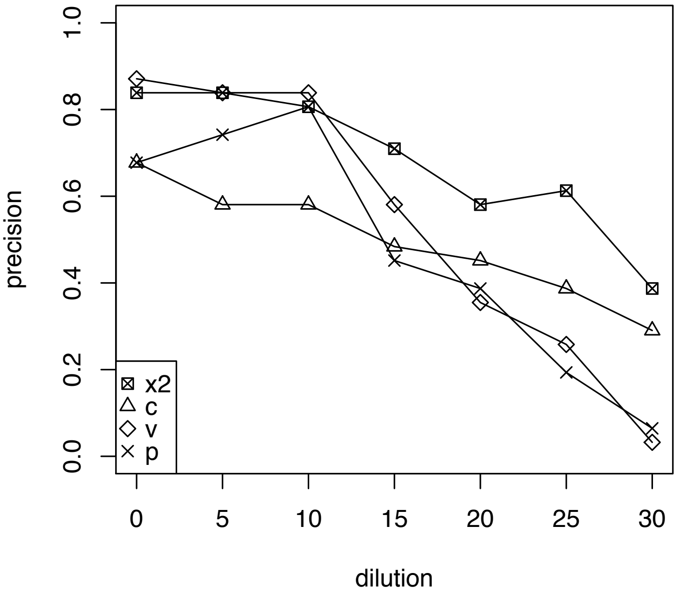


Figure 2.7: Precision of support measures on artificial networks with implanted motif pairs and edge density of 20%.

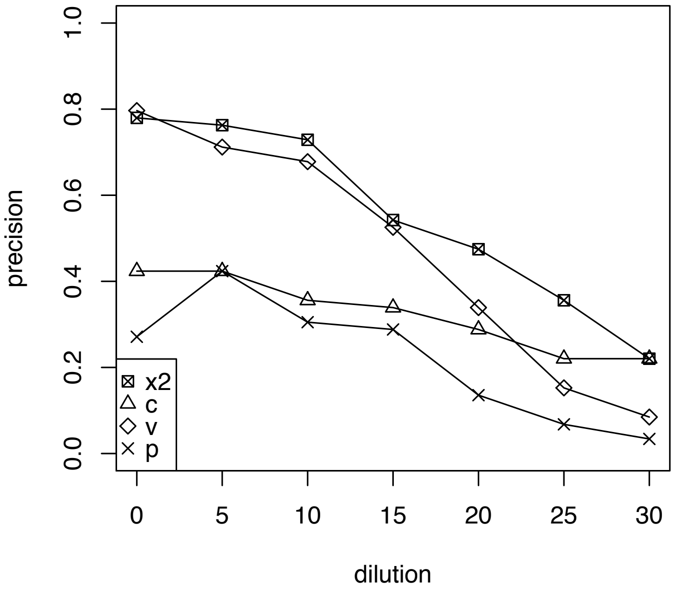


Figure 2.8: Precision of support measures on artificial networks with implanted motif pairs and edge density of 30%.

Thus, we can conclude this experimental section by saying that f_{χ^2} is superior to all other support measures considered on all merits.

2.8.3 Evaluation of neighborhood functions

We will now confirm that our neighborhood functions, which are based on a sliding window interpretation on the sequences, are superior to neighborhood functions which simply define small perturbations to explore the search space.

In particular, we define the following perturbations:

- letter change (LC, replace one non-wildcard character by another);
- swap adjacent (SA, swap an adjacent wildcard and non-wildcard character); and,
- swap (S, swap an arbitrary wildcard and non-wildcard character).

We denote neighborhood functions combining these perturbations by concatenating their abbreviations with boolean operators. For instance, LCandSA denotes the neighborhood function which requires a letter change *and* a swap adjacent perturbation. Finally, we consider a simple version of N^{mot}, denoted $N^{\mathrm{mot}}_{\ominus}$, which forces the motif to slide left or right by only allowing to change the leftmost (rightmost) non-wildcard character into a wildcard and demanding that the new non-wildcard character is added to the right (left) of the existing ones. The corresponding neighborhood functions on pairs of motifs are defined similarly: one motif is kept fixed, while the other is replaced by a neighbor. As a naive baseline, we also compare with the method Random, which evaluates random motif pairs using f_{χ^2}.

Figure 2.9 displays the precision of SA-CMM with each of these neighborhood functions on five implanted networks of density 10% and their noisy versions. The displayed precision is averaged over 5 SA-CMM runs. Runs on the networks of density 20 and 30% give similar results (data not shown). As the speed of SA-CMM is highly dependent on the chosen neighborhood function, we provided each run the same amount of time (10 minutes). In this way, faster neighborhood functions like LCorSA can process more randomly chosen starting motif pairs than slower functions like N^{mot} and N^{seq} (cf. the table in Figure 2.10). As can be seen from Figure 2.9, N^{seq}, and thereby SEQ-SLIDER, outperforms the other SA-CMM variants using other neighborhood functions, including M-SLIDER which is second.

For the sake of completeness, we also experimented with neighborhood functions on motif pairs where both motifs can be replaced with a neighboring one (in contrast to the previous neighborhood functions where one is kept fixed). Unfortunately, the precision was never larger than 10%, independent of

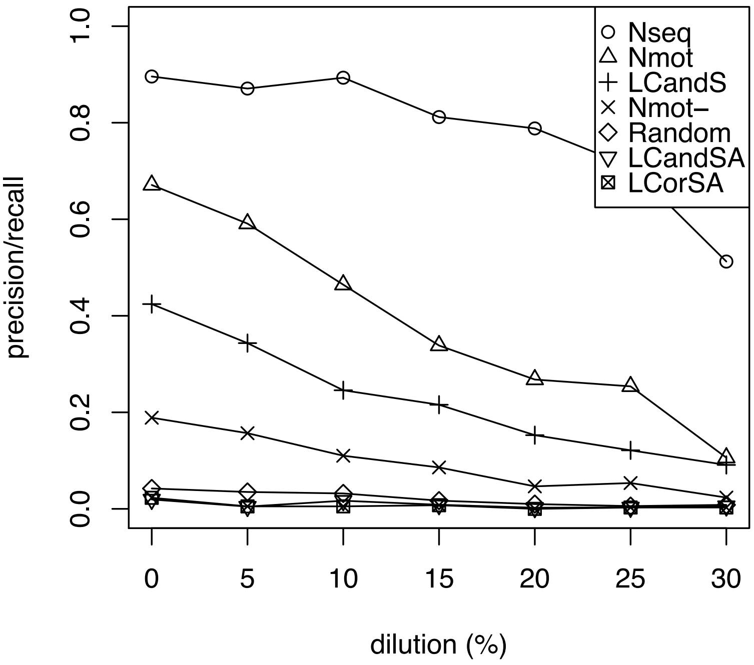


Figure 2.9: Precision of SA-cmm with different neighborhood functions on artificial networks with implanted motifs.

Neighbor func.	seeds
N^{seq}	90K
N^{mot}	277K
LCandS	784K
$N^{\mathrm{mot}}_{\ominus}$	1 986K
LCandSA	3 315K
LCorSA	3 643K
Random	15 924K

Figure 2.10: Average amount of randomly chosen initial motif pairs per run for each neighborhood function.

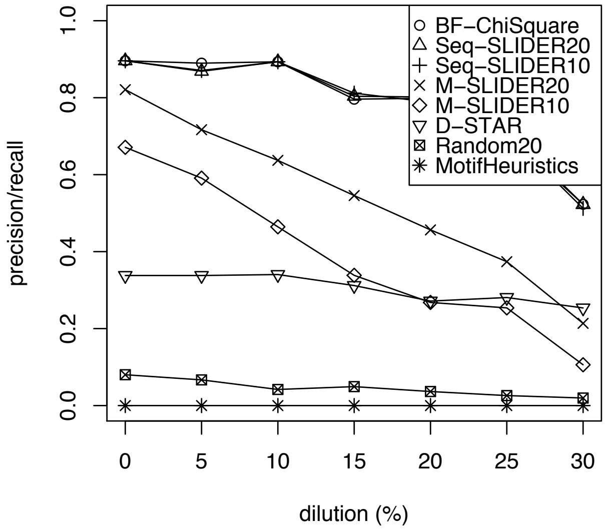


Figure 2.11: Precision of SLIDER compared with that of D-STAR, MotifHeuristics and Random on artificial networks.

the noise level, indicating that in those cases the merit of a larger neighborhood is overshadowed by the time it costs to search it.

2.8.4 Comparison with existing methods

D-STAR. Tan et al. introduced the first motif-driven method for CMM: D-STAR [THSN06]. In contrast with our approach, D-STAR uses (ℓ, d)-motifs in the *mismatch model*. In the mismatch model, an (ℓ, d)-motif is simply a string s of length ℓ and an amino acid sequence is said to contain the (ℓ, d)-motif s if it contains a substring of length ℓ that differs in at most d characters from s. D-STAR is based on the observation that two strings s_1 and s_2 which both differ at most d characters from s, differ at most in $2d$ characters from each other. Strictly spoken, D-STAR does not deliver (ℓ, d)-motifs. Instead it returns two strings s_X and s_Y, and two sets of proteins V_X and V_Y together with the indices of the substring of the amino acid sequence of each protein in V_X that differs at most $2d$ characters from s_X, and similarly for V_Y and s_Y. To construct the $\{V_X, V_Y\}$-pairs, D-STAR considers for each interaction $\{v, w\}$, each substring of length ℓ in $\lambda(v)$ and $\lambda(w)$ as the initial strings s_X and s_Y, determines V_X and V_Y, and evaluates $\{V_X, V_Y\}$ using f_{χ^2}. As the similarity in Section 2.8.1 is defined in terms of positions of substrings, we can directly use the returned subsets V_X and V_Y to compare with implanted motifs. Every run of D-STAR on the same network produced the same result, consequently the running time of D-STAR cannot be parameterized. We used the D-STAR implementation freely available on the web.

MotifHeuristics. Another method, called MotifHeuristics, proposed by Leung et al. [LSY$^+$07], derives (ℓ, d)-motifs directly within the wildcard model and introduced the probabilistically motivated f_p-support. Although the authors do not describe it as such, MotifHeuristics can be seen as a steepest ascent method in which the neighbors of a motif pair $\{X, Y\}$ are all motif pairs $\{X, Y'\}$ at odd steps and all motif pairs $\{X', Y\}$ at even steps. Because we could not obtain an implementation of MotifHeuristics, we implemented our own version based on the algorithmic description in [LSY$^+$07] and confirmed the correctness of the implementation by reproducing all results on the SH3-dataset from [LSY$^+$07].

Comparison. Given that each method relies on different principles, it is not easy to compare them directly. Both SLIDER variants and MotifHeuristics share the principle that they start from a random motif pair which is improved by local search principles. One could be tempted to compare them by looking at the results which each method obtains using a fixed number of motif pair seeds but such a comparison would favor a method which considers a larger neighborhood in each step, that is, has an expensive neighbor function. Moreover, D-STAR is a deterministic method and as such is unable to improve its results by using more time. For those reasons, we believe that comparing the results obtained by each method within a given time frame yields the fairest comparison as time is the most important constraint for large PPI-networks and any method method requires time to produce results.

The graph in Figure 2.11 depicts the precision of the various methods on the artificial network of density 10%, including Random as naive baseline. D-STAR took 5 minutes to finish. We let Random and both SLIDER variants run once for 10 and once for 20 minutes. To give our unoptimized implementation of MotifHeuristics a fair chance, we allowed it to run for 175 minutes. The underlying reason why MotifHeuristics takes such a long time is that for every search step a number of supports has to be calculated which approaches the total number of motifs. The graph makes it quite apparent that SEQ-SLIDER is vastly superior to all other methods — the precision obtained by both SEQ-SLIDER runs are so close to the precision obtained by brute force that they are almost indistinguishable in the figure. M-SLIDER is second as long as the network is not too noisy but loses to D-STAR as the networks become more noisy. It might be noteworthy that D-STAR finishes in more or less 5 minutes but, as mentioned earlier, its results cannot be improved by giving it more time. We also performed 5 minute M-SLIDER runs to make the comparison with D-STAR more fair and in that time frame M-SLIDER's precision is only better than D-STAR's for the original networks. On the other hand, if we give M-SLIDER more time it beats D-STAR on all noise levels. Somewhat surprisingly, Random performs better than MotifHeuristics. Calculating f_p-support for an enormous amount of neighbors takes so much time that our implementation of MotifHeuristics could handle only about 120 initial motif pairs in 175 minutes. Hence, this experiment indicates that even a random search using f_{χ^2} is a better approach to retrieve implanted motif pairs than a heuristic search using f_p.

Overall, both SLIDER variants are more effective and robust than its competitors although M-SLIDER needs more time to outclass D-STAR on these small networks.

We conclude the comparison by pointing out that both SLIDER variants obtain a precision>80% in 20 minutes on the original networks, which is quite fast in comparison with the 40 hours necessary to obtain the best motif pairs by brute force.

2.8.5 Biological validation

Next, we assess the effectiveness of SLIDER on two of the largest real-life PPI-networks: the yeast network and the human network (see Section 2.7.2).

Retrieving the best motif pairs.

We will first assess if M-SLIDER and SEQ-SLIDER are still capable of retrieving the best motif pairs on networks of this size. As the motif pairs which describe the interactions in the real PPI-networks are not known, we use the 1 000 best scoring motif pairs obtained by a brute force algorithm as the "ground truth". Hence, the notion "precision" is a bit misleading here because the real motifs describing the interactions are unknown and might not even exist because of the limitations of the (ℓ, d)-motif model. Nevertheless, from a purely theoretical point of view, calculating precision against the best scoring motif pairs is a correct and objective merit to assess the capability of our methods to find the best motif pairs *according to the model.* Moreover, because in this setting we are guaranteed to compare against all best scoring motif pairs, we do not have to rely on the positional similarity measure and can compare the two sets of motif pairs directly.

To give an idea, the brute force computation for (8,3)-motif pairs on the yeast network occupied about 100 nodes in the cluster spanning a period of 2 weeks.

We ran M-SLIDER and SEQ-SLIDER for 20 minutes exploiting all 8 cores of the Mac Pro. The average precision of the 1 000 best results returned by M-SLIDER over 5 runs is 14%, that of SEQ-SLIDER is 74.2%. The number implies that SEQ-SLIDER succeeds in recovering 742 of the 1000 best correlated motifs out of a search space of 6×10^{15} (8,3)-motif pairs after only a run of 20 minutes which is quite satisfactory. As SEQ-SLIDER returns a ranked list, these 742 motif pairs occur at the top.

Biological relevance of best motif pairs.

We will now assess the biological relevance of the results of the brute force algorithm, SEQ-SLIDER and MotifHeuristics on the yeast network and the human network. We used our own implementation of MotifHeuristics, but allowed it to run significantly longer. We did not assess D-STAR, because even though D-STAR terminated on our artificial networks within 5 minutes, the method does not scale to larger networks. In particular, Leung et al. [LSY+07] mention an experiment where they executed D-STAR on the yeast network and it did not finish in 5 days. We ourselves have run D-STAR on this network for a month without result. We took protein structures from the protein databank (PDB) [BBB+02] and selected only those that could be mapped to proteins in the human and yeast networks (using `pdb_homologs.tab` from `yeastgenome.org` for yeast and the GTOP database [KFH+02] for human), with blast e-value < 1E-10. We discarded any structures where no two separate chains of the structure could be mapped to two interacting proteins in one of the networks, or where one or both of those proteins didn't contain a motif from the result. Subsequently, we used NACCESS [HT93] to calculate the Relative Solvent Accessibility (RSA) of each residue in the PDB structures. The higher RSA, the more at the surface a residue is. Protein sequences were aligned with PDB protein sequences, and in this way the solvent accessibility of residues covered by a correlated motif was obtained (see example in the table in Figure 2.12). This was done two times for each residue: once in the structure of the complex (two chains bound to each other) and once in the free protein chain. The solvent accessibility of these residues in the single proteins was compared with that in the protein complex structure. Residues which have a smaller accessibility in the complex, are considered to be at the interaction site. For example, for the residues listed in the table in Figure 2.12, the first, second, fourth and eighth residue, respectively `R`, `D`, `P` and `F`, have accessibility 35.6, 39.2, 33.3 and 7.5 in the single chain, but only 1.2, 18.0, 6.0 and 0 in the complex, which implies that that they are indeed at the interaction site. You can see the positions of the motifs in that complex in Figure 2.13.

Unfortunately, because of the limited available structure information, none of the proteins of the human network survived both the PDB-mapping and motif-filtering phase for (8,3)-motifs obtained by SEQ-SLIDER. The number of

Position in protein	**321**	**322**	323	**324**	325	326	327	**328**
Residue	**R**	**D**	P	**P**	H	N	N	**F**
Position in PDB	**322**	**323**	324	**325**	326	327	328	**329**
Residue	**R**	**D**	P	**P**	H	N	N	**F**
RSA (single chain)	**35.60**	**39.20**	8.90	**33.30**	21.00	1.20	0.80	**7.50**
RSA (complex)	**1.20**	**18.00**	8.90	**6.00**	21.00	1.20	0.80	**0.00**

Figure 2.12: Mapping a motif hit of `RDxxxxNx` (rank 7, SEQ-SLIDER) in protein 18 010 of the human network to PDB 1Y8Q, chain C. The residues in bold are at the interaction site according to the RSA values. Its partner motif `GxGxxGxx` also occurs at the interface of the complex.

proteins remaining for yeast is also extremely small, as can be seen from Figure 2.14. For that reason, we ran the brute force method and SEQ-SLIDER using (the less informative) (8,5)-motifs where we used all 8 cores of our machine for an hour and 15 minutes (for an equivalent of 10 hours of computation on a single core) for both the yeast and human network to increase the number of motif hits for which RSA values can be obtained. Each of these results gave us 1 000 motif pairs ranked by their χ^2-support. We ran our own implementation of MotifHeuristics for the equivalent of a month of computation time.

To see if the current (real) motif pair interface coverage is statistically significant, we prepared 100 sets of random motif pair occurrences in the sequences from the interaction network and analyzed how many of them have more motif pair interface coverage than the real data. These datasets were generated from the original result set by choosing a random new position for each motif hit in the sequence in which it appears. Results of this comparison are shown in the tables in Figure 2.14 and Figure 2.15.

Proteins shows the amount of proteins that remains when both both proteins of a pair need to contain at least one motif from the result. *Motif hits* shows the number of motif-protein hits after filtering data such that only motif hits for which a complementary motif hit is present in an interacting protein (with both protein having an associated structure) are kept. *At site* shows the percentage of randomly generated motif hit datasets that have more hits at interaction sites than the result of the method. *Random* $\geq$ shows the percentage of random sets that have greater presence at the interaction site.

Both for the yeast and human network we have significantly more overlap than random with the interface. Notably, for the human network only 2 out of the 100 random sets have at least 45% of their motifs overlapping with the interface (as observed for the SEQ-SLIDER motifs). In this run, the average of the percentage of motif hits overlapping with the interface is 36.5 for the random motif hits and the standard deviation 4.5. The fact that SEQ-SLIDER

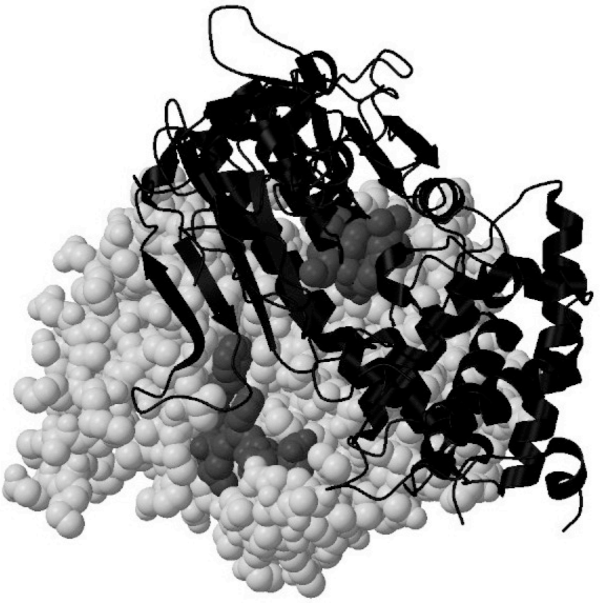


Figure 2.13: Two interacting chains C and D of PDB 1Y8Q in black and white and the two motif hits in gray.

Method parameters	Proteins	Motif hits	At site	Random ≥
Brute force χ^2, (8,3)	252	48	13 (27%)	37%
Brute force χ^2, (8,5)	949	5 335	2 103 (39%)	48%
SEQ-SLIDER χ^2, (8,5), 600min	949	1 157	335 (29%)	12%
MotifHeuristics p, (8,5), 1 month	949	615	224 (36%)	50%
SEQ-SLIDER (restricted: 400)	926	817	319 (39%)	6 %

Figure 2.14: Occurrences at surface and at interaction site compared to random sampling in the yeast high-confidence network (1620 proteins/9060 interactions).

Method parameters	Proteins	Motif hits	At site	Random ≥
Brute Force χ^2, (8,5)	229	188	61 (32%)	23%
SEQ-SLIDER χ^2, (8,5), 600min	229	137	62 (45%)	2%
MotifHeuristics p, (8,5), 1 month	208	14	8 (57%)	8%
SEQ-SLIDER (restricted: 24)	156	13	8 (62%)	4%

Figure 2.15: Occurrences at surface and at interaction site compared to random sampling in the human network (8872 proteins/34230 interactions).

has more overlap with the interaction site than brute force can be explained by the more complementary nature of the SEQ-SLIDER motif pairs; their motif hits cover more regions in the sequences (see Section 2.8.6).

We also ran MotifHeuristics on the large-scale networks. As the method did not return a single motif pair after ten hours, we allowed it to run for a full month, still producing less motif pairs than SEQ-SLIDER in a ten-hour run. We restrict the comparison to the same number of found motif pairs. SEQ-SLIDER still finds a larger overlap with the interface (See the tables in Figure 2.14 and Figure 2.15).

Using an additional cutoff for the interface (i.e. not only requiring change in RSA upon complexation but also that RSA in free protein is above a cutoff) does not change much in analysis (data not shown).

Conclusion.

We find significant overlap of motif hits with interface residues for SEQ-SLIDER, on both the yeast and human results. That being said, the results on human are remarkably better than those for yeast. Our experimental results seem to suggest that the model itself is better in describing the interactions in the human network than the interactions in the yeast network. A possible explanation for the skewness in these results is that the (ℓ, d) with χ^2-support model suffers more from false positives caused by indirect interactions, which are prominently present in the yeast network, than from false negatives, which are assumed to be common in the human network, as explained above.

It might be worth pointing out that, as far as we know, this is the first effort to assess if CMM is able to produce biologically meaningful results from genome-wide PPI-networks.

2.8.6 Performance comparison for SEQ-SLIDER and brute force computation

At first sight, it seems strange that on yeast and human SEQ-SLIDER outperforms the brute force computation (as seen in the tables in Figure 2.14 and Figure 2.15). The main reason is that brute force returns the 1 000 best motif pairs with the highest score even if several of them are very similar. To define similarity, we extend the definition of Section 2.8.1 as follows.

