

ON INTERACTION MOTIF INFERENCE FROM  
BIOMOLECULAR INTERACTIONS: RIDING THE  
GROWTH OF THE HIGH THROUGHPUT  
SEQUENTIAL AND STRUCTURAL DATA

HUGO WILLY

NATIONAL UNIVERSITY OF SINGAPORE

2010

ON INTERACTION MOTIF INFERENCE FROM  
BIOMOLECULAR INTERACTIONS: RIDING THE  
GROWTH OF THE HIGH THROUGHPUT  
SEQUENTIAL AND STRUCTURAL DATA

HUGO WILLY  
*B. Comp. (Hons.), NUS*

A THESIS SUBMITTED FOR THE DEGREE OF  
DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE  
NATIONAL UNIVERSITY OF SINGAPORE

2010



# Summary

Biochemical processes in the cell are mostly facilitated by (bio)catalysts commonly known as the enzymes. They have remarkable catalytic properties that enable a vast variety of chemical reaction to occur at high rates and specificity. There are currently two biomolecules that are known to act as enzymes in the cell; the protein and the RNA. The enzymatic property of these two are achieved by their ability to fold into a huge number of possible shapes and structures.

RNA can act as a messenger which passes information from DNA to protein. However, some RNA do not code for protein—collectively these are called the non-coding RNA. They instead catalyze cellular reactions much like proteins do. The base of RNA's catalytic ability is that RNA could form myriads of possible structures through self hybridization. Such structural RNA can be seen in the ribosome, the organelle responsible of translating the genetic code in the messenger RNA into proteins. Non-coding RNA are also involved in many other important cell processes, mostly related to gene transcription and translation processes, like mRNA splicing, gene expression regulation and chromosomal regulation.

The protein is the cellular workhorse. They function as enzymes, provide structural support, involved in cellular defense, transport biomolecules into and out of the cell, and, regulate the production of themselves or other proteins. In order to accomplish these functions, proteins often works together with another protein or RNA by forming a complex.

One interesting question is how do protein and RNA recognize their correct interaction partners? Based on our current understanding, they recognize a pattern, a motif, on the surface of its partner which it can specifically bind to. To bind those patterns, the protein or the RNA itself has a conserved region dedicated to recognition. We call these conserved patterns which are involved in the interaction between two biomolecules as the *interaction motif*. These patterns mostly form complementarily shaped surface areas within the two biomolecules. More often than not, the surface would also have complementary charge/chemical properties; ensuring strong and highly specific binding. From an evolutionary point of view, the interaction motif is under pressure to be con-

served so long as the interaction they mediate is crucial to the organism's survival. Such conservation mean, given enough data, one should be able to design a computational technique to recognize these patterns. This thesis presents a study on the interaction motifs underlying the interaction of RNA and protein with their partners and proposes several methods to discover them.

For RNA, it is known that the structure/shape of the RNA is generally more conserved than the sequence. One important example is the transfer RNA (tRNA) that exists in virtually all living organisms. All tRNA unfailingly exhibit the clover-leaf shaped structure while some of them have a low overall RNA sequence similarity (less than 50% similarity). One way to describe the structure of RNA is by describing the RNA's set of base pairings, that is, its secondary structure. We present an algorithm to infer RNA secondary structure of an RNA sequence given a known structure. We improved the current best method in terms of computational time and space complexity. These improvements are important as more non-coding RNA transcripts from different organisms will be sequenced by the most recent second generation nucleic acid sequencing technology. The space complexity improvement is also important because a group of longer non-coding RNA has also been identified. At the same time, the number of reference RNA structures in the Structural Database like the Protein Data Bank is steadily increasing over the years and we expect more structures will be available soon given the importance of the non-coding RNA.

On protein interaction motifs, many protein-protein interactions are known to be mediated by the binding of two large globular domain interfaces (domain-domain interactions). However, there also exists a class of transient interactions typically involving the binding of a protein domain to a short stretch (3 to 20) of amino acid residues which is usually characterized by a simple sequence pattern, i.e. *a short linear motif (SLiM)*. SLiMs are involved in important cellular processes like the signaling pathways, protein transport and post translational modifications.

We designed two programs, D-STAR and D-SLIMMER, to mine SLiMs from the current protein-protein interaction (PPI) data. Both programs are based on the concept of correlated motif, which basically state that a pair of (interaction) motif that enables interaction will have a significantly higher number of interaction between the proteins containing them. We show that our correlated motif approach, which is interaction

based, is more suitable for mining SLiMs from the PPI data. D-STAR was the pioneer program which used the correlated motif concept to find SLiMs from PPI data (earlier work was done on correlation between known protein domains). We showed that D-STAR is capable to find real biologically relevant SLiMs from the SH3 domain and TGF $\beta$  PPI data. We further improved D-STAR by designing D-SLIMMER. D-SLIMMER uses a mix of non-linear (protein domain) and linear (SLiM) interaction motif as correlated motifs. This important difference enables D-SLIMMER to outperform D-STAR and other programs like MotifCluster and SLIDER.

D-SLIMMER also proposes two possible novel SLiMs related to the Sir2 and SET domain respectively. The first SLiM is a acetylated lysine (K) motif, **AK.V.I** (K must be acetylated for recognition) which is correlated with a family of deacetylase proteins, Sir2. The second is a target of the SET methyltransferase family, **SK.KK..H** (the bold K is the methylation target). Both SLiMs have important implications in Histone modification and chromosomal regulation in general and we present supporting literature and structural evidences to show that the novel SLiMs are biologically viable. Given the significant growth of the protein-protein interaction data in the recent years, we expect that D-SLIMMER and other programs in this line would be of high importance for mining more SLiMs from the PPI data.

We designed another method, SLiMDiet, which collects all possible de-novo SLiMs from the structural data in the PDB database. We characterized 452 distinct SLiMs from the Protein Data Bank (PDB), of which 155 are validated by either literature validations or over-representation in high throughput PPI data. We further observed that the lacklustre coverage of existing computational SLiM detection methods could be due to the common assumption that most SLiMs occur outside globular domain regions. 198 of 452 SLiM that we reported are actually found on domain-domain interface; some of them are implicated in autoimmune and neurodegenerative diseases. We suggest that these SLiMs could be useful for designing inhibitors against the pathogenic protein complexes underlying these diseases. Our findings show that 3D structure-based SLiM detection algorithms can strongly complement current sequence-based SLiM mining approaches by providing a more complete coverage on the SLiMs on domain-domain interaction interfaces. Further experimental works is needed to validate the correctness of D-SLIMMER's and SLiMDiet's predicted SLiMs and we leave these as future works.



# Acknowledgement

I am deeply thankful to my supervisor Dr. Sung Wing Kin who have been patiently guiding me through my PhD years. His passion and dedication towards the work of research strongly inspires many people who work with him and I am privileged to have him as my mentor. I thank him for his strict requirement on my research results while being very supportive and helpful on all other things that I need. He made sure that I can focus on my study without needing to worry about other matters. I hope I could one day become a good teacher, a good researcher like him.

I am truly grateful to Dr. Ng See Kiong, my co-supervisor, who had given much support and direction during my early research years. There were many times when my work seems to meet a dead-end and he would give a good and clear overview on our situation and suggest yet another approach to attempt. I also admire his exceptional writing skill which I have yet to master even now.

In the middle of my PhD years, I started to move deeper into the field of Biology. The transition was not an easy one and I am fortunate to have worked with Dr. Tan Soon Heng in the second project presented in this paper. My contribution is on the program design; the biological problem formulation and the biological validations was designed by him. During the work, I learnt more about the biological side of the field of Bioinformatics especially on validating the computational results using the biological literature. The skill helped me a lot in the subsequent projects that I did and I am indebted to him for that.

I also wish to thank many friends and colleagues in the Computational Biology Lab for their interesting discussion and warm friendship. Huge thanks to Song Fushan who had worked so hard in the SLiMDiet project that we finally got a good publication for it. Also not forgetting my great "corner" friends who provided me great company and much entertainment during many sleepless nights of my paper deadlines. I thank the management staffs of School of Computing who had been helping me with many of the (tedious) paperworks involving my PhD study.

I wish to thank my parents who have supported me to pursue my own interest in research; to have loved and nurtured me from the very day I am born until now. To my



dearest sisters, thank you for taking care of our parents while I am away. I wish to give a special thanks to my love, Sun Lu, who has been on my side, giving unfailing support through my difficult times. Thank you so much for being there all this time.

My PhD study has been a prolonged one. Had it not been for my two supervisors' trust and guidance; had it not been for the help and support I received from so many wonderful people around me, I honestly doubt I could have accomplished my study. I truly thank you for all you have done for me.

Thank you.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	RNA and Protein: The two catalysts of the living cell . . . . .	1
1.2	Interaction motif . . . . .	2
1.3	RNA Secondary Structure . . . . .	3
1.3.1	Current approaches on finding RNA secondary structure . . . . .	4
1.3.2	Our contribution . . . . .	5
1.4	Protein-Protein Interaction Motif . . . . .	6
1.4.1	Existing computational methods on SLiM mining . . . . .	6
1.4.2	Our contributions . . . . .	7
1.5	Thesis organization . . . . .	10
<b>2</b>	<b>Background</b>	<b>11</b>
2.1	RNA: Ribonucleic acid . . . . .	11
2.1.1	The non-coding RNA . . . . .	12
2.1.2	RNA Secondary Structure in non-coding RNA . . . . .	15
2.1.3	Current RNA secondary structure data . . . . .	16
2.2	The proteins . . . . .	16
2.2.1	Protein-Protein Interaction Motif . . . . .	18
2.2.2	Protein Short Linear Motifs (SLiMs) . . . . .	20
2.2.3	The availability of the PPI and Protein Structural Data . . . . .	22
<b>3</b>	<b>Discovering Interacting Motifs in RNA: Predicting the RNA Secondary Structure</b>	<b>23</b>
3.1	Introduction . . . . .	23
3.2	Existing Method . . . . .	26

3.2.1	Preliminaries . . . . .	26
3.2.2	Algorithm Description . . . . .	28
3.3	Our Algorithm’s Description and Analysis . . . . .	30
3.3.1	Running Time Improvement through Sparsification on the Dynamic Programming . . . . .	30
3.3.2	Using Less Space in the Computation of the WLCS Score . . . . .	40
3.3.3	Tackling Both the Time and Space Complexity Bound: a Hirschberg-like Traceback Algorithm . . . . .	43
3.4	Conclusion . . . . .	50
3.5	List of publication . . . . .	50
<b>4</b>	<b>Discovering Interaction Motifs from Protein-Protein Interaction Data:</b>	
	<b>D-STAR</b>	<b>51</b>
4.1	Introduction . . . . .	51
4.2	Related works . . . . .	55
4.3	Methods . . . . .	55
4.3.1	Preliminaries . . . . .	55
4.3.2	Methods . . . . .	57
4.4	Results and discussion . . . . .	63
4.4.1	Artificial data with planted $(l, d)$ -motifs . . . . .	63
4.4.2	Biological data . . . . .	67
4.5	Conclusions . . . . .	74
4.6	List of publication . . . . .	76
<b>5</b>	<b>Discovering Interaction Motifs from Protein-Protein Interaction Data:</b>	
	<b>D-SLIMMER</b>	<b>77</b>
5.1	Introduction . . . . .	77
5.2	Materials and Methods . . . . .	80
5.2.1	Overview of the D-SLIMMER algorithm . . . . .	80
5.2.2	Preliminaries . . . . .	80
5.2.3	Mining SLiMs from each target domain’s PPIs . . . . .	81
5.2.4	Removing redundant $(L,W)$ -motif occurrences . . . . .	82

5.2.5	Filtering randomly occurring SLiMs using a 3 <sup>rd</sup> order markov chain background. . . . .	83
5.2.6	Scoring domain-SLiM interaction density: the chi-square function	84
5.2.7	Removing domain-SLiM redundancies . . . . .	85
5.3	Results and Discussion . . . . .	85
5.3.1	Comparative study between D-SLIMMER and existing methods	85
5.3.2	Scoring function analysis: Occurrence frequency vs. interaction density . . . . .	88
5.3.3	Biologically interesting SLiMs reported by D-SLIMMER . . . . .	89
5.4	Conclusions . . . . .	98
5.5	List of publication . . . . .	99

## **6 Discovering Interaction Motifs from Protein Structural Data: SLiMDiet 100**

6.1	Introduction . . . . .	100
6.2	Methods . . . . .	103
6.2.1	SLiMDiet's workflow . . . . .	103
6.2.2	Domain identification . . . . .	103
6.2.3	Interface extraction . . . . .	105
6.2.4	Pairwise structural alignment within each domain interface group	105
6.2.5	Hierarchical agglomerative clustering on the domain interfaces .	106
6.2.6	Quantification of the clustering performance . . . . .	107
6.2.7	SLiM extraction from the interface clusters . . . . .	107
6.2.8	Computing the statistical significance of the SLiM using PPI data	111
6.2.9	Computing the statistical significance of domain-domain SLiM .	112
6.3	Results . . . . .	114
6.3.1	Both known and novel SLiMs are discovered . . . . .	114
6.3.2	SLiMs with validations from the literature . . . . .	115
6.4	Discussion . . . . .	115
6.4.1	Different SLiM classes have different interface geometries . . . .	115
6.4.2	Known and Novel SLiMs are found on domain-domain interfaces	118
6.5	Conclusion . . . . .	123
6.6	List of publication . . . . .	124

**7 Conclusion** **125**  
7.1 Possible future works . . . . . 126

# List of Tables

5.1	Performance comparison between D-SLIMMER, MotifCluster, SLIDER and SLiMFinder. This table shows the best rank of each method’s detected SLiMs containing a reference SLiM for a domain. The best rank is chosen among all different species’ including the combined species dataset. Ties are resolved by reporting the median rank of the motifs sharing the same score. “–” is listed when a method has not detected any SLiM containing the reference SLiM within its top-50 SLiMs.	87
6.1	The benchmark interfaces and their classification based on the literature reference.	116
6.2	Clustering performance comparison of SLiMDiet and SCOWLP. We collected the interfaces of the SH2, SH3 and 14-3-3 domains whose domain-SLiM interaction class is defined in their respective reference papers. The grouping from the literature constitutes the reference clusters, against which the accuracy of both SLiMDiet and SCOWLP are computed. The cases where one method outperforms the other are printed in bold.	118

# List of Figures

2.1	The structure of RNA and its nitrogen bases . . . . .	12
2.2	The secondary structure of RNA. This figure is adapted from Molecular Biology of the Cell, 5E, © 2002, by permission of Garland Science LLC. Reproduced by permission of Garland Science/Taylor and Francis LLC. . . . .	13
2.3	The tertiary structure of RNA. This figure is adapted from Molecular Biology of the Cell, 5E, © 2002, by permission of Garland Science LLC. Reproduced by permission of Garland Science/Taylor and Francis LLC. . . . .	13
2.4	The secondary and tertiary structure of the transfer RNA (tRNA). The clover-like secondary structure is conserved in all domains of life. Some of the nucleotides are post-processed into a non-canonical nucleotides (T stands for Ribothymidine, $\psi$ for pseudouridine and the nucleotides with an 'm' sign are methylated in their ribose sugar). These figures are taken from the Wikimedia Commons.	14
2.5	Two examples of non-coding RNA secondary structure motifs. (A) The secondary structure of ATPC RNA motif conserved in certain cyanobacteria (RFAM ID:RF01067). We can see from the coloring that the sequence conservation of this structure is rather weak. (B) The structure of invasion gene associated RNA (also known as InvR). This is a small non-coding RNA involved in regulating one of the major outer cell membrane porin proteins in Salmonella species (RFAM ID:RF01384). The figures are taken from the RFAM database [1]. . . . .	15
2.6	(A) The 20 side chains of the known amino acids. (B) The diagram illustrates the atomic configuration of an amino acid. The same backbone atoms are used in all amino acids and the R part is where the different side chains are attached. These figures are taken from the Wikimedia Commons. . . . .	18

2.7	The illustrations of protein’s primary, secondary, tertiary and quaternary structures. This figure is taken from the Wikimedia Commons. . . . .	19
2.8	(A) A domain-domain interface and (B) a domain-SLiM interface. We can see that the SLiM (shown in sticks) is in an extended linear conformation while the domain surface ”wraps” around it. We also observe that the size of the interface is significantly larger for domain-domain as compared to domain-SLiM interface. This figure is generated by PyMOL [2]. . . . .	20
3.1	The algorithm from [3] described in terms of EXTEND, MERGE and ARC-MATCH operations. The two arc-annotated sequences $S_1$ and $S_2$ are of length $n$ and $m$ , respectively. $P_1$ is the arc-annotation of $S_1$ ; given the nested arc-annotation, the maximum number of arcs in $P_1$ are bounded by $O(n)$ . For any arc $u \in P_1$ , $u_l$ is its left endpoint and $u_r$ is its right endpoint. . . . .	30
3.2	Illustration of the set $S$ . The distinct scores in each row are highlighted in grey. From the figure we can see that $RowIP_{(i,i';2,8)} = \{2, 3, 5, 6, 7, 8\}$ ( $j = 2, j' = 8$ ). Then, as defined, we have $S_{(i,i',i'';2,8)} = \{3, 5, 6, 7, 8\}$ since $j' = 8$ and, for all $j^* \in \{3, 5, 6, 7\}$ , we have 8 inside the set $RowIP_{(i'+1,i'';j^*+1,8)}$ . . . . .	33
3.3	The pseudocode for the new MERGE operation. We have two DP tables, $DP_{(i,u_l-1)}$ is the currently computed DP table and $DP_{(u_l,u_r)}$ is the DP table of the arc $u$ we wish to combine into the former to compute the merged $DP_{(i,u_r)}$ . . . . .	35
3.4	The core-path $CP(c_1)$ is the ordered set $\{c_1, c_2, c_3\}$ . . . . .	36
3.5	An example of arc-annotation on which the algorithm in [3] requires $\Omega(nm^2)$ space to compute the <i>score-only</i> $WLCS(S_1, P_1, S_2)$ . Note that the <i>post-ordering</i> forces the algorithm to compute the DPs for all the leaves before the internal nodes. . . . .	41
3.6	The recursion on the partitioned continuous region by Lemma 3.3.14. The recursive call on the inner region is exactly the same as the the previous recursive level. The call on the outer region have a requirement that the concatenation point be aligned to each other. . . . .	44
3.7	The figure describes the partitioning of $S_1$ for the case where $g > c_r$ . For the sake of clarity, the regions are drawn connected to each other. Note that, actually, the regions $R_1, R_2, R_3$ and $R_4$ are disjoint (not sharing their endpoints). . . . .	47



4.1	A depiction of our approach for finding correlated motifs. The dotted lines indicates the interactions between the proteins. . . . .	53
4.2	The D-MOTIF-BASIC algorithm. $(s_i, s_j)$ is a pair of interacting protein from the PPI dataset $I$ . $s_i[u]$ ( $s_j[v]$ , resp.) is the length $l$ substring starting at position $u$ ( $v$ , resp.) in $s_i$ ( $s_j$ , resp.). $X_{s_i[u]}$ ( $X_{s_j[v]}$ , resp.) is the set of all length $l$ string which have at most $d$ mismatches with $s_i[u]$ ( $s_j[v]$ , resp.). The set $S_d(p)$ ( $S_d(p')$ , resp.) is the set of all proteins containing at least one length $l$ substring which has at most $d$ mismatches with $p$ ( $p'$ , resp.). The subset of $I$ containing the interactions between proteins in $S_d(p)$ and $S_d(p')$ is denoted as $I(p, p')$ . The set $S'_d(p)$ is the subset of $S_d(p)$ which has an interaction with another protein in $S'_d(p')$ given the interaction set $I(p, p')$ . $k_n$ and $k_i$ are minimum size of the interacting protein set and interaction set, respectively. $\chi(S_d(p), S_d(p'))$ is the chi-score computed for the pair $(p, p')$ . . . . .	58
4.3	The D-MOTIF algorithm. $X_{(s_i[u], s_j[v], s_k[w])}$ is a short notation for $X_{s_i[u]} \cap X_{s_j[v]} \cap X_{s_k[w]}$ . The algorithm's speed up is achieved by only considering $l$ substrings which have at least three other substrings with at most $d$ mismatches from it. .	59
4.4	The D-STAR algorithm. . . . .	60
4.5	Comparison of running time between D-MOTIF and D-STAR We observe that the running time of D-MOTIF increases rapidly as the input data grows and also as the $(l, d)$ -motif gets weaker. Experiments were run on a x86 Pentium 4 1.6GHz machine with 512MB of memory. . . . .	62
4.6	Comparison on specificity and sensitivity between D-MOTIF and D-STAR. This table shows that D-STAR runs orders of magnitude faster than D-MOTIF while sacrificing a small amount of accuracy in terms of sensitivity and specificity. .	62
4.7	Comparison between D-STAR and S-STAR(A variant of SP-STAR) in extracting planted $(l, d)$ -motifs. The motifs are arranged on the x-axis in decreasing order of motif strength. The number of planted motif instances in each dataset is 5 and the datapoint is the average over 10 runs. . . . .	66
4.8	Rank of sequence segment sets or sequence segment pair sets output by the various algorithms that express various known binding motifs of SH3 domains. ”-” denote the biological motif is not expressed within the top 50 sequence segment sets. . . . .	67

4.9	The P..P, P..P.[KR] and [KR]..P..P motifs and their associated motifs extracted by D-STAR. Lines between the sequence segments denote interaction between their parent proteins. The result is found from multiple runs of D-STAR with different combination of motif width $l = 6, 7, 8$ , distance $d = 1$ and $k_i = k_n = 5$ . We then rank all the outputs from the different runs by their $\chi$ -score. . . . .	69
4.10	Evidence from PDB structural data - SH3 domain vs P..P.R. The figure illustrates the 3D structure of a SH3 domain of FYN tyrosine kinase (PDB ID: 1AVZ) bound to with another protein. The sequence segments that express the P..P.R motif and G..P.NY motif (detected by D-STAR in this work) are highlighted in dark blue and orange respectively. The two segments correspond to actual interacting subsequences. . . . .	71
4.11	The best motif pair found in TGF $\beta$ . The highlighted proteins on the left belongs to the Kinase domain while those on the right contain the Kinase phosphorylation motifs (as checked by another program PhosphoMotif Finder [4]) . . . . .	73
4.12	The list of motifs of the phosphorylation sites that are over-represented in the segment set with the general pattern GKT[CIS][ILT][IL]. . . . .	74
4.13	The odd-ratio of known Kinase phosphorylation motifs found in D-STAR's motif pair. As the motifs are degenerate, we compared their actual number of occurrence with their expected random occurrence within any random segment set of the same size preserving the same amino acid distribution as the whole dataset's. . . . .	75
5.1	The flowchart of D-SLIMMER algorithm. . . . .	81
5.2	$P(D)$ ( $P(M)$ , respectively) is the set of protein containing domain $D$ (motif $M$ , respectively). $I(D, M)$ is the subset of the PPI data $I$ where one protein of the interaction contains the domain $D$ while the contains $M$ . $P(M I(D, M))$ is the subset of $P(M)$ which is involved in $I(D, M)$ . . . . .	82

5.3	(A) The PPI corresponding to EF hand domain and the SLiM A..IQ..WR found from the combined PPI data of BioGRID. The source organism are indicated in the protein names. The instances of A..IQ..WR are listed along with their position in their respective protein sequences. Among the 13 proteins with A..IQ..WR, 7 of them (the IQ motif sites are marked with asterisks (★)) are annotated to have IQ motif at the site of the SLiM by UNIPROT. Another 4 are annotated to have the Pfam domain regions of IQ motif (Pfam ID: PF00612) which describe EF hand binding sites (marked with +). The remaining two proteins are also annotated to have the IQ motifs at the occurrence site of A..IQ..WR [5]. (B) A similar IQ motif is also found in the BioGRID PPI dataset of <i>D. melanogaster</i> . The SLiM is AT..IQ...R which, upon inspection on the position directly before the last R, is actually AT..IQ..[FWY]R. Combining the (A) and (B) gave us the SLiM A..IQ..[FWY]R for Calmodulin. . . . .	91
5.4	The sequence alignment of 5 human KCNQ along with <i>D. melanogaster</i> 's KCNQ protein Q5PXF9 indicates that their IQ motif instances also missed the last position of the ELM's IQ motif (..[SACLIVTM]..[ILVMFCT]Q...[RK].{4, 5}[ <b>RKQ</b> ])—the matching positions for the [RKQ] residue are underlined. . . . .	92
5.5	(Top) The PPI corresponding to RB_B domain and the SLiM EG..DLFD. The instances of the SLiM (highlighted in bold) also match correctly against a known ELM SLiM ..[LIMV]..[LM][FY]D which is related to the RB_B domain (ELM: LIG_Rb_pABgroove.1). (Bottom) The sequence alignment of the C-terminal area of the target E2F proteins indicates that the SLiM region is highly conserved as compared to its neighboring positions. . . . .	93
5.6	The PPI corresponding to GYF domain and the SLiM PPPGL. . . . .	94
5.7	The PPI between 8 Sir2 proteins and 10 proteins containing the SLiM AK.V.I. The K is the predicted acetyllysine position. The SLiM AK.V.I fulfils the requirement of having an aliphatic residue at position +2 w.r.t the acetyllysine in [6]. . . . .	95

5.8	Location of the SLiM AK.V.I in Glyceraldehyde-3-phosphate dehydrogenase proteins. The left picture shows that the predicted acetyllysine position is pointing outward of the protein (PDB ID:2I5P). On the right, we show that both the dimeric (PDB ID:2I5P) and tetrameric complexes (PDB ID:2VYN) present the SLiM region (circled) at their outer peripheries. The figures are generated by PyMOL [2]. . . . .	96
5.9	The conservation of AK.V.I instances in Glyceraldehyde-3-phosphate dehydrogenase (GPDH) proteins from the UniREF50 database [7]. The sequences are at most 50% similar to one another. Our predicted SLiM is conserved in 11 out of 28 GPDH reference proteins and they are all aligned to the AK.V.I instances in the GPDH proteins found by D-SLIMMER (UniProt ID:P07487 and P00359). 5 GPDH proteins have the exact AK.V.I SLiM while another 6 have an approximate match to the SLiM. For approximate matching, position -1's Alanine (A) can be replaced by a similarly small Valine (V) residue. Position +2's Valine (V) can be replaced by other aliphatic residues like Leucine (L) and Isoleucine (I). We also allow the same replacement for the position +4's Isoleucine (I). The protein alignment is generated by MUSCLE [8]. . . . .	97
6.1	SLiMDiet's overview. The domain interfaces of each PFAM domain are clustered by their structural similarity. Next, from each cluster, the domain and partner faces are structurally aligned and we build a Gapped PSSM based on the contacts on the partner faces. The Gapped PSSM has flexible gaps defined by the minimum and maximum gaps observed between two PSSM positions. We define a Gapped PSSM as linear when the total length of its non-gap positions is three to twenty residues with gaps of at most four residues between any consecutive residue positions. To detect domain-SLiM interfaces, we collect domain interface clusters whose partner faces are covered by a linear Gapped PSSM. . . . .	104
6.2	An example of SLiMDiet's gapped PSSM. . . . .	107
6.3	Partner face alignment steps for finding the longest linear block. The latter is where we extract the SLiM from. . . . .	109
6.4	An illustration of SLiMDiet's gapped PSSM generation from a linear block computed from the multiple interface alignment. . . . .	110

- 6.5 P-value checking on the literature SLiMs and SLiMDiet's Gapped PSSM based SLiMs. The 'motif' column shows the literature's reference SLiM. We can see that 23 out of the 34 known SLiMs in ELM and MnM are enriched in our PPI data based on the hypergeometric p-value  $\leq 0.05$ . The p-values of 17 of SLiMDiet's Gapped PSSM are also  $\leq 0.05$  with 16 of them overlap with the 23 SLiMs from ELM and MnM with p-value  $\leq 0.05$ . . . . . 113
- 6.6 Domain-SLiM interface between Glyceraldehyde 3-phosphate dehydrogenase, C-terminal (Gp\_dh\_C, ID: PF02800) and Glyceraldehyde 3-phosphate dehydrogenase, N-terminal (Gp\_dh\_N, ID: PF00044). (A). The dimer of the Glyceraldehyde 3-phosphate dehydrogenase complex (PDB ID:1gd1). The blue part is the C-terminal domain and the red part mark the N-terminal domain. The C-terminal domain binds to a linear region on the N-terminal domain of the opposite chain (highlighted in ball-and-stick mode). SLiMDiet's predicted SLiM for this region is [YH].[KRQ][YH]D[ST] (B). The surface representation of the Gp\_dh\_C domain of Holo-glyceraldehyde-3-phosphate dehydrogenase from *Bacillus stearothermophilus* (PDB ID:1gd1). The linear region **HLLKYDSVHGR** of the opposite N-terminal domain bound to the domain is shown in ball-and-stick representation. (C). The structure of linear sequence **YQMKHDTVHGR** bound to the Gp\_dh\_C domain of *Leishmania mexicana*'s glycosomal glyceraldehyde-3-phosphate dehydrogenase (PDB ID:1a7k). This figure is generated by PyMOL [2]. . . . . 121
- 6.7 Domain-SLiM interfaces of TNF domain of BAFF proteins recognizing the SLiM D[LHS]L[LV][RH].[IV]. (A). The TNF interface from BAFF with a part of BAFF receptor protein (PDB ID:1oqe). The linear region is shown in ball-and-stick display, comprising the residues DLLVRHCV. (B). The structure between the TNF domain of BAFF complexed with only the minimal peptide DLLVRHWV (shown in ball-and-stick, PDB ID:1osg). This figure is generated by PyMOL [2]. 122

# Chapter 1

## Introduction

All cells on this earth share a strikingly similar set of biomolecules which are the building blocks of the process we called life. All known organisms use macromolecules like the *deoxyribonucleic acid* (DNA), *ribonucleic acid* (RNA), and proteins for their functioning. They also require the use of a group of simpler, yet essential, molecules like sugars, lipid, water, ions and some other organic compounds.

The central dogma of the Molecular Biology stated that DNA stores the genetic information of the organism which, by a process called transcription, is transferred into a messenger RNA and exported out of the cell's nucleus into the cytoplasm. The messenger RNA is then translated into its corresponding protein [9, 10]. The proteins constitute an overwhelming majority of the working machinery that runs the cell. Years of studies in the field have revealed a much more detailed and complicated view of the cell's processes. While the dogma still stands true, recent studies have elucidated that the entities in the dogma have highly complex behaviors and functions. Most of these emerging complexities originate from the interaction between these entities.