We say that two (ℓ, d)-motif pairs $\{X, Y\}$ and $\{X', Y'\}$ are (α, β)-similar in a PPI-network $G = (V, E, \lambda)$ if

$$s(\{X, Y\}, \{X', Y'\}, G) = \frac{|E_{X,Y} \cap_{pos} E_{X',Y'}|}{|E_{X,Y} \cup E_{X',Y'}|} \geq \alpha$$

(α, β)	SEQ-SLIDER	BF
(1.0, 8)	0	11
(0.75, 3)	3	161
(0.5, 1)	91	402

Figure 2.16: For both methods, the amount of motif pairs (out of 1 000) that are (α, β)-similar to a better-scoring motif pair in the result.

where $\{v, w\} \in E_{X,Y} \cap_{pos} E_{X',Y'}$ if there exists substrings s_v and s'_v in $\lambda(v)$ and s_w and s'_w in $\lambda(w)$ such that

- s_v matches with X and s_w with Y;
- s'_v matches with X' and s'_w with Y'; and,
- s_v and s'_v as well as s_w and s'_w overlap in at least β positions in $\lambda(v)$, respectively $\lambda(w)$.

Let $S = \{M_1, \ldots, M_n\}$ be a list of motif pairs, then we reduce S by deleting for every j from 1 to n, every M_i for $i > j$ such that $s(M_i, M_j) \geq \alpha$. We denote the reduced version of S by S^*.

The table in Figure 2.16 shows that for decreasing values of α and β both brute force and SEQ-SLIDER select more and more similar motif pairs on the human network. As we suspected, these tests show that a motif pair $\{X, Y\}$ appearing in the list of motif pairs with highest support is often accompanied with very similar motif pairs in the list in the sense that they select (almost) the same V_X, V_Y, $E_{X,Y}$ and also have the same or largely overlapping motif hits in the respective sequences. Consequently, if the motif hits of such a motif pair are not part of the interface, then the motif hits of the whole set of similar motif pairs are not at the interface.

This tendency to end up with a lot of similar motif pairs, is far less present in SEQ-SLIDER because in each run, it starts from a random point and moves to a local optimum without reporting the similar motif pairs it considered to reach this local optimum. Hence, although the average support of the motif pairs returned by SEQ-SLIDER is lower than those returned by brute force, the motif pairs are less likely to have the same or largely overlapping motif hits and thus more likely to cover more regions of the proteins than the motif pairs returned by a brute force computation.

For the same reason, the brute force results tend to explain less interactions and we will consider an approach that focuses on coverage of the interactions and/or regions of the proteins in Chapter 3.

The fact that the results for the brute force methods are much more similar, also explains that the brute force method has more motif hits, even though the same amount of proteins is selected (as seen in the tables in Figure 2.14 and Figure 2.15). A lot of those motif hits will only be a few amino acids apart.

So when a motif is found not to be at the interface, this means that all similar motifs are also not at the interface. So brute force is punished more severely than SEQ-SLIDER. That is why the randomly generated motif hits manage to outperform the brute force method so much more.

2.8.7 Additional simulated data

It is not likely that every interaction in a network is explainable by a motif pair following the (ℓ, d)-motif model. Therefore, we want simulated networks where only a small amount of edges could be explained by a motif pair, to verify whether our method could detect those that were present. We made two different kinds of simulated network and ran tests on them.

First, we implanted a single motif pair into a small existing network (120 proteins, 90 interactions), adding a single biclique that could be explained by one motif pair. To make sure we did not add a biclique of too large a size, we added both motifs a number of times that was smaller than the largest amount of proteins selected by a single motif occurring naturally within the network. We made 5 networks of this sort, and ran SEQ-SLIDER on them for 10 minutes. SEQ-SLIDER always managed to find the implanted motif pair as its top result.

Next, we made networks based on the known yeast network (1 620 proteins, 9 060 interactions). We added motif pairs in the same way as the simulated networks described in the paper until the amount of edges present had grown by 10%. We also made 5 of these networks. We ran SEQ-SLIDER on them for 60 minutes and found on average roughly half of the implanted motif pairs (exactly, not by positional information) (see Figure 2.17). After 300 minutes, this was over 70%. We ran SEQ-SLIDER with the p-score for the same amount of time. Even after 300 minutes, the precision did not rise above 10%.

2.9 Conclusion

Steepest ascent is not only the oldest, but also the simplest among the known metaheuristics for combinatorial optimization [BR03]. Several others exist that would avoid getting stuck in local optima and move on to a better, global optimum. We tried simulated annealing with several parameters for its starting temperature, annealing schedule and acceptance function and found no

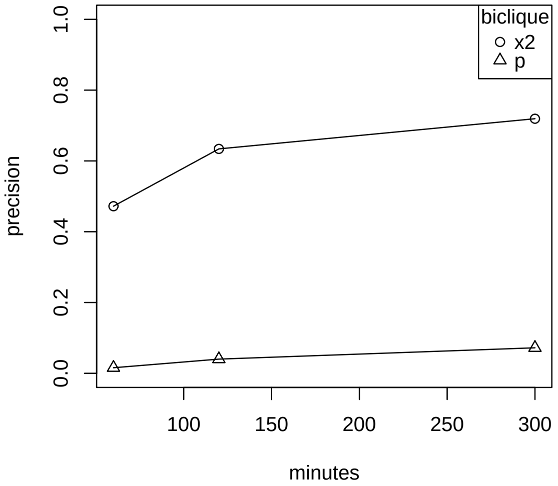


Figure 2.17: Percentage of implanted motif pairs recovered over time within the yeast network.

improvement upon our steepest ascent algorithm, we even found it to generate worse results. The more advanced metaheuristics improve upon steepest ascent by escaping a local optimum by taking a few steps in a direction that decreases the support to gain access to a region from where a better local optimum can be reached. We checked if such a search path is feasible for the neighborhood function N^{seq}. As N^{seq} always takes its motifs from two proteins, we can visualize the search space (for one starting seed) as a 2D plane. Each point (x, y) on this plane represents the best support out of all possible motif pairs $\{X, Y\}$ where X starts at position x and Y at position y. We have visualized these search spaces for several interacting motif pairs in the yeast network and found that the local maxima are too far away from each other to be reached by such an approach. We also observed that the search space consists mostly of positions where all neighbors have the same support. Steepest ascent would immediately stop at these points, where simulated annealing would continue to walk around randomly until it has moved its allotted steps. Hence, it appears that the search landscape of CMM is not suitable for these more advanced metaheuristics. An example of such a visualized search space is given in Figure 2.18.

At first sight the present work seems highly related to the mining of frequent patterns in sequences (as for instance in [GHZ07]). It is therefore tempt-

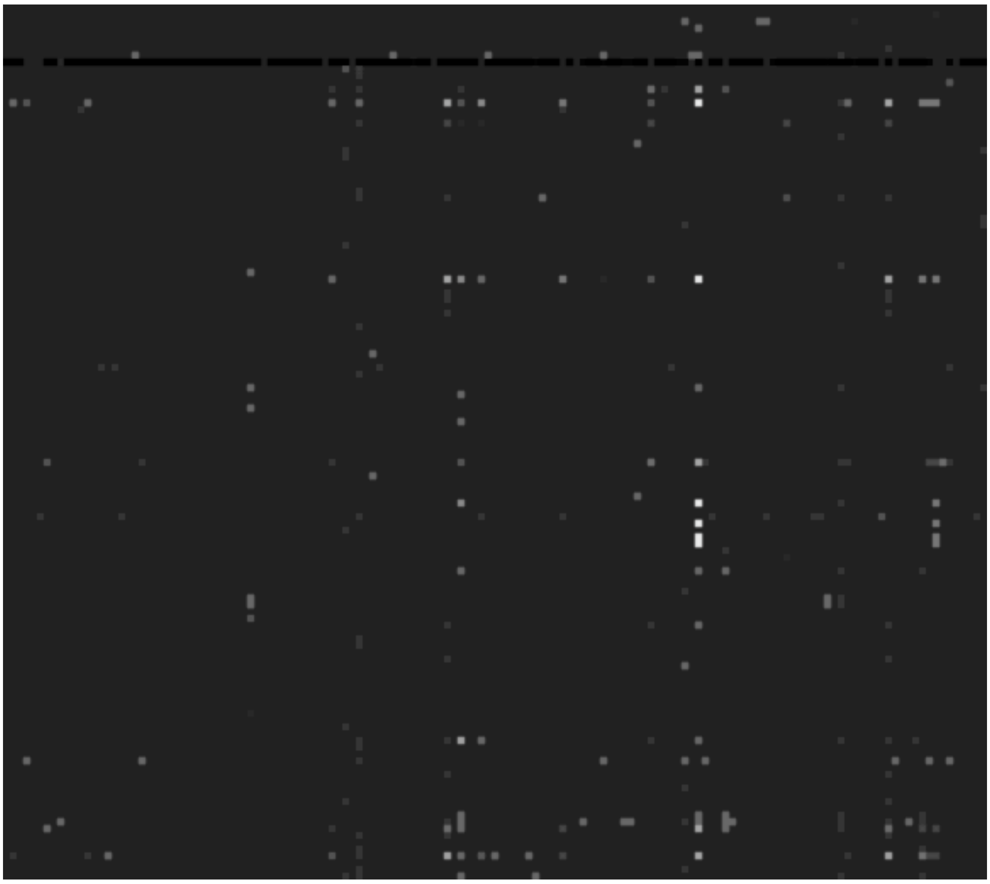


Figure 2.18: Search landscape between two proteins in the yeast high confidence network (lighter gray represents a better score).

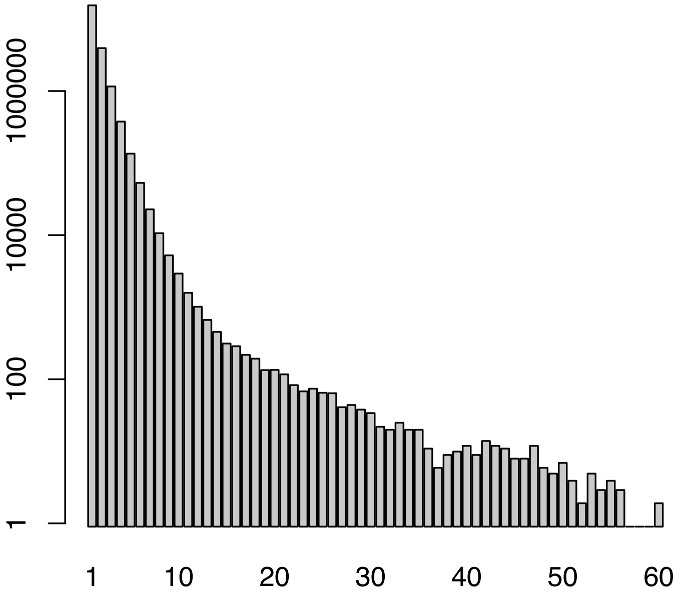


Figure 2.19: Number of (8,3)-motifs (y-axis) selecting a given number of proteins in the yeast network (x-axis).

ing to think about a method which first mines frequent motifs from protein sequences which are then paired together in a second step serving as candidates for high scoring correlated motifs. An examination of the 1 000 top correlated motifs in yeast, however, reveals that each of the participating motifs occur only in 3 to 10 proteins, whereas highly frequent motifs in yeast occur in up to 60 proteins as can be seen from the histogram in Figure 2.19. Therefore, mining correlated motifs is very different from mining frequent motifs.

We layed the foundation of motif-driven CMM by establishing an adequate support measure and determining the complexity of the general problem. The novel generic metaheuristic SLIDER based on the sliding window neighborhood function outperforms existing motif-driven CMM algorithms and shows a very promising behavior on real-world PPI-networks.

We could not confirm the claimed superiority in [LSY+07] of MotifHeuristics over D-STAR. In fact, our results clearly show that f_p is inferior to f_{χ^2} in recovering implanted motifs. These tests should be repeated on real world data, but as long as only few biological correlated motifs are known this is not possible.

The SLIDER-implementation and the data used in the experiments are available on `http://bioinformatics.uhasselt.be`.

3

Mining minimal motif pair sets maximally covering interactions

3.1 Introduction

As shown in the previous chapter, when we score a motif pair solely on its own merit within the network, we get a lot of very similar high-scoring motif pairs within the same dense part of the network. For example, among the best 1 000 $(8, 5)$-motif pairs found using a brute force approach in the PPI-network of Human (a network consisting of 8 872 nodes) only 598 refer to dissimilar subnetworks (see Section 2.8.6). This effect becomes even worse when considering more specific motif pairs (i.e., restricting the number of allowed wildcards). For example, in Yeast (a network consisting of 1 620 nodes) out of the 1 000 best $(8, 3)$-motif pairs only 382 refer to distinct networks. Every found motif pair selects a subnetwork identical to one of these, even to the point of having the same motif positions in the proteins. Moreover, using all 1 000 motif pairs, only 5% of all nodes and 8% of all edges of the Yeast network are described. Correlated motif covering (CMC) is the problem of finding a set of motif pairs in the sequences of proteins from a PPI-network which describe the interactions in the network as concisely as possible. In other words, a perfect solution for CMC would be a minimal set of motif pairs which describes the interaction behavior perfectly in the sense that two proteins from the network interact if

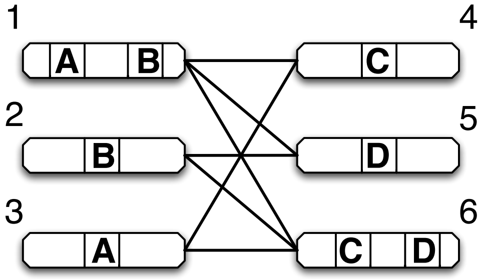


Figure 3.1: An example protein-protein interaction network (repeated).

and only if their sequences match a motif pair in the minimal set. We formally introduce the Correlated Motif Covering (CMC) problem in Section 3.2. In Section 3.3, we show that CMC is NP-hard, and that it is impossible to create a sub-square root approximation algorithm for it. We do this by reducing the Red-Blue Set Cover (RBSC) problem to CMC while using an Approximation Factor Preserving (AFP)-reduction, a form of reduction that allows us to transfer knowledge of approximation factors. We show the existence of a theoretical approximation algorithm for CMC by providing an AFP-reduction from CMC to WRBSC. We adapt the latter algorithm into a functional heuristic for CMC, called CMC-approx and then present two more naive algorithms in Section 3.4. First, we use SEQ-SLIDER as a naive baseline algorithm by interpreting the set of motif pairs it returns as a cover. Secondly, we present a greedy algorithm called CMC-greedy. In Section 3.5, we describe the used data sets, followed by a validation that the algorithms achieve their intended purpose (increased coverage) in Section 3.6. We perform several test to ensure the biological meaningfulness of our results in Section 3.7. We start by examining the usefulness of the returned motif pairs in the prediction of novel protein-protein interactions. Then, we compare the found locations to known binding sites from 3D-structure information. Our last experiment shows an application where the high coverage of CMC-approx is advantageous, namely in the comparison of results across multiple species. We provide evidence that if a site is returned as a binding site by our methods in multiple species, then the chance of it being an actual binding site increases. Finally, we conclude in Section 3.8.

3.2 Covering a graph with a set of motif pairs

Rather than scoring motif pairs on an individual basis as in CMM (see Section 2.2), we now formalize how to score the explicative power of a set of motif pairs as a whole. To this end, let $\mathcal{M}$ be a set of motif pairs. Then the graph $G_{\mathcal{M}} = (V_{\mathcal{M}}, E_{\mathcal{M}}, \lambda_{\mathcal{M}})$ induced by $\mathcal{M}$ on G is defined as

$$G_{\mathcal{M}} := \bigcup_{\{X,Y\}\in\mathcal{M}} (V_X \cup V_Y, V_X \times V_Y, \lambda_{|V_X \cup V_Y}),$$

where the union of two graphs $G_1 = (V_1, E_1, \lambda_1)$ and $G_2 = (V_2, E_2, \lambda_2)$ is simply $G_{1,2} = (V_1 \cup V_2, E_1 \cup E_2, \lambda_1 \cup \lambda_2)$. Note that $G_{\mathcal{M}}$ might contain edges not present in G, contrary to the case for G_M for a motif pair M. For example, if we were to take $G_{\{\{B,C\}\}}$ in Figure 3.1, we get the network containing proteins 1, 2, 4, 6, with four edges connecting those proteins, even though in G and $G_{B,C}$ only three of those edges are present.

For a result set $\mathcal{M}$ to have little redundancy and high coverage, we want $\mathcal{M}$ to be small while maximizing the similarity between $G_{\mathcal{M}}$ and G. Therefore, we adopt the Minimum Description Length (MDL) principle [Ris83] which embraces the slogan of *Induction by Compression.* More specifically, we can compress G by $G_{\mathcal{M}}$ encoded as $\mathcal{M}$. We then only need to list the false positive and false negative edges. That is, the edges present in $E_{\mathcal{M}}$ but not in E_G, and the edges missing in $E_{\mathcal{M}}$ but present in E_G. The size of $\mathcal{M}$, denoted $|\mathcal{M}|$ is simply counted as the number of motif pairs occurring in it. More formally, we calculate the size of the latter compression relative to three nonnegative numbers α, β, and γ as follows

$$\text{cost}_{\alpha,\beta,\gamma}(\mathcal{M}, G) := \alpha|\mathcal{M}| + \beta|E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}|.$$

The table below lists some example solutions for the network in Figure 3.1:

$\mathcal{M}$	$cost_{\alpha,\beta,\gamma}(\mathcal{M}, G)$
$\emptyset$	7γ
$\{\{B,C\}\}$	$\alpha + \beta + 4\gamma$
$\{\{B,C\},\{A,D\}\}$	$2\alpha + 2\beta + 2\gamma$
$\{\{A,C\},\{B,D\}\}$	2α

The last line contains a solution that describes the interactions perfectly, and, given reasonable values for α, β, and γ, should be the optimal solution.

In the experimental section, following the MDL principle, we will set α to the number of bits to represent a motif pair and $\beta = \gamma$ to the number of bits to represent an interaction.

We are now ready to define the problem central to this paper:

The *Correlated Motif Covering* problem (CMC)

- **Input:** A PPI-network G, numbers $\ell, d \in \mathbb{N}^+$ with $d < \ell$, and α, β, $\gamma \in \mathbb{Q}^+$
- **Output:** a set $\mathcal{M}$ of (ℓ, d)-motif pairs minimizing $\mathrm{cost}_{\alpha,\beta,\gamma}(\mathcal{M}, G)$.

3.3 Complexity and Approximation of CMC

We show that CMC is NP-hard and that there are limits to how well it can be approximated within polynomial time. On the positive side, we show that there are algorithms that provide approximation guarantees that are within those limits by establishing an Approximation Factor Preserving (AFP-)reduction to the Weighted Red-Blue Set Cover (WRBSC) problem.

3.3.1 Lower bounds

In this section, we show that CMC is NP-hard and that it is hard to approximate. Specifically, we show that, unless P = NP, it is impossible to achieve a polynomial-time algorithm for CMC with an $\mathcal{O}(2^{\log^{1-\delta}\sqrt{n}})$ approximation ratio, with $\delta = 1 / \log \log^c n$, for any constant $c < 1/2$, with n the size of the input.

We start by reducing the Red-Blue Set Cover (RBSC) problem to CMC using AFP-reductions.

First, we define RBSC [CDKM00]. The input consists of a universe U of elements which are either red or blue. In addition, a set $\mathcal{S}$ of subsets of U is given. The objective is to choose sets from $\mathcal{S}$ covering all blue elements while selecting as few red elements as possible.

Formally, we define the *Red-Blue Set Cover* (RBSC) problem [CDKM00] as follows:

- **Input:** Finite sets $U, B, R, \mathcal{S}$, with $U = B \cup R$, $B \cap R = \emptyset$, and $\mathcal{S} \subseteq 2^U$, with $\bigcup\limits_{S \in \mathcal{S}} S = U$.
- **Output:** a set $\mathcal{S}^* \subseteq \mathcal{S}$ such that $B \subseteq \bigcup\limits_{S \in \mathcal{S}^*} S$ minimizing the cost function

$$cost_{\text{RBSC}}(\mathcal{S}^*, U, B, R) := |\left(\bigcup_{S \in \mathcal{S}^*} S\right) \cap R|.$$

To obtain the lower bounds for CMC, we make use of a reduction from RBSC to CMC that preserves not only constant factor approximability but also the constant itself. Therefore, we will need the following notion:

Definition 3.1. [Vaz04] Let Π_1 and Π_2 be two minimization problems. An *Approximation Factor Preserving* (AFP-)*reduction* from Π_1 to Π_2 is a pair of PTIME functions (f, g) such that:

- for any instance I_1 of Π_1, $I_2 = f(I_1)$ is an instance of Π_2 such that $\mathrm{OPT}_2(I_2) \leq \mathrm{OPT}_1(I_1)$, where OPT_1 (resp. OPT_2) is the quality of an optimal solution to I_1 (resp. I_2), and
- for any solution s_2 to I_2, $s_1 = g(s_2)$ is a solution to I_1 such that $\mathrm{obj}_1(s_1) \leq \mathrm{obj}_2(s_2)$, where $\mathrm{obj}_1()$ (resp. $\mathrm{obj}_2()$) is a function measuring the quality of a solution to I_1 (resp. I_2).

By $\Pi_1 \leq_{AFP} \Pi_2$, we denote that there exists an AFP-reduction from Π_1 to Π_2.

Throughout the remainder of this section, obj_Π is the cost function of problem Π and $\mathrm{OPT}_\Pi(I)$ equals the cost of an optimal solution for instance I of problem Π.

The proofs in the rest of this section are based on the following lemma, which we state without proof [Vaz04]:

Lemma 3.2. *If there exists an approximation algorithm with approximation factor α for problem Π_1, and $\Pi_2 \leq_{AFP} \Pi_1$, then there exists an approximation algorithm with approximation factor α, for problem Π_2.*

In the remainder of this section, we will prove the following theorem.

Theorem 3.3. CMC *is* NP*-hard and unless* P = NP*, there is no polynomial-time algorithm for* CMC *with an $\mathcal{O}(2^{log^{1-\delta}\sqrt{n}})$ approximation ratio, with $\delta =$ 1 / log log$^c n$, for any constant $c < 1/2$, with n the size of the input.*

We need the following lemma:

Lemma 3.4. RBSC $\leq_{AFP}$ CMC.

Proof. Let $I_{\text{RBSC}} = (U, B, R, \mathcal{S})$ be an instance of RBSC. Let $\mathcal{S} = \{S_1, \ldots, S_{|\mathcal{S}|}\}$. We start by defining the function f mapping I_{RBSC} to $f(I_{\text{RBSC}}) = I_{\text{CMC}} = (G, \ell, d, \alpha, \beta, \gamma)$. Here, $\ell = |\mathcal{S}|$ and $d = |\mathcal{S}| - 1$. This means that motifs only have one non-wildcard character. Furthermore, set $\alpha = 0$, $\beta = 1$, and γ to one more than the maximal number of red elements occurring in a set in $\mathcal{S}$. That is, $\gamma = \max\limits_{S \in \mathcal{S}} |\mathcal{S} \cap R| + 1$. It remains to define $G = (V, E, \lambda)$. Define $V = U \cup \{c\} \cup O$

where c is a new element and O consists of $2\gamma|B|+1$ new elements. Here, c is the center of the graph connected to all elements in B. Furthermore, O, and R consist of isolated vertices. That is, set $E = \{\{b, c\} \mid b \in B\}$. Although Σ contains the 20 characters used to specify amino acids, G uses only three characters. For ease of exposition, we shall refer to three characters in Σ as 0, 1, and 2. Then, $\lambda(c)$ is a sequence of length $|\mathcal{S}|$ consisting only of 2-characters; for each $u \in U$, $\lambda(u) = c_1 \dots c_{|\mathcal{S}|}$, with $c_i = 1$ if $u \in S_i$ and 0 otherwise; and; for each $o \in O$, $\lambda(o)$ is a sequence of length $|\mathcal{S}|$ consisting only of 0-characters. Clearly, f is PTIME-computable.