### 1.1 RNA and Protein: The two catalysts of the living cell

Almost all processes in the cell involve one or more protein(s) while some other involve both the protein and RNA. These proteins and RNA interact with each other and form functional complexes. They either stay complexed to remain functional (we call them obligate complexes) or they dissociate back into their individual form after accomplishing a certain task (called the transient complexes). An example of an RNA-protein obligate

complex is the ribosomal complex which contain both folded RNA and proteins. On the other hand, a transient RNA-protein complex can be seen in the process called aminoacylation where the *aminoacyl transferase* enzyme attaches a specific amino acid to a particular tRNA based on the tRNA's specific codon. Once the amino acid is attached to the 3' of the tRNA, this enzyme-RNA complex dissociates and the enzyme finds another tRNA to work on.

On the protein side, obligate complexes can be seen in proteins that consist of multiple (possibly the same) protein chains. Each chain adopts a specific three dimensional structure (the protein's tertiary structure) and these individual structures are then arranged in a specific spatial configuration to form the fully functional proteins (the quaternary structure). For obligate complexes, the protein must stay in its complexed form to remain functional. Protein transient complexes, on the other hand, is ubiquitous in processes like the signal transduction where specific pair of proteins take turns to interact in a short period of time to pass specific cell signals across a cascade of interacting proteins.

## 1.2 Interaction motif

One important factor that enables interactions to occur simultaneously in the confined space within a cell is that these interactions are highly specific. To accomplish this, there must be some way for the proteins/RNA to recognize their interaction partner.

Studies had shown that each biomolecule maintains certain patterns (commonly named 'motifs' in the field of Bioinformatics) that are necessary for its interaction with its partner. These motifs are preserved throughout the evolution as long as the interaction is crucial for survival. Such motifs can be embedded inside the sequence of the biomolecule (sequence motif) or the motif is embedded in the three dimensional shape of the biomolecule (structural motif). Strictly speaking, there is no actual sequence motif. All interaction between biomolecules take place in a 3D space hence a sequence motif in a biomolecule is merely a type of 3D structural motif whose elements are localized to a short consecutive region in the biomolecule's sequence.

We propose the term 'interaction motif' to define a general class of biomolecular motif that is conserved for a specific purpose of maintaining one or more functional

interaction(s) between the biomolecule and its interaction partners. This thesis aims to study two instances of interaction motifs, one is found within the RNA and another in the proteins.

1. The RNA structure is found to have stronger implication on the function of the RNA as compared to its sequence content [11]. These structures are found to be recognized by other biomolecules and thus can be considered as a structural interaction motif. One way of representing the structure of RNA is using its secondary structure. We propose an efficient algorithm to infer the secondary structure of an unknown RNA sequence given a known template secondary structure.
2. The second type of motif studied is one class of protein's interaction motif called the Short Linear Motifs (SLiMs). This type of motif is a short sequence motif in proteins whose length is generally less than 20 amino acids. We design three different methods to mine SLiMs, two of them from the protein-protein interaction data and one from the protein structural data.

### 1.3 RNA Secondary Structure

RNA is a biopolymer of nucleotides Adenine (A), Cytosine (C), Guanine (G) and Uracil (U). These nucleotides can form specific pairwise hydrogen bonds where A would pair with U and C would pair with G. Furthermore, U can also pair with G, forming a *wobble pair* [12]. In the cell, DNA are mostly found in pairs of complimentary sequences; each pair forms a double helix. On the other hand, RNA are found as shorter single strands for most of their function in the cells. Single stranded RNA adopts a specific folding; achieved by specific base pairing between its own nucleotides.

Thanks to its ability to form different structures, RNA can function as catalyst and regulator in nucleic acid processing in addition to its commonly known intermediary role in DNA transcription and translation process. Collectively, they are called the *non-coding RNA (ncRNA)*. A study by Carninci *et al* showed that the number non-coding RNA transcripts in human is estimated to be around 35000 which is of the same order as the number of genes in human [13].

Non-coding RNA are mostly recognized by their structure rather than their nu-



cleotide sequence [11]. This implies that sometime the sequence similarity of non-coding RNA of similar function can be quite low yet they still adopt similar structure and perform similar function (nevertheless, some non-coding RNA that are involved in RNA interference process do require a conserved sequence for their function since they rely on accurate hybridization with their target messenger RNA). A simple comparison of all known tRNA sequences (whose length, on average, is around 80 nucleotides (nt)) of human revealed that the sequence similarity of different tRNAs can be lower than 50% yet the tRNAs invariably exhibit the tRNA L-shaped signature structure and all of them are viable in their interaction with the mRNA and ribosome. To model RNA's folding, one can start with the RNA's secondary structure. The latter is a listing of the nucleotide sequence of the RNA and the base pairings that is found in the folded structure of the RNA.

### 1.3.1 Current approaches on finding RNA secondary structure

As mentioned earlier, the secondary structure arises from the complimentary pairing between the bases within the RNA sequence. Currently, few methodologies can resolve the structure of an RNA sequence. Experimentally, the most reliable technique is to solve the 3D coordinates of the RNA sequence in question through X-ray crystallography or NMR spectroscopy. Most other methodologies are based on computational prediction. There are basically two different approaches to predict the RNA secondary structure. The first one, called the *free energy approach*, is based on searching for the most stable RNA folding configuration i.e. one that has the lowest free energy. The assumption is that the correct RNA structure would have the lowest free energy. Some prominent example of this approach is the Minimum Free Energy Algorithm by Zuker [14–16] and the Partition Function Algorithm by McCaskill [17].

The second approach is the *Comparative approach* which is further separated into two subclasses. One uses multiple sequence alignment of related RNA sequences and infers the secondary structure of the group based on the conservation pattern in the multiple alignment. Representatives of this subclass include Maximum Weighted Matching (MWM) [18–20] and Stochastic Context Free Grammars (SCFGs) [21–23].

Another subclass of the comparative approach uses an existing RNA secondary struc-

ture as a template and infers the structure of another RNA sequence. This line of approach is able to bypass the initial alignment problem of the other subclass since it has a valid RNA structure to start with. Our survey on the available RNA structures in the PDB database [24] shows that there has been a steady rise in the number of resolved RNA structures over the years.

Some methods in this line use the Arc-annotated sequence to model the RNA secondary structure. Briefly, an arc-annotated sequence is a string with additional information indicating related pairwise positions within the string. In such model, the string would represent the RNA's nucleic acid sequence and the arc annotation represents the base pairing. Bafna *et al* studied the problem and come up with an algorithm with  $O(n^2m^2 + nm^3)$  time and  $O(n^2m^2)$  space complexity [25] (where  $n$  is the length of the sequence with the known secondary structure and  $m$  is the length of the sequence to be inferred). The algorithm was subsequently implemented in the FASTR program [26] and was shown to be capable of efficiently and reliably inferring the secondary structures of a large number of non-coding RNA in the bacterial and archaeal genomes [26–28]. The algorithm performance was improved in [3] to  $O(nm^3)$  time and  $O(nm^2)$  space.

### 1.3.2 Our contribution

We designed an algorithm to infer the secondary structure motif of an RNA sequence given a known RNA structure template (i.e our method belongs to the second subclass of the Comparative approach). Our main contribution is on the theoretical complexity of the algorithm. Compared with the best algorithm by Zhang [3] (running in  $O(nm^3)$  time and  $O(nm^2)$  space), we improved both the asymptotic time and space complexity of the existing algorithms by an order of magnitude. Effectively, our algorithm runs in  $O(n^2m + nm^2)$  time and  $O(nm + m^2)$  space. These improvements are important since many biological results reported to date are based on the FASTR program (which is based on the  $O(n^2m^2 + nm^3)$  time and  $O(n^2m^2)$  space algorithm). By improving the time and space efficiency, we could infer the secondary structure inference of longer RNA sequences and also increase the throughput of computing the secondary structures of a larger number of RNA sequences.

## 1.4 Protein-Protein Interaction Motif

Protein interaction was previously modeled as "lock" and "key" mechanism where the properties of the interacting proteins complement each other's [29]. The model was improved to allow a more flexible induced fit between the lock and the key [30]. By our definition, these 'locks' and 'keys' are interaction motifs. Interaction motifs in proteins can be of two different types. One is a non-linear, structural motif which is known as the protein domain. A protein domain is an independent protein fold that is conserved in many different proteins. As interaction motif, a protein domain is capable to interact with another protein domain. More recently, it is found that protein domains can recognize a second type of interaction motif, called short linear motif (SLiM) on another protein [31–36]. The listing of all known SLiMs to date could be found in databases like ELM [37] and MiniMotif (MnM) [38, 39]. Some existing experimental methods to find SLiMs are site-directed mutagenesis and phage display. These are tedious and expensive methods to apply on the whole protein interaction data of a single organism (called the *interactome*). Thus it would be beneficial to have a high confidence set of SLiMs to reduce the number of validations. To this end, a number of computational predictions have been designed.

### 1.4.1 Existing computational methods on SLiM mining

As SLiMs are interaction-enabler entities, we expect them to be enriched in interacting proteins. This observation becomes the basis of the majority of the computational methods to mine for SLiMs. However, the main challenge of computing SLiMs lies on its length and motif degeneracy [33]. Their length is around 3–20 residues and the degeneracy implies that the conserved positions in these SLiMs can be quite few.

There are in general three approaches on computing the SLiMs *in silico*. The first approach mines motifs from a given set of related protein sequences. The relation among the sequences maybe established by prior biological knowledge like: sharing similar function, similar localization to a certain cell compartment, and sharing of interaction partners. Methods in this line, for example DILIMOT [40], SLiMDisc [41] and SLiMFinder [42, 43], use statistical analysis on the significance of each of their predicted SLiM. Often, they require a dataset that is compact enough such that a good number

of the sequences actually have the SLiM. When there are too many spurious sequences, the signal of the SLiM could be too weak to be detected from the other unrelated, yet conserved, patterns in the protein set.

The second approach is to mine SLiMs that are over-represented in the available protein interaction data. The difference between this approach and the previous one is that, instead of insisting statistical significance on the motif occurrence, the approach tries to compute the statistical significance of the co-occurrence of a SLiM within some proteins with another motif in their interacting partners. The methods in this class have two subclasses:

1. Methods finding bicliques [44] or quasi-bicliques [45] in the PPI network. These methods fall into the class of interaction driven approach [46] (where the methods start with finding dense bipartite network structure and then mine motifs from the proteins within the structure).
2. Methods finding SLiMs which are found within a statistically significant number of interactions e.g D-STAR [47], MotifCluster [48] and SLIDER [46]. They are categorized under the motif driven approaches (the methods starts from motifs and compute the statistical significance their co-occurrence in interacting proteins).

The third approach is mining SLiMs from the available protein complex data. As opposed to mining statistically significant motif, which may not directly translate into biologically significant ones, given a 3D structure, we can be sure to find our target SLiMs only from the interaction interfaces of proteins. While there have been quite a few methods which compute and characterize domain-domain interface in the structural data like SCOPPI [49] and SCOWLP [50], we only found one method, D-MIST [51], which specifically target SLiMs within the interfaces.

### 1.4.2 Our contributions

*D-STAR*. We designed the first interacting-motif based program, D-STAR [47], to find SLiMs directly from the PPI data. We showed that the interaction signal of the real SLiMs is better than the occurrence signal using two biological datasets, the SH3 and the TGF $\beta$  protein interaction data. More recently, D-STAR has been used in another

work to study TF-TF interaction [52]. As D-STAR was found to be less scalable to handle full genomic PPI data, it was further improved by some recently published programs like MotifCluster [48] and SLIDER [46].

*D-SLIMMER*. We found a significant limitation in the current interaction motif approaches. All interaction motif programs (D-STAR, MotifCluster and SLIDER) assume that both the interaction motifs are linear. However, based on our structural studies (which we will discuss next), this requirement may be too strict. When a domain recognizes a SLiM, the surface that binds to the SLiM is mostly constituted by residues that are not consecutive in the domain’s sequence. Thus, we designed a new algorithm, D-SLIMMER, which is specifically designed to find SLiMs that are recognized by certain protein domains. The critical difference of D-SLIMMER and the existing interaction motif based programs is that it computes the interaction density of the protein domain and the SLiM. Specifically, D-SLIMMER finds interaction motif pairs which consist of a non-linear motif (a protein domain) and a linear one (a SLiM).

We collected 34 reference SLiMs (taken from ELM [37] and MiniMotif database [38,39]) known to interact with 16 reference domains. For each domain, we generate two PPI dataset, one from the BioGRID database [53] and another one from the Human Protein Reference Database (HPRD) [4]. We show that D-SLIMMER significantly outperform the existing programs by finding twice as many experimental SLiMs (15 SLiMs, 6 of which are found in both datasets) from the PPI compared to the best performing program, MotifCluster (7 SLiMs, 2 of which are found in both datasets).

We further reported three variants of known SLiM and a candidate novel SLiM. The first of the three variant SLiM is A.IQ..[FWY]R, which is related to the the IQ motif ..[SACLIVTM]..[ILVMFCT]Q...[RK].{4,5} [RKQ] (ELM ID: LIG\_IQ); a known target of the EF hand domain. Our SLiM’s [FWY] position matches the requirement of the large hydrophobic side-chain just before the basic [RK] residues [54] and missed the last [RKQ] positions while still maintain its IQ motif functionality [5]. This suggests that our A.IQ..[FWY]R SLiM is a valid (variant) IQ motif *bona fide* for interaction.

The second variant SLiM EG..DLFD partially matches the ELM SLiM related to RB\_B domain: ..[LIMV]..[LM][FY]D (ELM: LIG\_Rb\_pABgroove\_1) while also including an acidic residue just before the conserved suffix [LM][FY]D; such acidic residue is

used by some adenovirus to mimic the E2F-Rb interaction [55]. The third variant SLiM PPPGL matches the recently reported PPG $\phi$  motif (where  $\phi$  = hydrophobic amino acid, except for tryptophan) for the GYF domain [56]. The SLiM PPG $\phi$  has only recently been published in the literature, hence it is yet to be included in the current SLiM databases.

Our proposed novel SLiM AK.V.I is associated with the Sir2 domain which is involved in repression of gene transcription in the telomeres, DNA repair process, cell cycle progression, chromosomal stability and cell aging [57]. One instance of our SLiM has been experimentally verified and the SLiM also satisfies the residue preference of Sir2 as mentioned in [6].

*SLiMDiet.* We present another result in which we looked into the available 3D structural data to mine for linear motif to complement our sequence based SLiM mining methodologies. In this setup, we computed and aligned all possible linear stretch of amino acids which are recognized by the same protein domain. Our program, named SLiMDiet, uses a pairwise interaction interface similarity algorithm which is tailored specifically for Domain-SLiM interfaces. We showed that the clusters which resulted from the use of our similarity algorithm was more accurate than those produced by the existing algorithm.

Our method found a list of 41 literature validated SLiMs, 61 SLiMs with peptide experiment validation and 61 high confidence novel linear motifs which are enriched in the current high throughput sequence interaction data. SLiMDiet covers significantly more literature SLiMs when compared to D-MIST [51]. A careful study on a few cases further reveals biologically significant novel motifs. We also study whether the coverage of the current PPI dataset is uniform over all known protein domains. We found that there are a sizable number of well validated domain-SLiM interaction that is under represented in the high throughput data, presumably because they are not amenable to the protein interaction detection protocol. This shows that structure based SLiM prediction is an important complement to the current sequence based SLiM mining methods. SLiMs produced by our method would also serve as validators (since they are all based on existing 3D structures) of predicted SLiMs from the sequence based approaches.

## 1.5 Thesis organization

This thesis is organized as follows. We first provide some background information on RNA secondary structure and protein Short Linear Motifs (SLiMs) in chapter 2. We discuss on our results on the RNA secondary structure prediction in chapter 3. Chapter 4 would provide a description on our first PPI SLiM mining algorithms, D-STAR. The theoretical concept and notation of the correlated motif approach are discussed. Chapter 5 is dedicated to D-SLIMMER which outperforms the accuracy of the other existing PPI SLiM mining approaches. The SLiMDiet algorithm and its biologically significant SLiMs are described in chapter 6. Finally, chapter 7 concludes this thesis with summary of our results and discussion on the possible avenues for future works.

# Chapter 2

## Background

This chapter aims to provide some background information on the two biomolecules that we study in this thesis, the RNA and the Proteins. We touch on the chemical building blocks of these molecules and how they form an ordered pattern to be recognized for interaction with one another.

### 2.1 RNA: Ribonucleic acid

RNA is known to be the template with which the information on the DNA sequence of an organism is translated into the proteins. These RNA are known as the messenger RNA (mRNA) which are copied from a gene (a region in DNA encoding a protein's sequence). The process is known as the *transcription* of DNA. The mRNA transcripts are then exported out of the nucleus into the cytoplasm for protein production. This process, called the translation of the mRNA, is done by a specialized organelle (a specific subunit with a specific function in a cell) called the ribosomes.

RNA is another member of the nucleic acids which is, like DNA, a biopolymer consisting of nucleotides. However, RNA molecules have several differences from the DNA:

1. It contains a ribose sugar as opposed to deoxyribose sugar in DNA. This results in an additional hydroxyl at the sugar's 2' which makes RNA less stable by its being more prone to hydrolysis and its ability to cleave the backbone.
2. RNA does not use the nucleotide Thymine, instead it uses the uracil base (the un-



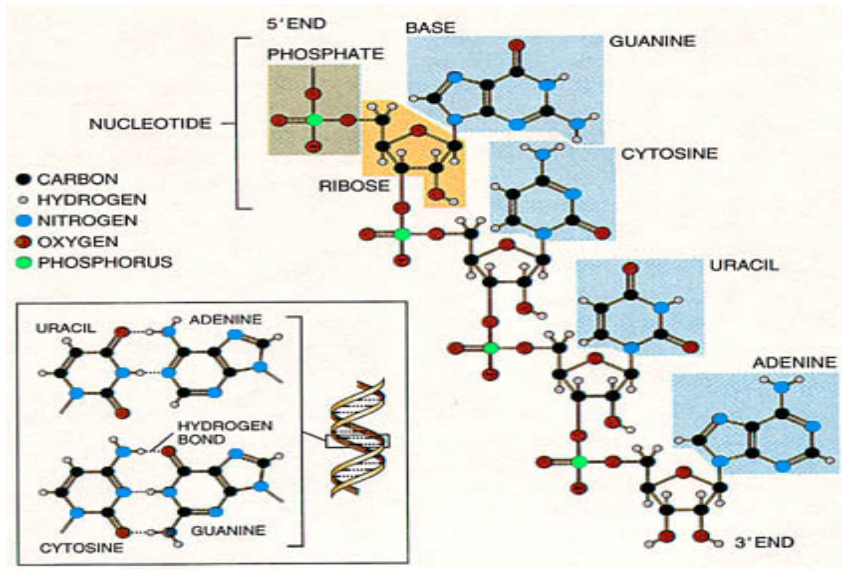


Figure 2.1: The structure of RNA and its nitrogen bases

methylated version of the thymine) which can pair with both adenine and guanine (called the wobble pair [12]) .

3. RNA is found as shorter single strands for most of its function in the cells (as opposed to long DNA double helix). Most of the time, RNA adopts a specific folding much like proteins.

An illustration of the RNA nucleotide pairings, the chemical structure its sugar and phosphate backbone is shown in Fig. 2.1. RNA can form secondary structures, by specific base pairing between its own nucleotides, forming *stems* (the region that is paired in the folded RNA) and *loops* (the region that is unpaired). Based on their positions, loops are further divided into *hairpins*, *bulges*, *internal loop* and *multi loop*. These secondary structures can be seen in Fig. 2.2. When unpaired bases from one loop is paired to the bases on another loop, they form the tertiary structures shown in Fig. 2.3.

### 2.1.1 The non-coding RNA

RNA's function is not limited to passing information from the DNA into the protein. In fact, some RNA do not code for proteins but functions as enzymes and regulators in many cell processes. This functionality comes from to RNA's ability to adopt different structures and its chemically more active nature [58]. This class of RNA is similarly

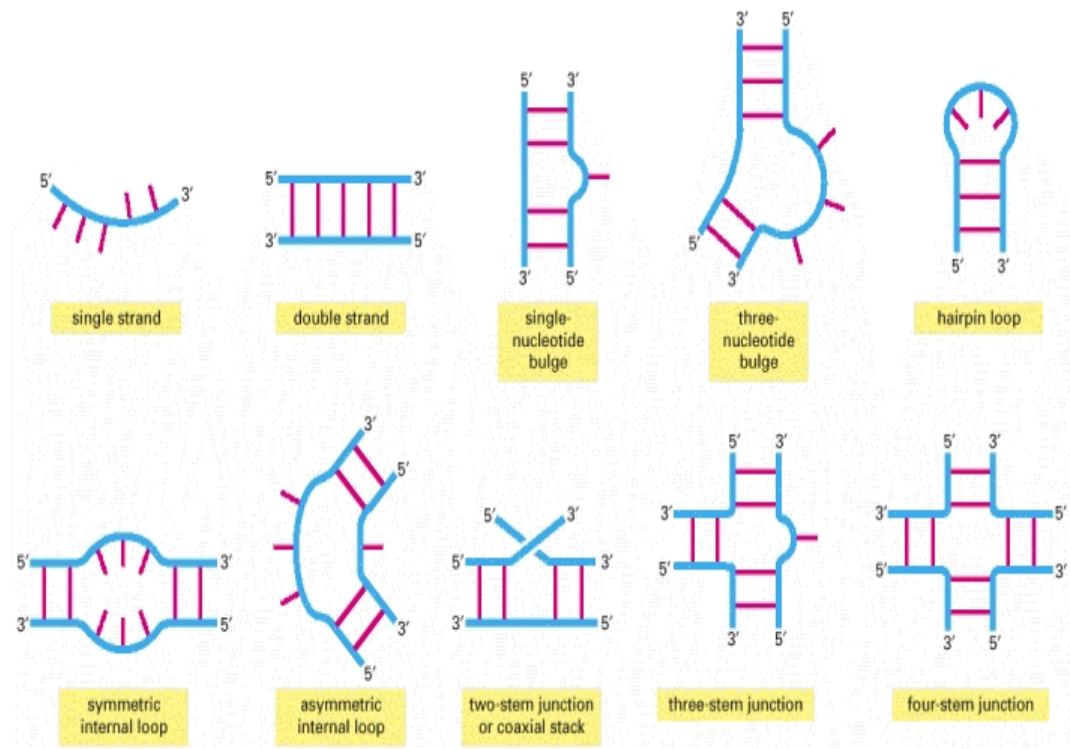


Figure 2.2: The secondary structure of RNA. This figure is adapted from Molecular Biology of the Cell, 5E, © 2002, by permission of Garland Science LLC. Reproduced by permission of Garland Science/Taylor and Francis LLC.

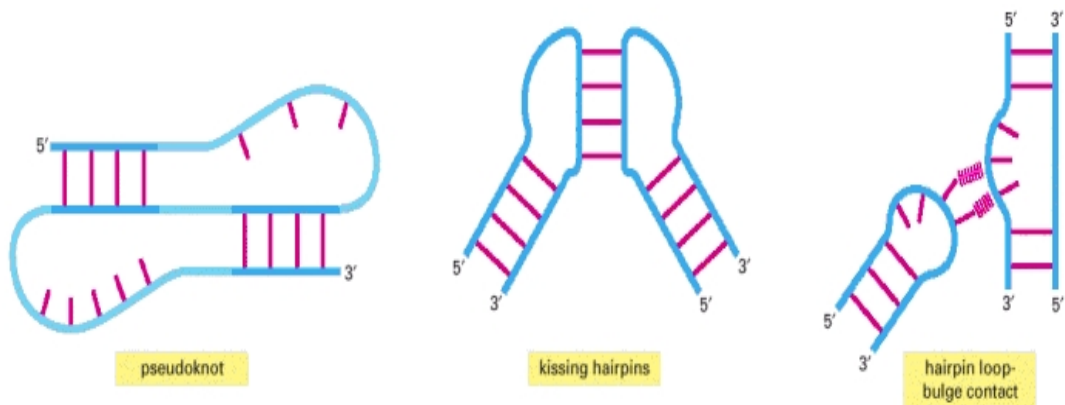


Figure 2.3: The tertiary structure of RNA. This figure is adapted from Molecular Biology of the Cell, 5E, © 2002, by permission of Garland Science LLC. Reproduced by permission of Garland Science/Taylor and Francis LLC.

transcribed from the DNA of the organism yet it lacks of any apparent open reading frame (ORF) thus incapable of producing functional proteins. Collectively, they are called the *non-coding RNA (ncRNA)* and they have been found ubiquitously in all three

---

Figure 2.4: The secondary and tertiary structure of the transfer RNA (tRNA). The clover-like secondary structure is conserved in all domains of life. Some of the nucleotides are post-processed into a non-canonical nucleotides (T stands for Ribothymidine,  $\psi$  for pseudouridine and the nucleotides with an 'm' sign are methylated in their ribose sugar). These figures are taken from the Wikimedia Commons.

domains of life (bacteria, archaea, and eukarya).

There are already many well studied ncRNA: the ribosomal RNA (rRNA) and transfer RNA (tRNA) which are involved in the protein translation machinery of the cell, the small nuclear RNA (snRNA) which splice off the introns from nascent messenger RNA into their mature form, and several others with important and specific regulatory roles (reviewed in [59]). More recently, other classes of small ncRNA such as microRNAs (miRNAs), CD box snoRNAs, small interfering RNAs (siRNAs), and small temporal RNAs (stRNAs) have been characterized based on transcription analysis and computational screening [60–66]. More detailed information on these newer non coding RNA are covered in excellent reviews like [67, 68].

The number of non-coding mRNA transcripts in human is estimated to be of the same order as the number of genes [13]. Such vast expanse of RNA functionalities give a strong support to an existing hypothesis that the earliest forms of life relied on RNA both to carry genetic information and to catalyze biochemical reactions-an RNA world [69, 70].

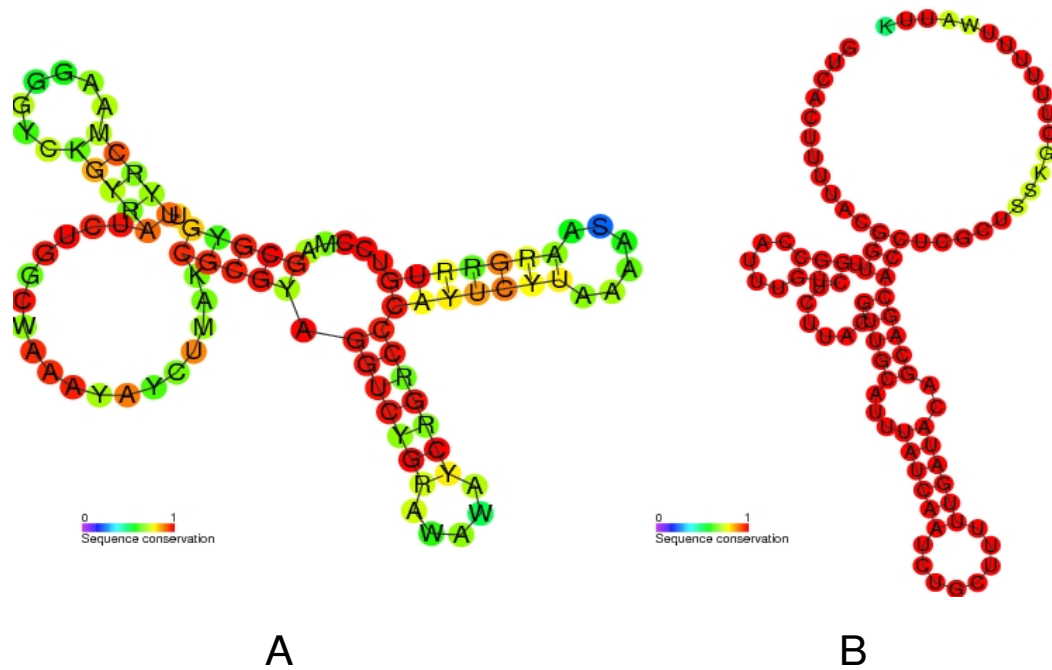


Figure 2.5: Two examples of non-coding RNA secondary structure motifs. (A) The secondary structure of ATPC RNA motif conserved in certain cyanobacteria (RFAM ID:RF01067). We can see from the coloring that the sequence conservation of this structure is rather weak. (B) The structure of invasion gene associated RNA (also known as InvR). This is a small non-coding RNA involved in regulating one of the major outer cell membrane porin proteins in *Salmonella* species (RFAM ID:RF01384). The figures are taken from the RFAM database [1].

### 2.1.2 RNA Secondary Structure in non-coding RNA

RNA often works with proteins to form a complex called the *ribonucleoproteins* (RNP) with a few exception like tRNA. Mostly, the RNA is used as the recognizing agent and the RNP usually targets other nucleic acid molecules (e.g DNA, RNA). In the ribosome, rRNA are bound by protein and make up the catalytic site. One part of the rRNA recognizes the sequence preceding the first codon to be translated in the mRNA, the latter is known as the *Shine-Dalgarno box* consisting of the sequence AGGAGG in prokaryotes [71]. A similar sequence in eukaryotes is named the *Kozak box* [72].

It has been suggested that catalytic RNA are mostly recognized by their shape as opposed to their sequence content. This implies that sometime the sequence similarity of these RNA of similar function can be quite low yet they still adopt similar structure. In a sense, the folding pattern of the RNA sequence is the determinant of its

interaction specificity with its partners. Such pattern can be captured by the RNA secondary structure which details all base pairings in an RNA structure. Indeed, a lot of non-coding RNA are found to have conserved secondary structures—yet have weaker sequence conservation. Fig. 2.4 and 2.5 depicts the tRNA structure and some known RNA secondary structure listed in the RFAM database [1] respectively. Note that some part of the secondary structure are not very conserved (indicated by the base’s coloring). Given the limited current knowledge on non-coding RNA and given the strong conservation on these non-coding RNA’s structures, we would need efficient methods for identifying RNA secondary structures given their sequence.

### 2.1.3 Current RNA secondary structure data

We propose a method which uses a template secondary structure to infer the secondary structure of another RNA sequence. Hence, we would need to show that there are enough such secondary structure to begin with. The best source of templates would be the 3D structures of RNA stored in the PDB database. Currently there are 1744 RNA structures (818 are RNA only structures (based on PDB statistics [24]) and 926 are protein-RNA complex structures [73]). The number of just 3 years ago in 2007 was 1142 RNA structures, of which 615 are RNA only [24] and 527 are protein-RNA structures [74], averaging about 200 new RNA structures per year. Another source of secondary structures would be the RFAM database [1]. It contains the multiple sequence alignment and the covariance profiles (constructed using the first subclass of the comparative approach) of many structural RNA (including non-coding RNA). The number of RNA families in the RFAM database is 1446. These two sources provide a significant amount of known secondary structures that can be used for our proposed method in the next chapter.

## 2.2 The proteins

Almost all function in the cells are performed by proteins. The catalyzing of various biochemical reactions, the scaffolding that gives shape and mechanical strength to the cell, the signaling process within and between the cell(s), the cascade of immune responses, the process underlying cell adhesion and the regulation of the cell cycle are but a few

of the essential tasks of proteins within the living cell. Proteins make up half the dry weight of an *Escherichia coli* cell, whereas other macromolecules such as DNA and RNA make up only 3% and 20%, respectively [75].