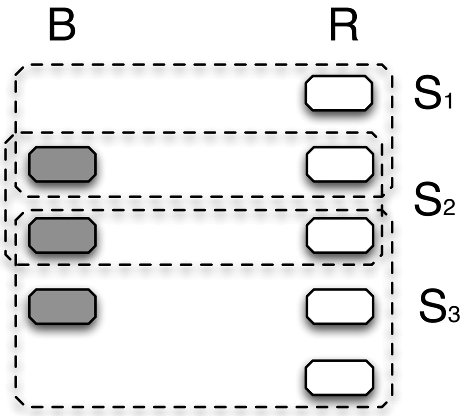


Figure 3.1: An example instance I_{RBSC} for RBSC. Blue elements are shaded in gray.

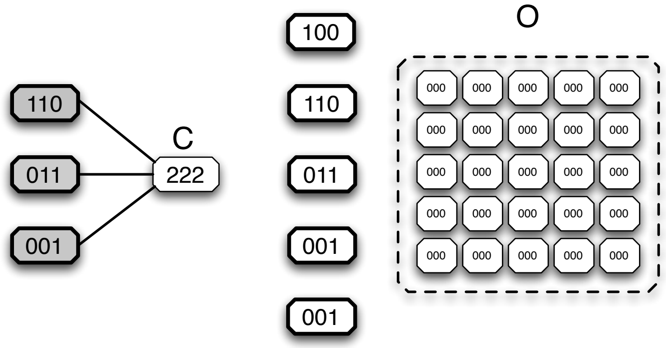


Figure 3.2: $f(I_{\text{RBSC}}) = I_{\text{CMC}}$. The newly created elements are indicated by thin edges.

Example 3.5. In Figure 3.1, we see a possible instance I_{RBSC} for RBSC. In Figure 3.2, we see the corresponding $f(I_{\text{RBSC}}) = I_{\text{CMC}}$. In I_{CMC}, as there are 3 sets in $\mathcal{S}$, $\ell = 3, d = 2$. We automatically set $\alpha = 0, \beta = 1$, and because the S_3 contains the most red elements, we set $\gamma = 4$, making $|O| = 2 \times 4 \times 3 + 1 = 25$. The first blue element gets a sequence of `110` as it is an element of S_1 and S_2, but not of S_3.

$\triangleleft$

We next argue that $\text{OPT}_{\text{CMC}}(I_{\text{CMC}}) \leq \text{OPT}_{\text{RBSC}}(I_{\text{RBSC}})$. First, we need some terminology concerning motif pairs. As $\ell = d + 1$, every motif contains precisely one non-wildcard character σ. We refer to such a motif as a σ-motif. We say that a motif pair $\{X, Y\}$ where X is a σ-motif and Y is a σ'-motif, is a motif pair of type $[\![\sigma, \sigma']\!]$.

Let $s_{\text{CMC}} = (\mathcal{M})$ be a solution to I_{CMC}. Then $\text{obj}_{\text{CMC}}(\mathcal{M}) = \alpha|\mathcal{M}| + \beta|E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| = |E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}|$ by choice of α and β. Assume $\mathcal{M}$ is optimal. Then $\mathcal{M}$ has to cover all edges. That is, $E_G \setminus E_{\mathcal{M}} = \emptyset$. Indeed, assume towards a contradiction that $\{b, c\} \in E_G \setminus E_{\mathcal{M}}$ and let $b \in S_i$ (b is always in at least one set, according to the definition of RBSC). Then, we can extend $\mathcal{M}$ with a motif pair selecting $\{b, c\}$ thereby decreasing its cost and contradicting the fact that $\mathcal{M}$ is optimal. Indeed, denote by M_i the motif pair $\{X_i, Y_i\}$ where X_i is the 1-motif with 1 on the i-th position and Y_i is the 2-motif with 2 on the i-th position. Then

$$\begin{aligned}
&\text{obj}_{\text{CMC}}(\mathcal{M} \cup \{M_i\}) = \\
&\quad = \quad |E_{\mathcal{M} \cup \{M_i\}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M} \cup \{M_i\}}| \\
&\quad \leq \quad (|E_{\mathcal{M}} \setminus E_G| + |S_i \cap R|) + (\gamma|E_G \setminus E_{\mathcal{M}}| - \gamma) \qquad (\dagger) \\
&\quad < \quad |E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \qquad (\ddagger) \\
&\quad = \quad \text{obj}_{\text{CMC}}(\mathcal{M}).
\end{aligned}$$

Here, (†) is an equality if S_i contains only one blue element not yet covered, and (‡) follows as $\gamma = \max_{S \in \mathcal{S}} |S \cap R| + 1 > |S_i \cap R|$. So, it follows that $E_G \setminus E_{\mathcal{M}} = \emptyset$ and that $\text{obj}_{\text{CMC}}(\mathcal{M}) = |E_{\mathcal{M}} \setminus E_G|$ when $\mathcal{M}$ is optimal.

Example 3.6. In the example in Figure 3.2, the set $\{\{\texttt{xx1}, \texttt{2xx}\}\}$ covers two out of three edges, and its cost is $3 + 4 = 7$. By adding the motif pair $\{\texttt{x1x}, \texttt{2xx}\}$, we reduce the cost to 5. A cost of γ was removed, and at most $\gamma - 1$ could be added.

$\triangleleft$

We distinguish the following kinds of motif pairs:

- We say that a motif pair is *good* when it is of type $[\![1, 2]\!]$. Note that every such motif pairs selects the subgraph consisting of the center and all elements in some set of S.
- We say that a motif pair is *bad* when it is of type $[\![0, 2]\!]$ Note that every such motif pairs selects the subgraph consisting of the center and all elements in the complement of some set of S, together with all the elements in O.
- Motif pairs which only contain σ-motifs with $\sigma \in \{0, 1, 2\}$ but which are not good nor bad are called *superfluous.* Specifically, they are of type $[\![0, 0]\!]$, $[\![1, 1]\!]$, $[\![2, 2]\!]$, and $[\![0, 1]\!]$. They are called superfluous as $E_M \cap E_G = \emptyset$ for every superfluous motif pair M.
- A motif pair $M = \{X, Y\}$ which is not good, bad, or superfluous is called *empty.* Specifically, it is of type $[\![\sigma, \sigma']\!]$ where $\sigma \notin \{0, 1, 2\}$ or $\sigma' \notin \{0, 1, 2\}$. They are called empty because at least one of V_X or V_Y is empty.

Example 3.7. In the example in Figure 3.2, $\{\mathtt{x1x}, \mathtt{2xx}\}$ is a good motif pair, it selects the center and the elements that were in S_2 in Figure 3.1. Also, $\{\mathtt{x0x}, \mathtt{2xx}\}$ is a bad motif pair, it selects the center and all the elements that were not in S_2, including all the elements of O. Next, $\{\mathtt{x0x}, \mathtt{0xx}\}$ is a superfluous motif pair, it selects elements, but never any edges. Finally, $\{\mathtt{x3x}, \mathtt{0xx}\}$ would be an empty motif pair, as one of its motifs never selects any elements, since it contains a character not present in any sequence.

$\triangleleft$

Next, we argue that the optimal solution $\mathcal{M}$ cannot contain bad motif pairs. Furthermore, we will show that for every optimal solution $\mathcal{M}$ there is a reduced optimal solution $\mathcal{M}'$ which does not contain empty or superfluous motif pairs and only a minimal number of good motif pairs which can be mapped in a one-to-one fashion on sets in $\mathcal{S}$.

Indeed, assume $\mathcal{M}_{bad}$ to be the non-empty set of bad motif pairs in any solution $\mathcal{M}$. Then

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(\mathcal{M} \setminus \mathcal{M}_{bad}) \\
&\quad = \; |E_{\mathcal{M}\setminus\mathcal{M}_{bad}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}\setminus\mathcal{M}_{bad}}| \\
&\quad \leq \; |E_{\mathcal{M}\setminus\mathcal{M}_{bad}} \setminus E_G| + \gamma|E_G| \\
&\quad \leq \; |E_{\mathcal{M}} \setminus E_G| - |O| + 2\gamma|E_G| && (\dagger) \\
&\quad < \; |E_{\mathcal{M}} \setminus E_G| && \text{(by def. of } O) \\
&\quad \leq \; |E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad = \; \mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}).
\end{aligned}$$

Here, (†) follows as the motif pairs in $\mathcal{M}_{bad}$ select at least all elements in O, and at most all the elements in $|E_G|$.

Also, assume $\mathcal{M}_{sup}$ to be the non-empty set of superfluous motif pairs in any solution $\mathcal{M}$. Then

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(\mathcal{M} \setminus \mathcal{M}_{sup}) \\
&\quad = \; |E_{\mathcal{M}\setminus\mathcal{M}_{sup}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}\setminus\mathcal{M}_{sup}}| \\
&\quad = \; |E_{\mathcal{M}} \setminus E_G| - |E_{\mathcal{M}_{sup}}| + \gamma|E_G \setminus E_{\mathcal{M}}| && (\dagger) \\
&\quad \leq \; |E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad = \; \mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}).
\end{aligned}$$

Here, (†) follows from the fact that $E_{\mathcal{M}_{sup}} \cap E_G = \emptyset$.

Finally, assume $\mathcal{M}_{empty}$ to be the non-empty set of empty motif pairs in any solution $\mathcal{M}$. Then

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(\mathcal{M} \setminus \mathcal{M}_{empty}) \\
&\quad = \; |E_{\mathcal{M}\setminus\mathcal{M}_{empty}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}\setminus\mathcal{M}_{empty}}| \\
&\quad = \; |E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad = \; \mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}).
\end{aligned}$$

So, given an optimal set of motif pairs $\mathcal{M}$, it cannot contain any bad motif pairs. Furthermore, let $\mathcal{M}'$ be obtained from $\mathcal{M}$ by removing all empty and superfluous motif pairs, then $\mathcal{M}'$ is also optimal and contains only good motif pairs. Therefore, if $\mathcal{M}$ is optimal $\mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}) = \mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}') = |E_{\mathcal{M}'} \cap \mathrm{Red}| = |E_{\mathcal{M}} \cap \mathrm{Red}|$ with $\mathrm{Red} = \{\{c, r\} \mid r \in R\}$. Thus, each motif pair in $\mathcal{M}'$ is of the form $\{X_i, Y_j\}$ where X_i is the 1-motif with 1 on the i-th position and Y_j is the 2-motif with 2 on the j-th position. We replace each motif pair $\{X_i, Y_j\}$, with the motif pair $M_i = \{X_i, Y_i\}$ and remove any duplicates. Since any 2-motif selects only the central node, the solution still selects the same node pairs, so its cost and thus its optimality are maintained.

Example 3.8. In the example in Figure 3.2, $\mathcal{M} = \{\{\mathtt{x1x}, \mathtt{2xx}\}, \{\mathtt{xx1}, \mathtt{x2x}\}, \{\mathtt{x1x}, \mathtt{0xx}\}, \{\mathtt{x1x}, \mathtt{3xx}\}\}$ is an optimal set of motif pairs, with cost 4. It does not contain any bad motif pairs. We transform it into the set $\mathcal{M}' = \{\{\mathtt{x1x}, \mathtt{x2x}\}, \{\mathtt{xx1}, \mathtt{xx2}\}\}$, which is still optimal, by removing the empty and superfluous motif pairs and changing the motif pairs to the form of $M_i = \{X_i, Y_i\}$.

$\triangleleft$

Each M_i selects exactly those $\{u, c\}$ with $u \in S_i$. Since we have shown that $E_G \setminus E_{\mathcal{M}} = \emptyset$, we can create a solution for I_{RBSC} by taking S_i for each $M_i \in \mathcal{M}'$. Also, because $\text{OPT}_{\text{CMC}}(I_{\text{CMC}}) = |E_{\mathcal{M}} \cap Red|$, this solution for I_{RBSC} has the same cost as the optimal solution to I_{CMC}. This solution is also optimal, because if there was a $\mathcal{S}^*$ with a lower cost we could create a set of motif pairs with that same cost (by taking M_i for each $S_i \in \mathcal{S}^*$), which would be in contradiction with $\mathcal{M}$ being optimal. This means $\text{OPT}_{\text{CMC}}(I_{\text{CMC}}) = \text{OPT}_{\text{RBSC}}(I_{\text{RBSC}})$, and so definitely $\text{OPT}_{\text{CMC}}(I_{\text{CMC}}) \leq \text{OPT}_{\text{RBSC}}(I_{\text{RBSC}})$.

Example 3.9. We turn our solution $\mathcal{M}' = \{\{\mathtt{x1x}, \mathtt{x2x}\}, \{\mathtt{xx1}, \mathtt{xx2}\}\}$ for I_{CMC} (in Figure 3.1) into the solution $\mathcal{S}^* = \{S_2, S_3\}$ for I_{RBSC} (in Figure 3.2). This solution also has cost 4 and is also optimal.

$\triangleleft$

Let $s_{\text{CMC}} = (\mathcal{M})$. We next define the function g mapping s_{CMC} to $g(s_{\text{CMC}}) = s_{\text{RBSC}} = (\mathcal{S}^*)$. Here, let $\mathcal{M}_{good}$ be the set of good motif pairs in $\mathcal{M}$. Define $\mathcal{S}_{good}$ as the set of sets S_i for which there is a motif pair $\{X, Y\}$ in $\mathcal{M}_{good}$, with X the 1-motif with 1 on its ith position. For every $b \in B$, let S_b be an arbitrary set in $\mathcal{S}$ containing b. Then, define

$$\mathcal{S}_{extra} := \bigcup_{\left(b \in B, b \notin \bigcup_{S \in \mathcal{S}_{good}} S\right)} S_b.$$

Finally, set

$$\mathcal{S}^* := \mathcal{S}_{good} \cup \mathcal{S}_{extra}.$$

Clearly, g is PTIME-computable.

We next argue that $\text{obj}_{\text{RBSC}}(s_{\text{RBSC}}) \leq \text{obj}_{\text{CMC}}(s_{\text{CMC}})$. Indeed,

$$
\begin{aligned}
&\mathrm{obj}_{\text{RBSC}}(s_{\text{RBSC}}) \\
&= |\bigcup_{S \in \mathcal{S}^*} S \cap R| \\
&\leq |\bigcup_{S \in \mathcal{S}_{good}} S \cap R| + (\max_{S \in \mathcal{S}} |\mathcal{S} \cap R|)|\mathcal{S}_{extra}| \\
&\leq |E_{\mathcal{M}_{good}} \cap Red| + \gamma |E_G \setminus E_{\mathcal{M}_{good}}| \\
&= \mathrm{obj}_{\text{CMC}}(\mathcal{M}_{good}) \\
&\leq \mathrm{obj}_{\text{CMC}}(s_{\text{CMC}}).
\end{aligned}
$$

Example 3.10. In the example in Figure 3.2, $\mathcal{M} = \{\{\mathtt{x1x}, \mathtt{2xx}\}\}$ is a (non-optimal) solution. We turn it into the (optimal) solution $\mathcal{S} = \{S_2, S_3\}$ by taking S_2 due to the motif pair in $\mathcal{M}$ and adding any set containing the blue elements not yet present. The cost of $\mathcal{S}$ is equal to or lower than that of $\mathcal{M}$ since the added cost of a new set is always lower than the cost in I_{CMC} for not covering the corresponding edge.

◁

Hence, the lemma follows.

□

Since an AFP-reduction is a polynomial-time reduction and RBSC is an NP-hard problem, this lemma already shows that CMC is NP-hard.

Lemma 3.11. *If there is a constant δ, with $0 < \delta < 1$, for which there exists a polynomial-time algorithm for* CMC *with an $\mathcal{O}(2^{log^{1-\delta}\sqrt{n}})$ approximation ratio, with n the size of the input, then there exists a polynomial-time algorithm for* RBSC *with an $\mathcal{O}(2^{log^{1-\delta}n})$ approximation ratio, with n the size of the input.*

Proof. A polynomial-time algorithm for RBSC with an $\mathcal{O}(2^{\log^{1-\delta}n})$ approximation ratio is achieved by composing the AFP-reduction from the proof of Lemma 3.4 and the polynomial-time algorithm for CMC.

It remains to discuss the approximation rate. First, we discuss the size of the obtained CMC instance, which is $|I_{\text{CMC}}| = |(G, \ell, d, \alpha, \beta, \gamma)| = \mathcal{O}(|\mathcal{S}||V| + \log(|\mathcal{S}|) + \log(|\mathcal{S}|) + 1 + 1 + \log(|R|)) = \mathcal{O}(|\mathcal{S}||V|)$, where $|G| = \mathcal{O}(|V| + |E| + |\lambda|) = \mathcal{O}(|V| + |\lambda|)$, because it is a sparse graph by construction and $|\lambda| = \mathcal{O}(|\mathcal{S}||V|)$. So, $\mathcal{O}(|\mathcal{S}||V|) = \mathcal{O}(|\mathcal{S}||U|^2)$, because of the size of O.

The size of the RBSC instance is $n = |I_{\text{RBSC}}| = |(U, B, R, \mathcal{S})| = \mathcal{O}(|U| + |U| + |U| + |\mathcal{S}||U|) = \mathcal{O}(|\mathcal{S}||U|)$.

Thus, the size of the CMC-instance created by the reduction is $\mathcal{O}(|\mathcal{S}||U|^2) = \mathcal{O}(n^2)$.

Since $\mathrm{OPT}_{\mathrm{CMC}}(I_{\mathrm{CMC}}) = \mathrm{OPT}_{\mathrm{RBSC}}(I_{\mathrm{RBSC}})$ and $\mathrm{obj}_{\mathrm{RBSC}}(s_{\mathrm{RBSC}}) \leq \mathrm{obj}_{\mathrm{CMC}}(s_{\mathrm{CMC}})$ (as shown by the proof of Lemma 3.4), it follows that our composed algorithm has an approximation rate of $\mathcal{O}(2^{\log((\sqrt{n})^2)^{1-\delta}}) = \mathcal{O}(2^{\log(n)^{1-\delta}})$. Take a solution s_{CMC} to an instance I_{CMC} from the theorized polynomial-time algorithm for CMC. The algorithm has a $\mathcal{O}(2^{\log^{1-\delta} n})$ approximation ratio which means

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{RBSC}}(s_{\mathrm{RBSC}}) \\
&\quad\leq \mathrm{obj}_{\mathrm{CMC}}(s_{\mathrm{CMC}}) \\
&\quad\leq \mathcal{O}(2^{\log^{1-\delta}\sqrt{n_{\mathrm{CMC}}}})\ \mathrm{OPT}_{\mathrm{CMC}}(I_{\mathrm{CMC}}) \\
&\quad= \mathcal{O}(2^{\log^{1-\delta}\sqrt{n_{\mathrm{CMC}}}})\ \mathrm{OPT}_{\mathrm{RBSC}}(I_{\mathrm{RBSC}}) \\
&\quad= \mathcal{O}(2^{\log^{1-\delta}\sqrt{(n_{\mathrm{RBSC}})^2}})\ \mathrm{OPT}_{\mathrm{RBSC}}(I_{\mathrm{RBSC}}) \\
&\quad= \mathcal{O}(2^{\log^{1-\delta} n_{\mathrm{RBSC}}})\ \mathrm{OPT}_{\mathrm{RBSC}}(I_{\mathrm{RBSC}}).
\end{aligned}$$

Hence, the lemma follows.

□

RBSC was shown to be $\Omega(2^{\log^{1-\epsilon} n})$-inapproximable[1] by Dinur et al. [DS04], Carr et al. [CDKM00], and Elkin et al. [EP07]. Dinur et al. [DS04] prove this inapproximability under the weakest assumption (P $\neq$ NP). Specifically, they proved unless P $=$ NP, it is impossible to achieve a polynomial-time algorithm for RBSC with an $\mathcal{O}(2^{\log^{1-\delta} n})$ approximation ratio, with $\delta \;=\; 1 \;/\; \log\,\log^c n$, for any constant $c < 1/2$, with n the size of the input. So, since we know a polynomial-time algorithm for RBSC with an $\mathcal{O}(2^{\log^{1-\delta} n})$ approximation ratio is impossible unless P $=$ NP, this means a polynomial-time algorithm with an $\mathcal{O}(2^{\log^{1-\delta}\sqrt{n}})$ approximation ratio is impossible for CMC unless P $=$ NP, which proves Theorem 3.3.

3.3.2 Upper bounds

In this section, we show that we can use existing approximation algorithms to solve CMC. Specifically, we show that CMC is polynomial time reducible to the weighted version of the Red-Blue Set Cover problem using AFP-reductions. This allows to transfer known approximation algorithms (such as the one given by Peleg [Pel07]) from WRBSC to CMC, while preserving any constant factor approximations. In section 3.3.1, we have shown that such algorithms are unlikely to exist, but in Section 3.4.3 we will show that we can still use this AFP-reduction to obtain an approximation algorithm with a non-constant approximation guarantee.

[1] $f(n) \in \Omega(g(n)) \iff \exists k > 0, n_0 : \forall n > n_0 : g(n)k \leq f(n)$

We start by introducing WRBSC which is the weighted version of RBSC (defined in Section 3.3.1), in which the red elements have a nonnegative cost assigned to them. By choosing elements from $\mathcal{S}$, the objective is to select all the blue elements (to which no cost is assigned), while incurring as little cost by selecting red elements as possible.

Formally, we define the *Weighted Red-Blue Set Cover* (WRBSC) problem [CDKM00] as follows:

- **Input:** Finite sets U, B, R, $\mathcal{S}$ with $U = B \cup R$ and $B \cap R = \emptyset$, $\mathcal{S} \subseteq 2^U$, with $\bigcup\limits_{S \in \mathcal{S}} S = U$, and a function $c : R \to \mathbb{Q}^+$.

- **Output:** a set $\mathcal{S}^* \subseteq \mathcal{S}$ such that $B \subseteq \bigcup\limits_{S \in \mathcal{S}^*} S$ and that minimizes the cost function

$$cost_{\text{WRBSC}}(\mathcal{S}^*, U, B, R, c) := \sum_{r \in (\bigcup\limits_{S \in \mathcal{S}^*} S \cap R)} c(r).$$

By restricting the length of motifs to be at most logarithmic in the size of the network, we can reduce CMC to WRBSC using approximation factor preserving reductions.

Theorem 3.12. *If $\ell \leq log(|G|)$,* CMC $\leq_{AFP}$ WRBSC.

The rest of this section is dedicated to proving this theorem.

First, we reformulate CMC on a more abstract level which facilitates the reduction to WRBSC. Let U be a finite set of elements divided into two disjoint subsets $U = C \cup I$. Here, C contains the correct elements while I contains the incorrect ones. Let $\mathcal{S}$ be a set of subsets of U. The relationship with CMC is as follows: Given a PPI-network $G = (V_G, E_G, \lambda_G)$ and numbers ℓ and d, U corresponds to the set of all protein pairs, C corresponds to the edges in G, while I are the non-edges (anti-edges). Finally, $\mathcal{S}$ contains precisely those sets of protein pairs defined by a (ℓ, d)-motif pair. That is,

$$\mathcal{S} = \bigcup_{\{X,Y\}} \{V_X \times V_Y\}.$$

We will show that if we restrict ℓ to its typical small values, CMC reduces to the following problem which we refer to as the MDL-cover (MDL-COVER) problem:

- **Input:** Finite sets U, C, I with $U = C \cup I$, $C \cap I = \emptyset$, $\mathcal{S} \subseteq 2^U$ and nonnegative numbers α, β, and γ
- **Output:** a set $\mathcal{S}^* \subseteq \mathcal{S}$ that minimizes the cost function

$$\alpha|\mathcal{S}^*| + \beta|I \cap \bigcup_{S \in \mathcal{S}^*} S| + \gamma|C \setminus \bigcup_{S \in \mathcal{S}^*} S|.$$

Lemma 3.13. *If* $\ell \leq$ *log*$(|G|)$, CMC $\leq_{AFP}$ MDL-COVER.

Proof. Let $I_{\text{CMC}} = (G, l, d, \alpha, \beta, \gamma)$ be an instance of CMC. Let $G = (V, E, \lambda)$. We start by defining the function f mapping I_{CMC} to $f(I_{\text{CMC}}) = I_{\text{MDL-COVER}} = (U, C, I, \mathcal{S}, \alpha, \beta, \gamma)$. Here, α, β, and γ remain the same. Let $C = E$, and $I = \{e \in (V \times V) \mid e \notin E\}$, then $U = C \cup I$. Lastly, define $\mathcal{S} = \bigcup_{\{X,Y\}} V_X \times V_Y$.

Example 3.14. In Figure 3.3, we depict a possible instance I_{CMC} for CMC. In Figure 3.4, we show the corresponding $f(I_{\text{CMC}}) = I_{\text{MDL-COVER}}$. For ease of exposition, we use an alphabet $\Sigma = \{A, B\}$, $\ell = 1$ and $d = 0$. In that case, the sets in Figure 3.4 correspond to the sets of protein pairs selected by all possible motif pairs.

◁

We argue that f is PTIME-computable. To construct $\mathcal{S}$, we need to generate all possible motif pairs. The amount of possible (ℓ, d)-motifs is given by the following equation

$$\binom{\ell - 1}{d}|\Sigma|^{(\ell - d)} \leq \sum_{i=0}^{\ell-1} \binom{\ell - 1}{i}|\Sigma|^{\ell} = 2^{\ell-1}|\Sigma|^{\ell}.$$

If $\ell \leq \log(|G|)$, then this amount is polynomial in $|G|$. Since the number of motif pairs is polynomial in the number of motifs, we can enumerate all motif pairs in time polynomial in the size of G. For each motif pair $\{X, Y\}$, we then need to find $\{V_X \times V_Y\}$. This can be done in polynomial time, by passing over all the sequences.

Let $s_{\text{MDL-COVER}} = \mathcal{S}$. We next define the function g mapping $s_{\text{MDL-COVER}}$ to $g(s_{\text{MDL-COVER}}) = s_{\text{CMC}} = (\mathcal{M})$. By construction, for any set $S_i \in \mathcal{S}$, there is at least one motif pair $M_i = \{X, Y\}$ with $V_X \times V_Y = S_i$. Therefore, define

$$\mathcal{M} = \bigcup_{S_i \in \mathcal{S}} M_i.$$

By the same reasoning given above, it can be argued that g is PTIME-computable.

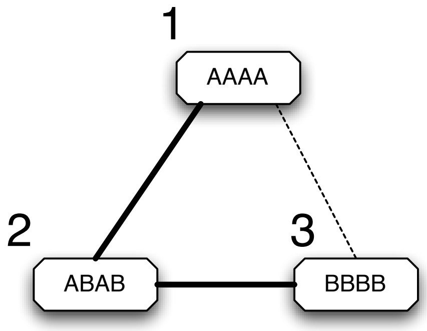


Figure 3.3: An example instance I_{CMC} for CMC. The anti-edge is indicated by a dotted line.

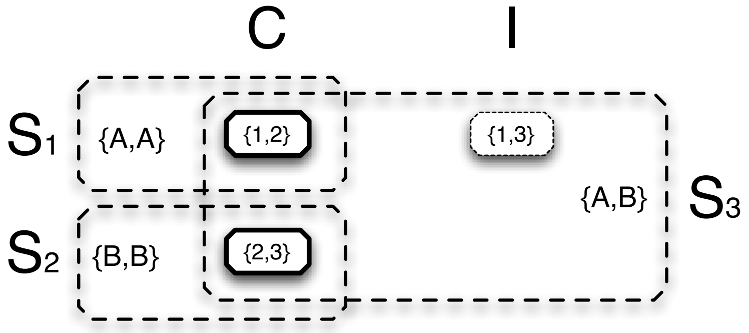


Figure 3.4: $f(I_{\text{CMC}}) = I_{\text{MDL-COVER}}$. Elements correspond to edges (thick lines) or anti-edges (dotted lines) in Figure 3.3.

Example 3.15. Let $\mathcal{S} = \{S_1, S_3\}$ be a solution to $I_{\text{MDL-COVER}}$ in Figure 3.4, we can find the corresponding set of motif pairs $\{\{A,A\},\{A,B\}\}$ by iterating over all possible motif pairs.

◁

We next argue that $\text{obj}_{\text{CMC}}(s_{\text{CMC}}) \leq \text{obj}_{\text{MDL-COVER}}(s_{\text{MDL-COVER}})$. By construction, the sets in $s_{\text{MDL-COVER}}$ contain the same correct and incorrect elements that are selected by the motif pairs in s_{CMC}. That is, $E_G = C$ and $V \times V \setminus E_G = I$. Further, $E_{\mathcal{M}} = \bigcup_{S \in \mathcal{S}^*} S$. It is however possible that multiple sets contain the same elements and thus get represented by the same motif pair, i.e., $|\mathcal{M}| \leq |\mathcal{S}^*|$. So,

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(s_{\mathrm{CMC}}) \\
&\quad = \alpha|\mathcal{M}| + \beta|E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad \leq \alpha|\mathcal{S}^*| + \beta|E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad = \alpha|\mathcal{S}^*| + \beta|\bigcup_{S\in\mathcal{S}^*} S \setminus C| + \gamma|C \setminus \bigcup_{S\in\mathcal{S}^*} S| \\
&\quad = \alpha|\mathcal{S}^*| + \beta|I \cap \bigcup_{S\in\mathcal{S}^*} S| + \gamma|C \setminus \bigcup_{S\in\mathcal{S}^*} S| \\
&\quad = \mathrm{obj}_{\mathrm{MDL\text{-}COVER}}(s_{\mathrm{MDL\text{-}COVER}}).
\end{aligned}$$

We next show that $\mathrm{OPT}_{\mathrm{MDL\text{-}COVER}}(I_{\mathrm{MDL\text{-}COVER}}) \leq \mathrm{OPT}_{\mathrm{CMC}}(I_{\mathrm{CMC}})$.

We first argue that if an optimal set of motif pairs $\mathcal{M}$ contains two motif pairs which select the same node pairs, we can remove one of them without removing optimality. We refer to such motif pairs as duplicates. Indeed, assume M to be a motif pair in $\mathcal{M}$ that selects the same node pairs as another motif pair in $\mathcal{M}$. Then,

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(\mathcal{M} \setminus \{M\}) \\
&\quad = \alpha|\mathcal{M} \setminus \{M\}| + \beta|E_{\mathcal{M}\setminus\{M\}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}\setminus\{M\}}| \\
&\quad = \alpha(|\mathcal{M}| - 1) + \beta|E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad \leq \alpha|\mathcal{M}| + \beta|E_{\mathcal{M}} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}}| \\
&\quad = \mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}).
\end{aligned}$$

So, for an optimal solution $\mathcal{M}$, we can create a solution $\mathcal{M}'$ which contains no duplicates and is still optimal.

Example 3.16. In the example in Figure 3.3, if we were to change the motifs to $\ell = 2, d = 1$, we could have a solution $\mathcal{M} = \{\{\mathtt{Ax}, \mathtt{Bx}\}, \{\mathtt{xA}, \mathtt{Bx}\}\}$ with cost $2\alpha + \beta$. If this solution is optimal, the solution $\mathcal{M}' = \{\{\mathtt{Ax}, \mathtt{Bx}\}\}$ is also optimal, with cost $\alpha + \beta$. Note that duplicates are only possible in an optimal solution if $\alpha = 0$.

$\triangleleft$

For convenience, we now construct the function g^{-1} that maps $\mathcal{M}'$ to $g^{-1}(\mathcal{M}') = \mathcal{S}^*$, as

$$g^{-1}(\mathcal{M}') = \mathcal{S}^* = \bigcup_{M_i \in \mathcal{M}^*} S_i.$$

Since $|\mathcal{M}'| = |\mathcal{S}^*|$,

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}') \\
&= \alpha|\mathcal{M}'| + \beta|E_{\mathcal{M}'} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}'}| \\
&= \alpha|\mathcal{S}^*| + \beta|E_{\mathcal{M}'} \setminus E_G| + \gamma|E_G \setminus E_{\mathcal{M}'}| \\
&= \alpha|\mathcal{S}^*| + \beta|\bigcup_{S\in\mathcal{S}^*} S \setminus C| + \gamma|C \setminus \bigcup_{S\in\mathcal{S}^*} S| \\
&= \alpha|\mathcal{S}^*| + \beta|I \cap \bigcup_{S\in\mathcal{S}^*} S| + \gamma|C \setminus \bigcup_{S\in\mathcal{S}^*} S| \\
&= \mathrm{obj}_{\text{MDL-COVER}}(\mathcal{S}^*).
\end{aligned}$$

This solution $\mathcal{S}^*$ to $I_{\text{MDL-COVER}}$ is also optimal. Assume towards a contradiction that there exists $\mathcal{S}_2^*$ with $\mathrm{obj}_{\text{MDL-COVER}}(\mathcal{S}_2^*) < \mathrm{obj}_{\text{MDL-COVER}}(\mathcal{S}^*)$. We have already shown $\mathrm{obj}_{\mathrm{CMC}}(g(s_{\text{MDL-COVER}})) \leq \mathrm{obj}_{\text{MDL-COVER}}(s_{\text{MDL-COVER}})$, so

$$\begin{aligned}
&\mathrm{obj}_{\mathrm{CMC}}(g(\mathcal{S}_2^*)) \\
&\leq \mathrm{obj}_{\text{MDL-COVER}}(\mathcal{S}_2^*) \\
&< \mathrm{obj}_{\text{MDL-COVER}}(\mathcal{S}^*) \\
&= \mathrm{obj}_{\mathrm{CMC}}(\mathcal{M}'),
\end{aligned}$$

which is in contradiction with $\mathcal{M}'$, and thus $\mathcal{M}$, being optimal. Therefore, $\mathrm{OPT}_{\text{MDL-COVER}}(I_{\text{MDL-COVER}}) \leq \mathrm{OPT}_{\mathrm{CMC}}(I_{\mathrm{CMC}})$. The lemma now follows.

□

Lemma 3.17. MDL-COVER $\leq_{AFP}$ WRBSC.

Proof. Let $I_{\text{MDL-COVER}} = (U_{\text{MDL-COVER}}, C, I, \mathcal{S}_{\text{MDL-COVER}}, \alpha, \beta, \gamma)$ be an instance of MDL-COVER. We start by defining the function f mapping $I_{\text{MDL-COVER}}$ to $f(I_{\text{MDL-COVER}}) = I_{\mathrm{WRBSC}} = (U_{\mathrm{WRBSC}}, B, R, \mathcal{S}_{\mathrm{WRBSC}}, c)$. Here, $B = C$, $R = I \cup \{r_T \mid T \in \mathcal{S}_{\text{MDL-COVER}}\} \cup \{r_b \mid b \in C\}$, where all r_t and r_b are new objects, and $U_{\mathrm{WRBSC}} = B \cup R$. Define

$$c(r) = \begin{cases} \alpha & r \in \{r_T \mid T \in \mathcal{S}_{\text{MDL-COVER}}\}; \\ \gamma & r \in \{r_b \mid b \in C\}; \text{and,} \\ \beta & r \in I. \end{cases}$$

For every $T \in \mathcal{S}_{\text{MDL-COVER}}$, we define $S_T = \{r_T\} \cup T$. For every $b \in B$, we define $S_b = \{r_b\} \cup \{b\}$. Finally,

$$\mathcal{S}_{\mathrm{WRBSC}} = \left(\bigcup_{T\in\mathcal{S}_{\text{MDL-COVER}}} S_T \right) \cup \bigcup_{b\in B} S_b. \qquad (\star)$$

Clearly, f is PTIME-computable.

Example 3.18. In Figure 3.5, we depict a possible instance $I_{\text{MDL-COVER}}$ for MDL-COVER. In Figure 3.6, we see the corresponding $f(I_{\text{MDL-COVER}}) = I_{\text{WRBSC}}$. Elements from I have gotten weight β. A red element with weight α has been added to every set. And new sets have been created for every blue element, containing it and a new red element with weight γ.

◁

We next define the function g mapping s_{WRBSC} to $g(s_{\text{WRBSC}}) = s_{\text{MDL-COVER}}$. Simply set

$$s_{\text{MDL-COVER}} = \{T \in \mathcal{S}_{\text{MDL-COVER}} \mid S_T \in s_{\text{WRBSC}}\}. \qquad (\star\star)$$

Clearly, g is PTIME-computable.

We next argue that $\text{obj}_{\text{MDL-COVER}}(s_{\text{MDL-COVER}}) \leq \text{obj}_{\text{WRBSC}}(s_{\text{WRBSC}})$. Indeed,

$$\begin{aligned}
&\text{obj}_{\text{WRBSC}}(s_{\text{WRBSC}}) \\
&= \sum\{c(r) \mid r \in (\bigcup_{S \in s_{\text{WRBSC}}} S \cap R)\} \\
&= \sum\{c(r) \mid T \in s_{\text{MDL-COVER}}, S_T \in s_{\text{WRBSC}}, r \in (S_T \cap R)\} \\
&\qquad + \sum\{c(r) \mid b \in C, S_b \in s_{\text{WRBSC}}, r \in (S_b \cap R)\} \\
&= \sum\{c(r) \mid T \in s_{\text{MDL-COVER}}, r \in ((T \cup \{r_T\}) \cap R)\} \\
&\qquad + \sum\{c(r) \mid b \in C, S_b \in s_{\text{WRBSC}}, r \in (\{r_b\} \cap R)\} \\
&= \sum\{c(r) \mid T \in s_{\text{MDL-COVER}}, r \in (\{r_T\} \cap R)\} \\
&\qquad + \sum\{c(r) \mid T \in s_{\text{MDL-COVER}}, r \in (T \cap R)\} \\
&\qquad + \sum\{c(r) \mid b \in C, S_b \in s_{\text{WRBSC}}, r \in (\{r_b\} \cap R)\} \\
&= \alpha|s_{\text{MDL-COVER}}| + \beta|\bigcup_{T \in s_{\text{MDL-COVER}}} T \cap I| + \gamma|\bigcup_{S_b \in s_{\text{WRBSC}}} \{r_b\}| \\
&\geq \alpha|s_{\text{MDL-COVER}}| + \beta|\bigcup_{T \in s_{\text{MDL-COVER}}} T \cap I| + \gamma|C \setminus \bigcup_{S \in s_{\text{MDL-COVER}}} S| \qquad (\dagger) \\
&= \text{obj}_{\text{MDL-COVER}}(s_{\text{MDL-COVER}}).
\end{aligned}$$

Here, (†) follows as a set S_b can be selected in s_{WRBSC}, while another set already selects b.

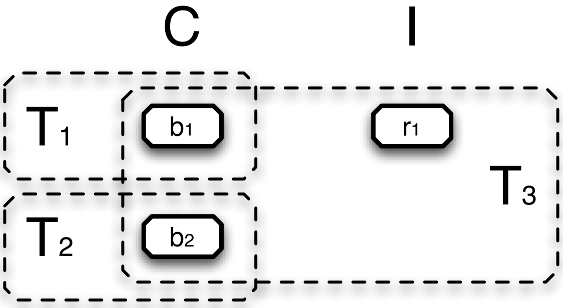


Figure 3.5: An example instance $I_{\text{MDL-COVER}}$ for MDL-COVER.

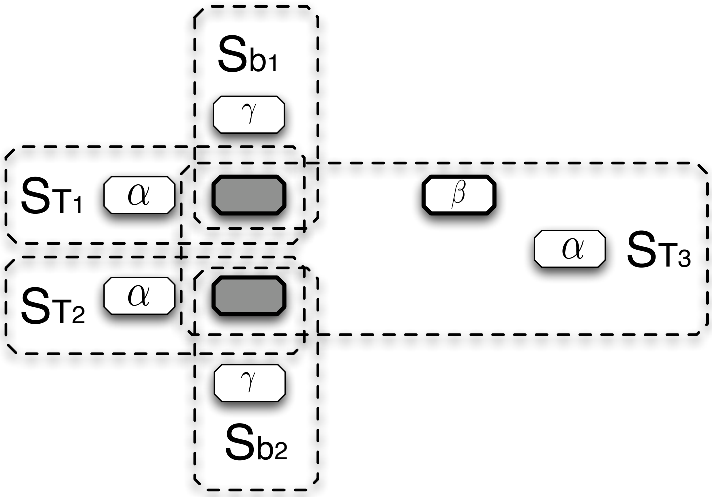


Figure 3.6: $f(I_{\text{MDL-COVER}}) = I_{\text{WRBSC}}$. Blue elements shaded in gray, red elements have their weight indicated. Newly created elements are indicated with thinner lines.

Example 3.19. In Figure 3.6, $\mathcal{S} = \{S_{T_3}, S_{B_1}\}$ is a solution with cost $\alpha+\beta+\gamma$. Then, $g(\mathcal{S}) = \{T_3\}$ with cost $\alpha + \beta$. In this case, the cost is reduced, due to the fact that b_1 was contained in S_{T_3}.

◁

We next argue that $\mathrm{OPT}_{\textsc{wrbsc}}(I_{\textsc{wrbsc}}) \leq \mathrm{OPT}_{\textsc{mdl-cover}}(I_{\textsc{mdl-cover}})$.

We first argue that if an optimal solution $\mathcal{S}^*_{\textsc{wrbsc}}$ for $I_{\textsc{wrbsc}}$ contains a set S_b and a set S_T with $b \in S_T$, we can remove S_b without removing optimality. Indeed, assume S_b to be in $\mathcal{S}^*_{\textsc{wrbsc}}$ with another set in $\mathcal{S}^*_{\textsc{wrbsc}}$ containing b, then

$$\begin{aligned}
&\mathrm{obj}_{\textsc{wrbsc}}(\mathcal{S}^*_{\textsc{wrbsc}} \setminus S_b) \\
&= \sum_{r\in\left(\left(\bigcup_{S\in\mathcal{S}^*_{\textsc{wrbsc}}\setminus S_b} S\right)\cap R\right)} c(r) \\
&= \sum_{r\in\left(\left(\bigcup_{S\in\mathcal{S}^*_{\textsc{wrbsc}}} S\right)\cap R\right)} c(r) \qquad - \gamma \\
&\leq \sum_{r\in\left(\left(\bigcup_{S\in\mathcal{S}^*_{\textsc{wrbsc}}} S\right)\cap R\right)} c(r) \\
&= \mathrm{obj}_{\textsc{wrbsc}}(\mathcal{S}^*_{\textsc{wrbsc}}).
\end{aligned}$$

So, for an optimal solution $\mathcal{S}^*_{\textsc{wrbsc}}$, we can create a solution ${\mathcal{S}^*_{\textsc{wrbsc}}}'$ that contains no S_b with $b \in S_T \in {\mathcal{S}^*_{\textsc{wrbsc}}}'$ and is still optimal.

For an instance $I_{\textsc{wrbsc}}$, we can use g to transform an optimal solution $\mathcal{S}^*_{\textsc{wrbsc}}$ into a solution $s_{\textsc{mdl-cover}}$ for $I_{\textsc{mdl-cover}}$ with $\mathrm{OPT}_{\textsc{wrbsc}}(I_{\textsc{wrbsc}}) = \mathrm{obj}_{\textsc{wrbsc}}(\mathcal{S}^*_{\textsc{wrbsc}}) = \mathrm{obj}_{\textsc{mdl-cover}}(s_{\textsc{mdl-cover}})$. The solution $s_{\textsc{mdl-cover}}$ is also an optimal solution for $I_{\textsc{mdl-cover}}$, because if there was a better solution, we could transform it into one for $I_{\textsc{wrbsc}}$ with the same score, by taking S_T for every T and S_b, for every element of C not covered. This would be in contradiction with $\mathcal{S}^*_{\textsc{wrbsc}}$ being optimal. Therefore, $\mathrm{OPT}_{\textsc{wrbsc}}(I_{\textsc{wrbsc}}) = \mathrm{OPT}_{\textsc{mdl-cover}}(I_{\textsc{mdl-cover}})$, which means that definitely $\mathrm{OPT}_{\textsc{wrbsc}}(I_{\textsc{wrbsc}}) \leq \mathrm{OPT}_{\textsc{mdl-cover}}(I_{\textsc{mdl-cover}})$.

The lemma now follows. □

Theorem 3.12 now follows from Lemma 3.13 and Lemma 3.17 and the fact that AFP-reductions are closed under composition [Vaz04].

3.4 Algorithms

In the next three subsections, we present three algorithms for CMC: SEQ-SLIDER, CMC-greedy, and CMC-approx.

3.4.1 SEQ-SLIDER

We use the SEQ-SLIDER algorithm (see Section 2.5.2) as a baseline. Using local search, the latter returns the set of the k best motif pairs (according to its χ^2-score) it can find, which is then interpreted as a cover.

3.4.2 CMC-greedy

We devise a naive greedy algorithm, called CMC-greedy, for CMC which repeatedly looks for a motif pair to add to $\mathcal{M}$ which reduces $cost_{\alpha,\beta,\gamma}(\mathcal{M}, G)$ the most. Since the space of possible motif pairs is far too large to consider every motif pair (see Section 2.2), we replace this step with a local search approach, similar to that of SEQ-SLIDER, that looks for the best motif pair to add to $\mathcal{M}$ during a set time period. The algorithm keeps adding motif pairs until it no longer finds one that reduces $cost_{\alpha,\beta,\gamma}(\mathcal{M}, G)$. The full algorithm is shown in Algorithm 2; the local search step is shown in Algorithm 3, with neighbors the neighbor function of SEQ-SLIDER. When used by CMC-greedy $cost_f(\mathcal{M}, M, G) = cost_{\alpha,\beta,\gamma}(\mathcal{M}, M, G)$, which is simply defined as $cost_{\alpha,\beta,\gamma}(\mathcal{M} \cup \{M\}, G)$.

3.4.3 CMC-approx

By Theorem 3.12, we can apply to CMC any constant factor approximation algorithm for WRBSC while preserving the approximation factor. However, no such algorithms are known, and by Theorem 3.3 are very unlikely to exist. We can however also apply the reduction to algorithms with known (but non-constant) approximation ratios and then deduce the approximation ratio of this algorithm for CMC. Therefore, we first give an overview of such algorithms.

WRBSC admits naive approximation algorithms with ratios $|B|$, $|R|$ or $|S|\log(|B|)$ [Pel07]. Carr et al. [CDKM00] suggest an algorithm that gives an approximation ratio of $2\sqrt{K_B|S|}$ or $\mathcal{O}(|S|^{1-1/K_R}\log(|S|))$ (with K_B (K_R) the maximum amount of blue (red) elements in a single set). These ratios are low when K_B or K_R are small, but may go up to $\Omega(\sqrt{|S||B|})$ or $\Omega(|S|\log(|S|))$. Peleg [Pel07] suggests a greedy algorithm with approximation factor $\Delta(S)\log(|B|)$ (with $\Delta(S)$ the maximum amount of sets that contain the same red element) and another algorithm with factor $2\sqrt{|S|\log(|B|)}$.