Proteins are biopolymers consisting of amino acids. There are twenty common amino acids that are used universally by all organisms known on earth. They are Alanine (A), Cysteine (C), Aspartate (D), Glutamate (E), Phenylalanine (F), Glycine (G), Histidine (H), Isoleucine (I), Lysine (K), Leucine (L), Methionine (M), Asparagine (N), Proline (P), Glutamine (Q), Arginine (R), Serine (S), Threonine (T), Valine (V), Tryptophan (W), and Tyrosine (Y). Sometimes cysteine is found with a selenium atom, forming the amino acid Selenocysteine (U). Different amino acids share the same backbone atoms with one another and have different side chain atoms. (the diagram of different side chains and the general structure of an amino acid are given in Fig. 2.6).

Amino acids are linked together by a peptide bond to form functional protein chains. These chains are also able to form local secondary structures which arise from the hydrogen bonding between the backbone atoms in the chain. The most commonly known secondary structures for protein are the *alpha helix* and the *beta sheet*. These structures, in turn, form a tertiary structure; a process which is driven by the long range residue interactions like the hydrogen bonding, hydrophobic and electrostatic interactions. Cysteine residues can also form a covalent bond between their sulphur atoms—called the disulfide bridge. The tertiary structure is fixed given a certain amino acid sequence in the protein chain (the primary structure of the protein) and a set of environmental parameter (like the pH and the ionic conditions). Several protein tertiary structures can also combine together to form the quaternary structure, which is the functional complexed form of the protein (also referred to as the biological unit of the protein). The primary, secondary, tertiary and quaternary structures of a protein are illustrated in Fig. 2.7.

Proteins are modular by nature. A functional protein tertiary structure may consist of two or more functional subunits. These subunits are sequentially conserved in many different proteins and are capable to fold into specific independent structures. Collectively, they are known as the protein domains. There exist quite a few databases which list a set of known protein domains like PFAM [76], InterPro [77], PROSITE [78] and PRODOM [79], which are derived from protein sequence data. Another group

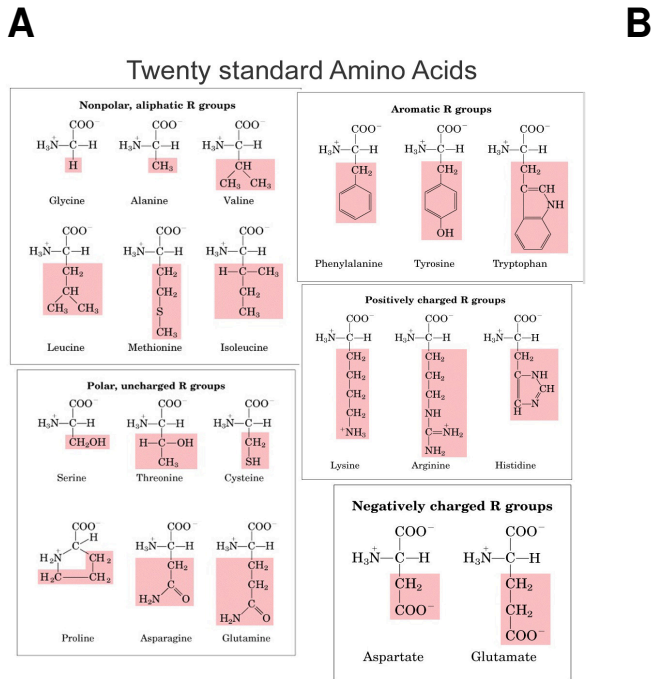


Figure 2.6: (A) The 20 side chains of the known amino acids. (B) The diagram illustrates the atomic configuration of an amino acid. The same backbone atoms are used in all amino acids and the R part is where the different side chains are attached. These figures are taken from the Wikimedia Commons.

of databases list protein domains which are derived from the increasingly larger protein structural data in the Protein Data Bank. Examples of the latter databases are SCOP [80] and CATH [81].

## 2.2.1 Protein-Protein Interaction Motif

Protein interaction plays an essential role in a vast number of known biological processes. It is responsible in the formation of functional protein complexes (the quaternary structure), signal transduction, cell regulation and immune response processes. The interaction partners of proteins are very diverse: (1) transcription factor proteins can bind specific DNA sequences to activate or repress transcription activity of a gene, (2) enzymes catalyze reactions involving sugars, lipids and inorganic metal ions, (3) protein cooperate with RNA with certain sequence and structure to form the Ribonucleoprotein complexes.

From the strength of the interaction, protein interaction can be a permanent inter-

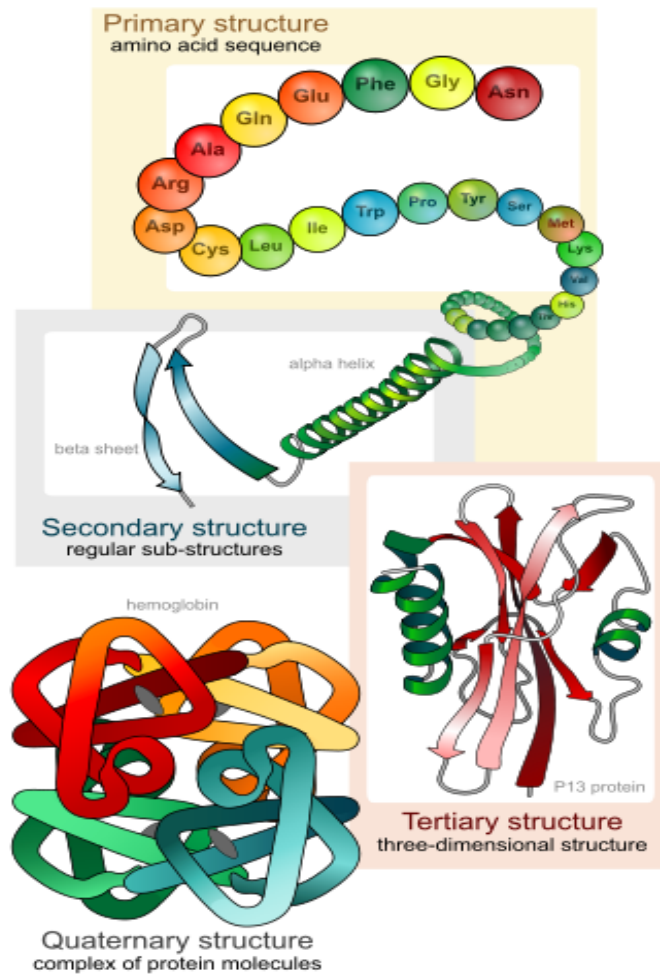


Figure 2.7: The illustrations of protein's primary, secondary, tertiary and quaternary structures. This figure is taken from the Wikimedia Commons.

action seen in the binding of different subunits of a functional protein complex (termed as obligate interaction). With its relatively high binding affinity, this type of interaction usually lasts throughout the protein's lifetime. The second type of interaction is a temporary, mostly of lower affinity, interaction (termed transient interaction) which forms and breaks in a cascade of biochemical reactions in the cell seen commonly in the cellular signal transduction [82, 83]

Based on the interaction motifs, there are two general types of protein interaction:

1. Interaction between two structural, non-linear (e.g. the protein domains) interaction motifs on the protein and,
2. Interaction between a non-linear interaction motif with a linear peptide interaction



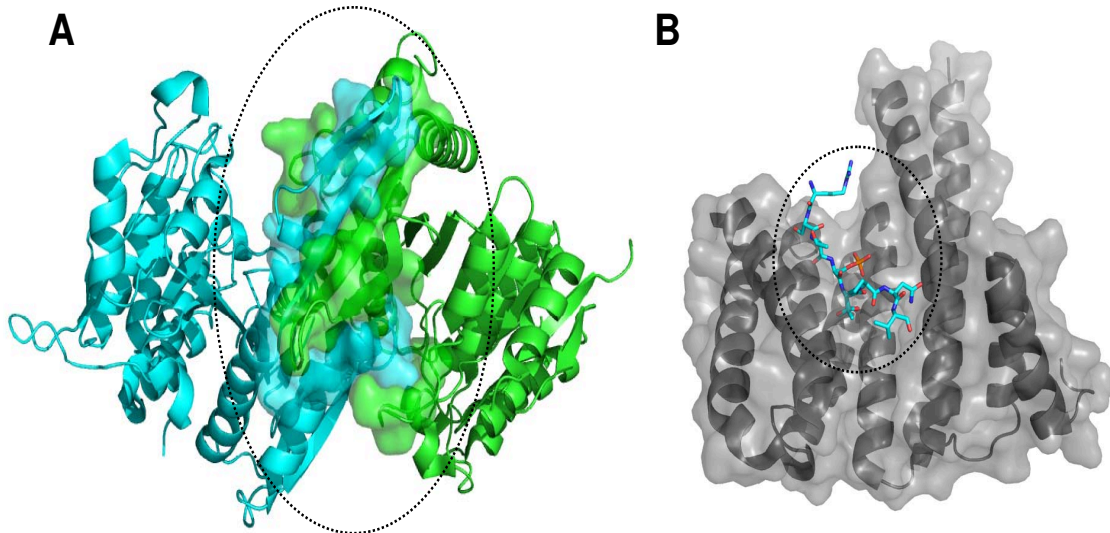


Figure 2.8: (A) A domain-domain interface and (B) a domain-SLiM interface. We can see that the SLiM (shown in sticks) is in an extended linear conformation while the domain surface "wraps" around it. We also observe that the size of the interface is significantly larger for domain-domain as compared to domain-SLiM interface. This figure is generated by PyMOL [2].

motif commonly known as the Short Linear Motifs (SLiMs).

Domain-domain interaction have been shown to be an important factor in protein-protein interactions. A number of studies had shown that domain-domain interactions are evolutionarily conserved among different species [84, 85]. Indeed, there are many protein-protein interaction prediction algorithms which are trained on the domain composition of the interacting proteins in the dataset [86–88]. Domain-domain interaction has also been used in protein complex study and predictions [89]. Based on the domain-domain interaction in the PPI data, Ng. *et. al.* created the InterDom database and provided a useful tool for predicting pairwise protein interaction and protein complex formation [90]. Some researchers mined the domain-domain interactions directly from the PDB structural database [24]; the databases in this line are iPFAM [91], 3DID [92], SCOPPI [49] and SCOWLIP [50].

### 2.2.2 Protein Short Linear Motifs (SLiMs)

As mentioned, domain-domain interaction is not the whole picture of protein-protein interaction. There is another class of interaction where one of the interaction motif is

a short linear stretch of peptide. This type of interaction motif is called protein *Short Linear Motifs* (SLiMs). Processes like cell signaling, post translational modification and protein transport are found to be dependent on SLiM recognition and binding [33,35,36]. Many SLiMs are recognized by a specialized protein domain; for example, the SH2, WW, 14-3-3, FHA, and PDZ domain [31–34].

Most domain-SLiM interaction are found to form transient complexes because of their smaller interaction interfaces [93]. Fig. 2.8 gives a picture that contrasts a domain-domain interaction interface against a domain-SLiM interaction one. The small binding areas on the SLiMs also make them better candidates for intervention by small molecules [34]. This makes finding SLiMs important for drug discovery as many domain-SLiM interactions have been implicated in disease pathways. For instance, the proline-rich motifs and glutamine-rich motifs have been linked to Alzheimer’s disease, Muscular Dystrophy [94] and Huntington’s disease [95]. Recently, Marti *et. al.* reported that the short linear sequence motif R.L.[QE] played a key role in the pathogenesis of malaria [96,97]. One example of a SLiM based drug is the cancer drug candidate compound Nutilin-3, which disrupts the p53-MDM2 complex by mimicking a peptide in p53, thus freeing p53 to respond to DNA damage [98,99]. A few other similar examples can be found in an excellent review by Vagner *et al* [100].

Experimental methods that are available for finding SLiMs are, for example, the site-directed mutagenesis and the phage display. One can also perform experiments to solve the 3D structure of a protein domain and a peptide containing a SLiM. However, we note that these techniques are all low-throughput in nature and also pretty expensive. The listing of all experimental SLiMs to date could be found in ELM [37] and MiniMotif (MnM) database [38,39]. Their number is around 500 based on the older ELM [37] and Minimotif 1.0 [38] listing. The newer Minimotif 2.0 database reported to contain 5089 protein-SLiMs interactions [39]—the database separately records interactions between different proteins against the same SLiM (no number of distinct SLiMs recorded is indicated). Unfortunately, we can only query against these SLiMs but not list them all.

SLiMs are expected to be enriched in pair of interacting proteins since they enable some of them. Thus, most computational methods mine SLiMs based on their over-representation in the PPI data. The main challenge of computing SLiMs lies on its length and motif degeneracy [33]. They can be as short as 3 amino acid residues and

rarely exceed 20 residues and these motifs have even fewer conserved positions within them. (one SLiM recognized by the SH2 is **Y.N.** (Y is phosphorylated)—a length 4 SLiM with only two defined positions). We propose to find SLiMs from the PPI using an interaction based approach—which scores a candidate SLiM based on the density of the interaction network between the SLiM and its partners.

Another approach would be to mine SLiMs directly from the structural data; a logical extension of finding domain-domain interaction in the structural data. Up to date, we only found D-MIST to attempt this approach [51]. However, we observe that it relies too little on the structural data and depends too much on the (sequential) PPI data (it uses just one domain-SLiM structural template and enrich it using the PPI data). By doing so, it suffers from the limitations of the current PPI data and we show that we can outperform D-MIST by finding structural SLiMs that are inherently hard to mine from the PPI data (details in chapter 6).

### **2.2.3 The availability of the PPI and Protein Structural Data**

PPI data have been continuously increasing over the years; starting around year 2004. The number of known interaction was below 5000 before 2004 and jumps to around 20000 in the early 2004. From then on, the number increases until it reaches  $\sim 150000$  known interactions today [101]. More interactions are identified in high throughput experiments (59.8%) as opposed to low throughput ones (40.2%) [53] and we expect that this would be the norm in the future. On the structural side, the number of protein complexes solved to date is 64353. There has been a yearly addition of 6000 structures, in average, since 2005 [24]. This wealth of data would be a good source of structural mining algorithms like SLiMDiet (chapter 6).

## Chapter 3

# Discovering Interacting Motifs in RNA: Predicting the RNA Secondary Structure

### 3.1 Introduction

Earlier, we have shown that RNA secondary structure prediction is important for determining the structure of an RNA which in turn determines the functionality and interaction of the RNA with its partners. We also briefly described the two classes of computational methods for predicting RNA secondary structure: the Energy Minimization and Comparative methods.

The (free) energy minimization of the RNA structure is based on some empirical thermodynamics study on short RNA sequences [102]. The approach assumes that the free energy of the base pairing and the loop structures within the RNA secondary structures are additive and the correct RNA structure is the one with the minimum free energy. A few prominent example of this approach is the Minimum Free Energy Algorithm by Nussinov and Jacobson [103] and by Zuker *et al* [14–16, 104]. Another examples is the Partition Function Algorithm by McCaskill [17]. These algorithms are implemented in the popular Vienna RNA package [105]. As the energy parameters and additivity assumption are approximations at best, the resulting lowest energy structures may not really be the actual folded structures. Thus, several recent approaches tried

to report several (good) structures that is within a range of free-energy values from the lowest [106].

The comparative methods are based on the assumption that: (1) Base pairing is the main stabilizing force of the RNA folding; they have to be conserved for the RNA to keep its folding configuration and, (2) furthermore, the base composition of the unpaired RNA sequence is also important for the RNA's interaction with its target (this interaction requirement cannot be easily modeled into the free energy approaches). These conservation pressures result in specific base and base-pairing retention over the course of the evolution. Hence, by comparing a few related RNA sequences, one can, in theory, observe such conservation and infer their secondary structure. Based on this, algorithms to align and compare RNA sequences and secondary structures were designed. The comparative approach is currently the best way to predict RNA structures [107, 108].

This approach is further divided into two subclasses. The first one takes a number of RNA sequences that are expected to share a similar structure and build a multiple sequence alignment from them. Based on the conservation pattern in the multiple sequence alignment, we can compute the consensus secondary structure among the sequences. A few examples of this subclass are the Mutual Information based algorithms [18–20] and the *Stochastic Context Free Grammar* (SCFG) based algorithms [21–23]. Since there is an evolutionary pressure to keep the base pairings in some positions, these positions would have detectable covariations which is indicative of complimentary base pairing. The Mutual Information based approach would utilize this observation to find base pairing regions within an alignment of RNA sequences. The SCFG approach is a natural extension of the *Hidden Markov Model* (HMM) used to model protein or nucleic acid sequences. The algorithms would start from an initial alignment of RNA sequence and its predicted consensus structure to align and predict the structure of another RNA sequence whose structure is unknown. Like the HMM, SCFG-based algorithms rely on a seed alignment and progressively add new sequence into the alignment.

Methods in this subclass would require a good initial alignment and this is usually done by finding a few homolog of the target RNA (using a sequence homology program like BLAST [109]). However, this method may give rise to an initial alignment that is too conserved sequentially which, when learned by the model, would fail to recognize remote structural homologs (those RNA with similar structure but low overall sequence

similarity). Moreover, when the number of homologous sequences is not large enough, the accuracy can be low.

When the structure of one RNA is known, the secondary structure of another similar RNA sequence can be predicted through structural inference [3, 25]. Consider two sequences  $S_1$  and  $S_2$  of length  $n$  and  $m$ , respectively. Assuming that the secondary structure of  $S_1$  is known, this method infers the secondary structure of  $S_2$  by aligning  $S_1$  and  $S_2$ . This inference approach is the second subclass of the comparative approach. The formal definition of the problem is given in Section 3.2.1. Bafna *et al* [25] propose a dynamic programming solution to this problem and solve it using  $O(n^2m^2 + nm^3)$  time and  $O(n^2m^2)$  space. Bafna *et al* had implemented the algorithm in the FASTR program [26]. and showed that the inference approach is capable to efficiently and reliably infer structures of a large number of non-coding RNA in the bacterial and archaeal genomes [26–28]. Since all of these results are built on the FASTR program, one could significantly improve the efficiency and extend the usability of these programs to longer sequences by improving the algorithm’s complexity. Zhang [3] was the first to report an algorithm that runs in  $O(nm^3)$  time and  $O(nm^2)$  space.

In this work, we further improve the running time to  $\min\{O(nm^2+n^2m), O(nm^2 \log n), O(nm^3)\}$  and at the same time bring down the space requirement to  $\min\{O(m^2 + mn), O(m^2 \log n + n)\}$ . Our algorithmic improvement in the running time stems from a dynamic programming sparsification technique. We observe that the entries in every row in the dynamic programming tables are monotonically increasing, enabling us to fill in a smaller number of entries in the tables without losing any information. We also present a new recursive dynamic programming algorithm that gives a better worst-case space requirement for the case of computing only the score of the optimal alignment of  $S_1$  and  $S_2$ . Finally, by incorporating the latter into an algorithm similar to Hirschberg’s traceback [110] together with a simple compression method, we can recover the optimal inferred structure from the table within the stated reduced space complexity.

## 3.2 Existing Method

### 3.2.1 Preliminaries

In our algorithm, we represent an RNA sequence and its secondary structure information using the arc-annotated sequence [111]. Let  $[a..b]$  represents a discrete interval bounded by the integers  $a$  and  $b$  where  $a \leq b$ . When  $a = b$ , the interval can be written as  $[a]$ . Consider a sequence  $S$  over a fixed alphabet  $\Sigma = \{A, C, G, U\}$ . We define  $S[i]$  to be the  $i^{\text{th}}$  character in  $S$  and  $S[i..j]$  to be the substring of  $S$  in positions between  $i$  and  $j$  (inclusive). For any  $x \in \Sigma$ , let  $\text{Complement}(x)$  be the complementary base(s) of  $x$  according to the Watson-Crick or Wobble (G-U) base pairing. Therefore,  $\text{Complement}(A) = \{U\}$ ,  $\text{Complement}(C) = \{G\}$ ,  $\text{Complement}(U) = \{A, G\}$ , and  $\text{Complement}(G) = \{C, U\}$ . An unordered pair of positions  $(i, j)$ , where  $i < j$ , indicates that  $S[i]$  and  $S[j]$  form a base pair in the RNA structure. Such a pair is called an *arc*. For RNA sequences, we require that, for any  $(i, j)$ ,  $S[j] \in \text{Complement}(S[i])$  and vice versa. A set  $P$  of arcs is called an *arc-annotation*, and the pair  $(S, P)$  is called an *arc-annotated* sequence. Arc-annotated sequences are well-studied [3, 111–117] and are commonly used in computational biology to represent the structure of RNA and protein sequences.

Since we are considering RNA secondary structures, we assume that the RNA sequences we are dealing with do not have any pseudoknots. The corresponding type of arc-annotation for RNA structures without pseudoknots is the *nested* arc-annotation [112, 115–117] where, for any two arcs, either one is within the other, or they are completely disjoint ( $\forall (i_1, j_1), (i_2, j_2) \in P, i_1 \in [i_2..j_2] \Leftrightarrow j_1 \in [i_2..j_2]$ ). For any arc  $u \in P$ , we denote  $u_l$  and  $u_r$  to be the left and the right endpoints of  $u$ , respectively. The *size* of an arc  $u$  is denoted by  $|u| = u_r - u_l + 1$ . We say that position  $i$  is *free* if  $i$  is not an endpoint of any arc in  $P$ . A position  $i$  is *covered* by an arc  $u$  if  $u_l < i < u_r$  and there exists no other arc  $u'$  such that  $u_l < u'_l < i < u'_r < u_r$ . The set of all positions covered by  $u$  is called the *arc cover* of  $u$ , denoted by  $C(u)$ .

Consider two arc-annotated sequences  $(S_1, P_1)$  of length  $n$  and  $(S_2, P_2)$  of length  $m$ . Let  $S_2$  is the sequence whose arc-annotation  $P_2$  is to be inferred from  $S_1$ 's arc-annotation  $P_1$ . Given two arc-annotated sequences, we can define the similarity of the sequences by aligning the bases and the arcs in them. We need to define a scoring function for each type of alignment. Let  $\chi$  be the function to score the alignment of unpaired bases in

the two sequences where, for  $a, b \in \{A, C, G, U, \sqcup\}$  (' $\sqcup$ ' denotes a blank character),

$$\chi(a, b) = \begin{cases} \beta & \text{if } a = b, a \neq \sqcup, b \neq \sqcup \\ 0 & \text{otherwise} \end{cases}$$

For any pair of position  $(u_1, u_2)$  in  $S_1$  and  $(v_1, v_2)$  in  $S_2$ , let  $\delta$  be a scoring function for arc alignment whose value is defined as:

$$\delta((S_1[u_1], S_1[u_2]), (S_2[v_1], S_2[v_2])) = \begin{cases} -\infty & \text{if } S_1[u_1] \notin \text{Complement}(S_1[u_2]) \text{ or} \\ & S_2[v_1] \notin \text{Complement}(S_2[v_2]), \\ \alpha_1 & \text{if } S_1[u_1] = S_2[v_1] \text{ and } S_1[u_2] = S_2[v_2], \\ \alpha_2 & \text{if } S_1[u_1] = S_2[v_1] \text{ and } S_1[u_2] \neq S_2[v_2] \text{ or} \\ & S_1[u_1] \neq S_2[v_1] \text{ and } S_1[u_2] = S_2[v_2], \\ \alpha_3 & \text{if } S_1[u_1] \neq S_2[v_1] \text{ and } S_1[u_2] \neq S_2[v_2]. \end{cases}$$

$\beta$ ,  $\alpha_1$ ,  $\alpha_2$  and  $\alpha_3$  are positive integer constants. Usually the parameters are set so that  $\beta < \alpha_3 < \alpha_2 < \alpha_1$  which reflects that an arc alignment ( $\alpha_1$ ,  $\alpha_2$  or  $\alpha_3$ ) takes precedence over a single base alignment ( $\beta$ ). Moreover, an arc alignment with exactly the same base pairs should score higher ( $\alpha_1$ ) since both the bases and their arcs are aligned. One can also have constraints on the arc width, for example, when  $|u|$  or  $|v|$  is less than some minimum arc width parameter, we can define  $\delta = -\infty$ . Now given the definition of the arc annotation and the scoring functions, we formally state our problem as follows.

The common substructure of two arc-annotated sequences  $(S_1, P_1)$  and  $(S_2, P_2)$  is defined as the alignment between  $S_1$  and  $S_2$  where free positions in  $S_1$  are aligned to free positions in  $S_2$  and (both endpoints of ) arcs in  $P_1$  are aligned to (both endpoints of) arcs in  $P_2$ . The common substructure score is the weighted sum of all bases' and arcs' individual alignment scores. The Weighted Largest Common Substructure(WLCS) score is then defined as the maximum common substructure score among all possible common substructures. The problem we address in this paper is: Given a nested arc-annotated sequence  $(S_1, P_1)$  and a plain sequence  $S_2$ , infer the nested arc-annotation  $P_2$  for  $S_2$  that maximizes their WLCS score.



### 3.2.2 Algorithm Description

This section reviews Zhang's algorithm (presented in [3]) for inferring the RNA secondary structure  $P_2$  for  $S_2$  that maximizes the WLCS score between  $(S_1, P_1)$  and  $(S_2, P_2)$ . Recall that  $|S_1| = n$  and  $|S_2| = m$ . Let  $DP_{(i,i')}[j, j']$ , where  $1 \leq i \leq i' \leq n$  and  $1 \leq j \leq j' \leq m$ , denote the optimal WLCS score between  $(S_1[i..i'], P_1)$  and  $(S_2[j..j'], P_2)$  among all possible  $P_2$ . Note that  $DP_{(i,i')}[j, j'] = 0$  whenever  $i > i'$  or  $j > j'$ . Zhang presented an algorithm to compute  $DP_{(1,n)}[1, m]$  that runs in  $O(nm^3)$  time and uses  $O(nm^2)$  space based on a two-step dynamic programming. Below are the three equations used to compute the two steps of the algorithm. Please refer to [3] for the correctness proofs.

**Lemma 3.2.1** (Lemma 4 in [3]) *If  $i'$  is free,*

$$DP_{(i,i')}[j, j'] = \max \begin{cases} DP_{(i,i'-1)}[j, j' - 1] + \chi(S_1[i'], S_2[j']), \\ DP_{(i,i'-1)}[j, j'] + \chi(S_1[i'], \sqcup), \\ DP_{(i,i')}[j, j' - 1] + \chi(\sqcup, S_2[j']). \end{cases}$$

**Lemma 3.2.2** (Lemma 5 in [3]) *For any arc  $u \in P_1$  and  $i < u_l$ ,*

$$DP_{(i,u_r)}[j, j'] = \max_{j-1 \leq j'' \leq j'} \{DP_{(i,u_l-1)}[j, j''] + DP_{(u_l,u_r)}[j'' + 1, j']\}.$$

**Lemma 3.2.3** (Lemma 3 in [3]) *For any arc  $u \in P_1$ ,*

$$DP_{(u_l,u_r)}[j, j'] = \max \begin{cases} DP_{(u_l+1,u_r-1)}[j+1, j' - 1] + \delta((S_1[u_l], S_1[u_r]), (S_2[j], S_2[j'])), \\ DP_{(u_l+1,u_r-1)}[j, j'], \\ DP_{(u_l,u_r)}[j+1, j'], \\ DP_{(u_l,u_r)}[j, j' - 1]. \end{cases}$$

Below we define three operations over the whole table  $DP_{(i,i')}$ , namely, EXTEND, MERGE, and ARC-MATCH.

**Definition 1** *If  $i'$  is free then given the table  $DP_{(i,i'-1)}$ ,  $DP_{(i,i')}$  can be computed by using Lemma 3.2.1. We define the computation of  $DP_{(i,i')}$  from  $DP_{(i,i'-1)}$  as the operation  $EXTEND(DP_{(i,i'-1)})$ .*

**Definition 2** Consider any arc  $s$ . The operation  $MERGE(DP_{(i,s_l-1)}, DP_{(s_l,s_r)})$  is defined to be the computation of the table  $DP_{(i,s_r)}$  given  $DP_{(i,s_l-1)}$  and  $DP_{(s_l,s_r)}$  using Lemma 3.2.2.

**Definition 3** Consider any arc  $s$ . The operation  $ARC-MATCH(DP_{(s_l+1,s_r-1)})$  is defined to be the computation of the table  $DP_{(s_l,s_r)}$  given  $DP_{(s_l+1,s_r-1)}$  using Lemma 3.2.3.

Fig. 3.1 describes the procedure  $WLCS(S_1, P_1, S_2)$  that computes  $DP_{(1,n)}[j, j']$  for all  $1 \leq j \leq j' \leq m$ . It is actually the algorithm in [3] expressed in terms of our defined operations on the DP tables. Given  $DP_{(1,n)}[j, j']$  and all its intermediary DP tables, an optimal alignment can then be retrieved via the standard traceback procedure.

The time and space complexity of  $WLCS(S_1, P_1, S_2)$  is analyzed by computing the contributions of the operations EXTEND, ARC-MATCH, and MERGE separately. First we analyze the time complexity of the algorithm. An EXTEND operation involves computing  $DP_{(i,i')}[j, j']$  from  $DP_{(i,i'-1)}[j, j']$  for all  $1 \leq j \leq j' \leq m$ . Since there are  $O(m^2)$   $(j, j')$  pairs to compute, each EXTEND operation takes  $O(m^2)$  time. Next, because EXTEND is applied only on free positions, whose number is bounded by  $O(n)$ , the total cost for all EXTEND operations is  $O(nm^2)$ . The analysis for the ARC-MATCH operation is similar to the one for EXTEND above except that ARC-MATCH is invoked only on arcs whose cardinality is also bounded by  $O(n)$  (since we assumed nested arc-annotation). Thus, it also takes  $O(nm^2)$  time for all ARC-MATCH calls. Each call to MERGE requires computing the maximum  $DP_{(i,i')}[j, j']$  by summing the values  $DP_{(i,i'')}[j, j'']$  and  $DP_{(i''+1,i')}[j''+1, j']$  where  $i''$  is fixed and  $j''$  is chosen from the range  $[j..j']$ . In the worst case, one would require  $O(m)$  time to compute  $DP_{(i,i')}[j, j']$  for a particular  $(j, j')$ . This yields  $O(m^3)$  time for a MERGE operation. Observing that the algorithm only invokes MERGE on arcs, the total contribution of MERGE is  $O(nm^3)$ . In total, the running time of the algorithm is  $O(nm^3)$ .