Algorithm 2 The algorithm CMC-greedy.

```
Input: PPI-network $G = (V, E, \lambda)$, $\ell, d \in \mathbb{N}^+$, $d < \ell$, and $\alpha$, $\beta$, $\gamma \in \mathbb{Q}^+$,
    TIME $\in \mathbb{N}^+$
Output: best cover $\mathcal{M}$ according to $cost_{\alpha,\beta,\gamma}$ found in $G$
$\mathcal{M} \leftarrow \emptyset$
cost $\leftarrow cost_{\alpha,\beta,\gamma}(\mathcal{M}, G)$
oldCost $\leftarrow \infty$
while cost $<$ oldCost do
    oldCost $\leftarrow$ cost
    bestMotifPair = undef
    startTime = getTime()
    while getTime() $<$ startTime + TIME do
        $M \leftarrow$ localSearch$(G, \ell, d, \mathcal{M}, cost_{\alpha,\beta,\gamma})$
        if $cost_{\alpha,\beta,\gamma}(\mathcal{M} \cup \{M\}, G) <$ cost then
            bestMotifPair $\leftarrow M$
            cost $\leftarrow cost_{\alpha,\beta,\gamma}(\mathcal{M} \cup \{M\}, G)$
    if cost $<$ oldCost then
        $\mathcal{M} \leftarrow \mathcal{M} \cup \{$bestMotifPair$\}$
```

Algorithm 3 The local search algorithm for CMC, with $cost_f$ as the cost function.

```
Input: PPI-network $G = (V, E, \lambda)$, $\ell, d \in \mathbb{N}$, $d < \ell$, set of motif pairs $\mathcal{M}$, a
    cost function $cost_f$
Output: best correlated motif pair $M^*$ to add to $\mathcal{M}$ according to $cost_f$ found
    in $G$
$M^* \leftarrow$ randomMotifPair()
cost $\leftarrow cost_f(\mathcal{M}, M^*, G)$
oldCost $\leftarrow \infty$
while cost $<$ oldCost do
    oldCost $\leftarrow$ cost
    $M \leftarrow M^*$
    for all $M' \in$ neighbors$(M)$ do
        if $cost_f(\mathcal{M}, M', G) <$ cost then
            $M^* \leftarrow M'$
            cost $\leftarrow cost_f(\mathcal{M}, M', G)$
```

Theoretically, the naive approximation algorithm with ratio $|B|$ is the most appealing as B in our setting corresponds to the number of edges and is far smaller than the number of possible motif pairs $|S|$. The naive algorithm selects for each blue element a set containing it with the least cost worth of red elements. For a blue element b, there is the option to take a set S_T or the set S_b (cf. equation ($\star$) in the proof of Lemma 3.17). Here, S_T has a cost of at least α and corresponds to a motif pair, while the set S_b contains b, has associated cost γ and does not correspond to a motif pair. As in our setting $\alpha > \gamma$, the naive algorithm will always select the sets S_b which results in returning the empty set of motif pairs as a solution for CMC (cf. equation ($\star\star$) in the proof of Lemma 3.17). So, although this naive algorithm has the best theoretical approximation guarantee, it can not be meaningfully applied towards CMC.

Therefore, we turn to Peleg's algorithm with approximation factor $2\sqrt{|S|\log(|B|)}$ [Pel07] which by composing the reductions in the proofs of Lemmas 3.13 and 3.17 results in a theoretical algorithm for CMC with an approximation factor of $2\sqrt{(\mathrm{MP}(\ell, d) + |E|)\log(|E|)}$. We next explain how the algorithm works. Peleg's approximation algorithm [Pel07], transforms an instance of WRBSC into an instance of the Weighted Set Cover (WSC) problem by assigning to each set the cost of the red elements in it and then removing the red elements from each set. He applies the greedy algorithm for WSC, which is known to have an approximation ratio of $\log(|B|)$ [Chv79], to this instance. The greedy algorithm works by selecting the most cost-efficient set at each time point. He proves this algorithm has approximation guarantee $2\sqrt{|S|\log(|B|)}$, when applied to an instance of WRBSC adapted from the original instance using Z, the maximum cost of red elements in a single set in the optimal solution to that instance. As Z is unknown, the algorithm is run repeatedly with every possible value and the best result is saved. The adaptation of the original instance starts by removing any set with total cost greater than Z. Then it removes from the remaining set all red elements still occurring in more than $\sqrt{|\mathcal{S}|/\log(|B|)}$ sets. Unfortunately, the latter algorithm is infeasible as it iterates over all elements in $\mathcal{S}$ corresponding to all possible motif pairs.

Therefore, we consider an adaption, called CMC-approx, which is shown in Algorithm 4. Rather than iterating over all possible motif pairs, CMC-approx employs local search to find the most cost-efficient motif pair to be added. Specifically, CMC-approx iterates over the possible values for Z, the maximum amount of cost for a single motif pair, and then uses local search utilizing a cost function $cost_{\text{WSC},\alpha,\beta,\gamma,Z}$. Here, $cost_{\text{WSC},\alpha,\beta,\gamma,Z}(\mathcal{M}, M, G)$ calculates the cost (as the inverse of cost efficiency) of adding M to $\mathcal{M}$ as follows:

$$cost_{\text{WSC},\alpha,\beta,\gamma,Z}(\mathcal{M}, M, G) :=$$
$$\begin{cases} \dfrac{\alpha + \beta|\hat{A}_M|}{|E_M \setminus E_{\mathcal{M}}|} & \text{if } \alpha + \beta|A_M| <= Z; \\ \infty & \text{otherwise;} \end{cases}$$

where $\hat{A}_M$ is the set of anti-edges A_M, but with the anti-edges selected by $\sqrt{(|\text{MP}(\ell, d)| + |E|)/\log(|E|)}$ or more motif pairs removed. We cannot determine the amount of motif pairs that select an anti-edge without knowledge of all the sets, but we can compute an upper bound. Indeed, for every protein p, we compute the number N_p of motifs selecting p (note that the number of motifs is significantly lower than the number of motif pairs). The number of motif pairs selecting an anti-edge $\{p, q\}$ is then bounded above by $N_p \times N_q$.

Algorithm 4 The adapted approximation algorithm for WRBSC applied to CMM.

```
Input: PPI-network $G = (V, E, \lambda)$, $\ell, d \in \mathbb{N}^+$, $d < \ell$, and $\alpha$, $\beta$, $\gamma \in \mathbb{Q}^+$,
    TIME $\in \mathbb{N}^+$
Output: best cover $\mathcal{M}$ according to $cost_{\alpha,\beta,\gamma}$ found in $G$
$\mathcal{M} = \emptyset$
for $Z = 1 \rightarrow \alpha + \beta|V \times V|$ do
    $\mathcal{M}' \leftarrow \emptyset$
    while $E \not\subset E_{\mathcal{M}'}$ do
        bestMotifPair = undef
        cost $\leftarrow \infty$
        startTime = getTime()
        while getTime() < startTime + TIME do
            $M \leftarrow$ localSearch($G, \ell, d, \mathcal{M}', cost_{\text{WSC},\alpha,\beta,\gamma,Z}$)
            if $cost_{\text{WSC},\alpha,\beta,\gamma,Z}(\mathcal{M}', M, G)$ < cost then
                bestMotifPair $\leftarrow M$
                cost $\leftarrow cost_{\text{WSC},\alpha,\beta,\gamma,Z}(\mathcal{M}, M, G)$
        $\mathcal{M}' \leftarrow \mathcal{M}' \cup \{M\}$
    if $cost_{\alpha,\beta,\gamma}(\mathcal{M}', G) < cost_{\alpha,\beta,\gamma}(\mathcal{M}, G)$ then
        $\mathcal{M} \leftarrow \mathcal{M}'$
```

3.5 Data

We ran our methods on the high-confidence PPI-network of yeast [CKX+07] consisting of 1 620 proteins and 9 060 interactions, to determine their performance and to check the coverage of the network obtained by the results.

Yu et al. [YGN$^+$10] present positive and negative example datasets created by balanced sampling for testing protein-protein interaction prediction. We took their files for yeast containing 4 972 physical interactions and 4 972 non-interactions and created from those a network with all the proteins contained in these files and all the physical interactions present. We ran our methods on cross-validation networks created from this network and used the resulting covers to predict protein-protein interactions.

Protein-protein interaction prediction is used in two different ways:

- Predicting interaction/non-interaction between proteins for which no previous observation exists at all, and
- predicting interaction/non-interaction for proteins for which some knowledge of their interactions has been obtained already.

Both scenarios are relevant in biology and are tested here.

For each of these setups, we made ten training and test sets for cross-validation purposes:

- In the first case, the focus lies on separating test-proteins from training-proteins. We divide the list of all proteins into ten sets randomly. For each of these sets, we took the original (non-)interaction-data set and put each (non-)interaction where one or more proteins are contained in the set, into the test set. What remains became the training set.
- In the second case, the focus lies on separating test-(non-)interactions from training-(non-)interactions. The positive and negative data sets were divided randomly into ten parts. Each part became a test set, with the other (non-)interactions forming the training set.

For every training set, a network was created with all the proteins remaining in the positive and negative training set and as interactions those in the positive training set. Practically, our algorithms did not have substantially different results depending on the case. Therefore, we treat this as a homogeneous set of interaction networks for prediction purposes.

To compare our results to known 3D-structures, to see if our methods locate actual binding sites, we took the high-confidence networks for human and yeast presented by Yu et al. [YGN$^+$10] and attempted to map the protein sequences (retrieved from uniprot.org and yeastgenome.org respectively) to structures in the PDB database. To perform the mapping of the protein sequences to PDB structures, we used a strategy similar to the one used by Yabuki at al. [YMSS04], where each protein sequence in the input dataset is searched against the PDB database by gapped BLAST. The protein structure was mapped to the protein sequence if the aligned regions present identity, coverage of aligned region on query sequences, coverage of aligned region on PDB sequences and bit score higher than 40%, 30%, 30% and 70, respectively. By doing so, it is guaranteed that at least 30% of the length of the protein sequence is mapped to the PDB structure, and the PDB structure is covered at least 30% by the alignment with identity of at least 40% with bit score of at least 70. After mapping, we obtained a human network containing 578 proteins and 547 interactions, and a yeast network containing 526 proteins and 263 interactions.

3.6 Evaluation

All experiments were run on a 3GHz Mac Pro using 2GB of RAM and 8 cores. Our Java-implementations of SEQ-SLIDER, CMC-greedy and CMC-approx can use as many processors as are available. We set out to check if the algorithms we present, fulfill the requirement we first laid out: the result set should be small and have a high coverage of the network.

We have performed a run of SEQ-SLIDER on each of the cross-validation networks described in Section 3.5 taking about 75 minutes requesting 10 000 result motif pairs. A run of CMC-greedy with 30 seconds for each iteration was also performed, as well as a run of CMC-approx with 15 seconds for each iteration. The run of CMC-greedy took on average 14 minutes while CMC-approx took 120 minutes. While in theory the results of SEQ-SLIDER can keep improving over time until they reach the same results as a brute force search, in practice, the gain of allowing additional time over 75 minutes of computation is negligible.

The results averaged over all twenty networks are shown in the table in Figure 3.7. Protein coverage (proteins) shows the percentage of proteins in the network that is covered by at least one motif in the result set. Interaction coverage (interactions) shows the percentage of interactions that is covered by at least one motif pair. Size is the average size of the result sets of the different methods (for SEQ-SLIDER this is constant). It is clear that even though the amount of results returned is much larger for SEQ-SLIDER, it does not cover the interactions in the network well. Even CMC-greedy, with only a fraction of the results covers more interactions. CMC-greedy covers less proteins than SEQ-SLIDER, though. The overall winner on coverage is CMC-approx which, with less than a tenth of the results of SEQ-SLIDER, manages to cover about three quarters of the interactions, and almost all the proteins. If we look at coverage per result, CMC-greedy and CMC-approx score about the same on interactions (and much better than SEQ-SLIDER), but CMC-approx covers much

Method	SEQ-SLIDER	CMC-greedy	CMC-approx
proteins	43.8%	23.4%	96.2%
interactions	11.8%	19.0%	73.5%
size	10 000	214.95	867.5

Figure 3.7: Coverage results of algorithms run on cross-validation networks.

Method	SEQ-SLIDER	CMC-greedy	CMC-approx
proteins	21.2%	44.9%	100%
interactions	23.1%	48.0%	99.9%
size	10 000	663	1 786

Figure 3.8: Coverage results of algorithms run on yeast high-confidence network.

more proteins.

We also ran all algorithms on the high-confidence yeast network. The results are shown in the table in Figure 3.8.

It is clear that the coverage of CMC-approx is far superior to that of both other algorithms. The latter comes at the expense of a longer running time and a larger set of results compared to CMC-greedy.

3.7 Biological validation

To further evaluate the biological relevance of our algorithms, we tested the effectiveness of the derived motif pairs in several scenarios. The first scenario is the prediction of protein interactions from 2D-sequences. In the second scenario, we investigate how well motif hit locations correspond to interface residues by a comparison with protein structure data. Finally, we investigate how cross-species motif pair mining can be used as a filtering step to increase accuracy.

3.7.1 Prediction

We used the weka [HFH+09] data mining software to do prediction using an SVM classifier on the twenty networks described in Section 3.5. The training and test data for the positive and negative data sets were created with the result motif pairs from the runs in Section 3.6 as binary attributes. Of the 10 000 results returned by SEQ-SLIDER only the 1 000 with the best score were used due to limitations of the classifier. In the rare case where CMC-approx returned more than 1 000 results, only the first 1 000 were used. CMC-greedy never returned more than 1 000 results.

The average AUC (Area Under the ROC Curve) of both SEQ-SLIDER and CMC-greedy are 0.502, while that of CMC-approx is 0.526, only a slight improvement. We performed the same predictions with a random forest classifier, but the results were nearly identical. While slightly improving over SEQ-SLIDER, the predicitve performance remains still only marginally different from random (AUC slightly above 0.5), confirming the analysis of Yu et al. [YGN$^+$10], showing that sequences do not contain sufficient information to be useful for predicting PPIs.

3.7.2 Comparison with protein structure data

To determine the overlap of our found motif pairs with actual interaction sites, we ran all three methods on the human and yeast network for which we have a mapping to 3D-structures as described in Section 3.5. The table in Figure 3.9 shows the results for the yeast network; the table in Figure 3.10 for the human network. We start with the amount of motif pairs returned by the method (motif pairs) and check how many unique motifs we have (unique motifs). We then check at how many positions those motifs occur in the proteins (motif hits), if a position occurs multiple times, we count it only once (unique hits). In addition, we also don't count a hit if it occurs within seven amino acids, so it has at least one residue overlap, of a hit we already found (dissimilar). The precision (precision) is the percentage of the remaining hits that is found at the interface. The decision of whether an amino acid is an interface amino acid is based on its Relative Solvent Accessibility (RSA) in the protein and in a complex, in the same way as described in Section 2.8.5. At last, we show the percentage of proteins (proteins) and interactions (interactions) covered.

The comparison with networks containing proteins mapped to protein structure data indicates that for the yeast as well as the human network, coverage of the network increases for CMC-approx, as would be expected, at the expense of only a slight reduction in precision compared to SEQ-SLIDER.

3.7.3 Cross-species comparison

We investigate the hypothesis that motif positions which are found in more than one species could be more likely to be interface residues. In that case, a cross-species comparison could be used to filter noise from the predictions. For the values $(\ell, d) = (8, 3)$, $(8, 4)$ and $(8, 5)$, resulting motifs for SEQ-SLIDER (75 minutes), CMC-approx (2 minutes per iteration), and CMC-greedy (4 minutes

Method	SEQ-SLIDER	CMC-greedy	CMC-approx
motif pairs	10 000	6	82
unique motifs	5 963	11	156
motif hits	8 547	40	340
unique hits	1 792	40	336
dissimilar (7)	388	40	319
precision	49.74%	65%	44.51%
proteins	21.13%	15.49%	100%
interactions	31.94%	20.91%	97.72%

Figure 3.9: Motif hits in 3D info for yeast.

Method	SEQ-SLIDER	CMC-greedy	CMC-approx
motif pairs	10 000	15	135
unique motifs	2 387	30	268
motif hits	10 774	104	972
unique hits	2 246	104	941
dissimilar (7)	393	100	879
precision	39.95%	53%	31.29%
proteins	9.39%	15.48%	100%
interactions	9.46%	23.56%	100%

Figure 3.10: Motif hits in 3D info for human.

per iteration) for the human and yeast networks for which we have a mapping to 3D-structures were mapped to the sequences. CMC-greedy never returned motif pairs for which a decent amount of proteins could be found, so its results are omitted. Subsequently, a cross-species comparison was performed between yeast and human. Proteins from these species were linked to each other by finding best BLAST hits and requiring a sequence identity of at least 50%. The yeast-human pairs were aligned to each other using Muscle [Edg04]. Subsequently, the positions of the motifs in each of the two sequences in such alignment were compared to each other to see if they overlapped or not. Note that this analysis was restricted to the networks for which we have a mapping to 3D-structures because we need structure data to asses the precision of the predicted residues.

The table in Figure 3.11 shows the results from this analysis. For each run, the table shows the number of proteins in which motifs are found and which are among the proteins linked to each other between species (Nprot). Next, it shows how many positions in those proteins contain motifs in both species (same), and how many of those are interface residues (sameTP). Then, the amount of positions in those proteins containing motifs in only a single species (diff), and how many of those are interface residues (diffTP). Finally, there is the fraction of positions that are found in both species, which are indeed interface residues (Fs) and the fraction of positions that are found in only one of the two species, which are indeed interface residues (Fd).

The data in the table supports the hypothesis to some extent. In five out of six cases, positions that are found in both species have a indeed a higher chance to be interface residues than positions that are found in only one species. These are the rows where Fs > Fd. The latter does not hold for SEQ-SLIDER (8,3) but the motifs from that run only select two proteins linked to each other between species. Besides the just described anomaly, there is not a clear difference between SEQ-SLIDER and CMC-approx in this respect, confirming that the CMC-approx motifs are of comparable quality as the SEQ-SLIDER motifs. What is also clear is that the number of proteins which could be used in this comparison was in most cases much higher for the CMC-approx results than for the SEQ-SLIDER results. This is obviously related to the fact that CMC-approx obtains a high network coverage; in this application, we compare two networks from different species which means that a method with a high network coverage is clearly advantageous.

To conclude, comparison of results from networks from various species seems to be a promising strategy to reduce noise in the predictions. The much higher network coverage that CMC-approx obtains compared to SEQ-SLIDER makes it well suited for such an approach.

Method	Nprot	same	sameTP	diff	diffTP	Fs	Fd
SEQ-SLIDER (8,3)	2	8	0	157	65	0.00	0.41
SEQ-SLIDER (8,4)	11	43	18	1 130	327	0.42	0.29
SEQ-SLIDER (8,5)	52	4 234	1 175	8 315	1 449	0.28	0.17
CMC-approx (8,3)	52	68	44	1 205	314	0.65	0.26
CMC-approx (8,4)	51	96	46	1 778	360	0.48	0.20
CMC-approx (8,5)	52	589	161	1 759	497	0.27	0.13

Figure 3.11: Cross-species comparison results for SEQ-SLIDER and CMC-approx on human and yeast networks for which we have a mapping to 3D-structures.

3.8 Conclusion

We sought to improve upon computational methods for deriving motif pairs that usually only find motif pairs that select the densest part of the network. Our new algorithms for CMC decrease the size of the result set while increasing the coverage of the network, having minimal impact on their precision. The much higher coverage of proteins by motifs is important in many applications such as cross-species comparison, where it increases the probability of finding interface residues. In addition, for experimental biologists who want to predict binding sites to perform further experimental studies, obviously it is very important to obtain predictions for as many proteins as possible.

The CMC-approx-implementation and the data used in the experiments are available on `http://bioinformatics.uhasselt.be`.

4

Applications and future work

4.1 Introduction

We now introduce some applications of our algorithms, and show that these provide some promising areas of future research. We start by extending SEQ-SLIDER with additional biological knowledge in Section 4.2 and show that this allows us to improve the accuracy of our results. In Section 4.3, we examine the possibility of extending SLIDER to find binding sites responsible for higher order complexes, by extending our motif pair model to a motif triplet model. In Section 4.4, we extend the experiments of Section 3.7.3 to use multiple species and show that the accuracy improves with each species added. Finally, we conclude in Section 4.5.

4.2 bioSLIDER

We adapt the SEQ-SLIDER algorithm to take as additional input several forms of biological information. This information determines when SEQ-SLIDER considers a motif to be present at a location. The types of information we include are the following:

- amino acid similarity;
- Relative Solvent Accessibility (RSA) values and, if those values are predicted, the confidence in those predictions; and,

- conservation scores.

4.2.1 Adaptation

The general SEQ-SLIDER alorithm (shown in Algorithm 1 in Section 2.5) considers an (ℓ, d)-motif to be present in a sequence, if there is a subsequence of length ℓ that matches it. Next, we describe how we use each type of data to change the definition of a motif being present.

Some amino acids are more similar than others (size, polarity,...). This also makes them more likely to replace each other in binding motifs without affecting the functionality of the protein. There are many different ways to qualify what the similarity between amino acids is. Therefore, we allow as input any matrix with similarity values expressed as a percentage. In practice, we use a modified version of the BLOSUM62 matrix [HH92]. The matrix is scaled to give similarities between amino acids ranging from 0 to 100%, as shown in Figure 4.1. We use the input matrix to determine the similarity between two amino acids by looking in the row of the amino acid in the sequence, and find the value in the column corresponding to the amino acid in the motif. We can use the similarities for amino acids to determine the similarity of a motif to a subsequence in two different ways:

- by averaging the similarities of the matching amino acids, or
- by multiplying the similarities.

In both cases, we ignore the positions of the motif containing a wildcard character. When using the similarity between amino acids as input, an (ℓ, d)-motif is considered to be present in a sequence, if there is a subsequence of length ℓ with similarity above a necessary threshold. This use of amino acid similarity allows for non-exact matches, and therefore increases the amount of matches found in a sequence. It is the only biological information we use that increases the amount of matches found at a location. As shown below, the use of the other data will limit the amount. How many non-exact matches are allowed can be determined by adjusting the threshold parameter.

For example, the sequence `ERLEELEKKEAQLTVTNDQIHILKKENELLHF` does not contain the exact $(8, 3)$-motif `AQxTITxx`. However, if we use the amino acid similarity matrix shown in Figure 4.1, we find that the motif is very similar to the subsequence `AQLTVTND`. If we average the similarity values, their similarity is 97.4%, if we multiply their similarities, the similarity is 85.7%, both of which would be above a reasonable threshold. By lowering the threshold, we would increase the amount of matches found in the sequence.