It is straightforward to see that  $EXTEND(DP_{(i,i'-1)})$  requires  $O(m^2)$  space as we only need  $O(m^2)$  space to store both  $DP_{(i,i'-1)}$  and the resulting  $DP_{(i,i')}$ . The same argument also applies to ARC-MATCH and MERGE (as for MERGE, we need space for three DP tables instead of two). But since [3] uses the standard traceback for inferring the secondary structure of the sequence  $S_2$ , one must store all intermediary

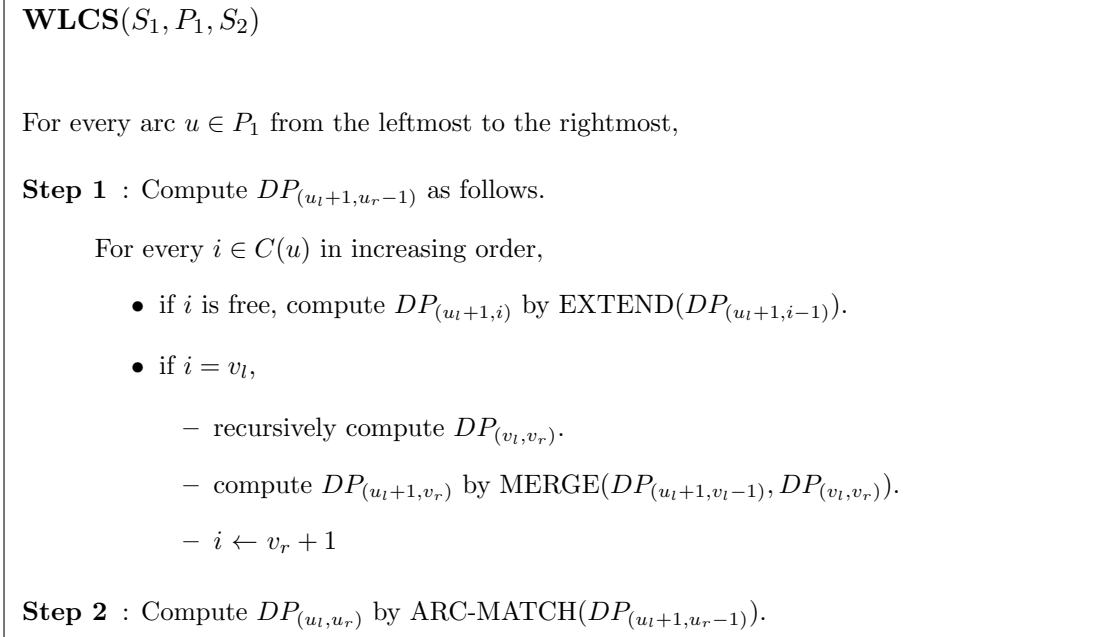


Figure 3.1: The algorithm from [3] described in terms of EXTEND, MERGE and ARC-MATCH operations. The two arc-annotated sequences  $S_1$  and  $S_2$  are of length  $n$  and  $m$ , respectively.  $P_1$  is the arc-annotation of  $S_1$ ; given the nested arc-annotation, the maximum number of arcs in  $P_1$  are bounded by  $O(n)$ . For any arc  $u \in P_1$ ,  $u_l$  is its left endpoint and  $u_r$  is its right endpoint.

DP tables computed by  $\text{WLCS}(S_1, P_1, S_2)$ . The size of the latter is bounded by  $O(nm^2)$  as the number of free positions and arcs are both bounded by  $O(n)$  and each DP table contains  $O(m^2)$  entries.

### 3.3 Our Algorithm’s Description and Analysis

#### 3.3.1 Running Time Improvement through Sparsification on the Dynamic Programming

The previous section shows that the bottleneck of the computation of the WLCS score is in the procedure MERGE. Here, we describe how to speed up the computation of MERGE by taking advantage of the properties of  $DP_{(i, i')}$ .

**Observation 1** For any  $i \leq i'$ ,  $DP_{(i, i')}$  satisfies the following properties.

1. In every row  $j$  of  $DP_{(i, i')}$ , the entries are monotonically increasing, i.e.,  $DP_{(i, i')}[j, j'] \leq DP_{(i, i')}[j, j' + 1]$ .

2. In every column  $j'$  of  $DP_{(i,i')}$ , the entries are monotonically decreasing, i.e.,  $DP_{(i,i')}[j, j'] \geq DP_{(i,i')}[j+1, j']$ .

The above observation motivates the following definition.

**Definition 4** [118] For every row  $j$  of  $DP_{(i,i')}$ , a position  $j^*$  satisfying  $j \leq j^* \leq m$  is called a row interval point if  $DP_{(i,i')}[j, j^* - 1] < DP_{(i,i')}[j, j^*]$ . (See Fig. 3.2)

**Definition 5** The set of row interval points  $j^*$  in the  $j^{\text{th}}$  row of  $DP_{(i,i')}$  that satisfy  $j^* \leq j'$  is denoted by  $RowIP_{(i,i';j,j')}$ .

**Lemma 3.3.1** For every  $j'' \in [j..j']$ , there exists a  $j^* \in RowIP_{(i,i';j,j')}$  such that  $DP_{(i,i')}[j, j^*] = DP_{(i,i')}[j, j'']$  and  $j^* \leq j''$ .

*Proof.* We know that the entries in any row of  $DP_{(i,i')}$  are monotonically increasing. Hence each new distinct entry will be greater than the entry preceding it. By its definition,  $RowIP_{(i,i';j,j')}$  covers all distinct entries in the interval  $[j..j']$ .  $\square$

**Lemma 3.3.2** Let  $\alpha = \max\{\beta, \alpha_1, \alpha_2, \alpha_3\}$ . Then  $|RowIP_{(i,i';j,j')}| \leq \min\{\alpha(i' - i + 1), (j' - j + 1)\}$ .

*Proof.* Since the row interval points are distinct,  $|RowIP_{(i,i';j,j')}|$  is clearly bounded above by  $j' - j + 1$ . Moreover, as we assume integer scores, the number of distinct interval points is also bounded above by the highest score possible from aligning  $S_1[i..i']$  with  $S_2[j..j']$ , which is equal to  $\min\{\alpha(i' - i + 1), \alpha(j' - j + 1)\}$ . By combining the terms, the lemma follows.  $\square$

In [3], for every  $(j, j')$  pair where  $j \leq j'$ , the procedure  $MERGE(DP_{(i,u_l-1)}, DP_{(u_l,u_r)})$  tries every possible  $j'' \in [(j-1)..j']$  to compute the one that maximizes the sum

$$DP_{(i,u_l-1)}[j, j''] + DP_{(u_l,u_r)}[j'' + 1, j']. \quad (3.1)$$

Given Lemma 3.3.2, we can see that there are at most  $(\min\{\alpha(i' - i + 1), (m - j + 1)\})$  row interval points in any row  $j$  of  $DP_{(i,i')}$ . The following lemma implies that it is unnecessary to consider all  $j'' \in [(j-1)..j']$  to find the maximum of (3.1).

**Lemma 3.3.3** *The equation from Lemma 3.2.2 can be computed using the following equation*

$$DP_{(i,u_r)}[j, j'] = \max_{j^* \in \{RowIP_{(i,u_l-1;j,j')} \cup \{j-1\}\}} \{DP_{(i,u_l-1)}[j, j^*] + DP_{(u_l,u_r)}[j^* + 1, j']\}.$$

*Proof.* Let us separate the range  $[(j-1)..j']$  into  $[(j-1)..(j-1)]$  and  $[j..j']$ . The lemma can be proven if we can show that, for every  $j'' \in [j..j']$ , there exists a  $j^* \in RowIP_{(i,u_l-1;j,j')}$  such that  $DP_{(i,u_l-1)}[j, j''] + DP_{(u_l,u_r)}[j'' + 1, j'] \leq DP_{(i,u_l-1)}[j, j^*] + DP_{(u_l,u_r)}[j^* + 1, j']$ . Note that, by Lemma 3.3.1, for each  $j'' \in [j..j']$ , there exists a  $j^* \in RowIP_{(i,u_l-1;j,j')}$  such that  $DP_{(i,u_l-1)}[j, j^*] = DP_{(i,u_l-1)}[j, j'']$  and  $j^* \leq j'' \leq j'$ . It follows that,

$$\begin{aligned} DP_{(i,u_l-1)}[j, j''] + DP_{(u_l,u_r)}[j'' + 1, j'] &= DP_{(i,u_l-1)}[j, j^*] + DP_{(u_l,u_r)}[j'' + 1, j'] \\ &\leq DP_{(i,u_l-1)}[j, j^*] + DP_{(u_l,u_r)}[j^* + 1, j'] \end{aligned}$$

since by Observation 1(2), we know that  $DP_{(u_l,u_r)}[j^* + 1, j'] \geq DP_{(u_l,u_r)}[j'' + 1, j']$ .  $\square$

Lemma 3.3.3 speeds up the computation time of  $DP_{(i,u_r)}[j, j']$  by considering only distinct values for the  $DP_{(i,u_l-1)}[j, j^*]$  terms (by choosing  $j^*$  from the set of  $RowIP_{(i,u_l-1;j,j')}$ ). We can still further improve the time complexity of MERGE by also considering only distinct values of  $DP_{(u_l,u_r)}[j^* + 1, j']$  given a particular choice of  $j^*$ .

Let us start with the following definitions.

**Definition 6** *Let us define the set*

$$\begin{aligned} S_{(i,i',i'';j,j')} &= \{j^* \in RowIP_{(i,i';j,j')} \cup \{j-1\} \mid j' \in RowIP_{(i'+1,i'';j^*+1,j')} \cup \{j^*\}\}, \\ S'_{(i,i',i'';j,j')} &= \{RowIP_{(i,i';j,j')} \cup \{j-1\}\} - S_{(i,i',i'';j,j')}, \end{aligned}$$

*Based on the set  $S$  and  $S'$  above, we define the following tables,*

$$\begin{aligned} P_{(i,i',i'')}[j, j'] &= \begin{cases} \max_{j^* \in S_{(i,i',i'';j,j')}} \{DP_{(i,i')}[j, j^*] + DP_{(i'+1,i'')}[j^* + 1, j']\} & \text{if } S_{(i,i',i'';j,j')} \neq \emptyset, \\ 0 & \text{otherwise} \end{cases} \\ P'_{(i,i',i'')}[j, j'] &= \begin{cases} \max_{j^* \in S'_{(i,i',i'';j,j')}} \{DP_{(i,i')}[j, j^*] + DP_{(i'+1,i'')}[j^* + 1, j']\} & \text{if } S'_{(i,i',i'';j,j')} \neq \emptyset, \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

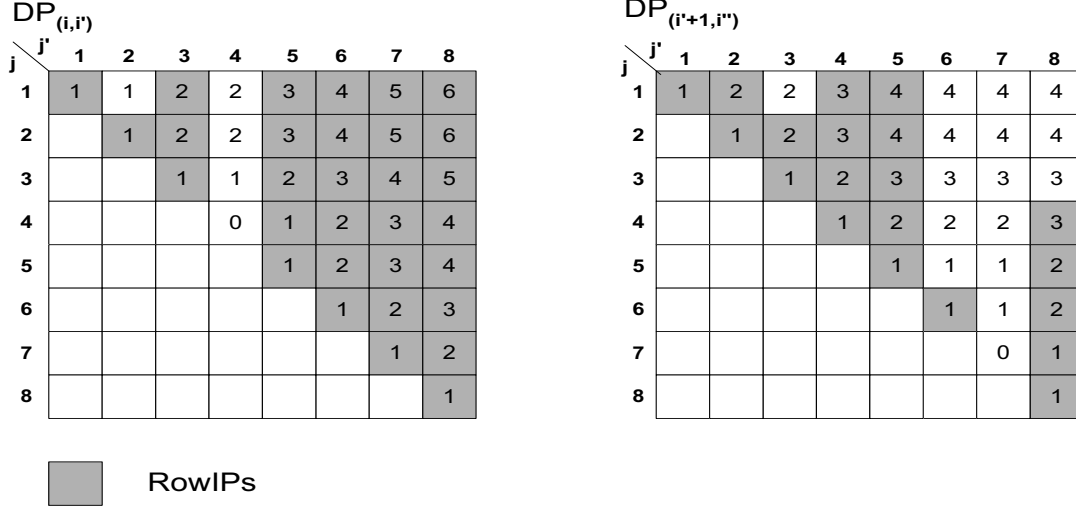


Figure 3.2: Illustration of the set  $S$ . The distinct scores in each row are highlighted in grey. From the figure we can see that  $RowIP_{(i,i';2,8)} = \{2, 3, 5, 6, 7, 8\}$  ( $j=2, j'=8$ ). Then, as defined, we have  $S_{(i,i',i'';2,8)} = \{3, 5, 6, 7, 8\}$  since  $j'=8$  and, for all  $j^* \in \{3, 5, 6, 7\}$ , we have 8 inside the set  $RowIP_{(i'+1,i'';j^*+1,8)}$ .

The set  $S_{(i,i',i'';j,j')}$  is actually a subset of the set  $RowIP_{(i,i';j,j')} \cup \{j-1\}$  where for each of its element  $j^*$ ,  $j'$  is in the set  $RowIP_{(i'+1,i'';j^*+1,j')} \cup \{j^*\}$ . Fig. 3.2 illustrates the definition of the set  $S_{(i,i',i'';j,j')}$ .

Given Definition 6 above, we can rewrite the equation in Lemma 3.3.3 into

$$DP_{(i,u_r)}[j, j'] = \max\{P_{(i,u_l-1,u_r)}[j, j'], P'_{(i,u_l-1,u_r)}[j, j']\}$$

In the following lemma, we claim that we only need to compute the value of  $DP_{(i,u_l-1)}[j, j^*] + DP_{(u_l,u_r)}[j^*+1, j']$  over  $j^* \in S_{(i,u_l-1,u_r;j,j')}$  instead of the whole  $RowIP_{(i,u_l-1;j,j')}$  for each  $j' \in [j..m]$ .

**Lemma 3.3.4** *When  $P_{(i,u_l-1,u_r)}[j, j'] \leq P'_{(i,u_l-1,u_r)}[j, j']$ , we have  $DP_{(i,u_r)}[j, j'] = DP_{(i,u_r)}[j, j' - 1]$ .*

*Proof.* Given  $P_{(i,u_l-1,u_r)}[j, j'] \leq P'_{(i,u_l-1,u_r)}[j, j']$ , we have  $DP_{(i,u_r)}[j, j'] = P'_{(i,u_l-1,u_r)}[j, j']$ . To prove this lemma we shall show that  $DP_{(i,u_r)}[j, j'-1] \leq P'_{(i,u_l-1,u_r)}[j, j']$  and  $P'_{(i,u_l-1,u_r)}[j, j'] \leq DP_{(i,u_r)}[j, j'-1]$ . The first one is trivial since, by Observation 1(1),  $DP_{(i,u_r)}[j, j'-1] \leq DP_{(i,u_r)}[j, j'] = P'_{(i,u_l-1,u_r)}[j, j']$ . Next we need to show that  $P'_{(i,u_l-1,u_r)}[j, j'] \leq DP_{(i,u_r)}[j, j'-1]$ . By its definition,  $\forall j^* \in S'_{(i,u_l-1,u_r;j,j')}$ , we have  $j^* < j'$  and  $j' \notin$

$RowIP_{(u_l, u_r; j^*+1, j')}$ . It follows that  $\forall j^* \in S'_{(i, u_l-1, u_r; j, j')}$ , we have  $DP_{(u_l, u_r)}[j^*+1, j'] = DP_{(u_l, u_r)}[j^*+1, j'-1]$ .

We further observe that

- $RowIP_{(i, u_l-1; j, j'-1)} = RowIP_{(i, u_l-1; j, j')} - \{j'\}$ ,
- $S'_{(i, u_l-1, u_r; j, j')} \subseteq RowIP_{(i, u_l-1; j, j')}$  and,
- $j' \notin S'_{(i, u_l-1, u_r; j, j')}$ ,

Thus, we can conclude that  $S'_{(i, u_l-1, u_r; j, j')} \subseteq RowIP_{(i, u_l-1; j, j'-1)}$ .

It follows that  $\max_{j^* \in S'_{(i, u_l-1, u_r; j, j')}} DP_{(i, u_l-1)}[j, j^*] + DP_{(u_l, u_r)}[j^*+1, j'] \leq \max_{j^* \in RowIP_{(i, u_l-1; j, j'-1)}} DP_{(i, u_l-1)}[j, j^*] + DP_{(u_l, u_r)}[j^*+1, j'-1]$ . Hence,  $P'_{(i, u_l-1, u_r)}[j, j'] \leq DP_{(i, i')}[j, j'-1]$ .  $\square$

**Corollary 3.3.5** *We can compute the value of  $DP_{(i, u_r)}[j, j']$  in Lemma 3.2.2 using the following equation*

$$DP_{(i, u_r)}[j, j'] = \max\{P_{(i, u_l-1, u_r)}[j, j'], DP_{(i, u_r)}[j, j'-1]\}.$$

The following lemma analyzes the complexity of the new MERGE operation.

**Lemma 3.3.6** *The complexity of the new MERGE operation is in  $O(\min\{\alpha(u_l - i), m\} \cdot \min\{\alpha|u|, m\} \cdot m) + O(m^2)$  time and  $O(m^2)$  space.*

*Proof.* By Corollary 3.3.5, we can compute  $DP_{(i, u_r)}[j, j']$  in constant time given that we have already computed the value of  $P_{(i, u_l-1, u_r)}[j, j']$ . A straightforward way to compute  $P_{(i, u_l-1, u_r)}[j, j']$  is, for a particular  $j'$ , compute the set  $S_{(i, u_l-1, u_r; j, j')}$  and use it to compute the former based on Definition 6. This would take  $O(\min\{\alpha(u_l - i), (j' - j)\} \cdot \min\{\alpha(u_r - u_l), (j' - j)\})$  time. Taking all possible  $j$  and  $j'$ , the running time will be in  $O(\min\{\alpha(u_l - i), (j' - j)\} \cdot \min\{\alpha(u_r - u_l), (j' - j)\}m^2)$ , which is unacceptable.

To avoid the need of computing  $S_{(i, u_l-1, u_r; j, j')}$ , we reverse the computational ordering of  $j^*$  and  $j'$ . Instead of computing the values of  $j^*$  for each  $j'$ ; for each  $j^* \in RowIP_{(i, u_l-1; j, m)} \cup \{j-1\}$ , we get the  $j' \in RowIP_{(u_l, u_r; j^*+1, m)} \cup \{j^*\}$  and, for all

```

MERGE( $DP_{(i,u_l-1)}$ ,  $DP_{(u_l,u_r)}$ )
1  Compute  $RowIP_{(i,u_l-1;j,m)}$  from  $DP_{(i,u_l-1)}$  for  $j = 1 \dots m$ 
2  Compute  $RowIP_{(u_l,u_r;j,m)}$  as above
3  Set  $P_{(i,u_l-1,u_r)}[j,j'] = 0$  for  $1 \leq j \leq j' \leq m$ 
4  for  $j = 1 \dots m$ 
5    for  $j^* \in RowIP_{(i,u_l-1;j,m)} \cup \{j-1\}$ 
6      for  $j' \in RowIP_{(u_l,u_r;j^*+1,m)} \cup \{j^*\}$ 
7         $P_{(i,u_l-1,u_r)}[j,j'] = \max\{P_{(i,u_l-1,u_r)}[j,j'], DP_{(i,u_l-1)}[j,j^*] + DP_{(u_l,u_r)}[j^*+1,j']\}$ 
8      endfor
9    endfor
10    $DP_{(i,u_r)}[j,j'] = \max\{P_{(i,u_l-1,u_r)}[j,j'], DP_{(i,u_r)}[j,j'-1]\}$ 
11 endfor
12 endfor

```

Figure 3.3: The pseudocode for the new MERGE operation. We have two DP tables,  $DP_{(i,u_l-1)}$  is the currently computed DP table and  $DP_{(u_l,u_r)}$  is the DP table of the arc  $u$  we wish to combine into the former to compute the merged  $DP_{(i,u_r)}$ .

such  $j'$ , update the value of  $P_{(i,u_l-1,u_r)}[j,j']$  whenever  $DP_{(i,u_l-1)}[j,j^*] + DP_{(u_l,u_r)}[j^*+1,j'] > P_{(i,u_l-1,u_r)}[j,j']$ . Effectively, for each  $j' \in RowIP_{(u_l,u_r;j^*+1,j')}$  for some  $j^* \in RowIP_{(i,u_l-1;j,j')}$ , the updating will compute the maximum value of  $DP_{(i,u_l-1)}[j,j^*] + DP_{(u_l,u_r)}[j^*+1,j']$  over all possible  $j^*$ . Note that we have to initialize the values in the table  $P_{(i,u_l-1,u_r)}$  to zero beforehand.

The number of such  $(j^*, j')$  pair is bounded by  $|RowIP_{(i,u_l-1;j,j')}| \cdot |RowIP_{(u_l,u_r;j^*+1,m)}|$  which is less than  $\min\{\alpha(u_l - i), m\} \cdot \min\{\alpha|u|, m\}$ . For each  $(j^*, j')$  pair, the sum  $DP_{(i,i')}[j,j^*] + DP_{(i'+1,i'')}[j^*+1,j']$  will only be computed once taking constant time. As there are  $m$  rows in  $P_{(i,u_l-1,u_r)}$ , its time complexity will then be in  $O(\min\{\alpha(u_l - i), m\} \cdot \min\{\alpha|u|, m\} \cdot m)$ .

The size of  $P_{(i,u_l-1,u_r)}$  is clearly in  $O(m^2)$ . Once we have computed  $P_{(i,u_l-1,u_r)}$ , we can compute the whole table of  $DP_{(i,u_r)}$  in  $O(m^2)$  time and space. By combining the complexity of the computation of both  $P_{(i,u_l-1,u_r)}$  and  $DP_{(i,u_r)}$ , the lemma follows.  $\square$

A MERGE operation can then be computed using the pseudocode in Fig. 3.3.



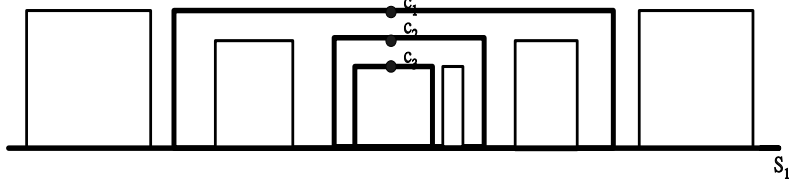


Figure 3.4: The core-path  $CP(c_1)$  is the ordered set  $\{c_1, c_2, c_3\}$

### Complexity Analysis of the Improved MERGE Operation

As the sparsification technique only optimized the MERGE operations, the computational resources required by all EXTEND and ARC-MATCH operations remain the same as in Zhang's algorithm (Fig. 3.1), i.e.,  $O(nm^2)$  for both time and space.

The previous section shows that each of the new  $MERGE(DP_{(i, u_l-1)}, DP_{(u_l, u_r)})$  operations requires  $O(\min\{\alpha(u_l - i), m\} \cdot \min\{\alpha|u|, m\} \cdot m) + O(m^2)$  time and  $O(m^2)$  space. We now consider the total time complexity of all MERGE operations. Let us start with some definitions to assist the analysis. The following is with respect to a *nested* arc-annotated structure.

**Definition 7** An arc  $u$  is a parent of an arc  $v$  (denoted by  $Parent(v)$ ) if  $u_l < v_l < v_r < u_r$  and there is no arc  $w$  such that  $u_l < w_l < v_l < v_r < w_r < u_r$ . Conversely,  $v$  is referred to as the child of the arc  $u$ . The set of children of an arc  $u$  is denoted by  $Children(u)$ .

**Definition 8** A terminal-arc is defined to be an arc that has no child. A core-arc, with respect to an arc  $u$ , is a child of  $u$  that has the biggest size (arbitrarily breaking ties). The latter is denoted as  $core-arc(u)$ . All other children of  $u$  are named the side-arcs and form the set  $side-arcs(u)$ .

**Definition 9** For any arc  $u \in P_1$ , the core-path  $CP(u)$  is an ordered set of core-arcs  $\{c_1, c_2, \dots, c_\ell\}$ , where  $c_1 = u$  and for any  $c_i, c_{i+1}$  is  $core-arc(c_i)$  (refer to Fig. 3.4).

**Lemma 3.3.7** For any arc  $u \in P_1$ , the time required by the MERGE operations on all of its children in  $Children(u)$  is in  $\min\{O(\alpha(|u| - |c|)x_u m) + O(|Children(u)|m^2), O(|Children(u)|m^3)\}$  where  $c$  is the core-arc of  $u$  and  $x_u = \min\{\alpha|u|, m\}$ .

*Proof.* The first observation is that MERGE only takes place when we encounter an arc as we try to extend the current DP table. Thus, the time required for applying MERGE on all arcs in  $Children(u)$  is (by Lemma 3.3.6)

$$\sum_{u' \in Children(u)} \left\{ O(\min\{\alpha(u'_l - u_l), m\} \cdot \min\{\alpha|u'|, m\} \cdot m) + O(m^2) \right\}.$$

The sum of the second term,  $O(m^2)$ , yields  $O(|Children(u)|m^2)$  while the sum of the first term ( $O(\min\{\alpha(u'_l - u_l), m\} \cdot \min\{\alpha|u'|, m\} \cdot m)$ ) gives several possible cases. When both  $\min\{\alpha(u'_l - u_l), m\} = m$  and  $\min\{\alpha|u'|, m\} = m$ , the first term is equal to  $O(m^3)$ . Summing over all children of  $u$  gives  $O(|Children(u)|m^3)$ .

Otherwise, let  $x_u = \min\{\alpha|u|, m\}$ . We need to show that the summation of the first term is equal to  $O(\alpha(|u| - |c|)x_um)$ . It is easy to show that  $O(\min\{\alpha(u'_l - u_l), m\} \cdot \min\{\alpha|u'|, m\} \cdot m)$  is bounded above by  $O(\alpha|u'|x_um)$ . Summing the value over  $Children(u)$  only gives the bound of  $O(\alpha|u|x_um)$ . To have a tighter bound, we separately consider the following cases;

1. The case when  $|c| \leq \frac{|u|}{2}$ . For this case,  $|u| - |c| > \frac{|u|}{2}$ . Hence,  $2(|u| - |c|) > |u|$  and we have  $O(\alpha|u|x_um) = O(\alpha(|u| - |c|)x_um)$ .
2. When  $|c| > \frac{|u|}{2}$ , applying MERGE on  $DP_{(u_l+1, c_l-1)}$  and  $DP_{(c_l, c_r)}$  will take at most  $O(\alpha(|u| - |c|)x_um)$  time since  $\min\{(c_l - u_l), m\} \leq \min\{(|u| - |c|), m\}$  and  $\min\{|c|, m\} \leq x_u$ . The remaining MERGE operations on the side-arcs will require at most  $O(\alpha(|u| - |c|)x_um)$  time too since their total size is bounded by  $|u| - |c|$ . Hence, in this case, the total time required is also bounded by  $O(\alpha(|u| - |c|)x_um)$ .

□

**Lemma 3.3.8** *The time required by all MERGE operations during the execution of  $WLCS(S_1, P_1, S_2)$  is in  $\min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}$ .*

*Proof.* For convenience of notation, let us include an imaginary arc  $r = (0, n+1)$  into  $P_1$ . Since the string  $S_1$  is indexed from 1 to  $n$ ,  $S_1[0]$  and  $S_1[n+1]$  are undefined and hence  $r$  will never be matched to any position in  $S_2$ . Note that  $r$  is the outermost arc and  $|r| = O(n)$ . Next, we define the set  $Arc(y)$ , where  $y \in P_1$ , to be the set  $\{u \in P_1 | y_l < u_l < u_r < y_r\}$ , that is, the set of all arcs in  $P_1$  whose span is within  $[y_l..y_r]$ .

Finally, based on Lemma 3.3.7, the time complexity  $T(y)$  of all MERGE operations during the computation of  $W LCS(S_1[y_l..y_r], P_1, P_2)$  can be computed by

$$T(y) = \sum_{\substack{u \in \text{Arc}(y) \\ c = \text{core-arc}(u)}} \min \left\{ O(\alpha(|u| - |c|)x_u m) + O(|\text{Children}(u)|m^2), O(|\text{Children}(u)|m^3) \right\}. \quad (3.2)$$

$$\begin{aligned} &= \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + \\ &\quad \sum_{\substack{u \in CP(y) \\ c = \text{core-arc}(u)}} \min \left\{ O(\alpha(|u| - |c|)x_u m) + O(|\text{Children}(u)|m^2), O(|\text{Children}(u)|m^3) \right\}. \end{aligned} \quad (3.3)$$

where  $x_u = \min\{\alpha|u|, m\}$ . We can derive (3.3) from (3.2) using the fact that

$$\text{Arc}(y) = CP(y) \cup \left( \bigcup_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} \text{Arc}(s) \right).$$

Next we need to examine the following possible values of  $\min \left\{ O(\alpha(|u| - |c|)x_u m) + O(|\text{Children}(u)|m^2), O(|\text{Children}(u)|m^3) \right\}$ .

1.  $\min \left\{ O(\alpha(|u| - |c|)x_u m) + O(|\text{Children}(u)|m^2), O(|\text{Children}(u)|m^3) \right\} = O(\alpha(|u| - |c|)x_u m) + O(|\text{Children}(u)|m^2)$ . For this case we have

$$\begin{aligned} T(y) &= \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + \sum_{\substack{u \in CP(y) \\ c = \text{core-arc}(u)}} O(\alpha(|u| - |c|)x_u m) + \sum_{u \in CP(y)} O(|\text{Children}(u)|m^2). \\ &\leq \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + \sum_{\substack{u \in CP(y) \\ c = \text{core-arc}(u)}} O(\alpha(|u| - |c|)x_y m) + \sum_{u \in CP(y)} O(|\text{Children}(u)|m^2). \end{aligned} \quad (3.4)$$

$$= \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + O(\alpha|y|x_y m) + \sum_{u \in CP(y)} O(|\text{Children}(u)|m^2). \quad (3.5)$$

We derive (3.5) from (3.4) by summing the telescoping series  $\sum_{\substack{u \in CP(y) \\ c = \text{core-arc}(u)}} O(\alpha(|u| - |c|)x_y m)$ . Next, depending on  $x_y$ ,

- (a)  $x_y = \alpha|y|$

$$T(y) = \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + O(\alpha^2|y|^2 m) + \sum_{u \in CP(y)} O(|\text{Children}(u)|m^2).$$

Since  $|s| \leq \frac{|y|}{2}$ ,  $\sum_s |s| < |y|$  and the combination of the  $\sum_{u \in CP(y)} O(|Children(u)|)$  in the whole recurrence tree is bounded above by the total number of arcs in  $y$  which is  $O(|y|)$ , the recurrence yields a decreasing geometric series that sums up to  $O(\alpha^2 |y|^2 m) + O(|y| m^2)$  time complexity.

(b)  $x_y = m$

$$T(y) = \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + O(\alpha |y| m^2) + \sum_{u \in CP(y)} O(|Children(u)| m^2).$$

As  $|s| \leq \frac{|y|}{2}$ , the depth of recursion tree for the recurrence above is at most  $O(\log |y|)$ . And since  $\sum_s |s| < |y|$ , each level in the recursion tree will require less than  $O(\alpha |y| m^2)$  time. Thus, in total, the time complexity of this case is  $O(\alpha |y| m^2 \log |y|)$

$$2. \min \left\{ O(\alpha(|u|-|c|)x_u m) + O(|Children(u)| m^2), O(|Children(u)| m^3) \right\} = O(|Children(u)| m^3).$$

In this case,

$$T(y) = \sum_{\substack{u \in CP(y) \\ s \in \text{side-arcs}(u)}} T(s) + \sum_{u \in CP(y)} O(|Children(u)| m^3). \quad (3.6)$$

which, by the bound on number of arcs under  $y$ , yields  $T(y) = O(|y| m^3)$ .

When  $y = r$ , that is,  $|y| = O(n)$ , we conclude that  $T(r) = \min\{O(\alpha^2 n^2 m + n m^2), O(\alpha n m^2 \log n), O(n m^3)\}$ .  $\square$

**Lemma 3.3.9** *Using the new MERGE operation,  $WLCS(S_1, P_1, S_2)$  runs in  $\min\{O(\alpha^2 n^2 m + n m^2), O(\alpha n m^2 \log n), O(n m^3)\}$  time and  $O(n m^2)$  space.*

*Proof.* As explained earlier, the operations EXTEND and ARC-MATCH both require  $O(n m^2)$  time while the time complexity of MERGE is  $\min\{O(\alpha^2 n^2 m + n m^2), O(\alpha n m^2 \log n), O(n m^3)\}$  by Lemma 3.3.8. Combining them will yield the time complexity stated in the lemma.

For the space complexity, assuming standard traceback, we have shown that EXTEND and ARC-MATCH operations will need  $O(n m^2)$  space. A single MERGE operation will need  $O(m^2)$  space as proven in Lemma 3.3.6. As MERGE is only applied on arcs, the total number of tables resulting from all MERGE operations is at most  $O(n)$ . The lemma thus follows.  $\square$

### 3.3.2 Using Less Space in the Computation of the WLCS Score

In some cases, one is only interested to find the WLCS score. In this case, one would naturally expect a more space-efficient version of the WLCS routine as it is unnecessary to store old DP tables for traceback. Let us name such procedure as *the score-only WLCS*( $S_1, P_1, S_2$ ). It turns out that, using the original algorithm of Zhang [3], the space complexity is still bounded by  $\Omega(nm^2)$  which is shown by the following lemma.

**Lemma 3.3.10** *Using the original algorithm in [3] combined with the newly improved MERGE operation, the score-only WLCS( $S_1, P_1, S_2$ ) requires  $\Omega(nm^2)$  space in the worst case.*

*Proof.* To compute the score-only WLCS( $S_1, P_1, S_2$ ), we only have to provide the space to perform the DP table operations, namely EXTEND, ARC-MATCH and MERGE and keep only the most current tables. As explained in Section 3.2.2, computing  $DP_{(i,i')} = \text{EXTEND}(DP_{(i,i'-1)})$  only requires  $O(m^2)$  space provided that  $DP_{(i,i'-1)}$  is already available when EXTEND is invoked. This condition is true for EXTEND and ARC-MATCH as we always compute  $DP_{(i,i'-1)}$  before  $DP_{(i,i')}$  and  $DP_{(u_l+1,u_r-1)}$  before  $DP_{(u_l,u_r)}$ .

But this is not quite the same for MERGE operations. As described in [3], the routine WLCS( $S_1, P_1, P_2$ ) computes the DP tables according to the *post-order* of the nodes in the tree representing the sequence with the secondary structure. Given the post-order, whenever we execute MERGE ( $DP_{(i,i'-1)}, DP_{(i',i'')}$ ), we would have computed  $DP_{(i,i'-1)}$  but not  $DP_{(i',i'')}$ . While computing the latter, one must temporarily store  $DP_{(i,i'-1)}$  in order to be able to finish the execution of the MERGE operation later. Note that the same kind of event could also take place during the computation of  $DP_{(i',i'')}$ . In the case of a skewed tree (see Fig. 3.5), the number of temporarily stored DP tables can reach  $\Omega(n)$  (around  $\frac{n}{3}$ ). Hence,  $\Omega(nm^2)$  space is required.  $\square$

### Space Complexity Improvement using Recursive DP on the Core Paths

This subsection will introduce a more-space efficient algorithm  $\text{WLCS}_r(S_1, P_1, S_2)$  that computes the WLCS score using a carefully designed recursive dynamic programming algorithm. This improved algorithm guarantees that each MERGE operation is applied

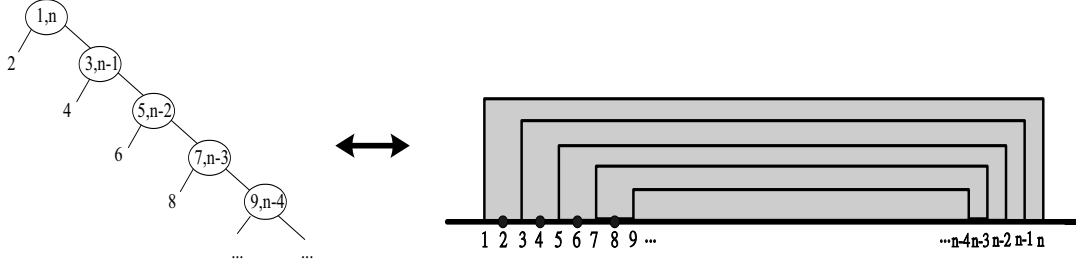


Figure 3.5: An example of arc-annotation on which the algorithm in [3] requires  $\Omega(nm^2)$  space to compute the *score-only*  $WLCS(S_1, P_1, S_2)$ . Note that the *post-ordering* forces the algorithm to compute the DPs for all the leaves before the internal nodes.

only to *side-arcs* where, by definition, the size of each side arc is at most half of the size of its parent.

$WLCS_r(S_1, S_2)$  first finds the largest arc  $u$  in  $[1..n]$  and processes every core-arc  $c \in CP(u)$  from the innermost to the outermost. As a special case, for the innermost core-arc  $t \in CP(u)$  (which is a terminal arc),  $DP_{(t_l, t_r)}$  can be computed without the MERGE operation. For the remaining core-arcs  $c$ ,  $DP_{(c_l, c_r)}$  will be computed using a *two-partition computation*. Let  $c'$  be core-arc( $c$ ) for an arc  $c$ . Due to the bottom-up ordering,  $DP_{(c'_l, c'_r)}$  is computed before  $DP_{(c_l, c_r)}$ . We first compute the value of  $DP_{(c_l+1, c'_l-1)}$  (the *LEFT Part* phase) using EXTEND and MERGE operations. Given  $DP_{(c_l+1, c'_l-1)}$ , we proceed to compute  $DP_{(c'_l, c_r-1)}$  (the *RIGHT Part* phase). In both phases, whenever we encounter a side-arc  $s$ , we first compute  $DP_{(s_l, s_r)}$  by recursively calling  $WLCS_r(S_1[s_l..s_r], P_1, S_2)$ . Then we apply MERGE to combine  $DP_{(s_l, s_r)}$  into the currently computed DP table. Having completed the computation of both phases, we apply MERGE on  $DP_{(c_l+1, c'_l-1)}$  and  $DP_{(c'_l, c_r-1)}$  to compute  $DP_{(c_l+1, c_r-1)}$ . Finally,  $DP_{(c_l, c_r)}$  is obtained by  $ARC-MATCH(DP_{(c_l+1, c_r-1)})$ .

If  $(1, n) \in P_1$ , then the largest arc  $u$  must be  $(1, n)$  and we are done. Otherwise, we need to compute  $DP_{(1, n)}$  using the same two-part computation technique: first compute  $DP_{(1, u_l-1)}$ , followed by  $DP_{(u_l, n)}$ , and then obtain  $DP_{(1, n)}$  by  $MERGE(DP_{(1, u_l-1)}, DP_{(u_l, n)})$ .

**Lemma 3.3.11** *Computing  $WLCS_r(S_1, P_1, S_2)$  requires  $\min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}$  time.*

*Proof.* As EXTEND and ARC-MATCH are still applied on free positions and arcs

in  $S_1$ , respectively, the running time complexity of both operations are still the same as the one in Lemma 3.3.9 which are both in  $O(nm^2)$ .

Note that MERGE is now invoked on all arcs that belong to the set  $\text{side-arc}(u)$  for some arc  $u \in P_1$  and on the merging of the LEFT part and the RIGHT part of all non-terminal arcs. Lemma 3.3.6 has showed that  $\text{MERGE}(DP_{(i,u_l-1)}, DP_{(u_l,u_r)})$  takes  $O(\min\{\alpha(u_l - i), m\} \cdot \min\{\alpha|u|, m\} \cdot m) + O(m^2)$  time to compute. Include an imaginary arc  $r = (0, n + 1)$  into  $P_1$ . Defining  $T(u)$  ( $u \in P_1$ ) as the total time complexity of MERGE during the computation of  $\text{WLCS}_r(S_1[u_l..u_r], P_1, P_2)$ , we can compute the total time complexity of all MERGE invocation in  $\text{WLCS}_r(S_1, P_1, S_2)$  by

$$T(r) = \sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} \left( T(s) + O(\min\{\alpha(|c| - |s|), m\} \cdot \min\{\alpha|s|, m\} \cdot m) + O(m^2) \right)^* + \sum_{\substack{c' = \text{core-arc}(c) \\ c \in CP(r)}} \left( O(\min\{\alpha(c'_l - c_l), m\} \cdot \min\{\alpha(c_r - c'_l), m\} \cdot m) + O(m^2) \right)^\dagger \quad (3.7)$$

$$\begin{aligned} &\leq \sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} T(s) + \sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} O(\min\{\alpha n, m\} \cdot \min\{\alpha|s|, m\} \cdot m) + \\ &\quad \sum_{\substack{c' = \text{core-arc}(c) \\ c \in CP(r)}} O(\min\{\alpha(c'_l - c_l), m\} \cdot \min\{\alpha n, m\} \cdot m) + \\ &\quad \sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} O(m^2) + \sum_{\substack{c' = \text{core-arc}(c) \\ c \in CP(r)}} O(m^2). \end{aligned} \quad (3.8)$$

$$= \min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}. \quad (3.9)$$

To obtain (3.9) from (3.8), we make use the following facts

1.  $\sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} |s| = \sum_{\substack{c' = \text{core-arc}(c) \\ c \in CP(r)}} (c'_l - c_l) = O(n)$  since, in all recursion level, all side-arcs  $s \in \text{side-arc}(c)$ , where  $c \in CP(r)$ , and the ranges  $[c_l..c'_l]$  are non-overlapping. Hence, the sums of the term  $\sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} O(\min\{\alpha n, m\} \cdot \min\{\alpha|s|, m\} \cdot m)$  and  $\sum_{\substack{c \in CP(r) \\ c' = \text{core-arc}(c)}} O(\min\{\alpha(c'_l - c_l), m\} \cdot \min\{\alpha n, m\} \cdot m)$  in all recursion level would both be bounded by  $\min\{O(\alpha^2 n^2 m), O(\alpha nm^2 \log n), O(nm^3)\}$  (following a similar proof as in Lemma 3.3.8).

---

\*The time needed to compute  $DP_{(s_l, s_r)}$  of side-arc  $s$  recursively and applying  $\text{MERGE}(DP_{(i, s_l-1)}, DP_{(s_l, s_r)})$ .

†The time needed to compute the merging of the tables computed by the two partition computation *i.e.*  $\text{MERGE}(DP_{(c_l, c'_l-1)}, DP_{(c'_l, c_r)})$ .

2. We can see that  $\sum_{\substack{s \in \text{side-arc}(c) \\ c \in CP(r)}} m^2 + \sum_{\substack{c' = \text{core-arc}(c) \\ c \in CP(r)}} m^2 = \sum_{c \in CP(r)} |\text{Children}(c)| m^2$ .  
Summing the term  $\sum_{c \in CP(r)} |\text{Children}(c)| m^2$  over all recursion level will yield the bound of  $O(nm^2)$ .

□

**Lemma 3.3.12**  $WLCS_r(S_1, P_1, S_2)$  uses  $\min\{O(m^2 \log n), O(m^2 + \alpha mn)\} + O(n)$  space.

*Proof.*

Referring back to Lemma 3.3.10, we only need  $O(m^2)$  to store the information needed to accomplish all EXTEND and ARC-MATCH operations. As for the MERGE operations, when there is no recursive call involved (the execution of MERGE on the LEFT and RIGHT parts), the space requirement is also in  $O(m^2)$ . In the recursive call, we now have managed to enforce a new computational ordering instead of using the original post-order (Lemma 3.3.10). Using the ordering given by the core-path in the annotation tree, Lemma 3.3.11 had shown that the latter guarantees  $O(\log n)$  recursion level. Hence the number of temporarily stored  $DP(i, s_l - 1)$  ( $s$  is a side-arc) during the recursive call to compute  $DP_{(s_l, s_r)}$  will not exceed  $O(\log n)$  as well. Storing only the row interval points takes  $O(\min\{\alpha(s_l - i)m, m^2\})$  space (by Lemma 3.3.2) (with  $O(m^2)$  time overhead for computing the set RowIP from/to the DP table). When  $O(\min\{\alpha(s_l - i)m, m^2\}) = O(m^2)$ , the space complexity is  $O(m^2 \log n)$ . For the other case, we further claim that the space required is smaller than  $O(\alpha nm)$  since, in each recursion level  $x$ , we only store  $DP_{(i_x, s_{l_x} - 1)}$  where all of the intervals  $[i_x \dots s_{l_x} - 1]$  are disjoint. Hence,  $\sum_x O(\alpha(s_{l_x} - i_x)m) \leq O(\alpha nm)$ . Combining the two cases along with the space complexity of EXTEND, ARC-MATCH, we have  $\min\{O(m^2 \log n), O(m^2 + \alpha mn)\}$ . Finally, we add the space needed to store  $S_1, S_2$  and  $P_1$  which is in  $O(n + m)$ . The lemma follows. □

### 3.3.3 Tackling Both the Time and Space Complexity Bound: a Hirschberg-like Traceback Algorithm

The previous section presents an algorithm  $WLCS_r(S_1, P_1, S_2)$  to compute the WLCS score in  $\min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}$  time and  $\min\{O(m^2 \log n + n), O(m^2 + \alpha mn)\}$  space. Following the idea of Hirschberg in [110], this section presents an algorithm that computes the optimal WLCS alignment between  $(S_1, P_1)$  and  $(S_2, P_2)$



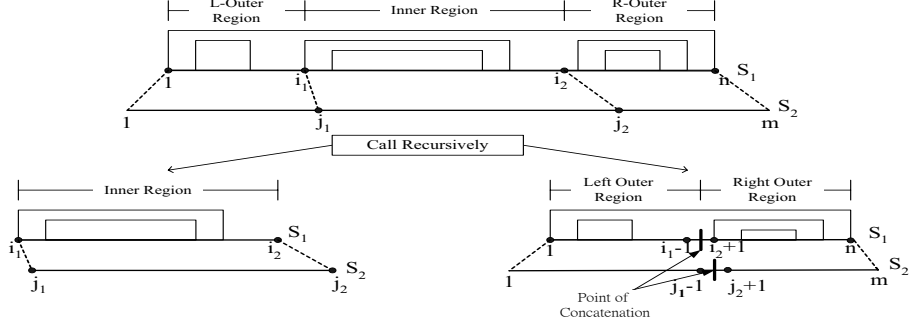


Figure 3.6: The recursion on the partitioned continuous region by Lemma 3.3.14. The recursive call on the inner region is exactly the same as the the previous recursive level. The call on the outer region have a requirement that the concatenation point be aligned to each other.

among all possible  $P_2$  within the same time and space complexity. The outline of the algorithm is as follows.

1. Divide  $S_1$  into a constant number of non-overlapping regions  $S_{11}, S_{12}, \dots, S_{1c}$ .
2. For each region  $S_{1i}$ , find the region  $S_{2i}$  in  $S_2$  such that the optimal WLCS alignment will align  $S_{1i}$  to  $S_{2i}$ .
3. Recursively compute the optimal WLCS alignments between  $S_{1i}$  and  $S_{2i}$  for  $i = 1, 2, \dots, c$ .

To do the first step, since  $S_1$  is arc-annotated, we divide  $S_1$  in such a way that we do not break any arc in  $P_1$ . The solution is to divide  $S_1$  into *inner* and *outer* regions so that, for any particular arc, both of its endpoints are in the same region. Given two points  $i_1$  and  $i_2$ ,  $1 \leq i_1 \leq i_2 \leq n$ , the *inner* region with respect to  $i_1$  and  $i_2$  is  $S_1[i_1..i_2]$  and the *outer* region is the concatenation of  $S_1[1..(i_1 - 1)]$  and  $S_1[(i_2 + 1)..n]$  (see Fig. 3.6). The latter is also referred as a *gapped* region since it has a discontinuous interval ( $S_1[i_1..i_2]$  is removed). Let  $\star$  be a special character that represents the gap in the sequence such that the gapped region can be written as  $S_1[1..(i_1 - 1)] \star S_1[(i_2 + 1)..n]$ . If a region has no gap in it, we say it is *continuous*. We shall show that we can bound the size of each region by  $\phi n$  for some constant  $\phi$ ,  $0 < \phi < 1$ .

**Lemma 3.3.13** *Given a nested arc-annotated sequence  $S_1$  of length  $n$ , we can compute two positions  $i_1$  and  $i_2$ ,  $1 \leq i_1 \leq i_2 \leq n$  in  $O(n)$  time and space, such that  $i_1$  and  $i_2$  satisfy*

1.  $\frac{n}{3} \leq i_2 - i_1 + 1 \leq \frac{2n}{3}$ ,
2.  $i_1$  and  $i_2$  are covered by the same arc  $u$ , or both are not covered by any arc,
3.  $i_1$  is either a free position or the left endpoint of some arc  $u' \in \text{Children}(u)$
4.  $i_2$  is either a free position or the right endpoint of some arc  $u'' \in \text{Children}(u)$ .

*Proof.* Define an imaginary arc  $r = (0, n + 1)$ . Find a pair of core-arcs  $c, c' \in CP(r)$  such that  $c' = \text{core-arc}(c)$ ,  $|c'| \leq \frac{2n}{3}$  and  $|c| > \frac{2n}{3}$  ( $c$  could be  $r$ ). When  $c$  is a terminal arc,  $i_1$  and  $i_2$  can be computed directly by choosing any two positions with distance at least  $\frac{n}{3}$  and at most  $\frac{2n}{3}$  in  $[c_l..c_r]$ .

Otherwise, if  $\frac{n}{3} \leq |c'| \leq \frac{2n}{3}$ , then we can use  $c'_l$  and  $c'_r$  as  $i_1$  and  $i_2$  (they are both covered by the core-arc  $c$ ,  $i_1$  is a left endpoint, and  $i_2$  is a right endpoint). Else if  $|c'| < \frac{n}{3}$ , we first set  $i_1$  and  $i_2$  to  $c'_l$  and  $c'_r$  and increase the range  $[i_1..i_2]$  by either increasing  $i_2$  or decreasing  $i_1$ . Let us consider the case of increasing  $i_2$ . Suppose  $i_2 + 1$  is a free position, then we can increase  $i_2$  by 1. Else if  $i_2 + 1$  is a left endpoint of some side-arc  $s \in \text{side-arc}(c)$ , then setting  $i_2 = s_r$  will increase  $i_2$  by  $|s|$ . Since  $|s| < |c'| < \frac{n}{3}$ , we guarantee that  $|s| + |c'| < \frac{2n}{3}$ .

Within this level of granularity, we can always extend the range  $[i_1..i_2]$  until we have  $\frac{n}{3} \leq i_2 - i_1 + 1 \leq \frac{2n}{3}$ . At the same time, we will satisfy the remaining constraints since  $i_2$  are chosen only from  $C(c)$  and  $i_2$  is never the left endpoint of any arc. The case of decreasing  $i_1$  can be proven similarly. The time required by the steps above is at most  $O(|CP(r)|) + O(|c|) = O(n)$  since finding  $c$  and  $c'$  takes  $O(|CP(r)|)$ , finding  $i_1$  and  $i_2$  takes  $O(|c|)$  time and both  $O(|CP(r)|)$  and  $O(|c|)$  are at most in  $O(n)$ . All these operations can be performed in  $O(n)$  space since we only need to store  $S_1$  and  $P_1$ .  $\square$

**Lemma 3.3.14** *We can always partition a continuous region into 2 non-overlapping subregions, where one of them is continuous and the other is gapped, in  $O(n)$  time and space. Every subregion's size is at most  $\frac{2}{3}$  of the original region.*

*Proof.* The proof of this lemma follows directly from Lemma 3.3.13.  $\square$

**Definition 10** *Let the ancestors of an arc  $u$  be defined as the ordered set  $A(u) = \{u_1, u_2, u_3, ..u_\ell\}$  where  $u_1 = u$  and  $u_{i+1} = \text{Parent}(u_i)$ . Let the least common ances-*

tor of the arcs  $u$  and  $v$ , denoted by  $LCA(u, v)$ , be the arc  $w \in A(u) \cap A(v)$  where  $|w|$  is minimal.

**Lemma 3.3.15** *We can always partition a gapped region into at most 4 non-overlapping subregions in  $O(n)$  time and space. Every subregion's size is at most  $\frac{2}{3}$  of the original region.*

*Proof.* Let  $S_1[i_1..i_2]$  be a gapped region. First, as in Lemma 3.3.14, we compute the points  $i'_1$  and  $i'_2$  such that  $\frac{(i_2-i_1+1)}{3} \leq i'_2 - i'_1 + 1 \leq \frac{2(i_2-i_1+1)}{3}$ . Having computed such  $i'_1$  and  $i'_2$ , we can guarantee that the size of  $(i_2 - i_1 + 1) - (i'_2 - i'_1 + 1) \leq \frac{2(i_2-i_1+1)}{3}$ . Let  $c$  and  $c'$  be the core-arcs where  $c' = \text{core-arc}(c)$  and  $c_l < i'_1 \leq c'_l < c'_r \leq i'_2 < c_r$ . Further, let the position of the special gap character ' $\star$ ' in  $S_1[i_1..i_2]$  be denoted by  $g$ . Based several possible position of  $g$  with respect to  $i'_1$ ,  $i'_2$  and  $c$ ; we have the following possible cases:

- $i'_1 \leq g \leq i'_2$ . We have two *gapped* subproblems,  $S_1[i'_1..i'_2]$  with  $g$  in it and  $S_1[i_1..(i'_1 - 1)] \star S_1[(i'_2 + 1)..i_2]$ .
- $c_l < g < i'_1$  or  $i'_2 < g < c_r$ . We will have one *continuous* region and two *gapped* regions. It is quite clear that the *continuous* region is  $S_1[i'_1..i'_2]$ . As for the gapped region, let us first consider the case where  $c_l < g < i'_1$ . If  $g \in C(c)$ , that is,  $g$  is a free position covered by  $c$ , then we have the *gapped* region  $S_1[g..(i'_1 - 1)]$  and  $S_1[i_1..(g - 1)] \star S_1[(i'_2 + 1)..i_2]$ . Else, if  $g$  is covered by some arc  $s$ , that is  $g \in C(s)$ , we find the ancestor of  $s$  that is a child of  $c$ . The latter is the arc  $s'$  such that  $s' \in (A(s) \cap \text{Children}(c))$ . Then the first *gapped* region will be  $S_1[s'_l..(i'_1 - 1)]$  and the second will be  $S_1[i_1..(s'_l - 1)] \star S_1[(i'_2 + 1)..i_2]$ . The case where  $i'_2 < g < c_r$  can be handled similarly.
- $g < c_l$  or  $g > c_r$ . In this case, we will have one *continuous* region,  $S_1[i'_1..i'_2]$ . In addition, we have three *gapped* regions. Suppose  $g < c_l$ . Let  $s$  be the arc that covers the position  $g$ . Let  $c'' = LCA(s, c)$ . It is clear that  $c''$  is a core-arc too. Next, let  $c''' = \text{core-arc}(c'')$  and  $s'$  be the arc in  $A(s) \cap \text{Children}(c'')$ . Now, we can readily define the gapped subproblems to compute in the next recursion. They are  $S_1[c'''_l..(i'_1 - 1)] \star S_1[(i'_2 + 1)..c'''_r]$ ,  $S_1[s'_l..(c'''_l - 1)]$  and  $S_1[i_1..(s'_l - 1)] \star S_1[(c'''_r + 1)..i_2]$ .

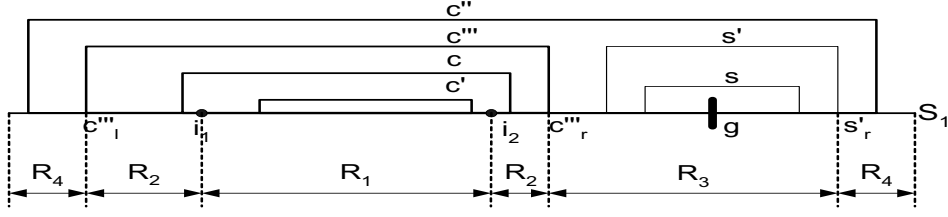


Figure 3.7: The figure describes the partitioning of  $S_1$  for the case where  $g > c_r$ . For the sake of clarity, the regions are drawn connected to each other. Note that, actually, the regions  $R_1$ ,  $R_2$ ,  $R_3$  and  $R_4$  are disjoint (not sharing their endpoints).

Again, the case where  $g > c_r$  can be computed in the same fashion. Fig. 3.7 illustrates the partitioning of  $S_1$  in the case of  $g > c_r$ .

The running time of this case is still bounded by  $O(n)$  since finding  $i'_1, i'_2$ , and the LCA of any two arcs requires at most  $O(n)$  and they are executed in constant number of times. For the space requirement, again we will only need  $O(n)$  space to store  $S_1$  and  $P_1$ .  $\square$

From Lemmas 3.3.14 and 3.3.15, we can conclude that the computational complexity of the first step of our new algorithm is  $O(n)$ . After dividing  $S_1$  into at most 4 subregions, where each is denoted by  $S_{1i}$  for  $i \leq 4$ , we now need to compute the regions  $S_{2i}$  in  $S_2$  to which the subregions  $S_{1i}$  is aligned by the optimal WLCS alignment. To do that, we will compute the positions in  $S_2$  where the boundaries of each region are aligned to. We shall first show that we can compute such an alignment for one single position  $p$  in  $S_1$ . By definition,  $DP_{(i,i')}[j, j']$  is the WLCS score produced by the optimal alignment between  $S_1[i..i']$  and  $S_2[j..j']$ . Now, for each entry  $DP_{(i,i')}[j, j']$  in the table  $DP_{(i,i')}$  where  $i \leq p \leq i'$ , we compute the position  $q$ ,  $j \leq q \leq j'$ , such that either  $p$  is aligned to  $q$  or  $p$  is aligned to  $\sqcup$  and  $[i..p-1]$  is aligned to  $[j..q]$ . We store such positions in a two dimensional table  $A_{(i,i')}^p$  which is defined as follows,

**Definition 11** For  $i \leq p \leq i'$  and  $j \leq q \leq j'$ , we define

$$A_{(i,i')}^p[j, j'] = \begin{cases} q & \text{if } p \text{ is aligned to } q \text{ by } DP_{(i,i')}[j, j'], \\ -q & \text{if } p \text{ is aligned to } \sqcup \text{ and } [i..p-1] \text{ is aligned to } [j..q] \text{ by } DP_{(i,i')}[j, j'], \\ 0 & \text{if } DP_{(i,i')}[j, j'] \text{ does not align } [i..p] \text{ to any position in } S_2[j..j']. \end{cases}$$

During the computation of WLCS, the only time we will align a position  $p$  with some position  $q$  in  $S_2$  is when we apply either  $\chi(S_1[p], S_2[q])$  (when  $p$  is free),  $\delta((S_1[p], \dots), (S_2[q], \dots))$ , or  $\delta(\dots, S_1[p]), (\dots, S_2[q])$  (when  $p$  is an arc endpoint).

**Lemma 3.3.16** *If  $p$  is free, then, for all  $1 \leq j \leq j' \leq m$ , we have*

$$A_{(i,p)}^p[j, j'] = \begin{cases} j' & DP_{(i,p)}[j, j'] = DP_{(i,p-1)}[j, j'-1] + \chi(S_1[p], S_2[j']), \\ -j' & DP_{(i,p)}[j, j'] = DP_{(i,p-1)}[j, j'] + \chi(S_1[p], \sqcup), \\ A_{(i,p)}^p[j, j'-1] & DP_{(i,p)}[j, j'] = DP_{(i,p)}[j, j'-1] + \chi(\sqcup, S_2[j']). \end{cases}$$

*Proof.* The first case in the recurrence is quite obvious since the optimal score  $DP_{(i,p)}[j, j']$  is obtained by adding  $DP_{(i,p-1)}[j, j'-1]$  with the score of aligning  $p$  with  $j'$  (by applying  $\chi(S_1[p], S_2[j'])$ ). As for the second case, we know that  $p$  is aligned to  $\sqcup$  and the alignment between  $S_1[i..p]$  and  $S_2[j..q]$  is actually the alignment corresponding to the score  $DP_{(i,p-1)}[j, j']$ . By Definition 11, we have  $A_{(i,p)}^p[j, j'] = -j'$ . Lastly, since the current  $j'$  is not included in the alignment, we must find the alignment of  $p$  in  $S_2[j..j'-1]$ .

□

The case when  $p$  is not free (ARC-MATCH operation) can be handled similarly. Finally, for the case of MERGE operation and the case where  $i < p < i'$ ,  $A_{(i,i')}^p[j, j']$  is equal to  $A_{(i'',i''')}^p[j', j'']$  where we have  $i \leq i'' \leq p \leq i''' \leq i'$  and  $DP_{(i,i')}[j, j'] = DP_{(i'',i''')}[j'', j'''] + X$  for  $X$  equals to some (probably empty) term that does not involve  $p$  (e.g the  $\chi(S_1[i'], S_2[j'])$ ,  $\delta((S_1[i], S_1[i']), (S_2[j], S_2[j'])),$  or  $DP_{(i'''+1,i')}[j'''+1, j']$ ).

**Lemma 3.3.17** *Given any position  $p$ ,  $1 \leq p \leq n$ , we can compute the position  $q$ ,  $1 \leq q \leq m$ , such that the optimal alignment between  $(S_1, P_1)$  and  $S_2$  aligns either  $S_1[1..p]$  to  $S_2[1..q]$  or  $S_1[1..p-1]$  to  $S_2[1..q]$ , within the same time and space complexity of the score-only  $WLCS_r(S_1, P_1, S_2)$ .*

*Proof.* Observe that the operation to compute the entry  $A_{(i,i')}^p[j, j']$  can be done right after the computation of one particular  $DP_{(i,i')}[j, j']$ . Next, the recurrences above show that  $A_{(i,i')}^p[j, j']$  can be computed in constant time. Hence computing  $A_{(1,n)}^p[1, m]$  yields the same time complexity as computing  $DP_{(1,n)}[1, m]$  which is the time complexity of  $WLCS_r$ .