To be a binding motif, a motif has to be at or near the surface of the protein. Therefore, we discard motif occurrences that have very low RSA values. When

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	100.0	23.1	23.1	7.7	30.8	7.7	7.7	7.7	0.0	7.7	7.7	7.7	7.7	23.1	23.1	23.1	23.1	15.4	15.4	15.4
S	28.6	100.0	57.1	28.6	57.1	42.9	57.1	42.9	42.9	42.9	28.6	28.6	42.9	28.6	14.3	14.3	14.3	14.3	14.3	0.0
T	28.6	57.1	100.0	57.1	28.6	57.1	42.9	57.1	42.9	42.9	42.9	28.6	42.9	28.6	14.3	14.3	14.3	14.3	14.3	0.0
P	9.1	27.3	45.5	100.0	27.3	18.2	27.3	27.3	27.3	27.3	18.2	18.2	27.3	18.2	9.1	9.1	18.2	0.0	9.1	0.0
A	42.9	57.1	28.6	28.6	100.0	42.9	28.6	14.3	28.6	28.6	14.3	28.6	28.6	28.6	28.6	28.6	14.3	14.3	14.3	0.0
G	10.0	40.0	50.0	20.0	40.0	100.0	20.0	30.0	20.0	20.0	20.0	20.0	20.0	10.0	0.0	0.0	40.0	10.0	10.0	20.0
N	10.0	50.0	40.0	20.0	20.0	40.0	100.0	50.0	40.0	40.0	30.0	40.0	40.0	20.0	10.0	10.0	10.0	10.0	20.0	0.0
D	10.0	40.0	50.0	30.0	20.0	30.0	50.0	100.0	60.0	40.0	30.0	20.0	30.0	10.0	10.0	0.0	10.0	10.0	10.0	0.0
E	0.0	44.4	44.4	33.3	33.3	22.2	44.4	66.7	100.0	66.7	44.4	44.4	55.6	22.2	11.1	11.1	11.1	11.1	22.2	11.1
Q	0.0	37.5	37.5	25.0	25.0	12.5	37.5	37.5	62.5	100.0	37.5	50.0	50.0	37.5	0.0	12.5	12.5	0.0	25.0	12.5
H	0.0	18.2	27.3	9.1	9.1	9.1	36.4	36.4	27.3	27.3	100.0	27.3	18.2	9.1	0.0	0.0	9.1	18.2	45.5	9.1
R	0.0	25.0	25.0	12.5	25.0	12.5	37.5	12.5	37.5	50.0	37.5	100.0	62.5	25.0	0.0	12.5	0.0	0.0	12.5	0.0
K	0.0	37.5	37.5	25.0	25.0	12.5	37.5	25.0	50.0	50.0	25.0	62.5	100.0	25.0	0.0	12.5	0.0	0.0	12.5	0.0
M	25.0	25.0	25.0	12.5	25.0	0.0	12.5	0.0	12.5	37.5	12.5	25.0	25.0	100.0	50.0	62.5	12.5	37.5	25.0	25.0
I	37.5	25.0	25.0	12.5	37.5	0.0	12.5	12.5	12.5	12.5	12.5	12.5	12.5	62.5	100.0	75.0	62.5	50.0	37.5	12.5
L	37.5	25.0	25.0	12.5	37.5	0.0	12.5	0.0	12.5	25.0	12.5	25.0	25.0	75.0	75.0	100.0	87.5	50.0	37.5	25.0
V	28.6	14.3	14.3	14.3	42.9	0.0	0.0	0.0	14.3	14.3	0.0	0.0	14.3	57.1	85.7	57.1	100.0	28.6	28.6	0.0
F	20.0	20.0	20.0	0.0	20.0	10.0	10.0	10.0	10.0	10.0	30.0	10.0	10.0	40.0	40.0	40.0	30.0	100.0	70.0	50.0
Y	10.0	10.0	10.0	0.0	10.0	0.0	10.0	0.0	10.0	20.0	50.0	10.0	10.0	20.0	20.0	20.0	20.0	60.0	100.0	50.0
W	13.3	6.7	6.7	0.0	6.7	13.3	0.0	0.0	6.7	13.3	13.3	6.7	6.7	20.0	6.7	13.3	6.7	33.3	40.0	100.0

Figure 4.1: scaled BLOSUM62 matrix

using RSA values as input, we consider an (ℓ, d)-motif to be present, if there is a subsequence of length ℓ that matches it, and whose average RSA value is above a set threshold. If exact RSA values are unknown, predictions can be used. In that case, we want to take into account how much confidence we have in the prediction for each single amino acid. So, we only take the predicted RSA value of an amino acid into account for averaging, if its confidence is above a set threshold. If none of the predictions for a subsequence have a high enough confidence, we ignore the RSA data and consider a motif to be present if it matches. To predict the RSA values of the amino acids, we use the SABLE [APM05] software that predicts solvent accessibilities from sequences alone using a neural networks algorithm trained on PDB structures. The SABLE algorithm outputs an integer value for each residue, ranging from 0 to 9, representing prediction of a fully buried amino acid to prediction of a fully exposed amino acid, respectively.

When part of a protein is conserved over time, it is more likely to be a functional part, such as a binding site, of that protein. When using conservation values as input, we consider an (ℓ, d)-motif to be present, if there is a subsequence of length ℓ that matches it, and whose average conservation value is above a set threshold. To calculate the conservation values for amino acids in a certain protein, we need to perform three steps:

1. select a group of homologous proteins;

2. align the protein sequence to those homologous proteins; and,

3. quantify the conservation of each amino acid in the alignment.

For the purpose of selecting groups of homologs, we use OrthoMCL (Version 2.0) [CMVR07] to assign each protein to a OrthoMCL-DB group. Next,

amino acid	E	R	L	E	**E**	L	**E**	K	K	**E**	A	**E**	L	T	V	T	N	D	Q	I	H	I	L	K	K	**E**	N	**E**	L	L	H	F
RSA	0	3	2	4	**8**	2	**8**	3	8	**4**	2	**1**	4	2	9	5	7	3	5	8	2	0	1	2	4	**6**	2	**4**	4	2	5	3
conservation	2	3	8	4	**5**	9	**0**	2	1	**7**	2	**8**	4	5	2	0	1	0	4	2	8	3	5	2	8	**7**	5	**9**	0	0	1	7

Figure 4.2: Example bioSLIDER input.

we use Clustal [LBB+07] to align the protein sequence with the sequences of all members of the OrthoMCL-DB group. Finally, we use the AL2CO software [PG01] to obtain a conservation score for each position in the multiple sequence alignments. The AL2CO algorithm performs its calculation in two steps: first amino acid frequencies at each position in the alignment are estimated, then a score is calculated from these frequencies. We use the methods unweighted-frequencies and entropy-based in the first and second steps, respectively. To assign a conservation score to each residue in the protein sequence, we use the integer conservation indices resulting from the AL2CO calculation. The AL2CO integer conservation indices range from 0 to 9, representing low to high conservation, respectively.

It is possible with the bioSLIDER algorithm to combine as many (or as few) of these options as wanted for each run.

For example, in Figure 4.2 we see an example input for the bioSLIDER algorithm. We look for the $(3, 1)$-motif ExE, with RSA threshold 5 and conservation threshold 4. We see that the motif occurs three times, but only once (the last occurrence) does it have both a high enough average RSA value and conservation value to be considered.

4.2.2 Experiments

The bioSLIDER algorithm requires the user to set values of parameters that determine the thresholds for degree of similarity, conservation and relative solvent accessibility. The performance of various parameter settings is analyzed by comparing our sequence-based predictions with available protein structure data. This analysis allows us to assess the significance of the inclusion of the biological information in bioSLIDER and, furthermore, to obtain a default set of parameters.

We use the datasets of human and yeast that were mapped to 3D-structures in Section 3.5. For the threshold of the allowed degree of similarity between motif and protein sequence, we test five different values ([none,0.4;0.5;0.6;0.7], where none stands for not having used the modification). For the thresholds of conservation and residue surface accessibility, we test six different values ([none,3;4;5;6;7]). For each combination of parameters (180 in total), we execute bioSLIDER on the structurally mapped datasets for the three species using the following configuration: $\ell = 8$; $d = 3$; execution time of 60 minutes and 1 000 motif pairs returned.

To analyze the results, we define two measures to quantify the quality of the predictions: accuracy (of predicted motifs) and coverage (of protein-protein interfaces). Here, accuracy is defined as the number of motifs correctly predicted to be in the interface as a fraction of all result motifs. A motif is said to be in the interface, if at least one of its residues is identified to be in the interface of its assigned complex structure. Coverage stands for the number of protein pairs that contain at least one motif mapped to their interface, as fraction of the total number of interacting pairs in the interaction data. Thus, accuracy reflects the predictive power of the algorithm toward finding motifs that are indeed located in the interface, and coverage reflects its predictive power towards finding motifs explaining the largest number of interactions. The overall performance of the predictions is measured via the F-score, defined as

$$\frac{2 * \text{accuracy} * \text{coverage}}{\text{accuracy} + \text{coverage}}.$$

The results are shown in Figure 4.3. The x-axis represents the coverage of protein-protein interfaces and the y-axis represents the accuracy of predicted motifs. The dots represent the results of bioSLIDER using each of the 180 tested sets of parameters, for human, and yeast structurally mapped subsets. The grey arrows indicate the dot corresponding to the results of the SEQ-SLIDER algorithm; while the black arrows indicate the dot corresponding to the results of bioSLIDER using the combination of parameters selected to be the default parameters: Amino acid similarity (using average) = 0.6; Conservation = 6; RSA = 7. This setting shows the best F-score over the different networks.

We observe that for most of the parameter settings, the bioSLIDER results are better than the SEQ-SLIDER results, in terms of both accuracy and coverage (Figure 4.3). Depending on the values of the thresholds, bioSLIDER could predict motifs with accuracy up to 58%, and 96%, respectively for the human, and yeast subsets. Likewise, the values of coverage were up to 43%, and 22%.

Of practical consequence here is that the highest thresholds for conservation and surface accessibility would adjust the algorithm to only keep motifs that are strictly conserved across species, respectively fully exposed at the surface. Perhaps the most serious disadvantage here is that the performance of the bioSLIDER algorithm relies, partially, on the correct calculation of residue conservation score and correct prediction of residue surface accessibility score from sequence. However, assuming the correctness of the methods that calculate the a priori information, we expect that the bioSLIDER predicts conserved motifs located at the surface of the proteins, thus improving the relevance of the results.

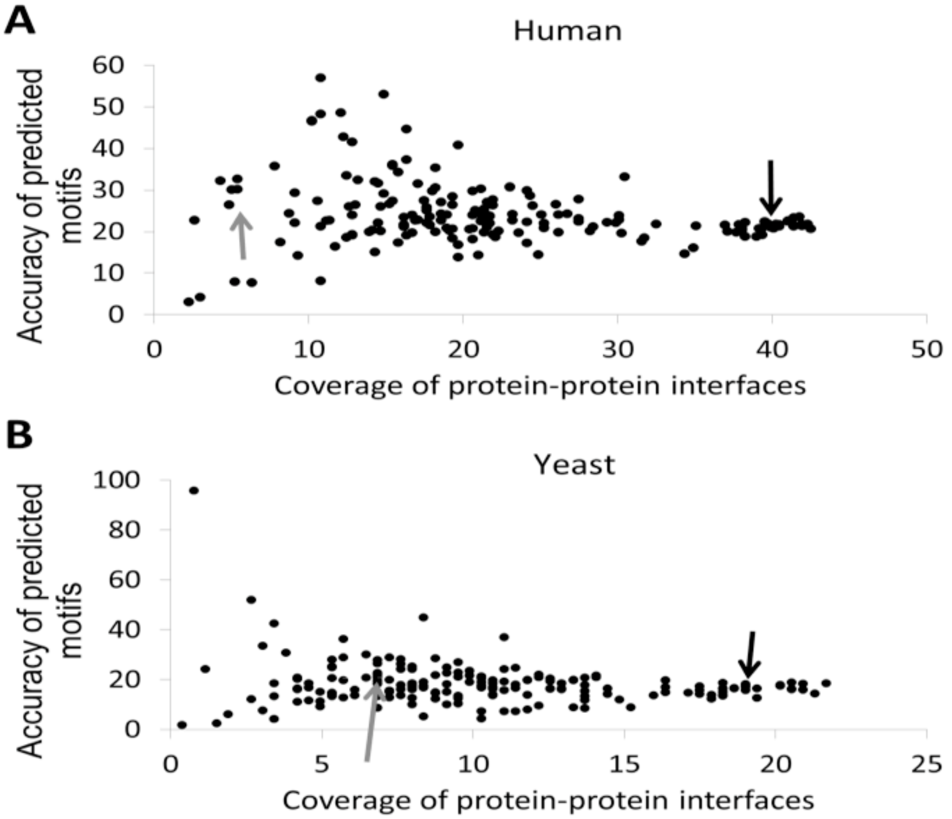


Figure 4.3: Accuracy and coverage (in %) of bioSLIDER on the human (A) and yeast (B) network.

4.3 Binding sites in higher order complexes

MADS domain proteins are involved in various developmental processes in plants [NY01, PdFK+03]. These proteins form dimers, but various experiments also indicate the importance of higher order complexes (consisting of more than two MADS molecules). It has been suggested that these complexes are tetramers (as suggested by the quartet model) which might be dimers of dimers [TS01, MVT08]. Because the MADS transcription factors can only fulfill their role in activating or repressing gene expression after forming the correct complex, and because different complexes have different functions, the specificity of interaction in those complexes is of great biological significance. Here, we investigate how higher order interaction specificity is encoded in the sequences of those proteins, using a variant of the SEQ-SLIDER method. We experimentally test our predictions.

Very limited knowledge exists on which sites are important for higher order complex formation. In particular, the two splice variants of ABS, ABS-I and ABS-II, which are different by a 5-amino acid insertion/deletion, have the same dimeric interactions as obtained by Yeast 2-hybrid (Y2H) but differ substantially in Yeast 3-hybrid (Y3H) interactions [KAST05]. In addition, Yang et al. [YJ04] found three mutations in PI (L121P, L143P and S145P) that interfered with or attenuated higher order complex formation. In tomato, a clear difference in higher order complex formation was found between the two PI-like proteins LePI and TPI: the former participated in various complexes, the latter in none [LEW+08]. There are only three different amino acids between those two proteins.

4.3.1 Adaptation

Higher order complexes

So far, we have always used motif pairs to accomodate for interactions between pairs of proteins. To accommodate the Y3H interactions, we change our approach from motif pairs to motif triplets. In the case of triplets, a PPI-network is defined as follows: $G = (V, E, \lambda)$, with V and λ as before, but now with a set of *hyperedges* $E \subseteq \{(\{u, v\}, w) \mid u, v, w \in V\}$. Remark that the form of the hyperedge means that if the interaction $(\{u, v\}, w)$ is present, this implies the interaction $(\{v, u\}, w)$ is as well, but it implies nothing of $(\{u, w\}, v)$ or any of the other possible permutations. This has implications, for example, on the maximum amount of unique edges that can be present in a subnetwork. We can also represent an edge with an adaptation of the standard notion of hyperedges by using sets $\{u, v, w\}$ or we can use fully ordered tuples (u, v, w), but we choose to focus on $\{(\{u, v\}, w)$ because of the following biologically inspired reason. The tuple $\{(\{u,v\},w)$ models that u and v interact first, and are then later joined by protein w. A motif triplet $(\{X,Y\},Z)$ selects a set of hyperedges as follows: $E_{X,Y,Z} = \{(\{u,v\},w) \in E \mid u \in V_X, v \in V_Y, w \in V_Z\}$.

To evaluate motif triplets, we adapt the χ^2-support measure as follows:

$$f_{\chi^2}((\{X,Y\},Z),G) = \begin{cases} \frac{(|E_{X,Y,Z}|-\overline{E_{X,Y,Z}})^2}{\overline{E_{X,Y,Z}}} & \text{if } |E_{X,Y,Z}| > \overline{E_{X,Y,Z}}; \\ 0 & \text{if } |E_{X,Y,Z}| \leq \overline{E_{X,Y,Z}}; \end{cases}$$

with $\overline{E_{X,Y,Z}}$ the expected number of interactions between V_X, V_Y, and V_Z. The value $\overline{E_{X,Y,Z}}$ is calculated by assuming a uniform density of edges:

$$\overline{E_{X,Y,Z}} = \mathrm{ed}(G) E^{\max}_{V_X,V_Y,V_Z} \ .$$

Here $\mathrm{ed}(G)$ is defined as $\frac{|E|}{|V|\binom{|V|}{2}}$, and $E^{\max}_{V_X,V_Y,V_Z}$ is the maximum number of possible interactions between the three sets of proteins:

$$E^{\max}_{V_X,V_Y,V_Z} = \left(|V_X||V_Y| - \binom{|V_X \cap V_Y|}{2} - |V_X \cap V_Y|\right)|V_Z| \ .$$

To use SEQ-SLIDER with this support measure, we also need a neighbor function. So we define,

$$\begin{aligned} N^{\mathrm{seq}}_{u,v,w}((\{X,Y\},Z)) = \{(\{X',Y\},Z) \mid X' \in N^{\mathrm{seq}}_u(X)\} \\ \cup \{(\{X,Y'\},Z) \mid Y' \in N^{\mathrm{seq}}_v(Y)\} \\ \cup \{(\{X,Y\},Z') \mid Z' \in N^{\mathrm{seq}}_w(Z)\}. \end{aligned}$$

Our implementation of SEQ-SLIDER for triplets can also use edge sets of the forms $E \subseteq \{(u,v,w) \mid u,v,w \in V\}$ or $E \subseteq \{\{u,v,w\} \mid u,v,w \in V\}$. Similar formulas for those forms can be readily derived.

Interaction probabilities

Methods of interaction detection introduce errors, some more than others. Also, some proteins are known to not interact, as opposed to those where it is not known if they interact, e.g. because the combination has not been tested. So instead of using a binary true/false metric for interactions, we want to use a probability of having an interaction that reflects the likelihood of an interaction actually being present when indicated by a certain method, counter-indicated by a method, or not checked at all. This probability could then be set depending on the source(s) that suggest (non-)interaction between a pair of proteins.

The χ^2-based support measure, easily translates into one that takes such probabilities into account:

$$f_{\chi^2}(\{X,Y\},G) = \begin{cases} \dfrac{\left(\sum\limits_{e \in E_{X,Y}} p(e) - \overline{E_{X,Y}}\right)^2}{\overline{E_{X,Y}}} & \text{if } \sum\limits_{e \in E_{X,Y}} p(e) > \overline{E_{X,Y}}; \\ 0 & \text{if } \sum\limits_{e \in E_{X,Y}} p(e) \leq \overline{E_{X,Y}}; \end{cases}$$

with $p(e)$ the probabilty of edge e, and $\overline{E_{X,Y}}$ the expected number of interactions (which is not necessarily an integer) between V_X and V_Y. The value $\overline{E_{X,Y}}$ is calculated in the same way as before, by assuming a uniform *density* of edges:

$$\overline{E_{X,Y}} = \mathrm{ed}(G) E^{\max}_{|V_X|,|V_Y|,|V_X \cap V_Y|} ,$$

but with $\mathrm{ed}(G) = \sum\limits_{e \in E} p(e) / \binom{|V|}{2}$.

It is clear that if all probabilities are set to 1, this defaults to the regular SLIDER algorithm.

The adaptation of the support measure using probabilities instead of binary interaction data to such a support measure used for motif triplets is straightforward.

4.3.2 Experiments

Arabidopsis MADS Y3H data, as obtained by Immink et al. [ITdF+09], are used. These data essentially contain triplets of proteins which interact, and triplets of proteins which do not interact. Out of these, we restrict ourselves to the type II MADS domain proteins (the majority, and best studied so far in general).

There are also some additional (smaller) datasets available, in particular there are gerbera [RNA+10], tomato [LEW+08], and a very small set from rice [SPP+10]. We use the gerbera and tomato data in addition to the Arabidopsis data.

On this data, we perform runs of SEQ-SLIDER for triplets where interaction data is considered in two ways: either unordered ($\{u, v, w\}$) or semi-ordered $((\{u, v\}, w))$; and where unknowns are either treated as if they were non-interactions, or are given a probability equal to the probability observed for all triplets measured to be interacting among all observed triplets (0.02754). We use the following combinations for ℓ and d: $(8, 5), (8, 4), (7, 4)$ and $(7, 3)$, resulting in total in 16 different sets of motif triplets.

The resulting lists of motif triplets suggest possible locations for binding sites on the proteins in the network. We now analyze if these predicted binding sites are indeed responsible for the interaction. This is done by taking the original protein and then changing the predicted binding site, using a process called mutagenesis. Then, we test if this newly formed protein still forms the same interactions using a new Y3H experiment. If the protein no longer forms those interactions, we are sure that the targeted location was indeed a binding site.

Mutation experiments

First, we must select which amino acids in the proteins we want to change. Positions of the motifs from the SEQ-SLIDER runs are visually analyzed for the various settings, using only the top 10 motif triplets. All cases with $(7, 4)$ and $(8, 5)$ show motifs all over the sequence; since these results do not allow to distinguish between interface and non-interface amino acids, they are discarded. Similarly, for $(7, 3)$ and $(8, 4)$ the cases with "probability" to treat the unknowns result in motifs all over the sequence; these results are also discarded. For the remaining results, it seems that $(7, 3)$ and $(8, 4)$ yield more or less similar results so only the latter are further analyzed. To make a selection out of the proteins containing these motifs, it is also taken into account whether a protein is interesting for biologists (because of known function for example). From the locations of the motifs on these proteins, only amino acids on positions that are not wildcards are chosen to be mutated.

Then, a mutation is introduced in two different proteins, targeting amino acids in two different SEQ-SLIDER motifs. In one case, four complexes (triplets of proteins) are tested which could be formed with the original protein, where for two of them the prediction was that they would be lost upon introducing the mutation, whereas for the other two the prediction was that they would not be lost (because the SEQ-SLIDER motif was not predicted to be involved in those two). Experimentally, all four complexes are still formed by the mutated version of the protein (the predicted loss of interaction is not observed). For the second protein, there are three complexes for which loss of interaction was predicted, and two for which it was predicted that the mutated protein would retain the interaction. In this case, the three predicted losses are not observed experimentally. Moreover, out of the two complexes for which it was predicted that they would be retained, one was experimentally observed to be lost.

In addition to the SEQ-SLIDER-based mutations, a mutation is introduced in ten other proteins at sites which are not covered by SEQ-SLIDER motifs. Here, in some cases, clear losses or gains of interations are observed.

Unfortunately, the results of these experiments indicate that SEQ-SLIDER for triplets is unable to locate actual binding sites. We will discuss options to improve the method in Section 4.5.

4.4 Cross-species evaluation

Our hypothesis, as first suggested in Section 3.7.3, is that combining interaction site predictions obtained for several PPI networks would improve the performance of such predictions. The basic assumption here is that proteins which have similar sequences (orthologs) also have interaction sites at similar locations. Hence, if for certain proteins interaction sites would be predicted at similar locations in various species, one would expect that those are more likely to be actual binding sites than when in each species the interaction site is predicted at a different location. This idea could be used to filter predictions from various species and combine them into one integrated set. To test this idea, we use data from various species. Subsequently, we map the proteins in various species to each other (using predicted orthology relationships) to combine the predicted interaction sites. Finally, we validate the predicted interaction sites by mapping results to available protein complex structures.

4.4.1 Data

For our experiments, we require data from several species. We can acquire this data in two ways. One choice would be to use interaction networks which have been experimentally determined in those species. Alternatively, given an experimentally determined network in one species, the network in other species could simply be predicted using orthologous sequences.

We use data from the STRING repository [vMJS$^+$05], which contains both known and predicted protein-protein associations, based on various experimental datasets and prediction methods. In addition, interactions are transferred across species using orthology information. No distinction was made between experimentally determined annotations and annotations of interactions based on orthology, because if we were to focus on using only experimentally determined interactions, only a sparse amount of data would be available. This is the case in particular because we need to map orthologs across species to compare our predicted interaction sites. Note that for the orthology relationships we use the orthology groups provided by STRING. We use only those interactions with a combined confidence score higher than the typically used threshold value of 0.7.

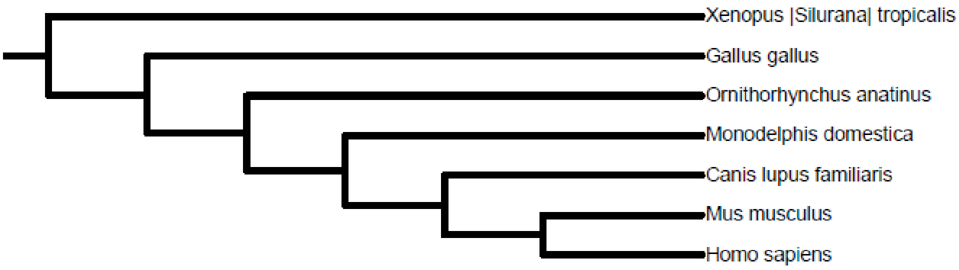


Figure 4.4: Species used from STRING.

	Complete network		Restricted network	
	Nprot	Nint	Nprot	Nint
Homo sapiens	15 010	459 198	4 486	111 190
Mus musculus	11 997	311 490	3 880	67 994
Canis lupus	5 541	63 168	2 705	30 686
Monodelphis domestica	4 790	55 776	2 536	30 980
Ornithorhynchus anatinus	3 305	30 644	2 090	19 038
Gallus gallus	4 088	44 416	2 309	24 562
Xenopus (Silurana)	3 636	31 162	2 007	16 780

Figure 4.5: Number of proteins and number of interactions.

Out of the hundreds of species for which STRING has data, we select a subset of seven species, including human and the six species most closely related to it (see Figure 4.4). In addition to filtering with the confidence score, we also require that proteins should be in an orthology group that has at least one member in each of the seven species. The table in Figure 4.5 contains the number of proteins (Nprot) and the number of interactions (Nint), for the networks restricted to the most reliable subset with combined confidence score > 0.7 (Complete network), as well as those for the networks where each protein must have orthologs in each of the other species (Restricted network).

4.4.2 Experiments

Interaction sites for the human proteins are obtained by mapping them to available PDB structures as described in Section 3.5. These sites are all combined, such that for each human protein we have a list of interface residues (proteins for which no interface residue was found at all are ignored in the subsequent performance assessment step). Sequences of members of the same orthology group are aligned with MUSCLE [Edg04], and the positions of the interface residues are mapped to alignment columns. For each species, a run of CMC-approx ($(8,3)$, 1 minute) is performed to gather predicted interaction sites. Subsequently, predicted interaction sites (motif locations) for members of the same orthology group are all mapped to alignment columns. Here, orthology groups for which no interaction site was predicted at all (i.e. for none of the proteins in the orthology group a motif hit the sequence) are ignored. For each alignment column, a prediction score is obtained as the fraction of sequences in the alignment for which this particular column was predicted as an interface residue.

In total, 2 016 different orthology groups are present in the different networks we analyzed. By construction, we focus on orthology groups for which proteins are present in each of the seven species. However, for some of those families, in some species the proteins do not have any interaction with proteins which pass the filter, and hence for a number of families there is not a protein present in the final network in each of the seven species. Out of a total of 2 016 orthology groups that passed the filter, this was the case for 234. We ignore those orthology groups in the performance prediction and focus on the remaining 1 782 for which at least one protein is present in the final interaction network in each of the seven species. In total, 30 346 proteins are present in those orthology groups.

We also separately analyze orthology groups with at least a minimum number of proteins. When requiring at least 15 proteins, the remaining number of orthology groups is 521 (with a total number of proteins 19 254); when requiring at least 20 proteins, the remaining number of orthology groups is 333 (with 16 112 proteins) and when requiring at least 25 proteins, the remaining number of orthology groups is 234 (with 13 953 proteins). Note that though the amount of orthology groups drops rapidly, the amount of proteins covered is still very large. For each of the orthology groups for which structural information is available for the human sequences, prediction scores are obtained for each sequence alignment column, and these are compared with the known interaction sites. This is performed using either only prediction in human (Homo sapiens), or in human and mouse (Mus musculus), and so on, progressively adding one species at a time. The performance is assessed using the AUC (class label to be predicted is 1/0 for interface/non-interface; prediction score is fraction of sequences in the alignment in which the position is predicted to be an interface residue, i.e. covered by a motif). From the AUC values (see the table in Figure 4.6) it is clear that adding additional species improves the prediction. It can also be seen that combining sequences within a species helps as well (the more proteins are present in an orthology group, the better the prediction performance). Figure 4.7 shows some example ROC curves.

Species used	All groups	< 15	≥ 15	≥ 20	≥ 25
1	0.579	0.507	0.649	0.679	0.719
2	0.597	0.515	0.669	0.702	0.745
3	0.609	0.516	0.684	0.715	0.757
4	0.619	0.520	0.693	0.726	0.765
5	0.628	0.520	0.703	0.735	0.771
6	0.635	0.522	0.709	0.741	0.778
7	0.642	0.522	0.717	0.748	0.785

Figure 4.6: AUC values for prediction of interface residues. Prediction can be done with a different number of species and with a different minimum number of proteins per orthology group.

4.5 Conclusion

We have shown the potential of bioSLIDER to increase both the accuracy and coverage of our results. We could consider additional types of, preferably plentifully available or easily acquired, biological data which might be added to further improve the results. For instance, most interactions between proteins occur between two domains. Therefore, we could look for motifs that fall completely within such a domain. bioSLIDER is already capable of using domain data as input. Such data would for each amino acid indicate in which domain (if any) it is located. When using domain data as input, we would consider an (ℓ, d)-motif to be present, if there is a subsequence of length ℓ that matches it, and that is contained within a single domain. The effect of using this data is so far untested.

Generalizing from a motif pair to a motif triplet is the most naive adaptation of SEQ-SLIDER possible for discovering higher order binding sites. Apparently, the motif triplets model is too much of a simplification of what is really going on to be useful. We might still improve our results by introducing a more powerful model into SLIDER. To capture the true biological background, we may need to consider pairs of motif pairs, one responsible for binding the first two proteins, and the second for binding the third protein to the complex. Maybe we even need triplets of motif pairs, where the last two pairs are responsible for binding the third protein to parts of both original proteins. Either way, we would need to devise a more powerful search algorithm to tackle the highly increased size of the search space this would entail. We must also consider that our current (ℓ, d)-motif model is not powerful enough for this application.

In the cross-species comparison, we have successfully demonstrated the possibility of filtering our results by using results from multiple closely related

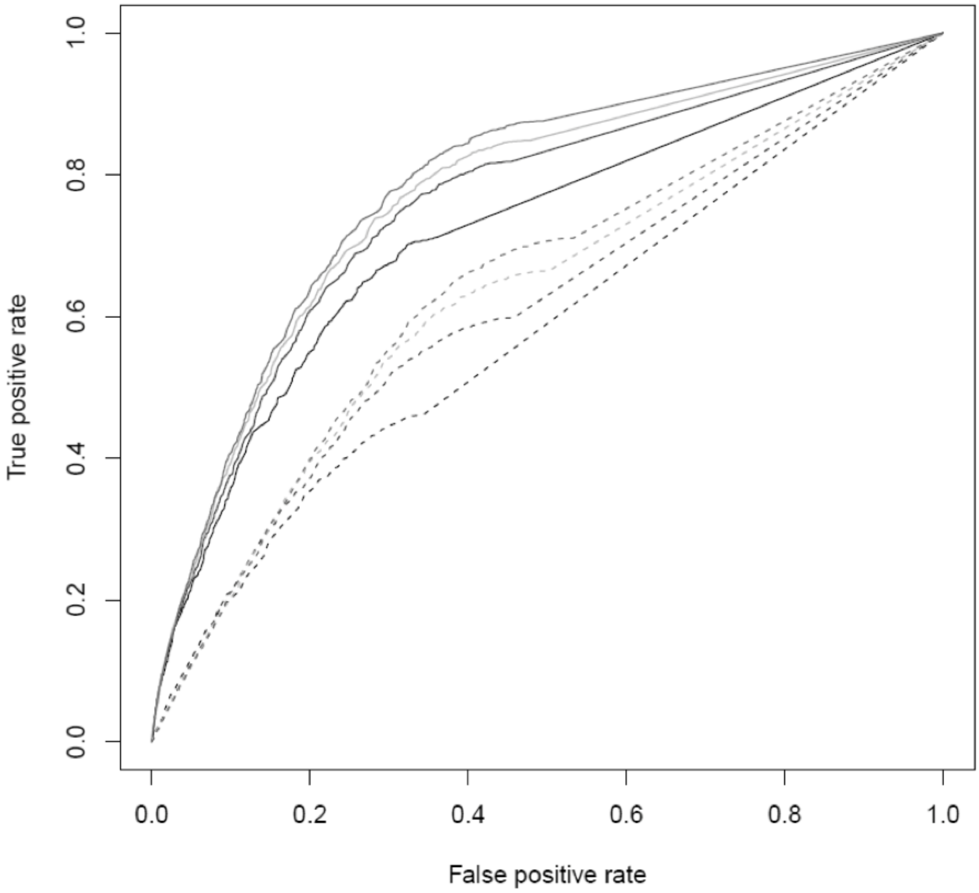


Figure 4.7: ROC curves for prediction of human interface residues. A dashed line indicates all orthology groups were used, a continuous line, only orthology groups with at least 25 proteins. For each set of lines the lowest is the one where only human network data were used to obtain predictions; second: three species were used; third: five species were used; top: all seven species were used.

species. It is still possible to investigate how necessary it is that the species be closely related. Also, note that the mapping of results between species could have been performed in different ways. Currently, we just compared predicted interaction sites for a set of orthologous sequences. On the other hand, we could also take into account differences in interactions between the species, and only compare predicted interaction sites for pairs of sequences where the interaction sites derive from a motif pair covering that pair of sequences, to see if this offers further improvement.

5 Conclusion

Discovering binding sites using biological experiments takes a considerable amount of time and money. Therefore, it is very interesting to see, if these methods can be complemented by computational approaches to guide the process. First, motif discovery algorithms were used to find motifs in groups of proteins, known to have similar interactions. This required knowledge of these protein groups and usually only provided the binding site in one of the interacting proteins. Therefore, methods for Correlated Motif Mining (CMM) were developed, two of which followed the motif-driven approach, namely MotifHeuristics [LSY$^+$07] and D-STAR [THSN06]. Of those, only D-STAR [THSN06] was freely available. D-STAR, however, has a severe limitation on the size of the network it can handle. It can not perform on networks consisting of more than 250 proteins, which is very small compared to the size of actual biological networks. Though MotifHeuristics can handle networks of larger size than D-STAR, it uses a probability-based support measure called the *p*-score, which we have experimentally shown to be inferior to the χ^2-based support measure of D-STAR.

By reducing the Balanced Complete Bipartite Subgraph (BCBS) problem to the decision problem associated with CMM, we proved the NP-hardness of CMM for a broad class of support measures. This shows that, given current mathematical knowledge, an exact algorithm for CMM is impossible. Since the decision problem associated with CMM is in NP, we can efficiently compare the support of two motif pairs, making it possible to tackle CMM as a combinatorial optimization problem.

To provide a highly scalable method, we created SLIDER, a local search method using a χ^2-based support measure, which can handle biological networks consisting of thousands of proteins. We have shown that SLIDER, SEQ-SLIDER in particular due to its unique neighbor function, has a higher precision and more resistance to noise in the data than any other existing method on artificial networks. On biological networks, we have found significant overlap of motif hits of SEQ-SLIDER results with interface residues. In general, SEQ-SLIDER closely approaches the results of a brute force run in a fraction of the time.

Though SEQ-SLIDER provides a way to solve CMM (near) optimally, there is a large amount of motifs that covers a very similar, if not identical, part of the network. As a result, the best scoring motif pairs often refer to highly similar selected networks and to get a decent coverage of the network, we would need to incorporate an unmanageable amount of motif pairs. So, even though SEQ-SLIDER is successful in finding the best motif pairs according to a support measure, the resulting binding sites all occur in proteins in the small, densest part of the network. Though knowledge of these binding sites is useful, it is important for biological tests to know as many as possible throughout the entire network.

The use of additional biological data, with bioSLIDER, shows promise in improving our results, especially increasing the coverage. However, it is still limited by the problem definition of CMM, which allows many result motif pairs to cover the same part of the network.

Therefore, we chose to move to a new problem, which we called Correlated Motif Covering (CMC), which focuses on the predictive power of the complete set of results instead of that of the single results. By demanding a motif pair adds predictive value to the set of already selected motif paris, we make an intelligent choice from any set of motif pairs that have large overlap in the interactions they describe.

Using reductions between CMC and the (Weighted) Red-Blue Set Cover ((W)RBSC) problem, we proved the NP-hardness of CMC and that it belongs to a class of problems for which it is practically impossible to find an approximation algorithm with a sub-square root approximation rate. There are some approximation algorithms for WRBSC with a higher approximation rate, and we adapted one of these algorithms to solve CMC.

We created CMC-approx, which uses data mining techniques to drastically increase the coverage of the output motif pairs with only a minor drop in precision, when compared to SEQ-SLIDER.

Using data from multiple species promises to boost the precision of our methods. When we run CMC-approx on the data of multiple species and combine the results, we get results with high accuracy and coverage.

We identify several directions for future work such as:

- Investigating candidate generation for motif pairs. Currently motif pairs are generated purely at random from the data. It might be interesting to see what could be accomplished by heuristically chosen starting motif pairs. For example, the motifs for a starting pair could be chosen based on the amount of motifs they select (though we have shown in Section 2.9 that a large motif does not necessarily equal the best motif pair). Alternatively, they could be based on the best motif pair out of a set of randomly chosen ones. These heuristic starting pairs could be used alone or mixed with randomly generated starting pairs to compensate for the bias induced by the heuristic.

- Though we favored the motif-driven approach, a detailed comparison with interaction-driven approaches [LLW06, LLLW07, LSLW08] should be done. Maybe ideas from both paradigms can be successfully combined into a hybrid method.

- Since the β parameter of CMC-approx has such a natural correspondence to noise, and the γ parameter to missing data, it might be interesting to find out if we can compensate for those factors by adjusting their respective values.

- Most importantly, we have only considered the admittedly very simple model of (ℓ, d)-motifs and our results suggest that this model has some drawbacks. It suffers from false positives caused by indirect interactions. Also, as we have shown when applying motif pairs to predict interactions, the prediction using $(\ell, d)-$motif pairs as binary attributes is hardly an improvement over random. We noticed that while the mined motif pairs cover the training set well, they hardly cover any of the proteins in the test set. In this sense, (ℓ, d)-motifs are too selective.

  Though (ℓ, d)-motifs are very common in the field of bioinformatics, it could be worthwhile to investigate how we can deviate from the simple $(\ell, d)-$motif model. We have already presented one possible adaptation by using the similarities between amino acids and we could consider others, such as, for instance, adding disjunctions. There also exist more expressive models (e.g., Position Weight Matrix or Hidden Markov Model). Of course, we should take care to balance the expressiveness, and generality, of motifs with the computational effort required to mine them.

Bibliography

[AL97] E. Aarts and J.K. Lenstra, editors. *Local Search in Combinatorial Optimization.* John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1997.

[APM05] R. Adamczak, A. Porollo, and J. Meller. Combining prediction of secondary structure and solvent accessibility in proteins. *Proteins: Structure, Function, and Bioinformatics*, 59(3):467–475, 2005.

[AR04] P. Aloy and R.B. Russell. Ten thousand interactions for the molecular biologist. *Nature Biotechnology*, 22(10):1317–1321, October 2004.

[Bay98] R.J. Bayardo. Efficiently mining long patterns from databases. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93, New York, NY, USA, 1998. ACM.

[BBB$^+$02] H.M. Berman, T. Battistuz, T.N. Bhat, W.F. Bluhm, P.E. Bourne, K. Burkhardt, Z. Feng, G.L. Gilliland, L. Iype, S. Jain, P. Fagan, J. Marvin, D. Padilla, V. Ravichandran, B. Schneider, N. Thanki, H. Weissig, J.D. Westbrook, and C. Zardecki. The Protein Data Bank. *Acta Crystallographica Section D: Biological Crystallography*, 58(Pt 6 No 1):899–907, June 2002.

[BNVD$^+$09] P. Boyen, F. Neven, D. Van Dyck, A.D.J. van Dijk, and R.C.H.J. van Ham. SLIDER: Mining Correlated Motifs in Protein-Protein Interaction Networks. In *ICDM 2009, The Ninth IEEE International Conference on Data Mining, Miami, Florida, USA, 6-9 December 2009*, pages 716–721, 2009.

[BNVD$^+$on] P. Boyen, F. Neven, D. Van Dyck, F.L. Valentim, and A.D.J. van Dijk. Mining minimal motif pair sets maximally covering interactions in a protein-protein interaction network. submitted for publication.

[BR03] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35:268–308, September 2003.

[BVDN$^+$11] P. Boyen, D. Van Dyck, F. Neven, R.C.H.J. van Ham, and A.D.J. van Dijk. SLIDER: A Generic Metaheuristic for the Discovery of Correlated Motifs in Protein-Protein Interaction Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, January 2011.

[BW05] J.R. Bradford and D.R. Westhead. Improved prediction of protein-protein binding sites using a support vector machines approach. *Bioinformatics*, 21(8):1487–1494, April 2005.

[BZ09] B. Bringmann and A. Zimmermann. One in a million: picking the right patterns. *Knowledge and Information Systems*, 18:61–81, 2009.

[CDKM00] R.D. Carr, S. Doddi, G. Konjevod, and M. Marathe. On the red-blue set cover problem. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, SODA '00, pages 345–353, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

[Chv79] V. Chvatal. A Greedy Heuristic for the Set-Covering Problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[CKX$^+$07] S.R.R. Collins, P. Kemmeren, Xue, J.F.F. Greenblatt, F. Spencer, F.C.C. Holstege, J.S.S. Weissman, and N.J.J. Krogan. Towards a comprehensive atlas of the physical interactome of Saccharomyces cerevisiae. *Molecular & Cellular Proteomics*, January 2007.

[CMVR07] F. Chen, A.J. Mackey, J.K. Vermunt, and D.S. Roos. Assessing performance of orthology detection strategies applied to eukaryotic genomes. *PLoS ONE*, 2(4):e383, 2007.

[DD07] M. Das and H.K. Dai. A survey of DNA motif finding algorithms. *BMC Bioinformatics*, 8(Suppl 7):S21+, 2007.

[DS04] I. Dinur and S. Safra. On the hardness of approximating label-cover. *Information Processing Letters*, 89:247–254, March 2004.

[Edg04] R.C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–1797, March 2004.

[EP07] M. Elkin and D. Peleg. The Hardness of Approximating Spanner Problems. *Theory of Computing Systems*, 41:691–729, December 2007.

[GGM04] F. Geerts, B. Goethals, and T. Mielikäinen. Tiling Databases. In *Lecture Notes in Computer Science*. Springer, 2004.

[GHZ07] K. Gouda, M. Hassaan, and M.J. Zaki. Prism: A Primal-Encoding Approach for Frequent Sequence Mining. In *ICDM 2007, The Seventh IEEE International Conference on Data Mining, Omaha, Nebraska, USA, 28-31 October 2007*, pages 487–492, 2007.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability; A Guide to the Theory of* NP*-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[HFH$^+$09] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations Newsletter*, 11:10–18, November 2009.

[HH92] S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, 89(22):10915–10919, November 1992.

[HT93] S. Hubbard and J. Thornton. NACCESS, 1993. Computer Program, Department of Biochemistry and Molecular Biology, University College London.

[ITdF$^+$09] R. Immink, I. Tonaco, S. de Folter, A. Shchennikova, A. van Dijk, J.B. Lange, J. Borst, and G. Angenent. SEPALLATA3: the 'glue' for MADS box transcription factor complex formation. *Genome Biology*, 10(2):R24+, feb 2009.

[KAST05] K. Kaufmann, N. Anfang, H. Saedler, and G. Theissen. Mutant analysis, protein-protein interactions and subcellular localization of the Arabidopsis B sister (ABS) protein. *Molecular Genetics & Genomics*, 274(2):103–18, 2005.

[KFH$^+$02] T. Kawabata, S. Fukuchi, K. Homma, M. Ota, J. Araki, T. Ito, N. Ichiyoshi, and K. Nishikawa. GTOP: a database of protein structures predicted from genome sequences. *Nucleic acids research*, 30(1):294–298, January 2002.

[KPGK$^+$09] T.S. Keshava Prasad, R. Goel, K. Kandasamy, S. Keerthikumar, S. Kumar, S. Mathivanan, D. Telikicherla, R. Raju, B. Shafreen, A. Venugopal, L. Balakrishnan, A. Marimuthu, S. Banerjee, D.S. Somanathan, A. Sebastian, S. Rani, S. Ray, C.J. Harrys Kishore, S. Kanth, M. Ahmed, M.K. Kashyap, R. Mohmood, Y.L. Ramachandra, V. Krishna, B.A. Rahiman, S. Mohan, P. Ranganathan, S. Ramabadran, R. Chaerkady, and A. Pandey. Human Protein Reference Database - 2009 update. *Nucleic Acids Research*, 37(suppl 1):D767–D772, January 2009.

[LBB$^+$07] M.A. Larkin, G. Blackshields, N.P. Brown, R. Chenna, P.A. McGettigan, H. McWilliam, F. Valentin, I.M. Wallace, A. Wilm, R. Lopez, J.D. Thompson, T.J. Gibson, and D.G. Higgins. Clustal W and Clustal X version 2.0. *Bioinformatics (Oxford, England)*, 23(21):2947–2948, nov 2007.

[LEW$^+$08] C.H. Leseberg, C.L. Eissler, X. Wang, M.A. Johns, M.R. Duvall, and L. Mao. Interaction study of MADS-domain proteins in tomato. *Journal of Experimental Botany*, 59(8):2253–2265, may 2008.

[LLLW07] J. Li, G. Liu, H. Li, and L. Wong. Maximal Biclique Subgraphs and Closed Pattern Pairs of the Adjacency Matrix: A One-to-One Correspondence and Mining Algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 19:1625–1637, December 2007.

[LLW06] H. Li, J. Li, and L. Wong. Discovering motif pairs at interaction sites from protein sequences on a proteome-wide scale. *Bioinformatics*, 22:989–996, April 2006.

[LSLW08] J. Li, K. Sim, G. Liu, and L. Wong. Maximal Quasi-Bicliques with Balanced Noise Tolerance: Concepts and Co-clustering Applications. In *Proceedings of the SIAM International Conference on Data Mining, SDM 2008, April 24-26, 2008, Atlanta, Georgia, USA*, pages 72–83, 2008.

[LSY$^+$07] H.C. Leung, M.H. Siu, S.M. Yiu, F.Y. Chin, and K.W. Sung. Finding linear motif pairs from protein interaction networks: a probabilistic approach. *Computational systems bioinformatics*

/ Life Sciences Society. Computational Systems Bioinformatics Conference, 6:111–119, 2007.

[LVT07] G. López, A. Valencia, and M.L. Tress. *Firestar* - prediction of functionally important residues using structural templates and alignment reliability. *Nucleic Acids Research*, 35(Web-Server-Issue):573–577, 2007.

[LZLZ06] S. Liang, C. Zhang, S. Liu, and Y. Zhou. Protein binding site prediction using an empirical scoring function. *Nucleic Acids Research*, 34(13):3698–3707, January 2006.

[MJ06] Y. Murakami and S. Jones. SHARP2: protein–protein interaction predictions using patch analysis. *Bioinformatics*, 22:1794–1795, July 2006.