As we only need to compute the value  $A_{(1,n)}^p[1, m]$  for each position  $p$ , we do not have to store all of the intermediary tables  $A_{(i,i')}^p$ . Instead, as in the case of the score-

only  $WLCS_r(S_1, P_1, S_2)$ , we only store those needed in the computation of the current  $A_{(i,i')}^p[j, j']$ . Consider the EXTEND operation. In computing  $DP_{(i,p)} = \text{EXTEND}(DP_{(i,p-1)})$ , we need to store  $DP_{(i,p-1)}$ . Correspondingly, computing  $A_{(i,p)}^p[j, j']$  only requires the values in  $A_{(i,p-1)}^p$ . This also applies on the ARC-MATCH and MERGE operations.

Since, at any point of time, we only need the entries  $A_{(i,i')}^p[j, j']$  from a constant number of  $(i, i')$  pairs (one pair for EXTEND and ARC-MATCH, two pairs for MERGE), we only need to store a constant number of such tables. Hence, the space needed by the  $A_{(i,i')}^p$  table is also  $O(m^2)$ .  $\square$

Lemma 3.3.17 had shown that finding the alignment of a single point can be done within the same time and space complexity of the score-only  $WLCS_r(S_1, P_1, S_2)$ . Therefore, as the number of points to compute is at most a constant, the complexity of the second step of our algorithm is equal to the score-only  $WLCS_r(S_1, P_1, S_2)$ 's.

While applying the third step of our new algorithm (the recursive call) on the continuous region is straightforward, the gapped region needs a bit of extra care. In this case,  $\star$  in  $S_{1i}$  must be aligned to  $\star$  in  $S_{2i}$  because they represent the subregion pair(s) computed in the other recursive call(s). To implement such constraint, we add into the base scoring function the following cases:  $\chi(\star, \star) = 0$  and  $\chi(\star, x) = \chi(x, \star) = -\infty$  for  $x \in \{A, C, G, U, \sqcup\}$ .

**Lemma 3.3.18** *Our new algorithm can recover the optimal WLCS alignment in  $\min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}$  time and  $\min\{O(m^2 \log n), O(m^2 + \alpha mn)\} + O(n)$  space.*

*Proof.* Let  $T(n, m)$  be the time complexity of the new algorithm. Let  $R_i$  denote the  $i^{\text{th}}$  region in  $S_1$  on which the algorithm is recursively applied. Along with each region  $R_i$ , define  $R'_i$  to be the region in  $S_2$  it is aligned to. We have earlier shown that the time complexity of the first and second step of our algorithm is in  $\min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}$ , hence we can formulate the recurrence

$$T(n, m) = \sum_i T(|R_i|, |R'_i|) + \min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}.$$

where  $i \leq 4$ ,  $\sum_i |R_i| = n$ ,  $\sum_i |R'_i| = m$  and  $|R_i| \leq \frac{2}{3}n$ . By inspection, we can see that the time complexity is still bounded by  $\min\{O(\alpha^2 n^2 m + nm^2), O(\alpha nm^2 \log n), O(nm^3)\}$ .

As for the space complexity, we define  $S(n, m)$  to denote the space requirement of the algorithm. Each time after the second step of our algorithm, we must store the alignments computed in the latter. This requires a dedicated  $O(n)$  space that can be accessed from all recursive calls. Observe that the space used by the current recursive call can be reused in the next one as we only need to store the alignments of the regions in the current computation. Therefore,

$$S(n, m) = \max\{\max_i S(|R_i|, |R'_i|), \min\{O(m^2 \log n), O(m^2 + \alpha mn)\} + O(n)\}.$$

Again, by inspection, we show that the complexity of  $S(n, m) = \min\{O(m^2 \log n), O(m^2 + \alpha mn)\} + O(n)$ . The lemma thus follows.  $\square$

### 3.4 Conclusion

Suppose we are given two homologous RNA sequences  $S_1$  and  $S_2$  where  $S_1$  has a known structure. This paper studies the problem of inferring the structure of  $S_2$  such that the WLCS score between the two structures is maximized. In the case of positive integer scoring, we designed an algorithm using dynamic programming sparsification technique that gives better time and space complexity than the brute-force approach.

Our techniques presented in this paper can be applied to *the longest arc-preserving common subsequence problem* (LAPCS) (see, e.g., [111, 113, 115]). Assuming similar scoring scheme (with the arc matching case removed, as the plain sequence would have no arc), we can also solve the LAPCS(nested, plain) problem in  $\min\{O(nm^2 + n^2m), O(nm^2 \log n), O(nm^3)\}$  time and  $\min\{O(m^2 + mn), O(m^2 \log n + n)\}$ , thus improving the currently best known time and space complexity bounds for this problem ( $O(nm^3)$  and  $O(nm^2)$ , respectively [115]). Our algorithm would improve the speed and scalability of existing programs like FASTR which in turn enable them to tackle larger RNA and more data at a given time.

### 3.5 List of publication

1) Jansson J, Ng S K, Sung W K and Hugo W. A Faster and More Space-Efficient Algorithm for Inferring Arc-Annotations of RNA Sequences Through Alignment. Initial publication at *WABI 2004*, full version is published at *Algorithmica*, 223–245, 2006.

## Chapter 4

# Discovering Interaction Motifs from Protein-Protein Interaction Data: D-STAR

### 4.1 Introduction

Some important biological processes, such as the signaling pathways, require protein-protein interactions that are designed for fast response to stimuli. These interactions are usually transient, easily formed and disrupted, and specific. These transient interactions typically involve the binding of a protein to a short stretch (3 to 20) of amino acid residues which is usually characterized by a simple sequence pattern, i.e. *a short linear motif (SLiM)*. These are short, functional regions on the proteins that conform to particular sequence patterns; a well-known example is the set of peptides expressing a P..P consensus (where . represent any arbitrary amino acid) that bind SH3 protein domains [119, 120].

SLiMs are discovered by biological experiments, such as *site-directed mutagenesis* and *phage display*, which are laborious and expensive. Since SLiMs are entities enabling protein interaction and given the availability of more protein-protein interaction data, many researcher start to study on different ways of finding SLiMs from the PPI. Given a set of protein-protein interaction data, binding motifs can be discovered computationally as follows: (i) group protein sequences that interact with the same protein, and (ii)



for each set of protein sequences grouped, extract the motifs using motif discovery algorithms like MEME [121], Gibbs Sampler [122], PRATT [123] and TEIRESIAS [124]. For example, to computationally detect any possible motif binds by protein *Crk*, we could input protein sequences interacting with *Crk* to motif discovery programs. The underlying assumption is that if *Crk* binds its interaction partners through a SLiM, it should be over-represented in the partners. For ease of discussion, we denote such approach as One-To-Many (OTM) since we start with one protein to derive a group of multiple proteins associated with it for motif extraction.

The OTM approach is effective only when the protein we start with have enough number of interacting partners for motif extraction. In reality, many proteins have limited interacting partners [125]. This means that for many of the proteins, the signals from the few and short motif instances would be too weak for detection by the existing motif discovery algorithms. The scenario is actually worse when we further consider the high noise levels in interaction data [126] and the inherent heterogeneity of protein interactions—not all the real interacting partners of a protein necessarily carry the same binding motif.

Sometimes, it is possible to use some common feature of a protein groups to increase the number of its partners for motif extraction. For example, if individual copies of the SH3 domain bind limited protein partners, we could pool all sequences that bind any SH3 domain proteins to increase the P.P motif’s instances for its discovery. We denote this approach as the Many-to-Many (MTM) approach since we derived a set of sequences for motif extraction from another set of sequences (protein group). Reiss and Schwikowski adopted an MTM-based method with a modified Gibbs sampling algorithm to enhance motif finding on proteins with limited binding partners and successfully extracted more motifs than the OTM-based approaches [127]. In another work, Neduva *et. al.* complement the OTM approach with MTM approach to find novel linear motif from protein interaction data [40].

Both OTM and MTM approaches are *occurrence based* i.e. they rely on the significant occurrences of the SLiMs to mine them. However, this may be problematic when the interaction partners contain some naturally similar short regions like those found in a protein domain or region of low complexity. A high occurrence of a SLiM within such regions may have nothing to do with interactions since the occurrences are caused by

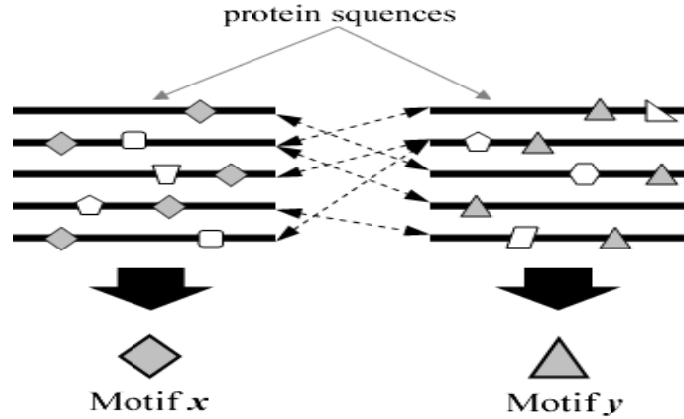


Figure 4.1: A depiction of our approach for finding correlated motifs. The dotted lines indicates the interactions between the proteins.

homology. This is the reason why OTM and MTM approaches would mask out the domain region and regions of low complexity (as done in [40]).

In this work, we present another approach to mine SLiMs from the PPI data. We propose that, in order to enable an interaction, both interacting proteins should have a conserved motif associated with each other (we defined these earlier as the interaction motifs). The interaction motifs describes specific regions within related pairs of interacting proteins that directly or indirectly enable the interaction between the two. We are interested in the case where the interacting motifs are SLiMs. The SLiMs either bind directly or interact indirectly with each other (by being a part of a domain that binds the other SLiM). This is reasonable because, for modular interaction domains, it is often the subregions, rather than the entire domains, that are involved in mediating protein-protein interactions.

Formally, suppose a set of protein-protein interactions occurs between proteins containing the SLiM  $X$  and proteins containing the SLiM  $Y$ . Our approach will simultaneously find both motifs  $X$  and  $Y$  directly from the PPI data. The algorithm is based on the intuition that if a set of interactions were indeed mediated by  $X$  and  $Y$ , those proteins containing  $X$  and  $Y$  would have significantly more interaction as opposed to random. We termed  $X$  and  $Y$  as a *correlated motif*. The term "correlated" indicates that the motif pair may not necessarily be directly binding each other but their co-occurrences in interacting sequences are significant. Our new approach offers the

following advantages:

1. In contrast to both OTM and MTM’s occurrence based approach, our approach is *interaction density based* since we target over representation of interaction between two candidate motif pairs. This difference is important because the motifs in a motif pair may not by itself have significant occurrences but together they have significant co-occurrence in interacting proteins.
2. Like the MTM approach, it increases the number of motif instances for detection.
3. By finding pairs of correlated motifs in the interaction data instead of single motifs in protein sequence data, our approach is more stringent and hence more resilient against noise since it is less likely for two spurious noise-induced motifs to co-occur in the interaction data more frequently than the true ones. This affords our program to do away with domain/low complexity masking while still retaining its accuracy (shown in the Result section).

To model the SLiMs, we adopted the  $(l, d)$ -motif model which had been used frequently to model motifs in biological sequences thanks to its simplicity [128–133]. In the  $(l, d)$ -motif model, the actual motif and motif instances are strings of length  $l$  and each instance differs by no more than  $d$  mismatches from the actual motif. Thus any two motif instances would have at most  $2d$  mismatches. Consequently, a set of very similar substrings can be modeled as a  $(l, d)$  motif with a small  $d$  while a more diverse substring set need to be modeled with a larger  $d$ . We then formulated our approach as an  $(l, d)$ -motif pair finding problem, and presented an exact algorithm, D-MOTIF, as well as its approximation algorithm, D-STAR to solve the problem.

Our benchmark analysis shows that D-STAR’s performance is comparable to D-MOTIF’s with a substantially shorter running time. Thus, in evaluation experiments, we compare only D-STAR with other existing algorithms so that we can run extensive tests on both simulated and real biological datasets. The results confirm that the correlated motif approach is more robust than OTM and MTM in extracting motifs from sparse but noisy interaction data.

Evaluation on real biological datasets further demonstrates that our D-STAR algorithm is able to extract correlated motifs that are biologically relevant. On a SH3

domain interaction dataset [119], D-STAR extracted P..P.[KR] and G..P.NY as correlated motifs; the two motifs were subsequently validated to actual interacting interfaces in the structural data of SH3 domain and its ligand (see Fig. 4.10). P..P.[KR] is known as the SH3 binding motif class 2 (as defined in the ELM [37]). D-STAR also extracted the SLiM [KR]..P..P, the SH3 binding motif class 1, that was not detected by any existing algorithms tested in this study(see Fig. 4.4.2 and Table 4.4.2). Application of D-STAR on the TGF $\beta$  signaling pathway [134] extracted correlated motifs that mapped to putative phosphorylation sites and kinase subregions in proteins respectively. Our results are published in [47].

## 4.2 Related works

There are existing works [86,135–137] that also find over-represented pairs of co-occurring sequence patterns from protein-protein interaction data, but most focused on discovering interaction correlations between existing protein domains like those in Pfam, InterPro and Prosite. These methods are also geared towards finding novel *interactions*, not novel *motifs*.

For SLiMS, currently only about 200 SLiMs out of some few thousands that possibly exist [33] have been listed in public databases (e.g ELM [37]). The correlated motif approach outlined in this work is a *de-novo* motif finding method which can potentially discover novel motifs as well as their correlations from the increasingly abundant protein interaction data. Our algorithms can also be applied on biological pathways or protein networks directly to detect the most significant co-occurring motif pairs in these pathways. Such functionality is important for studying pathways known to be mediated by recurring domains and motifs, such as those found in the various signaling pathways [93, 138].

## 4.3 Methods

### 4.3.1 Preliminaries

Let  $s = a_1a_2a_3..a_n$  be a length- $n$  protein sequence defined over the alphabet  $\Sigma$  of 20 amino acids, and  $s[u, v]$  as the substring of the string  $s$  starting at position  $u$  up to

position  $v$ . When the substring's length  $l$  is fixed, we simply write  $s[u]$  for  $s[u, u + l - 1]$ . We will call such a substring the  $l$ -substring at position  $u$ .

### The $(l, d)$ -motif finding problem

The definition of  $(l, d)$ -motif was originally proposed in [128] to model motifs in biological sequences. Consider a set  $S = \{s_1, s_2, s_3, \dots, s_t\}$  of  $t$  protein sequences of length  $n$ . A length- $l$  pattern  $p$  is an  $(l, d)$ -motif in  $S' \subseteq S$  if all sequences  $s_i \in S'$  have at least one  $l$ -substring  $s_i[u]$  which differs from  $p$  by at most  $d$  mismatches. Such  $s_i[u]$ 's are termed as the *instances* of  $p$ . In their work, Pevzner *et. al.* [128] computed for the  $(l, d)$ -motif  $p$  that has at least one instance in each sequence in  $S$ . In our work, it is important to find motifs from a significantly large subset  $S'$  of  $S$  since, in some case, there is no guarantee that every input sequence would contain an instance of the motif. In other words, for a given  $(l, d)$ -motif  $p$ , let  $\mathcal{S}_d(p)$  be  $\{s \in P \mid s \text{ contains an } l\text{-substring of distance at most } d \text{ from } p\}$ . Given the minimum number of instance threshold  $k_n$ , we then define the general  $(l, d)$ -motif finding problem as finding all  $(l, d)$ -motif  $p$  in  $S$  such that  $|\mathcal{S}_d(p)| \geq k_n$ .

### The $(l, d)$ -motif pair finding problem

We extend the problem of finding  $(l, d)$ -motifs in a set of sequences into one for finding motif *pairs* in a set of sequence *pairs* for mining interaction motifs in a set of protein-protein interactions. Given a protein interaction dataset  $I \subseteq S \times S$  of size  $m$  over the set of proteins  $S$  where for any  $(s_i, s_j) \in I$  we have  $i \leq j$ , we want to find a pair of  $(l, d)$ -motifs which is over-represented in  $I$ . That is, we want to find an  $(l, d)$ -motif pair  $(X, Y)$  that have the following characteristics:

- (1) Let  $I_{(X,Y)}$  be the set of interactions between  $\mathcal{S}_d(x)$  and  $\mathcal{S}_d(y)$ , namely,  $I_{(x,y)} = I \cap (\mathcal{S}_d(x) \times \mathcal{S}_d(y))$ . We require that  $|I_{(x,y)}| \geq k_i$  for a minimum number of interaction threshold  $k_i$ .
- (2) Let  $\mathcal{S}'_d(x)$  be a subset of  $\mathcal{S}_d(x)$  containing sequences that interact with those in  $\mathcal{S}_d(y)$ . Similarly, let  $\mathcal{S}'_d(y)$  be a subset of  $\mathcal{S}_d(y)$  with interacting sequences with  $\mathcal{S}_d(x)$ . We also require that  $|\mathcal{S}'_d(x)|, |\mathcal{S}'_d(y)| \geq k_n$ .

We call this problem the  $(l, d)$ -motif pair finding problem. For every  $(s_i, s_j) \in I_{(x,y)}$ , we want find  $(s_i[u], s_j[v])$  which are instances of  $X$  and  $Y$ . Biologically,  $(s_i[u], s_j[v])$  may correspond to the functional regions in the proteins  $s_i$  and  $s_j$  that mediate their interaction.

### Scoring function

It is likely for many  $(l, d)$ -motif pairs  $(x, y)$  to exist within a given interaction dataset  $I$  over the set of proteins  $S$ . We define here a scoring function to rank them systematically.

Let  $O(S_X, S_Y)$  be the observed number of interactions between two protein sets  $S_X$  and  $S_Y$  containing the motifs  $X$  and  $Y$  respectively. Let  $E(S_X, S_Y)$  be the expected number of interactions between  $S_X$  and  $S_Y$ . We estimate  $E(S_X, S_Y)$  based on the assumption that interactions occur at random. Since the probability of any interaction occurring between two random proteins in  $S$  is  $\frac{|I|}{\binom{|S|}{2}}$ , we have

$$E(S_X, S_Y) = \frac{|I|}{\binom{|S|}{2}} \left[ |S_X||S_Y| - \binom{|S_X \cap S_Y|}{2} - |S_X \cap S_Y| \right]$$

where the term in the brackets computes the total number of interactions possible between the proteins in  $S_X$  and  $S_Y$ . Based on the idea of chi-square statistic, we formulate the following function  $\chi$  to score a given pair of  $(x, y)$ -motif containing protein sets  $S_X$  and  $S_Y$  as

$$\chi(S_X, S_Y) = \frac{[O(S_X, S_Y) - E(S_X, S_Y)]^2}{E(S_X, S_Y)}$$

### 4.3.2 Methods

For illustration, we will first give an exact algorithm D-MOTIF to find co-occurring motifs in  $I$ . Then, we will present our approximation algorithm, D-STAR, that can offer significant speed-up at the cost of slight accuracy degradation. The use of D-STAR for scaling up is necessary for dealing with the large input datasets in practice.

#### D-MOTIF algorithm

The basic idea of the exact algorithm is to enumerate all possible  $(l, d)$ -motif pairs and then check if they have enough instances to satisfy the minimum size threshold  $k_n$  and  $k_i$ . Note that any  $(l, d)$ -motif pair must be of hamming distance  $d$  from some  $(l, d)$ -motif

## D-MOTIF-BASIC

```

1  for  $(s_i, s_j) \in I$ 
2      for  $(s_i[u], s_j[v]) \in (s_i, s_j)$ 
3          for  $(p, p') \in X_{s_i[u]} \times X_{s_j[v]}$ 
4              Compute  $\mathcal{S}_d(p)$  and  $\mathcal{S}_d(p')$ 
5               $I_{(p,p')} = I \cap (\mathcal{S}_d(p) \times \mathcal{S}_d(p'))$ 
6              Compute  $\mathcal{S}'_d(p)$  and  $\mathcal{S}'_d(p')$ 
7              if  $|I_{(p,p')}| \geq k_i$  and  $|\mathcal{S}'_d(p)|, |\mathcal{S}'_d(p')| \geq k_n$ 
8                  Store  $(p, p')$  sorted by  $\chi(\mathcal{S}_d(p), \mathcal{S}_d(p'))$  in list  $L$ .
```

Figure 4.2: The D-MOTIF-BASIC algorithm.  $(s_i, s_j)$  is a pair of interacting protein from the PPI dataset  $I$ .  $s_i[u]$  ( $s_j[v]$ , resp.) is the length  $l$  substring starting at position  $u$  ( $v$ , resp.) in  $s_i$  ( $s_j$ , resp.).  $X_{s_i[u]}$  ( $X_{s_j[v]}$ , resp.) is the set of all length  $l$  string which have at most  $d$  mismatches with  $s_i[u]$  ( $s_j[v]$ , resp.). The set  $\mathcal{S}_d(p)$  ( $\mathcal{S}_d(p')$ , resp.) is the set of all proteins containing at least one length  $l$  substring which has at most  $d$  mismatches with  $p$  ( $p'$ , resp.). The subset of  $I$  containing the interactions between proteins in  $\mathcal{S}_d(p)$  and  $\mathcal{S}_d(p')$  is denoted as  $I(p, p')$ . The set  $\mathcal{S}'_d(p)$  is the subset of  $\mathcal{S}_d(p)$  which has an interaction with another protein in  $\mathcal{S}'_d(p')$  given the interaction set  $I(p, p')$ .  $k_n$  and  $k_i$  are minimum size of the interacting protein set and interaction set, respectively.  $\chi(\mathcal{S}_d(p), \mathcal{S}_d(p'))$  is the chi-score computed for the pair  $(p, p')$ .

pair instance. Given a string  $p$  of length  $l$ , we define  $X_p$  to be all strings  $p'$  of length  $l$  with hamming distances at most  $d$  from  $p$ . The algorithm named D-MOTIF-BASIC in Fig. 4.3.2 describes the most straightforward brute force approach on the problem.

Observe that the instances of any  $(l, d)$ -motif  $X$  is of distance  $2d$  from one another. Pevzner *et. al.* [128] described a method to compute all instances of an  $(l, d)$ -motif by transforming the problem into finding cliques in a  $t$ -partite graph  $G$ . In this graph, all  $l$ -substrings in all  $s_i \in S$  are the nodes and any two of them will be connected by an edge if (a) they originate from distinct proteins and (b) they are at most  $2d$  apart. Thus, finding the  $(l, d)$ -motifs having at least  $k_n$  instances is equivalent to finding cliques of size at least  $k_n$  in  $G$ , which is an NP-hard problem.

We attempt to reduce the complexity of the problem by assuming that  $k_n \geq 3$  and try to find all cliques of size 3 first. In other words, we first find three  $l$ -substrings,  $(s_i[u], s_j[v], s_k[w])$ , from distinct sequences  $s_i$ ,  $s_j$ , and  $s_k$  and then only try those candidate  $(l, d)$ -motifs  $p \in X_{s_i[u]} \cap X_{s_j[v]} \cap X_{s_k[w]}$ . For convenience, we call the string triplet  $(s_i[u], s_j[v], s_k[w])$  a *triangle* within  $s_i$ ,  $s_j$ , and  $s_k$  and we denote the set intersection

## D-MOTIF

```

1  for  $\langle s_i, s_{i'} \rangle, \langle s_j, s_{j'} \rangle, \langle s_k, s_{k'} \rangle \in I_d$  where  $i \neq j \neq k$  and  $i' \neq j' \neq k'$ 
2    for  $(s_i[u], s_j[v], s_k[w]) \in (s_i, s_j, s_k)$ 
3      Compute and store  $X_{(s_i[u], s_j[v], s_k[w])}$  in  $T_l$ 
4    for  $(s_{i'}[u'], s_{j'}[v'], s_{k'}[w']) \in (s_{i'}, s_{j'}, s_{k'})$ 
5      Compute and store  $X_{(s_{i'}[u'], s_{j'}[v'], s_{k'}[w'])}$  in  $T_r$ 
6    for  $(X_l, X_r) \in T_l \times T_r$ 
7      for  $(p, p') \in X_l \times X_r$ 
8        Compute  $\mathcal{S}_d(p)$  and  $\mathcal{S}_d(p')$ 
9         $I_{(p, p')} = I \cap (\mathcal{S}_d(p) \times \mathcal{S}_d(p'))$ 
10       Compute  $\mathcal{S}'_d(p)$  and  $\mathcal{S}'_d(p')$ 
11       if  $|I_{(p, p')}| \geq k_i$  and  $|\mathcal{S}'_d(p)|, |\mathcal{S}'_d(p')| \geq k_n$ 
12         Store  $(p, p')$  sorted by  $\chi(\mathcal{S}_d(p), \mathcal{S}_d(p'))$  in list  $L$ .

```

Figure 4.3: The D-MOTIF algorithm.  $X_{(s_i[u], s_j[v], s_k[w])}$  is a short notation for  $X_{s_i[u]} \cap X_{s_j[v]} \cap X_{s_k[w]}$ . The algorithm's speed up is achieved by only considering  $l$  substrings which have at least three other substrings with at most  $d$  mismatches from it.

$X_{s_i[u]} \cap X_{s_j[v]} \cap X_{s_k[w]}$  by  $X_{(s_i[u], s_j[v], s_k[w])}$ .

In the case of interaction data, we have to find all interaction triplets  $(s_i, s_{i'}), (s_j, s_{j'}), (s_k, s_{k'})$  and compute the triangles from  $(s_i, s_j, s_k)$  and  $(s_{i'}, s_{j'}, s_{k'})$ . But as interaction is commutative (at least in our current consideration) i.e.  $(s_i, s_j)$  is equivalent to  $(s_j, s_i)$ , we also have to consider the latter configuration when we choose the interaction triplets. As such, we let  $I_d$  be the set of *ordered pair* which contains both  $\langle s_i, s_j \rangle$  and  $\langle s_j, s_i \rangle$  for each  $(s_i, s_j) \in I$ . The algorithm can then start by choosing the ordered pair triplets from  $I_d$  ( $|I_d| \approx 2m$ ). The complete listing of the algorithm, D-MOTIF, is presented in Fig. 4.3.

In practice, D-MOTIF runs much faster when compared to the straightforward brute force algorithm (which we have also implemented as a benchmark). However, the memory requirement of D-MOTIF could be much larger than the latter as we have to store the sets  $X$  for the different triangles in the set  $T_l$  and  $T_r$  to avoid redundant computations. When  $d$  is large relative to  $l$ , there would be a lot of candidate  $(l, d)$ -motifs to check given a triangle. When the number of triangles is also large, even D-MOTIF would soon run at a crawling speed. In view of that, we propose the following approximation algorithm, D-STAR. Before we start, let us define the  $(l, d)$ -star pair finding problem



## D-STAR

```

1  for  $(s_i, s_j) \in I$ 
2    for  $\langle s_k, s_\ell \rangle \in I_d - \langle s_i, s_j \rangle$ 
3      Perform a pairwise sequence comparison to find all positions in
         $s_i$  which has a neighbor of distance  $2d$  in  $s_k$ . Let the positions
        be  $P_1 = \{u_1, u_2, \dots, u_g\}$ .
4      Do the same for  $s_j$  and  $s_\ell$  and get the list of positions in  $s_j$ 
        which is  $P_2 = \{v_1, v_2, \dots, v_h\}$ .
5      if  $P_1 \neq \emptyset$ 
6        for all  $u \in P_1$  add  $s_k$  into  $\mathcal{S}_{2d}(s_i[u])$ 
7      if  $P_2 \neq \emptyset$ 
8        for all  $v \in P_2$  add  $s_\ell$  into  $\mathcal{S}_{2d}(s_j[v])$ 
9      for  $(u, v) \in P_1 \times P_2$ ,
10         Add the non-ordered pair  $(s_k, s_\ell)$  into  $I_{(s_i[u], s_j[v])}$ .
11  for  $(u, v)$  whose  $|\mathcal{S}'_{2d}(s_i[u])|, |\mathcal{S}'_{2d}(s_j[v])| \geq k_n$  and  $|I_{(s_i[u], s_j[v])}| \geq k_i$ .
12    Compute  $\chi(\mathcal{S}_{2d}(s_i[u]), \mathcal{S}_{2d}(s_j[v]))$  and put the  $(l, d)$ -star
         $(\mathcal{S}_{2d}(s_i[u]), \mathcal{S}_{2d}(s_j[v]))$  into the sorted list  $L$ .

```

Figure 4.4: The D-STAR algorithm.

and show how it approximates for the  $(l, d)$ -motif pair finding problem.

### The $(l, d)$ -star pair finding problem

For any given pair of  $l$ -substrings  $(s_i[u], s_j[v])$  from some interaction  $(s_i, s_j)$ , there may be an exponential (with respect to  $d$ ) number of possible  $(l, d)$ -motifs  $(x, y)$  which is within distance  $d$ . Hence, even after speeding-up the algorithm with filtering, D-MOTIF can only handle relatively small-sized problems. In our proposed algorithm D-STAR, we will aim to find only the *instances* of a motif pair  $(x, y)$  instead of finding the motif  $(x, y)$  themselves since they may not even occur in  $S$ .

### D-STAR algorithm

Recall that given an  $(l, d)$ -motif  $X$ , any two instances of  $X$ ,  $X_i$  and  $X_j$ , are at most  $2d$  apart. Hence, if we manage to get one instance  $X_i$  of  $X$ , all the other instances of  $X$  would surely be in  $\mathcal{S}_{2d}(x_i)$ . In the context of interaction data, we first get all  $l$ -substring pairs  $(s_i[u], s_j[v])$  from each interacting proteins  $(s_i, s_j) \in I$ . Next, we find those  $(s_i[u], s_j[v])$  that satisfy two conditions (1) There are more than  $k_i$  interactions between  $\mathcal{S}_{2d}(s_i[u])$  and  $\mathcal{S}_{2d}(s_j[v])$ . (2) Let the set of the interactions be denoted similarly by  $I_{(s_i[u], s_j[v])}$ , and we require that both  $|\mathcal{S}'_{2d}(s_i[u])|, |\mathcal{S}'_{2d}(s_j[v])| \geq k_n$ . The pair of protein

set  $(\mathcal{S}'_{2d}(s_i[u]), \mathcal{S}'_{2d}(s_j[v]))$  is denoted as an  $(l, d)$ -star pair. Our simulation experiments indicate that D-STAR yields a good approximation of the exact solution while being much more efficient when the dataset is large. The complete listing of the algorithm is in Fig. 4.3.2.