[MVT08] R. Melzer, W. Verelst, and G. Theissen. The class E floral homeotic protein SEPALLATA3 is sufficient to loop DNA in 'floral quartet'-like complexes in vitro. *Nucleic Acids Research*, pages gkn900+, nov 2008.

[NRS04] H. Neuvirth, R. Raz, and G. Schreiber. ProMate: A Structure Based Prediction Program to Identify the Location of Protein-Protein Binding Sites. *Journal of Molecular Biology*, 338(1):181–199, April 2004.

[NSO$^+$07] S.S. Negi, C.H. Schein, N. Oezguen, T.D. Power, and W. Braun. InterProSurf: a web server for predicting interacting sites on protein surfaces. *Bioinformatics*, 23(24):3397–3399, December 2007.

[NY01] M. Ng and M.F. Yanofsky. Function and evolution of the plant mads-box gene family. *Nature reviews. Genetics*, 2(3):186–95, 2001.

[OKA$^+$05] U. Ogmen, O. Keskin, A.S. Aytuna, R. Nussinov, and A. Gursoy. PRISM: protein interactions by structural matching. *Nucleic acids research*, 33(Web Server issue), July 2005.

[PBTL99] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering Frequent Closed Itemsets for Association Rules. In *Proceedings of the 7th International Conference on Database Theory*, ICDT '99, pages 398–416, London, UK, 1999. Springer-Verlag.

[PdFK$^+$03] L. Pařenicová, S. de Folter, M. Kieffer, D.S. Horner, C. Favalli, J. Busscher, H.E. Cook, R.M. Ingram, M.M. Kater, B. Davies, G.C. Angenent, and L. Colombo. Molecular and phylogenetic analyses of the complete MADS-box transcription factor family in arabidopsis: New openings to the MADS world. *The Plant Cell*, 15:1538–1551, jul 2003.

[Pel07] D. Peleg. Approximation algorithms for the Label-Cover$_{MAX}$ and Red-Blue Set Cover problems. *Journal of Discrete Algorithms*, 5:55–64, March 2007.

[PG01] J. Pei and N.V. Grishin. AL2CO: calculation of positional conservation in a protein sequence alignment. *Bioinformatics*, 17(8):700–712, 2001.

[PM07] A. Porollo and J. Meller. Prediction-based fingerprints of protein-protein interactions. *Proteins: Structure, Function, and Bioinformatics*, 66(3):630–645, February 2007.

[Ris83] J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11:416–431, 1983.

[RNA$^+$10] S. Ruokolainen, Y. Ng, V. Albert, P. Elomaa, and T. Teeri. Large scale interaction analysis predicts that the gerbera hybrida floral e function is provided both by general and specialized proteins. *BMC Plant Biology*, 10(1):129, 2010.

[SGG03] E.W. Stawiski, L.M. Gregoret, and Y.M. Gutfreund. Annotating Nucleic Acid-Binding Function Based on Protein Structure. *Journal of molecular biology*, 326(4):1065–1079, 2003.

[SPNW04] A. Shulman-Peleg, R. Nussinov, and H.J. Wolfson. Recognition of functional sites in protein structures. *Journal of molecular biology*, 339(3):607–633, June 2004.

[SPP$^+$10] H. Seok, H. Park, J. Park, Y. Lee, S. Lee, G. An, and Y. Moon. Rice ternary MADS protein complexes containing class B MADS heterodimer. *Biochemical and Biophysical Research Communications*, 401(4):598–604, 2010.

[STdS$^+$08] M.P.H. Stumpf, T. Thorne, E. de Silva, R. Stewart, H.J. An, M. Lappe, and C. Wiuf. Estimating the size of the human interactome. *Proceedings of the National Academy of Sciences*, 105(19):6959–6964, May 2008.

[THSN06] S.H. Tan, W. Hugo, W.K. Sung, and S.K. Ng. A correlated motif approach for finding short linear motifs from protein interaction networks. *BMC bioinformatics*, 7(502), November 2006.

[TQZ07] H. Tjong, S. Qin, and H. Zhou. PI2PE: protein interface/interior prediction engine. *Nucleic Acids Research*, 35(Web Server):W357–W362, May 2007.

[TS01] G. Theissen and H. Saedler. Plant biology. floral quartets. *Nature*, 409(6819):469–71, 2001.

[Vaz04] V.V. Vazirani. *Approximation Algorithms.* Springer, March 2004.

[vDMF$^+$10] A.D.J. van Dijk, G. Morabito, M. Fiers, R.C.H.J. van Ham, G.C. Angenent, and R.G.H. Immink. Sequence Motifs in MADS Transcription Factors Responsible for Specificity and Diversification of Protein-Protein Interaction. *PLoS Computational Biology*, 6(11):e1001017+, November 2010.

[VLS11] J. Vreeken, M. Leeuwen, and A. Siebes. Krimp: mining itemsets that compress. *Data Mining and Knowledge Discovery*, 23:169–214, July 2011.

[vMJS$^+$05] C. von Mering, L.J. Jensen, B. Snel, S.D. Hooper, M. Krupp, M. Foglierini, N. Jouffre, M.A. Huynen, and P. Bork. STRING: known and predicted proteinprotein associations, integrated and transferred across organisms. *Nucleic Acids Research*, 33(suppl 1):D433–D437, 2005.

[vMKS$^+$02] C. von Mering, R. Krause, B. Snel, M. Cornell, S.G. Oliver, S. Fields, and P. Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417(6887):399–403, 2002.

[vTV09] M. Šikić, S. Tomić, and K. Vlahoviček. Prediction of Protein-Protein Interaction Sites in Sequences and 3D Structures by Random Forests. *PLoS Computational Biology*, 5(1):e1000278, 2009.

[XJFD08] Y. Xiang, R. Jin, D. Fuhry, and F.F. Dragan. Succinct summarization of transactional databases: an overlapped hyperrectangle scheme. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 758–766, New York, NY, USA, 2008. ACM.

[YGN+10] J. Yu, M. Guo, C.J. Needham, Y. Huang, L. Cai, and D.R. Westhead. Simple sequence-based kernels do not predict protein-protein interactions. *Bioinformatics*, 26(20):2610–2614, 2010.

[YJ04] Y. Yang and T. Jack. Defining subdomains of the K domain important for protein-protein interactions of plant MADS proteins. *Plant Molecular Biology*, 55(1):45–59, 2004.

[YMSS04] Y. Yabuki, Y. Mukai, M.B. Swindells, and M. Suwa. GENIUS II: a high-throughput database system for linking ORFs in complete genomes to known protein three-dimensional structures. *Bioinformatics*, 20:596–598, March 2004.

[ZQ07] H.X. Zhou and S. Qin. Interaction-site prediction for protein complexes: a critical assessment. *Bioinformatics*, 23(17):2203–2209, September 2007.

[ZS01] H.X. Zhou and Y. Shan. Prediction of protein interaction sites from sequence profile and residue neighbor list. *Proteins*, 44(3):336–343, August 2001.

A

Software

The implementation in Java of our novel methodology as a tool is available on `http://bioinformatics.uhasselt.be`.

As the memory requirements for holding all possible motifs in memory are quite large, it is necessary to set the maximum memory for Java.

Example A.1. `java -Xms250M -Xmx2G -jar CorrelatedMotifs.jar`

will run the program. As no parameters are set, the program will output some help text.

◁

All parameters described in the rest of this Appendix are case insensitive.

A.1 Parameters

There are several parameters that are required for any run of the software:

- A method name (-m <methodName>, see Section A.2),
- ℓ, the length of the motifs (-l <integer>),
- d, the amount of wildcards (-d <integer>),
- a fasta-file with sequence data (-seq <filename>),
- a file containing the interaction information (-int <filename>), and

- a filename without extension for the output (-o <filename>).

Example A.2. `java -Xms250M -Xmx2G -jar CorrelatedMotifs.jar -m BruteForce -l 8 -d 3 -seq seq.fasta -int prot.int -o output`
would start a run of the brute force algorithm.

◁

For all methods except the brute force, a time in minutes (-min <number>) is required. CMC-greedy and CMC-approx use it as the amount of time allowed per iteration, for the other methods it is the total allotted time for the run. In both cases, the time will be divided among threads as appropriate.

Example A.3. `java -Xms250M -Xmx2G -jar CorrelatedMotifs.jar -m SeqSlider -min 600 -l 8 -d 3 -seq seq.fasta -int prot.int -o output`
would start a run of the SEQ-SLIDER algorithm for 10 hours.

◁

In addition there are several optional parameters. Some of these are very general and have default settings, which have been implicitly used in the previous examples:

- The amount of threads to be used (-t <integer>), which defaults to the amount of available processors.
- The support measure (-st <supportMeasure>), which default to the χ^2-based support measure. Different support measures will be enumerated in Section A.3.
- The amount of top scoring motif pairs to keep (-a <integer>), which defaults to 1 000.

There are numerous other optional parameters. Specifically, there are the parameters for bioSLIDER:

- Amino acid similarity (-simFile <filename>) reads the similarities between amino acids from the given file. When this parameter is used, the parameter to set the minimum similarity threshold (-var <number>) for motifs is also required. The parameter which sets the type of similarity (-varType avg or -varType product) is optional, as the algorithm defaults to the product type of similarity.
- The various types of data that indicate the likelihood of an amino acid to be part of a binding site, such as RSA values and conservation scores,

can be added using a single type of flag (-metaX <filename>, with X a number starting at 1). For each such flag, another flag (-metaValX <number>, with X the same) is required to set the minimum threshold. Optionally, confidence in the values can be added (-metaConfX <filename>), along with the corresponding minimum threshold for confidence (-metaConfValX <number>). Any type of data following the same format can be added using this flag.

- Domain data (-domain <filename>) can also be read from a file.

Example A.4.
```
java -Xms250M -Xmx2G -jar CorrelatedMotifs.jar
 -m SeqSlider -min 600 -l 8 -d 3 -seq seq.fasta -int prot.int
 -o output
 -simFile blosum62.txt -var 0.7 -varType product
 -meta1 conservation.txt -metaVal1 7.8
 -meta2 RSAprediction.txt -metaVal2 6.5
  -metaConf2 RSAconfidence.txt -metaConfVal2 5.0
 -domain domains.txt
```
is simply an extended version of a SEQ-SLIDER call.

◁

These are the parameters for using SEQ-SLIDER for triplets:

- Probabilities (-prob true) can be used instead of binary interaction information. This allows to indicate for each triplet if it is an interaction, a non-interaction or an unknown. Using probabilities allows for the use of the following parameters:
  - The positive value (-pos <number>, default 1.0) sets the probability for interactions.
  - The negative value (-neg <number>, default 0.0) sets the probability for non-interactions.
  - The unknown value (-unknown <number>, default 0.5) sets the probability for unknowns.
- You can set the order triplets are interpreted in. There are flags for fully ordered (X, Y, Z) (-ordered true) or unordered $\{X, Y, Z\}$ triplets (-unordered true), otherwise it defaults to the semi-ordered $(\{X, Y\}, Z)$ triplet case.

SEQ-SLIDER for triplets only works with the χ^2-based support measure, so this is automatically selected. If the probabilities option is selected, it automatically uses the adapted χ^2-based support measure.

Example A.5. `java -Xms250M -Xmx2G -jar CorrelatedMotifs.jar`

```
 -m SliderTriplets -min 600 -l 8 -d 3
 -seq seq.fasta -int prot.int -o output
 -prob true -pos 0.9 -unknown 0.05 -ordered true
```

Is a potential call for SEQ-SLIDER for triplets.

◁

And then there are a few remaining optional parameters:

- Read the motif data from a file (-motif <filename>), created with the saveMotifData method (see Section A.2), instead of parsing it from the sequences.
- It is also possible to skip the preprocessing step entirely (-reading true). In this case, no data of which motif occurs in which proteins is saved, and this information is recalculated during the running of the algorithm.
- There are several different neighbor functions (-f <neighborFunction>) for the steepest ascent method. The different choices are explain in Section A.4.

The order in which parameters are given is irrelevant.

A.2 Methods

The package allows the execution of all the algorithms described in this work. To indicate what method to run, a flag (-m <methodName>) is added to the call followed by the parameter in parentheses behind each of these algorithms:

- M-SLIDER (MSlider),
- SEQ-SLIDER (SeqSlider),
- CMC-greedy (CMCgreedy),
- CMC-approx (CMCapprox),
- bioSLIDER (SeqSlider, see Section A.1),
- SEQ-SLIDER for triplets (SliderTriplets),
- a brute force algorithm (BruteForce), and
- a steepest ascent algorithm (SteepestAscent).

There is one additional method (saveMotifData) possible, which for one network and its settings for motifs, such as those for bioSLIDER, loads all possible motifs and their occurrences in memory and then saves these to a file. This file can then later be loaded for use by any of the other methods to save time on the preprocessing step.

A.3 Support measures

It is possible to use the different support measures explained in this work (-st <supportMeasure>):

- The χ^2-based support measure,
- the p-score (p),
- f_c (cover),
- f_v (weightedv).

When no support measure is specifically selected, it defaults to the χ^2-based support measure.

A.4 Neighbor functions

If the steepest ascent method (SteepestAscent) is selected, there are multiple choices for neighbor function (-f <neighborFunction>). Their function is described in Section 2.5.1 and Section 2.8.3:

- Letter Change (LC),
- Swap Adjacent (SA),
- Letter Change and Swap (LCandS),
- Letter Change or Swap Adjacent (LCorSA),
- Letter Change and Swap Adjacent (LCandSA),
- N^{mot} (Slide),
- $N^{\mathrm{mot}}_{\ominus}$ (SimpleSlide),
- All (All) is the neighbor function implicitly used by MotifHeuristics, which fixes one motif and attempts to pair it with all possible motifs.

A.5 File formats

Our algorithms take several files of several types as input. Now follows a quick overview of their necessary formats.

The sequence information (-seq <filename>) should be in a file in the FASTA-format.

Example A.6.

```
>ID1
ERLEELEKKEAQLTVTNDQIHILKKENELLHF
>ID2
EYVKCLENRVAVLENQNKTLIEELKTLKDLYSNKSV
>ID3
VWVQSLEKKAEDLSSLNGQLQSEVTLLRNEVAQLKQLLLAHKDC
```

◁

In the interaction file (-int <filename>), each line represents an interaction. The two, or three, id's, from the FASTA-file, are separated by a space.

Example A.7.

```
ID1 ID2
ID1 ID3
```

◁

In the case of triplets, a + or − can optionally be added at the end of the lines, also separated by a space, to indicate interaction and non-interaction. In that case, triplets not in the file will be considered unknowns.

Example A.8.

```
ID1 ID2 ID3 +
ID3 ID1 ID2 -
```

◁

The file which indicates the similarities between amino acids (-simFile <filename>) should adhere to the following format. The first line should start with a tab and then contain a tab-separated list of all amino acids. Then each following line contains an amino acid and a tab-separated list of similarities to the amino acids in the order of the first list. The similarities are given as percentage values going from 0.0 to 100.0.

Example A.9.

```
     C     S     T
C    100.0 23.1  23.1
S    28.6  100.0 57.1
T    28.6  57.1  100.0
```

is a partial similarity matrix.

◁

The biological data for bioSLIDER (-metaX <filename> or -metaConfX <filename>) is to be given in a format similar to FASTA, only instead of amino acids, we have the corresponding values.

Example A.10.

```
>ID1
24799779597999455777675977579963
>ID2
777576466877667757996575777757316534
>ID3
34432777893532221321212144746334594388515845
```

◁

The domain data (-domain <filename>) is to be given as follows. There is a line for each domain present, containing the protein-id, the domain id, and the start and end position of the domain in the amino acid sequence. These are tab-separated, and the positions are 1-indexed.

Example A.11.

```
ID1    DOMID1    9     59
ID1    DOMID2    77    171
ID2    DOMID1    9     59
ID3    DOMID3    93    172
```

◁

B

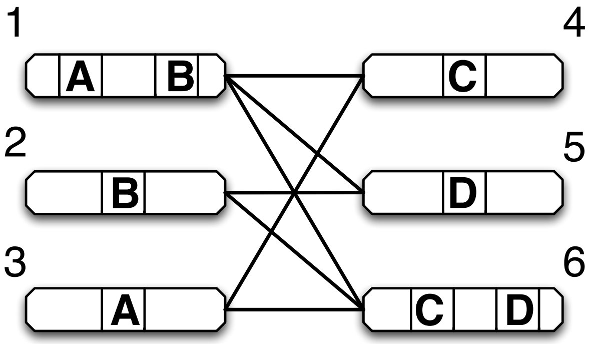
Samenvatting

B.1 Motivatie

Proteïnen zijn de belangrijkste onderdelen van alle levende organismes. Samen vervullen ze de basisfuncties van levende cellen. Om hun functies te vervullen, moeten proteïnen vaak samenwerken door aan elkaar te binden, waardoor ze een complex vormen. We zeggen dan dat deze proteïnen interageren, of dat ze een interactie hebben. Als we verschillende interacties van de proteïnen binnen eenzelfde organisme verzamelen, vormen ze samen een proteïne-proteïne interactienetwerk. Voor verschillende organismes zijn er grootschalige biologische netwerken beschikbaar, die een steeds groeiend aantal van deze interacties omvatten [STdS$^+$08]. Deze netwerken geven aan hoe een proteïne functioneert als deel van het netwerk, maar bieden geen inzicht in hoe de interactie gecodeerd is in de sequenties van aminozuren waaruit proteïnen opgebouwd zijn.

Interacties vinden plaats door middel van chemische verbindingen tussen delen van de sequentie van beide proteïnen. Aminozuren die betrokken zijn bij interacties noemen we bindresidu's of interactieresidu's, en samen vormen ze een interactiesite. Kennis van interactiesites is essentieel voor het voorspellen van tot nog toe ongekende proteïne interacties, het begrijpen van de evolutie van proteïne interacties, of het aanmaken van geneesmiddelen die een specifiek proteïne, en op die manier een specifieke functie, willen beïnvloeden.

Spijtig genoeg zijn er om deze interactiesites te ontdekken uitgebreide en kostelijke biologische experimenten nodig. Gebaseerd op het huidige tempo waarmee proteïnestructuren bepaald worden, wordt geschat dat er nog 20 jaar



Figuur B.1: Overeenkomstige interactiesites {A, C} en {B, D}, weergegeven als gecorreleerde motieven in sequenties.

nodig zal zijn om alle interactietypes te bepalen met behulp van de huidige experimentele technieken [AR04]. Bovendien zou men, zelfs als dit bereikt is, nog altijd voor elke interactie moeten voorspellen van welk type ze is.

Daarom hebben we een nieuwe methode ontwikkeld om mogelijke interactiesites computationeel te ontdekken. Een implementatie hiervan wordt besproken in Appendix A.

B.2 Probleemstelling

Er zijn verschillende computationele methoden voorgesteld om interactiesites op te sporen door middel van het *minen* van paren van patronen, motieven genaamd, die vaker voorkomen in de sequentie dan verwacht [LSY+07, LLW06, LLLW07, LSLW08, THSN06]. Correlated motif mining (CMM) is een methode om interactiesites te vinden door een consensuspatroon te zoeken in een groep proteïnen waarvan (bijna) alle proteïnen interageren met (bijna) alle proteïnen uit een groep die aan een ander consensuspatroon voldoet. Als we dergelijke patronen vinden, stellen deze waarschijnlijk een deel van het proteïneoppervlak voor dat interacties mogelijk maakt. Bijvoorbeeld, in Figuur B.1 stellen de patronen {A, C} en {B, D} dergelijke gecorreleerde motieven voor. Er is een interactie tussen twee proteïnen als de ene het motief A bevat, en de andere het motief C (gelijkaardig voor motieven B en D). Ondanks de ontwikkeling van verschillende algoritmes is het niet duidelijk hoeveel interactietypes beschreven kunnen worden door gecorreleerde motieven, maar de resultaten van deze methoden tonen wel aan dat gecorreleerde motieven informatie over interactietypes bevatten [LSY+07, LLW06, LLLW07, LSLW08, THSN06].

We merken dat verschillende hoog scorende motiefparen vaak naar zeer gelijkaardige subnetwerken verwijzen. Daardoor worden door methodes voor CMM enkel interactiesites gevonden in een klein deel van het interactienetwerk. Daarom onderzoeken we ook een andere methode waarbij we de motiefparen niet op individuele basis beschouwen, maar we de beschrijvende kracht van een verzameling van motiefparen evalueren. We noemen dit het Correlated Motif Covering (CMC) probleem. In essentie zoeken we een zo klein mogelijke verzameling motiefparen die een zo groot mogelijk deel van het netwerk dekt.

B.3 Bijdragen

Onze eerste bijdrage is een grondig onderzoek naar verschillende mogelijke scorefuncties voor gecorreleerde motieven. We vergelijken de *precision* en *recall* op artificiële netwerken waarin motieven ingeplant zijn met verschillende niveaus van ruis. Deze experimenten tonen aan dat de χ^2-scorefunctie duidelijk beter is dan de alternatieven in het vinden van beschrijvende motiefparen.

Vervolgens bewijzen we, onder redelijke aannames over de scorefunctie, dat het Correlated Motif Mining (CMM) probleem NP-hard is en dat het overeenkomstige beslissingsprobleem NP-compleet is. Daarom benaderen we het probleem als een combinatorisch beslissingsprobleem. We stellen de metaheuristiek SLIDER voor, die twee *steepest ascent* methoden bevat. Deze methoden zijn gebaseerd op het idee dat een motief dat in de buurt van een binding site ligt, er stap voor stap naartoe moet kunnen glijden.

De vergelijking van SLIDER met andere bestaande methoden toont aan dat SLIDER er beter in slaagt om motiefparen terug te vinden die in artificıele netwerken ingeplant zijn. De experimenten tonen ook aan dat SLIDER grootschalige netwerken aan kan.

We tonen aan dat het Correlated Motif Covering (CMC) probleem NP-hard is en dat het tot een klasse van problemen behoort waarvoor het ook moeilijk is om een “goed” benaderingsalgoritme te vinden. Daarom introduceren we de heuristiek CMC-approx die gebaseerd is op een benaderingsalgoritme van Peleg [Pel07] voor het Weighted Red-Blue Set Cover (WRBSC) probleem.

We vergelijken CMC-approx experimenteel met SEQ-SLIDER en een *greedy* algoritme, dat we CMC-greedy noemen, op biologische netwerken. De dekking van het netwerk door de resultaten van CMC-approx is veel hoger, maar heeft meer tijd nodig. De resultaten van CMC-approx zorgen ook voor een iets betere voorspelling van nieuwe interacties. Wanneer we de overlap tussen de gevonden motiefparen en echte interactiesites bekijken in netwerken met proteïnen waarvoor de 3D-structuur bekend is, merken we dat de resultaten van CMC-approx een drastisch grotere dekking hebben dan die van de andere methoden, ten kost van slechts een licht verlaagde precisie. De verhoogde dekking verhoogt het nut voor experimentele biologen enorm aangezien zij duidelijk interactiesites willen voorspellen voor zo veel mogelijk proteïnen.

Ten slotte stellen we dat we de resultaten van verschillende organismes kunnen gebruiken om de nauwkeurigheid te verhogen. Onze experimenten met proteïnen die in meerdere organismes voorkomen, tonen aan dat, als een deel van een sequentie teruggegeven wordt door onze methoden bij meerdere soorten, het ook inderdaad een hogere kans heeft om effectief een interactiesite te zijn.

De gebruikte data en de java-implementatie van SLIDER en CMC-approx bevinden zich op `http://bioinformatics.uhasselt.be`.