### Time complexity

The loop in line 1 takes  $O(m)$  time. The next loop in line 2 takes another  $O(m)$  time. Both pairwise sequence comparisons in step 3 and 4 require  $O(n^2)$  time. Each time, the number of position pairs  $(u, v)$  in  $P_1 \times P_2$  could also reach  $O(n^2)$ . Updating  $I_{(s_i[u], s_j[v])}$ ,  $\mathcal{S}'_{2d}(s_i[u])$ ,  $\mathcal{S}'_{2d}(s_j[v])$ , can all be done in constant time with a lookup table (one could save space using hash-sets, but the updating will take *amortized* constant time instead). The loop in line 11 would require at most  $O(n^2)$  time for all entries  $[u, v]$ , each requiring at most  $O(t)$  time to build  $(\mathcal{S}_{2d}(s_i[u]), \mathcal{S}_{2d}(s_j[v]))$ , from  $(\mathcal{S}'_{2d}(s_i[u]), \mathcal{S}'_{2d}(s_j[v]))$  for computing the chi-square score. Therefore, in the worst case, D-STAR would run in  $O(m^2n^2 + mtn^2)$ . We also need to be mindful that the memory requirement to store the matrix and arrays is  $\max\{O(mn^2), O(tn)\}$ .

### Comparison between D-MOTIF and D-STAR

First, we investigate the effect of data size on the performance of our two approaches. We ran our evaluation on 5 different datasets containing artificial interaction sets  $I$  of size ranging from 10 to 150 (note that for some weaker motifs, we did not evaluate up to size 150 as the running time of the D-MOTIF became too slow to be measured). In each interaction set, the protein sequences in all interaction are distinct; in other words,  $|S| = 2|I|$ . We also planted the  $(l, d)$ -motif pair in only half of the interactions in  $I$  to effect a fixed  $\epsilon = 0.50$  on all datasets.

Evaluation was performed here by checking if the planted motifs were reported as the *best* motif by the motif finding algorithm. Table 4.3.2 shows the average result over 10 datapoints ( $I = 10, 20, \dots, 100$ ) in each of the 5 evaluation datasets. Fig. 4.3.2 displays the running time of both algorithms on different data size averaged over the 5 datasets. We use an x86 Pentium 4M 1.6GHz machine with 512MB of memory for running the comparison. We observed that when the  $(l, d)$ -motifs get less specific and  $k_n$  is small, the planted motifs could be masked out by other signals present in the protein sequences.

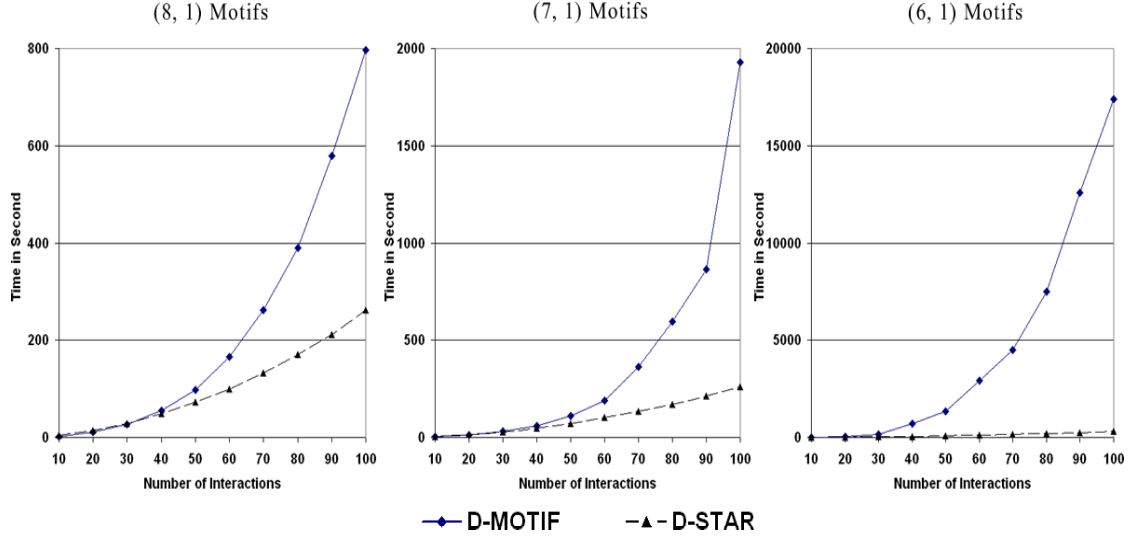


Figure 4.5: Comparison of running time between D-MOTIF and D-STAR. We observe that the running time of D-MOTIF increases rapidly as the input data grows and also as the  $(l, d)$ -motif gets weaker. Experiments were run on a x86 Pentium 4 1.6GHz machine with 512MB of memory.

$(l, d)$	D-MOTIF		D-STAR	
	Spec	Sens	Spec	Sens
(6, 1)	99.69%	100%	95.16%	99.1%
(7, 1)	100%	100%	99.89%	100%
(8, 1)	100%	100%	100%	100%

Figure 4.6: Comparison on specificity and sensitivity between D-MOTIF and D-STAR. This table shows that D-STAR runs orders of magnitude faster than D-MOTIF while sacrificing a small amount of accuracy in terms of sensitivity and specificity.

This happened in one of the datapoints of (6, 1)-motifs with  $|I| = 10$ , in which D-STAR failed to have 100% sensitivity rate.

Overall, it is clear that D-STAR performs only slightly worse than D-MOTIF while the running time of D-STAR is much better than D-MOTIF for larger datasets. The running time of D-MOTIF is also highly influenced by the strength/specificity of the  $(l, d)$ -motif. As compared to D-STAR, the running time of D-MOTIF increases much more rapidly when the motif gets less specific. For example, for  $|I| = 100$ , the running time of D-MOTIF on (8, 1), (7, 1), (6, 1) motifs are 797.4 s, 1930.7 s and 17385.2 s, respectively. For the same datapoints, D-STAR only required 253 s, 266.5 s, and 306.1 s, respectively. Indeed, this observation was further confirmed when we tried D-MOTIF

on our real biological dataset later—it was still running after 10 hours while D-STAR terminates in less than 20 minutes.

## 4.4 Results and discussion

In the following discussion, we compared our algorithms (D-STAR and D-MOTIF) against the existing algorithms, run in either OTM or MTM mode. This is because, to our knowledge, there is no existing algorithm based on our approach. Recall that in the  $(l, d)$ -motif model, the motif (a consensus string) and its instances are strings of length  $l$  and each instance differs by no more than  $d$  mismatches from the actual motif. The  $l$  and  $d$  are two parameters to the algorithms. Users can either input specific  $l$  and  $d$  into the algorithms or input a range of values for  $l$  and  $d$  instead. In the latter, the algorithms will extract the different  $(l, d)$ -motif pairs and output them, ranked based on their significance. At the same time, user must provide two additional parameters  $k_i$  and  $k_n$  for more directed search:  $k_i$  specifies the minimum number of interactions that  $(l, d)$ -motif pairs must co-occur in while  $k_n$  dictate the minimum of interacting proteins that must express each of the  $(l, d)$  motif.

In short, our algorithms tries to cluster the interaction data into groups of interaction which express some statistically significant  $(l, d)$ -motif pair; it look for pairs of similar substring set (defined by the  $(l, d)$  motif model) occurring across pairs of interacting proteins, and rank them based on their co-occurrence statistical significance. The exact algorithm D-MOTIF would find all possible motif pairs which satisfy the threshold given while D-STAR would allow a bit of inaccuracy for the sake of speed. We performed a preliminary experiment on D-MOTIF and D-STAR to compare their accuracy and efficiency, and found out that D-MOTIF is only modestly more accurate than D-STAR while running several orders of magnitude slower than the latter. The details of the comparison can be found in the Methods section. For efficiency, we therefore only ran D-STAR in our following evaluation experiments.

### 4.4.1 Artificial data with planted $(l, d)$ -motifs

We evaluate the robustness of D-STAR against noise in input data using simulated data with planted  $(l, d)$ -motifs. Another goal of the study is to investigate the performance of

D-STAR when dealing with problems involving weak motifs. This will provide insights to the user on how the latter influences D-STAR’s accuracy.

### Simulation setup

We follow the simulation setup devised in [128], where the authors planted well-defined artificial  $(l, d)$ -motifs into random sequences to create artificial datasets for evaluation. Here, we use actual yeast protein sequences with planted  $(l, d)$ -motifs and then randomly pair them up to generate artificial interaction datasets. For each pair of  $(l, d)$ -motifs  $(x, y)$ , five instances of motif  $X$  and five instances of motif  $Y$  are inserted into ten randomly selected protein sequences. Let us denote the five sequences with planted motif  $X$  as sequence set  $P_X$ , and the five sequences with planted motif  $Y$  as sequence set  $P_Y$ . We set  $|P_X| = |P_Y| = 5$  in our current simulations.

We simulate the real protein interactions by pairing every sequences in  $P_X$  to  $\mathcal{N}$  sequences in  $P_Y$ , and vice versa. A *spurious* interaction is modeled by pairing a protein in  $P_X(P_Y, \text{resp.})$  with a random yeast protein not in  $P_Y(P_X, \text{resp.})$ . Given that a protein interacts with an average of 5.8 other proteins (interaction statistics in DIP [139]), and that the high throughput yeast two-hybrid technique is known to have at least 50% error [126], we would expect at most 2.9 true interactions per protein. Being conservative, we set  $\mathcal{N} = 2$  here. Let  $\epsilon$  be the noise level defined as the fraction of the spurious interactions within all interactions that belong to one particular protein. We investigate the performance of the algorithms with  $\epsilon = 0.50$  as well as  $\epsilon = 0.60$ . For instance, when  $\mathcal{N} = 2$  and  $\epsilon = 0.50$ , the proteins in  $P_X$  and  $P_Y$  will be involved in (on average) 4 interactions; two of which are spurious.

### The algorithms and parameter settings

We applied D-STAR, as well as other known motif extraction algorithms such as MEME and Gibbs Sampler to see whether they can extract instances of both planted motifs amongst its motif pairs with the highest scores from the noisy input datasets. We also implemented an algorithm, S-STAR, to find single  $(l, d)$ -motifs in subsets of protein sequences based on the well-established SP-STAR algorithm [128]. We ran MEME, Gibbs Sampler and S-STAR using the MTM approach since  $\mathcal{N} = 2$  is too low for an OTM-based approach to detect the motifs. We assume that all the algorithms using the

MTM-approach will be ran only on the proteins that interact with those in  $P_Y$  when trying to find motif  $X$  (and vice versa for  $Y$ ). The average of the two cases is the reported performance. Note that this effectively provides the existing algorithms with prior knowledge on the underlying groupings of the protein sequences; the knowledge of sequence groups  $P_X$  and  $P_Y$ .

To search for the set of planted  $(l, d)$ -motifs, we set the parameters for the various algorithms as follows. For MEME, the parameters are: *Mode* = *ZOOPS* (option in MEME when not every input sequences are guaranteed to contain a motif of interest) and *Motif Width* =  $l$ . For Gibb Sampler, the parameters are: *Mode* = *Motif Sampler* (option in Gibbs Sampler when not all input sequences are guaranteed to contain a motif of interest), *Motif Width* =  $l$  and *Expected Motif Occurrence* = 5. For D-STAR and S-STAR, being  $(l, d)$ -motif searching algorithms, the first two parameters are  $l$  and  $d$ . We set the minimum number of motif occurrences in the sequences,  $k_n = 5$ . For D-STAR, the minimum number of interactions between the instances of the correlated motifs,  $k_i$  is also set to 5 as well.

## Evaluation metrics

We evaluate the relative performance of the algorithms using the following metrics:

$$\begin{aligned} \text{Specificity} &= \frac{TP_X + TP_Y}{TP_X + TP_Y + FP_X + FP_Y} \\ \text{Sensitivity} &= \frac{TP_X + TP_Y}{TP_X + TP_Y + FN_X + FN_Y} \\ \text{F-Measure} &= \frac{(2 \times \text{Specificity} \times \text{Sensitivity})}{(\text{Specificity} + \text{Sensitivity})} \end{aligned}$$

where  $TP_X$  ( $TP_Y$ , resp.) is the number of correctly recovered planted motifs  $X$  ( $Y$ , resp.)  $FN_X$  ( $FN_Y$ , resp.) is the number of instances of the planted motif  $X$  ( $Y$ , resp.) the algorithm fails to recover. Lastly,  $FP_X$  ( $FP_Y$ , resp.) is the number of spurious motifs included by the algorithm as a candidate instance of  $X$  ( $Y$ , resp.).

## Results

We applied D-STAR and all the other algorithms on numerous sets of simulated interaction data with different planted  $(l, d)$ -motifs, namely the  $(8, 1)$ ,  $(7, 1)$ ,  $(9, 2)$ ,  $(6, 1)$  and  $(8, 2)$ -motifs (listed in decreasing order of motif strength). For each combination of motif

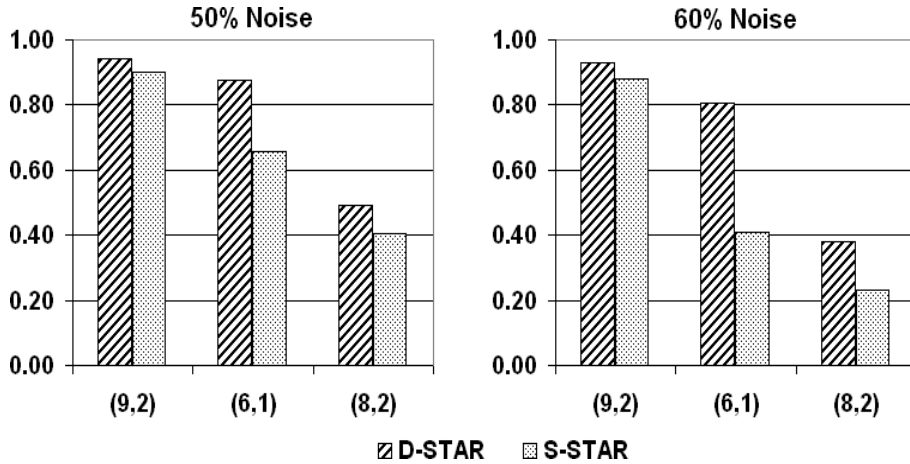


Figure 4.7: Comparison between D-STAR and S-STAR(A variant of SP-STAR) in extracting planted  $(l, d)$ -motifs. The motifs are arranged on the x-axis in decreasing order of motif strength. The number of planted motif instances in each dataset is 5 and the datapoint is the average over 10 runs.

and  $\epsilon$  value, we generated 10 random datasets and compute the average performance of the algorithms in discovering correct motif. Our results showed that that MEME and Gibbs Sampler performed quite poorly. Even for a relatively strong  $(8, 1)$ -motif, MEME can only achieve F-Measures of 0.49 and 0.35 for  $\epsilon = 0.50$  and 0.60, respectively (As for Gibbs Sampler, the F-Measures were 0.58 and 0.29 respectively). However, since both of these algorithms used different motif models, they may not be optimized to search for  $(l, d)$ -motifs. Instead, we will compare their relative performance on real biological data later. An noteworthy observation, however, is increased noise in input data can drastically decrease the performances of the algorithms.

Not surprisingly, both D-STAR and S-STAR attained very high average F-Measure of 0.99 for relatively stronger  $(8, 1)$  and  $(7, 1)$ -motifs on all values of  $\epsilon$  (data not shown). Fig. 4.4.1 shows the comparison of F-Measures of D-STAR and S-STAR on the weaker  $(9, 2)$ ,  $(6, 1)$  and  $(8, 2)$  motifs. Observe that D-STAR performed consistently better than S-STAR on all the cases, and furthermore, the performance margins were higher when there were more noise in the data. This study validates that even without having the prior knowledge of the motifs contained in the interaction data, D-STAR is able to handle noise much better than the other algorithms. This is of practical importance since real interaction data are often highly noisy data containing many interactions

Algorithm	PxxP	PxxPx[KR]	[KR]xxPxxP
D-STAR	1 <sup>st</sup>	1 <sup>st</sup>	8 <sup>th</sup>
S-STAR	1 <sup>st</sup>	-	-
MEME	1 <sup>st</sup>	-	-
GIBBS	3 <sup>rd</sup>	3 <sup>rd</sup>	-

Figure 4.8: Rank of sequence segment sets or sequence segment pair sets output by the various algorithms that express various known binding motifs of SH3 domains. "-" denote the biological motif is not expressed within the top 50 sequence segment sets.

between unknown domains and/or motifs.

#### 4.4.2 Biological data

In this section, we apply our algorithm on two biologically significant datasets: SH3 domain interaction data and TGF $\beta$  signaling pathway data. We show that our approach can better discover real biological motifs than the other methods.

##### SH3 domain interaction data

SH3 domains are conserved amino acid segments (of length  $\approx 60$  amino acids) found across multiple proteins. Through various biological experiments, SH3 domains have been determined to bind short sequence segments expressing the general motif P..P [119]. The interactions between SH3 proteins and the P..P motif mirror our motif pair  $(X, Y)$  (in this case, one of the motifs should correspond to parts of SH3 domain). For evaluation, we use the same dataset derived by Tong *et. al.* to find the interacting partners of SH3 domain proteins [119]. This dataset, which we called SH3-[PxxP]-Tong, was downloaded from BIND online database. It consists of 233 protein-protein interactions among 146 yeast proteins of which 23 are SH3 domain proteins (as determined using HMMER program from Pfam). We will first assess whether the known SH3 binding motifs can be extracted among the top motifs by each algorithm. Next, we investigate the biological relevance of the correlated motifs extracted by D-STAR.



## The algorithms and parameters

We ran D-STAR on the SH3-[P..P]-Tong dataset multiple times with different combinations of  $l = 6, 7, 8$ ,  $d = 1$  and  $k_n = k_i = 5$ . The outputs from the different runs were then systematically ranked using their  $\chi$ -scores. Note again that in the case of our D-STAR algorithm, the motifs were mined without having to separate the SH3 domain proteins and the non-SH3 domain proteins, unlike the other MTM motif extraction methods which require such prior knowledge. For comparison, we also attempted to extract the P..P-like motifs with MEME (*ZOOPS mode*, *Motif Width = 4 – 9*), Gibbs Sampler (*Motif Sampler mode*, *Motif Width = 4 – 8*, *Expected Motif Number  $\geq 5$* ) and SP-STAR ( $l = 6, 7, 8$ ,  $d = 1$  and *Minimum Motif Number=5*) from the 130 sequences in the dataset that bind to any SH3 proteins (*the MTM approach*).

## Validation

Without the luxury of experimentally validating the motifs extracted, it is hard to determine the accuracy of the various algorithms correctly. However, we reasoned that a good algorithm should at least extract most of the known motifs. In other words, when applying D-STAR on the interaction data of SH3 proteins, we should expect it to extract some P..P-like motifs on one side and another motif that occurs consistently in SH3 domains on the other side. We consider here the well-known SH3-binding motifs P..P, P..P.[RK] and [RK]..P..P. For each of these three motifs, we check whether it was “expressed” within the top 50 motifs reported (we assume that usually user would not check beyond this number). We define a set of protein sequence segments reported by an algorithm to be expressing a motif if at least 50% of the sequence segments match the pattern.

## Results

Table 1 shows the results for D-STAR, S-STAR, MEME, and Gibbs Sampler. The generic P..P motif was extracted among the top outputs by all algorithms. However, only our D-STAR algorithm managed to extract both P..P.[KR] and [KR]..P..P motifs (within the top 50 motifs output of each algorithm). In fact, only two instances of the P..P.[KR] motif are found in the segments extracted within the top 50 sets of segments

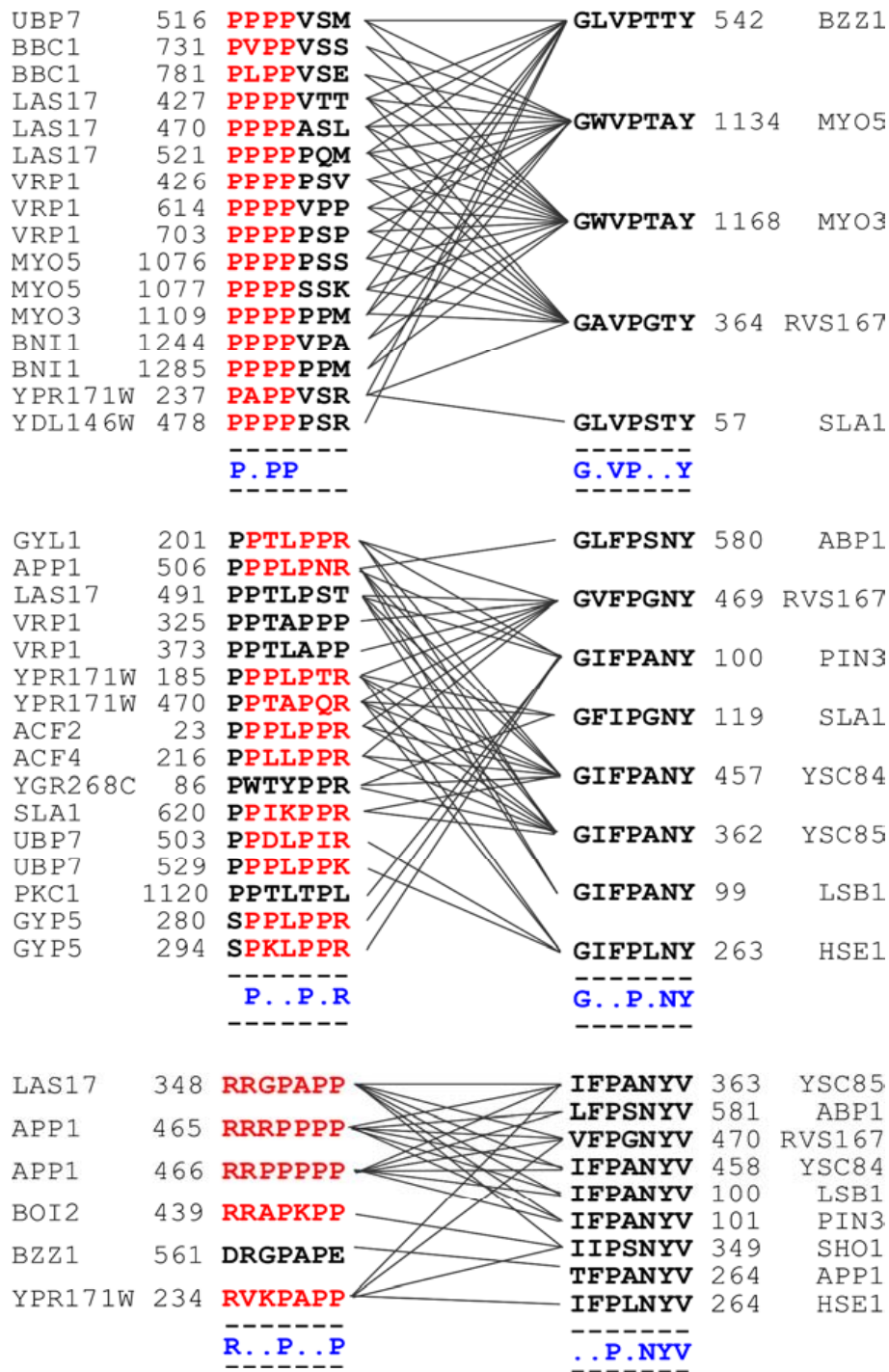


Figure 4.9: The P..P, P..P.[KR] and [KR]..P..P motifs and their associated motifs extracted by D-STAR. Lines between the sequence segments denote interaction between their parent proteins. The result is found from multiple runs of D-STAR with different combination of motif width  $l = 6, 7, 8$ , distance  $d = 1$  and  $k_i = k_n = 5$ . We then rank all the outputs from the different runs by their  $\chi$ -score.

extracted by MEME. No [KR]..P..P motif instance was extracted. To be sure, we re-ran MEME on the same 130 sequences with more specific motif lengths = 6-7 (instead of motif length = 4-9) but to no avail. This confirmed that MEME with the MTM approach has indeed missed out the more specific variants. As for S-STAR, the limited instances of the P..P.[KR] and [KR]..P..P motifs extracted were overwhelmed by the more general P..P motif. D-STAR, despite having no access to prior grouping knowledge unlike the other algorithms, was the only algorithm that was able to extract the specific SH3-binding motifs.

One might argue that since the MTM-algorithms were applied on the set of all SH3-binding sequences which contained either of the motifs P..P.[KR] and [KR]..P..P, it may be unsurprising that only the general P..P motif was extracted instead of the more specific motifs. The OTM approach may be more suitable for extracting the specific motifs since it does not consider the SH3-binding sequences in a “wholesale” manner as the MTM approach. As such, we applied MEME, Gibbs Sampler and S-STAR on the interacting protein partners of each individual SH3 protein in the SH3-[PxxP] dataset. In total, the OTM approach can be applied on the 22 SH3 proteins that bind more than 1 protein sequence. We used the same parameters used in the MTM approach for each algorithm except that the *Minimum Motif Occurrence*= 2. We deemed a motif to be extracted successfully if more than 50% of a segment set within the top 50 sets extracted expressed the motif and that 50% should comprise of at least 2 instances. For MEME, P..P motif was extracted for 3 SH3 proteins (Abp1,Rvs167,Bzz1) and P..P.[KR] motif was extracted for 2 other SH3 proteins (Ysc84,Myo3). Gibbs Sampler extracted the P..P and P..P.[KR] motifs for 1(Sho1) and 2 SH3 proteins (Yfr024c,Ysc84) respectively. Finally, for S-STAR, the P..P motif was extracted for 8 SH3 proteins (Fus1,Bbc1,Rvs167,Hse1,Bzz1,Myo3,Hof1,Nyo5) and the P..P.[KR] motif was extracted for 2 other SH3 proteins (Yfr024c,Ysc84). Again, all the algorithms failed to extract [KR]..P..P motif within the top 50 output for any of the SH3 proteins. In comparison, D-STAR extracted the specific P..P.[KR] and [KR]..P..P for more SH3 proteins (Fig. 4.4.2).

Since D-STAR extracts correlated motifs, it is interesting to further analyze the extracted associated sequence segments of the three proline-rich motifs as shown in Fig. 4.4.2. We were intrigued to discovered that all associated sequence segments

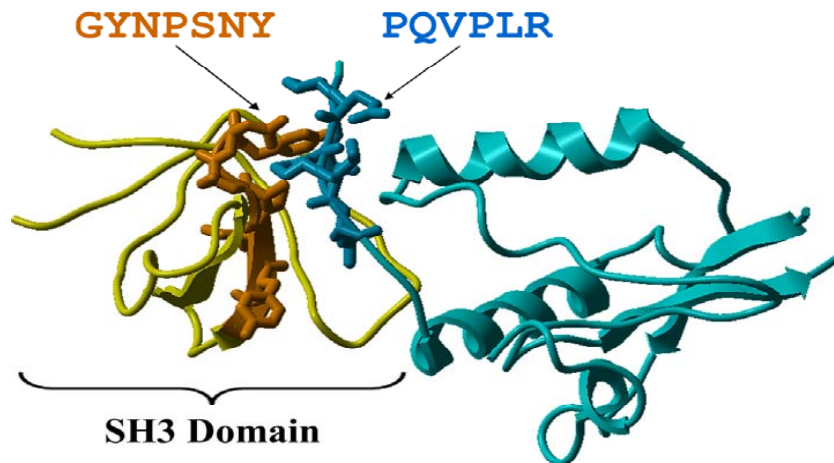


Figure 4.10: Evidence from PDB structural data - SH3 domain vs P..P.R. The figure illustrates the 3D structure of a SH3 domain of FYN tyrosine kinase (PDB ID: 1AVZ) bound to with another protein. The sequence segments that express the P..P.R motif and G..P.NY motif (detected by D-STAR in this work) are highlighted in dark blue and orange respectively. The two segments correspond to actual interacting subsequences.

extracted together with P..P, P..P.[RK] and [RK]..P..P by D-STAR were found within SH3 domains. In addition, we also discovered that all associated sequence segments of the three proline-rich motifs expressed a P.NY general consensus. Specifically, D-STAR extracted G..P.NY as the associated motif of P..P.[KR] motif. A further check into the PDB structure 1AVZ of an experimentally determined interaction between an SH3 protein and a protein expressing a P..P.[KR] motif reveals that the sequence segment in SH3 domain expressing the G..P.NY motif indeed forms a binding interface with the segment expressing the P..P.[RK] motif (Fig. 4.10). Hence, in this particular case, D-STAR has extracted correlated motifs that actually are binding motifs.

### TGF $\beta$ signaling pathway

Next, we applied D-STAR on the interaction network of TGF $\beta$  signaling pathway that was derived using LUMIER [134]—an automated high-throughput protein interaction detection technology that can detect phosphorylation-dependent interactions. Note that the original experiment was not specifically geared toward detecting interactions of any particular protein domain or motif. Hence, unlike the SH3-P..P dataset, it is not immediately apparent whether any relevant motif pairs can be found in the interaction

network. We applied D-STAR on this interaction dataset to see whether we can extract any interesting motif pairs. The dataset was retrieved from BIND database and consists of 446 interactions among 214 proteins. D-STAR was applied on the dataset with the same parameters used for SH3-P..P dataset. As we do not know what to expect as correct answer, we focused on validating the top motif pair extracted.

Interestingly, D-STAR extracted a motif pair, with general consensus patterns [TA]E [LI]Y[NQ]T and GKT[CIS][ILT][IL] (see Fig. 4.11), from 87 unique interactions as our top output. For ease of discussion, let us denote the motif pair as  $(X, Y)$ . First, we verified that  $(X, Y)$  is not likely to occur by chance as the estimated probability ( $p$ -value) of getting the motif pair with the same interaction set size is less than 0.001 (by testing the motif pair on 1000 randomly generated interaction data with the same network topology and sequences). Hence, we conjectured that the motif pair is a possible key interaction mechanism in the TGF $\beta$  signaling pathway.

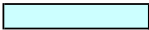
We also found that the sequence segment set of motif  $Y$  is enriched in known kinase phosphorylation motifs (27 sites in 50 segments, based on result from PhosphoMotif Finder [4]—see Fig. 4.12). To determine the significance of finding 27 sites in the segment sets, we generate 1000 segments sets, each containing 50 segments randomly selected from the same protein set. We found out that none of them contain at least 27 segments with the phosphorylation motifs, implying an estimated  $p$ -value  $< 0.001$ .

We listed the over-represented phosphorylation motifs in Table 4.13. Further analysis also showed that 5 out of 6 associated sequence segments of motif  $X$  were also found within kinase protein domains (determined using HMMER from Pfam [76]). Such biological characterization of our extracted motif pair  $(X, Y)$  with  $X$  as kinase motifs and  $Y$  as phosphorylation motifs is indeed in concurrence with the fact that signalling pathways are typically regulated by kinases through protein phosphorylation. This further indicates that our method have extracted a biologically feasible motif pair from the TGF $\beta$  interaction dataset.

We also investigated whether such kinase phosphorylation motifs may also be extracted using the OTM approach. For each kinase protein found in  $Y$  by D-STAR, we submitted their binding partners to MEME (*ZOOPS mode*, *Motif Width* = 4–8), Gibbs Sampler (*Motif Sampler mode*, *Motif Width* = 4–8, *Expected Motif Number*  $\geq 2$ ) and S-STAR ( $l = 6, 7, 8$ ,  $d = 1$  and  $k_n = 5$ ). We found that over-represented phosphorylation

Kinase Protein Set		
Proteins	Position	Segment
GI:4502431	244	TEIYQT
GI:40254649	248	TELYNT
GI:4501895	248	TELYNT
GI:6678323	245	AEIYQT
GI:4759226	245	AEIYQT
GI:4885457	191	TETYST

Phosphorylation Motifs Set		
Proteins	Position	Segment
GI:11024714	9	GKTITL
GI:11024714	85	GKTITL
GI:11024714	161	GKTITL
GI:11641237	20	GKTSII
GI:11967981	30	GKSSLA
GI:11967981	62	GATSLK
GI:13786127	73	SKRSLL
GI:13786129	44	GKTCLT
GI:16445426	149	GDTSLS
GI:19526471	85	GKTSRR
GI:19923750	33	GKTSFL
GI:21389385	17	GKTSLA
GI:22027525	769	SKTSIL
GI:30520350	27	GKTTIL
GI:34147073	199	LKNSLL
GI:41149704	277	GKRSTL
GI:41327767	32	GKTSLL
GI:4505571	262	GKRSRL
GI:4506713	9	GKTITL
GI:4506713	111	GKISRL
GI:4507449	28	GKTTFI
GI:4507761	9	GKTITL
GI:4757770	15	GKTSLL
GI:5031817	577	GCTSLK
GI:51036601	24	GKTSII
GI:56243590	904	QKTPLL
GI:7656900	28	GKTSLL
GI:10835049	16	GKTCLL
GI:10864013	118	WKTALL
GI:12849714	113	QYTSLL
GI:13786127	47	GDTSFL
GI:16903164	32	GKTCLL
GI:16903164	65	GKQHLL
GI:21361884	17	GKSCLL
GI:22003858	176	GNTMLL
GI:22027525	280	EVTSLI
GI:22218619	357	GQGSLL
GI:24111250	290	NKTDLL
GI:24586657	784	GLLSLL
GI:30039692	363	GGGSLL
GI:31543537	63	GKTCLI
GI:4502741	272	GKDLLL
GI:4505451	47	GETCLL
GI:4505487	92	YGTSLL
GI:4506363	19	GKTCLI
GI:4506381	14	GKTCLL
GI:46249393	14	GKTCLL
GI:47717139	87	GKTMLN
GI:9966809	15	GKTAIL
GI:9966861	23	GKTNLL

  
**The green-highlighted proteins are the proteins with real Kinase domains according to HMMER (5/6)**


  
**The red-highlighted proteins are the proteins with the phosphorylation sites as predicted by PhosphoFinder (27/50)**

Figure 4.11: The best motif pair found in TGF $\beta$ . The highlighted proteins on the left belongs to the Kinase domain while those on the right contain the Kinase phosphorylation motifs (as checked by another program PhosphoMotif Finder [4])

motifs can be found within the top 10 output segment sets for only 2 out of the 5 kinase proteins by all MEME, Gibbs Sampler and S-STAR (based on result from PhosphoMotif Finder).

Protein GI	Position	String	Site	Motif	Type
GI:11024714	9	GKTITL	KTIT	[R/K]xx[pS/pT]	PKC Kinase motif
GI:11024714	85	GKTITL	KTIT	[R/K]xx[pS/pT]	PKC Kinase motif
GI:11024714	161	GKTITL	KTIT	[R/K]xx[pS/pT]	PKC Kinase motif
GI:11024714	9	GKTITL	KTIT	Kxx[pS/pT]	PKA Kinase motif
GI:11024714	85	GKTITL	KTIT	Kxx[pS/pT]	PKA Kinase motif
GI:11024714	161	GKTITL	KTIT	Kxx[pS/pT]	PKA Kinase motif
GI:11641237	20	GKTSII	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:11967981	62	GATSLK	SLK	[pS/pT]x[R/K]	PKC and PKA Kinase motif
GI:11967981	30	GKSSLA	KSS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:13786127	73	SKRSLI	SKR	[pS/pT]x[R/K]	PKC and PKA Kinase motif
GI:13786127	73	SKRSLI	SKRS	[pS/pT]xx[S/T/Y]	CK2 Kinase motif
GI:13786127	73	SKRSLI	SKRS	[pS/pT]xxS	CK1 Kinase motif
GI:13786127	73	SKRSLI	KRS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:13786127	73	SKRSLI	SKRS	pSxx[E/pS/pT]	CK2 and Casein Kinase motif
GI:13786129	44	GKTCLT	TCLT	[pS/pT]xx[S/T/Y]	CK2 Kinase motif
GI:13786129	44	GKTCLT	KTCLT	Kxxx[pS/pT]	PKA Kinase motif
GI:16445426	149	GDTSLI	TSLI	[pS/pT]xx[S/T/Y]	CK2 Kinase motif
GI:16445426	149	GDTSLI	TSLI	[pS/pT]xxS	CK1 Kinase motif
GI:19526471	85	GKTSRR	TSR	[pS/pT]x[R/K]	PKC and PKA Kinase motif
GI:19526471	85	GKTSRR	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:19526471	85	GKTSRR	KTSRR	[R/K]x[pS/pT]x[R/K]	PKC Kinase motif
GI:19923750	33	GKTSFL	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:21389385	17	GKTSLA	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:22027525	769	SKTSIL	SKTS	[pS/pT]xx[S/T/Y]	CK2 Kinase motif
GI:22027525	769	SKTSIL	SKTS	[pS/pT]xxS	CK1 Kinase motif
GI:22027525	769	SKTSIL	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:22027525	769	SKTSIL	SKTS	pSxx[E/pS/pT]	CK2 and Casein Kinase motif
GI:30520350	27	GKTITL	KTIT	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:34147073	199	LKNSSL	KNS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:41149704	277	GKRSTL	KRST	[R/K][R/K]x[pS/pT]	PKA Kinase motif
GI:41149704	277	GKRSTL	KRST	[R/K][R/x]x[pS/pT]	PAKs phosphorylation motif
GI:41149704	277	GKRSTL	KRS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:41149704	277	GKRSTL	KRST	[R/K]xx[pS/pT]	PKC Kinase motif
GI:41149704	277	GKRSTL	KRST	Kxx[pS/pT]	PKA Kinase motif
GI:41327767	32	GKTSLL	KTS	[R/K]x[pS/pT]	PKC Kinase motif
GI:4505571	262	GKRSRL	KRS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:4506713	111	GKISRL	KIS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:4506713	9	GKTITL	KTIT	[R/K]xx[pS/pT]	PKC Kinase motif
GI:4506713	9	GKTITL	KTIT	Kxx[pS/pT]	PKA Kinase motif
GI:4507449	28	GKTIFL	KTT	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:4507761	9	GKTITL	KTIT	[R/K]xx[pS/pT]	PKC Kinase motif
GI:4507761	9	GKTITL	KTIT	Kxx[pS/pT]	PKA Kinase motif
GI:4757770	15	GKTSLL	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:5031817	577	GCTSLK	SLK	[pS/pT]x[R/K]	PKC and PKA Kinase motif
GI:51036601	24	GKTSLI	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif
GI:56243590	904	QKTPLL	TP	[pS/pT]P	Proline-directed Kinase motif
GI:56243590	904	QKTPLL	KTP	[R/K][pS/pT]P	Growth-associated histone H1 Kinase motif
GI:7656900	28	GKTSLL	KTS	[R/K]x[pS/pT]	PKC and PKA Kinase motif

Figure 4.12: The list of motifs of the phosphorylation sites that are over-represented in the segment set with the general pattern GKT[CIS][ILT][IL].

## 4.5 Conclusions

Discovery of novel binding motifs acting as interaction switches for biological circuits can lead to invaluable insights for important applications such as drug discovery, as various short binding motifs have been found to be associated with disease pathways. However, such motifs have also been known to be hard to find both experimentally and

Motif	Expected	Observed	Odd-Ratio
$[R/K]_x[S/T]$	3.15	17	5.40
$K_{xx}[S/T]$	1.22	6	4.92

Figure 4.13: The odd-ratio of known Kinase phosphorylation motifs found in D-STAR’s motif pair. As the motifs are degenerate, we compared their actual number of occurrence with their expected random occurrence within any random segment set of the same size preserving the same amino acid distribution as the whole dataset’s.

computationally [33].

The recently available protein-protein interaction data present a rich data source to aid in such important discoveries through motif discovery algorithms. The efforts can be hindered by sparse and noisy nature of existing protein interaction data, as well as the inadequacy of current biological knowledge. In this paper, we have proposed a novel approach of mining correlated *de-novo* motifs from interaction data. We formulated our approach as an  $(l, d)$ -motif pair finding problem for which we gave an exact algorithm, D-MOTIF, as well as its approximation algorithm, D-STAR. The approach is more robust in extracting motifs from noisy interaction data. Of course, since D-STAR is devised for finding linear sequence motifs, it would fail if one of the correlated motifs is a structural one. However, it may still be used to identify short conserved sequence regions that formed parts of such structural motifs. Given that existing protein structural data is still very limited when compared to available protein-protein interaction data, short conserved sequence regions identified by D-STAR could facilitate further biological experiments like mutagenesis studies.

While we have presented an approximation algorithm D-STAR to speed up the extraction of motif pairs from interaction data, more work will need to be done in order to scale up the approach to handle genome-wide interaction data or the larger DNA-protein interaction data. Also, as real biological motifs can be of varying lengths, we will also need to extend our current approach to discover binding motifs that are not of any predefined lengths. We leave these as future works.



## 4.6 List of publication

- 1) Tan S H, Hugo W, Sung W K, Ng S K. A correlated motif approach for finding short linear motifs from protein interaction networks. *BMC Bioinformatics*, 7:502, 2006.

## Chapter 5

# Discovering Interaction Motifs from Protein-Protein Interaction Data: D-SLIMMER

### 5.1 Introduction

We have shown in the earlier chapter that our interaction based method, D-STAR [47], performed better in finding SLiMs in the PPI data than existing motif occurrence based methods like MEME [121] (used by DILIMOT [40]) and Gibbs Sampler [122]. As D-STAR was found to be less scalable to handle full genomic PPI data, it was further improved by some recently published programs like MotifCluster [48] and SLIDER [46].

Despite these improvements, we observe that the current interaction motif approaches have a few limitations:

1. All interaction motif approaches to date have been targeting interacting pairs of SLiMs—these algorithms assume that the interaction can be explained by the presence of a pair of SLiMs. However, our structural study shows that roughly half of all domain-SLiM interfaces could not be represented by a SLiM. That is to say, they are non-linear.
2. The observation does not completely invalidate the results of D-STAR, MotifCluster and SLIDER since there are real domain-SLiM instances where both sides of the interface are linear. However, it does reveal their limitation. Specifically, cur-

rent programs would require that the protein domain that interacts with a SLiM **also contains** another conserved SLiM within it. This constraint is satisfied by the examples presented by D-STAR, most notably the signature SLiM G..P.NY of the SH3 domain. However, as we shall show later, this constraint limits the coverage of these methods.

Hence we designed a new interaction motif based approach which computes the interaction density between a non-linear motif, a protein domain, with a SLiM. The program, called D-SLIMMER (stands for **Domain-SLiM MinER**), somehow resembles the many-to-many (MTM) approach of DILIMOT described in the previous section because it collects the interaction partners of a protein domain for SLiM mining. However, it has one important difference: the score of the SLiMs are based on interaction-density as opposed to occurrence frequency. We also implemented rigorous statistical and homology filtering to ensure that the SLiMs are not mere random or homology artifacts.

To validate the effectiveness of D-SLIMMER, we checked if it can find real SLiMs from currently available PPI data. We also would like to know if it performs any better than the existing programs. To this end, we collected a reference set of experimentally verified SLiMs along with their recognition domain from the ELM and MiniMotif database [37–39]. Our benchmark contains 38 reference domains which are known to interact with a total of 113 different SLiMs (some of the domain recognize a few classes of SLiMs). For each benchmark domain, we generate ten PPI datasets, nine from the BioGRID database [53] and another one from the Human Protein Reference Database (HPRD) [4]. These datasets are split according to the source organism of the PPI data (since protein interaction across different species is unlikely to be included in the databases). We then run D-SLIMMER, SLIDER, MotifCluster and the latest occurrence based program, SLiMFinder, on the PPI data of the reference domains to see if the programs can find the SLiMs associated with the reference domains. D-STAR was not included in the comparison because of its scalability issue on some of the domain’s PPI.

D-SLIMMER managed to mine significantly more reference SLiMs compared to the other three methods. It manages to find 29 out of the 113 reference SLiMs while the best existing method only managed to find 9 of them. We also show that, in PPI data,

the real SLiM’s interaction density signal is stronger than its occurrence signal. Hence, we propose that interaction based approach is more suitable in PPI SLiM mining.

We also present four examples from D-SLIMMER’s predicted SLiMs which have significant biological supports. The first SLiM is A..IQ..[FWY]R, which is predicted by D-SLIMMER as a target of the EF hand domain. Our SLiM partially matches the IQ motif **..[SACLIVTM]..[ILVMFCT]Q...[RK].{4,5} [RKQ]** (ELM ID: LIG\_IQ) which is a known target to EF hand. Furthermore, our SLiM’s [FWY] position matches the requirement of the large hydrophobic side-chain just before the basic [RK] residues [54]; this is not reflected in LIG\_IQ. We note that our SLiM lacks any matching to the last [RKQ] residues of ELM’s LIG\_IQ. Nonetheless, we verified that most of our A..IQ..[FWY]R instances were either annotated as IQ motif in the UNIPROT database [7], the PFAM database [76] or in the literature [5]—indicating that they are valid sites. Interestingly, the SLiM A..IQ..[FWY]R’s occurrence sites within the KCNQ proteins that we reported (one human KCNQ2 protein and one *D. melanogaster*’s), as well as in all 4 other human KCNQ proteins, are annotated to be functional IQ motifs but they also missed the last [RKQ] positions [5]. This suggests that our A..IQ..[FWY]R SLiM is a valid IQ motif that is also viable for interaction.

The second example SLiM is EG..DLFD, a SLiM predicted for the Retinoblastoma B domain (in short, RB\_B) to bind to the E2F family of transcription factors. It partially matches the suffix of an ELM SLiM related to RB\_B domain: **..[LIMV]..[LM][FY]D** (ELM: LIG\_Rb\_pABgroove\_1). Our D-SLIMMER’s EG..DLFD SLiM included an acidic residue just before the conserved suffix [LM][FY]D of LIG\_Rb\_pABgroove\_1. This acidic residue is also found in all binding regions used by Adenovirus’ E1A protein to block the E2F-Rb interaction [55]. We therefore propose that the acidic residue before the [LM][FY]D positions is important for E2F specific interaction.

The third example SLiM PPPGL is a target SLiM of the GYF domain which is found in a versatile adaptor domain that recognizes proline-rich sequences (PRS). Our detected SLiM PPPGL matches the recently reported PPG $\phi$  motif (where  $\phi$  = hydrophobic amino acid, except for tryptophan) for the GYF domain [56]. The SLiM PPG $\phi$  has only recently been published in the literature, hence it is yet to be included in the current SLiM databases.

Our final example is a (predicted) novel SLiM detected from the PPI data of the

Sir2 domain (Pfam ID: PF02146). Sir2 is found in a family of protein deacetylase involved in important biological processes such as repression of gene transcription, DNA repair process, cell cycle progression, chromosomal stability and cell aging [57]. Our D-SLIMMER detected a novel SLiM AK.V.I which agrees with the binding preferences of Sir2 [6] (the K is the acetylated lysine position). Additional biological significance of this novel SLiM is suggested by the occurrences of the SLiM in Glyceraldehyde-3-phosphate dehydrogenase (GPDH) proteins. The interactions of Sir2 proteins and GPDH have been found in two species, *D. melanogaster* and *S. cerevisiae*. We also confirmed that the AK.V.I SLiM is strongly preserved in the GPDH proteins of multiple species. From the 3D structures, we further showed that the SLiM is indeed located on the surface of GPDH, that is, it is physically accessible for recognition. Based on these evidences, we propose that the interaction between Sir2 and GPDH is mediated by the novel SLiM AK.V.I.

## 5.2 Materials and Methods

### 5.2.1 Overview of the D-SLIMMER algorithm

Our D-SLIMMER algorithm works as follows. First, we collect the interaction data involving a protein domain of interest  $D$  from a set of PPI data  $I$ . From the set of interacting partners of  $D$  extracted from  $I$ , we find the statistically over-represented motifs. We then rank each such motif  $M$  with a score based on its interaction density with the domain  $D$ . The workflow of D-SLIMMER is depicted in Fig. 5.1.

### 5.2.2 Preliminaries

Consider a set of non-homologous PPI data  $I$  and a target domain  $D$ . Let  $P$  be the set of the proteins in  $I$ . We use  $P(D)$  to denote the set of proteins in  $P$  containing the domain  $D$ , and  $P'(D)$  the set of interaction partners of  $P(D)$  in  $I$ . Similarly, for any linear motif  $M$ , let  $P(M)$  denote the set of proteins containing  $M$  in  $P$ .

Let  $I(D, M)$  be the subset of  $I$  where one protein in the PPI contains the domain  $D$  while the other protein contains the motif  $M$ . Mathematically,  $I(D, M) = I \cap \{P(D) \times P(M)\}$ . We denote  $P(M|I(D, M)) = \{p_m \in P(M) | \exists p_d \in P(D) : (p_d, p_m) \in I\}$ —that is, the subset of  $P(M)$  involved in the interaction set  $I(D, M)$ . 5.2 depicts an example

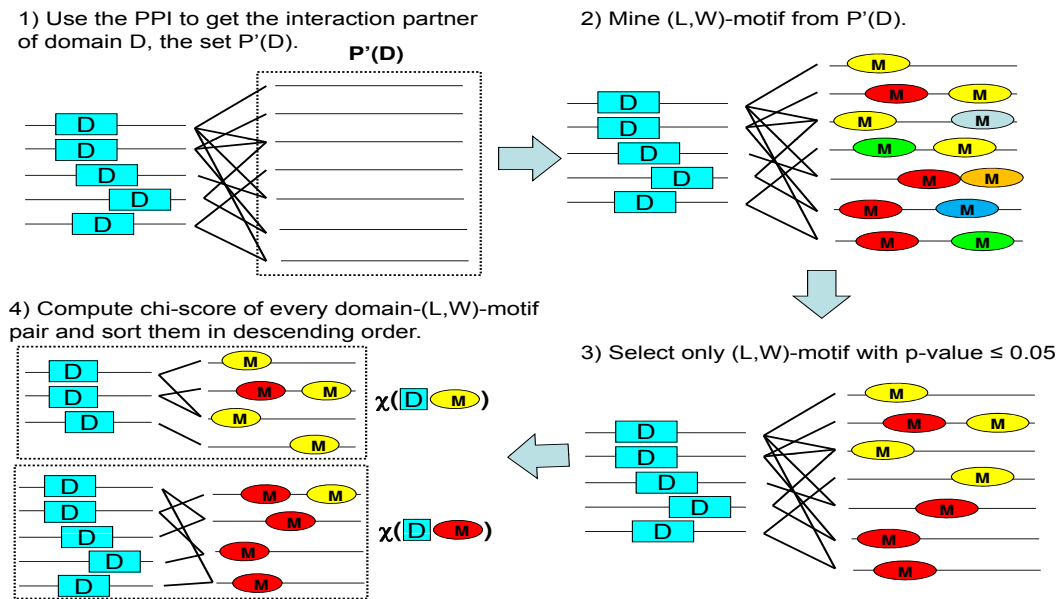


Figure 5.1: The flowchart of D-SLIMMER algorithm.

to illustrate the relationships between  $P(D)$ ,  $P(M)$ ,  $I(D, M)$ , and  $P(M|I(D, M))$ .

We compute the statistical significance of each such motif  $M$  in  $P(M|I(D, M))$  based on a Markov Chain background model. For motifs whose p-value is less than or equal to 0.05, D-SLIMMER then computes the density of interactions between  $P(M)$  and  $P(D)$  using a  $\chi$ -squared function which quantifies the significance of the number of observed interactions between  $D$  and  $M$  (defined as the size of the set  $I(D, M)$ ) as compared to the expected number based on random interactions. Finally, D-SLIMMER ranks the detected  $(D, M)$  pairs based on their interaction density scores.

Next, we provide further details on (1) how to collect the PPI data  $I$  over the protein set  $P$ , (2) how to obtain a candidate motif  $M$ , and (3) define the chi-square score between a domain  $D$  and a motif  $M$ .

### 5.2.3 Mining SLiMs from each target domain's PPIs

We use the  $(L, W)$ -motif to model the SLiMs in this work. An  $(L, W)$  protein motif is a string over the alphabet  $\Sigma \cup \{.\}$  where  $\Sigma = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}$  (the 20 amino acid residues) and '.' is the wildcard character. There is no restriction on the length of an  $(L, W)$ -motif  $M$  but for any length- $W$  substring  $M' = M[i, i + W - 1]$  of  $M$ , if  $M'$  starts with a non-wildcard character, then  $M'$  contains at least  $L$  non-wildcard characters.

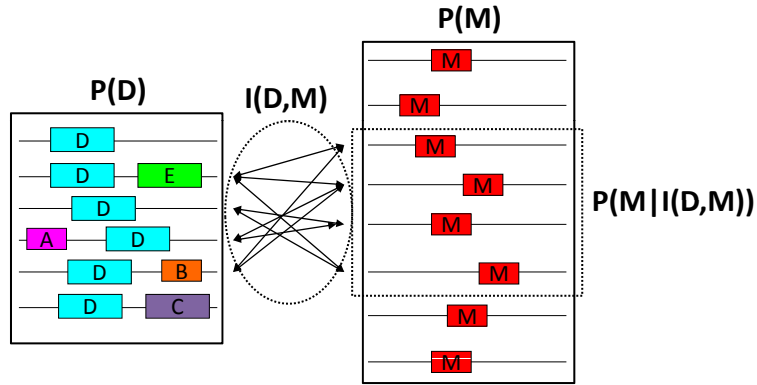


Figure 5.2:  $P(D)$  ( $P(M)$ , respectively) is the set of protein containing domain  $D$  (motif  $M$ , respectively).  $I(D,M)$  is the subset of the PPI data  $I$  where one protein of the interaction contains the domain  $D$  while the contains  $M$ .  $P(M|I(D,M))$  is the subset of  $P(M)$  which is involved in  $I(D,M)$ .

For each PPI dataset  $I$ , we obtain the subset of interactions involving proteins with the target domain  $D$ . We then collect the candidate SLiMs from the interaction partners of  $P(D)$ , i.e  $P'(D)$ , using the TEIRESIAS program [124]. Its inputs are the parameters  $L$ ,  $W$  and a minimum motif occurrence  $k$ . We chose  $L = 4$  and  $W = 8$  (MotifCluster [48] and SLIDER [46] used a similar parameter (4 defined positions over length 8 motif) but their motif model is of fixed length). Since SLiMs with high interaction densities may not always be amongst those with the highest occurrences [46], we set  $k = 5$  (which is a rather weak occurrence threshold so that we do not to miss any true SLiM).

#### 5.2.4 Removing redundant $(L,W)$ -motif occurrences

We observed that there were many redundant occurrences of the  $(L,W)$ -motifs arising from sequence homology. We apply homology clustering to avoid double-counting these occurrences.

We call the the homology clustering of a protein set  $P$  containing the  $(L,W)$ -motif  $M$  as the clustering  $C(P, M)$ . It is defined as the partitioning of the set  $P$  such that any two proteins from two different clusters in  $C(P, M)$  must satisfy: 1) the full sequences of the two proteins share less than 70% sequence identity, and 2) if the instances of  $M$  occurred in a particular domain  $X$ , the instances of  $X$  in the two proteins must also share less than 70% sequence identity.

The last condition was based on our observation that a motif  $M$  could occur in

several instances of a domain  $X$ . Consider the case where  $X$  occurs in three proteins  $P_1$ ,  $P_2$  and  $P_3$ . Suppose  $P_1$  contains the domains  $(A, X)$ ,  $P_2$  contains  $(B, X)$  and  $P_3$  contains  $(C, X)$ . Because of the presence of the other domains along with  $X$ , the full sequences of  $P_1$ ,  $P_2$  and  $P_3$  may share less than 70% sequence identity. However, the instances of  $X$  in  $P_1$ ,  $P_2$  and  $P_3$  may share more than 70% sequence identity. In this case,  $M$ 's occurrences in them are considered redundant.

In this work, we used CD-HIT version 4.0 [140] to compute both the homology clustering of the proteins in  $P$  and the Pfam domain instances in  $P$ . D-SLIMMER only accept those  $(L, W)$ -motif  $M$  satisfying  $|C(P(M|I(D, M)), M)| \geq 5$ .

### 5.2.5 Filtering randomly occurring SLiMs using a 3<sup>rd</sup> order markov chain background.

We also need to filter away SLiMs whose occurrences are not statistically over-represented given the background. Let us consider a domain  $D$  and an  $(L, W)$ -motif  $M$ . Suppose  $M$  (independently) occurs  $n$  times in  $P'(D)$ . We gauge the likelihood of this event being a random occurrence by computing  $M$ 's p-value. We model the (background) occurrence probability of  $M$  using a 3<sup>rd</sup> order Markov Chain model.

First, the frequencies of all amino acid 4-mers are computed from the set of all non-homologous proteins in the whole PPI  $I$ . Then, given an  $(L, W)$ -motif  $M$  of length  $|M|$ , its (background) probability of occurring at a single position in a protein sequence with a 4-mer distribution similar to  $I$  would be

$$\mathcal{P}(M) = \mathcal{P}(M_0) \prod_{i=1}^{|M|-4} \mathcal{P}_{cond}(M_i)$$

where  $M_i$  denote the 4-mer that starts at the  $i^{th}$  position in  $M$ ,  $\mathcal{P}(M_0)$  indicates the frequency of  $M_0$  in the background sequences and  $\mathcal{P}_{cond}(M_i)$  is the frequency of seeing  $M_i$  given its 3-mer prefix. The frequency of any 4-mer with wildcard position(s) is computed by summing up the marginal probabilities over all 20 amino acids on that position.

Let  $S$  be a protein of length  $\ell$  from  $I$ . The probability of an  $(L, W)$ -motif  $M$  to (randomly) occur in  $S$  at least once is

$$\mathcal{P}_{1+}(M, \ell) = 1 - (1 - \mathcal{P}(M))^{\ell - |M| + 1}$$



Next, given a protein set  $P$  from  $I$  proteins of length  $\ell$  each, the probability of an  $(L, W)$ -motif  $M$  to occur in at least  $n$  proteins in  $P$  is the sum of binomial probabilities

$$\mathcal{P}(M, |P|, n, \ell) = \sum_{i=n}^{|P|} \binom{|P|}{i} \mathcal{P}_{1+}(M, \ell)^i (1 - \mathcal{P}_{1+}(M, \ell))^{|P|-i} \quad (1)$$

A statistically over-represented motif  $M$  is then defined as one with occurrences such that  $\mathcal{P}(M, |C(P'(D))|, |C(P(M|I(D, M)))|, \ell) \leq 0.05$  ( $\ell$  is approximated using the average sequence length of protein in  $I$ ). We use this occurrence-based function  $\mathcal{P}$  to filter away randomly occurring  $(L, W)$ -motifs (as opposed to using it for ranking the significance of the  $(L, W)$ -motifs, as done in occurrence-based methods).

### 5.2.6 Scoring domain-SLiM interaction density: the chi-square function

Now, for each candidate  $(D, M)$  pair where  $D$  is a domain and  $M$  a statistically over-represented  $(L, W)$ -motif  $M$ , we compute the significance of  $I(D, M)$  given the size of  $P(D)$  and  $P(M)$ . We only count non-homologous interactions in  $I(D, M)$ . Two pairs of interacting proteins  $(P_1, P_2)$  and  $(P_3, P_4)$  are homologous when  $P_1$  and  $P_3$  ( $P_4$ , respectively) are homologous and  $P_2$  and  $P_4$  ( $P_3$ , respectively) are homologous. We use the notation  $|I(D, M)|$  to define the number of *non-homologous interactions* in  $I(D, M)$ .

First, we require  $|I(D, M)| \geq 7$  to avoid artificially small cases which required only a few interactions to reach high density. Next, we assume a background interaction distribution with a uniform density over the whole PPI. We define the background density by  $\rho_{bg} = \frac{|I|}{\binom{|P|}{2}}$  which computes the density of interaction in  $I$  as compared to a full clique (in which case  $\rho_{bg} = 1.0$ ). Given a PPI  $I$  over the protein set  $P$ , we define the density of the interaction between two protein sets containing the domain  $D$  and motif  $M$  to be  $\rho(D, M, I) = \frac{|I(D, M)|}{MaxInt(D, M)}$ .  $MaxInt(D, M)$  is the maximum number of interactions between the two protein sets  $P(D)$  and  $P(M)$ , assuming they form a complete bipartite graph, i.e.  $MaxInt(D, M) = |P(D)||P(M)| - \binom{|P(D) \cap P(M)|}{2} - (|P(D) \cap P(M)|)$ . Given the above definitions, we score the significance of the interaction density between  $D$  and  $M$  by

$$\chi(D, M, I) = \begin{cases} MaxInt(D, M) \frac{(\rho(D, M, I) - \rho_{bg})^2}{\rho_{bg}} & \rho(D, M, I) \geq \rho_{bg} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

### 5.2.7 Removing domain-SLiM redundancies

Finally, we sort all statistically over-represented  $(L, W)$ -motifs by their  $\chi$ -square score in a non-increasing order. However, this list may still have redundant motifs (i.e. there are two motifs  $M$  and  $M'$  such that  $I(D, M)$  and  $I(D, M')$  are highly overlapping). We perform the following additional step to remove such redundancy. For any motif  $M$  with rank  $R_M$ , when there exists another  $M'$  with a better rank  $R_{M'} < R_M$  and  $\frac{|I(D, M) \cap I(D, M')|}{|I(D, M)|} \geq 0.75$  (i.e. more than 75% of  $I(D, M)$  are also in  $I(D, M')$ ),  $M$  is deemed redundant with respect to  $M'$  and we remove it from the final list.

## 5.3 Results and Discussion

### 5.3.1 Comparative study between D-SLIMMER and existing methods

We compare D-SLIMMER with three existing SLiM finding programs: MotifCluster [48], SLIDER [46] and SlimFinder [42]. As a benchmark, we collected 113 experimentally verified SLiMs which are known to be recognized by 38 different domains (termed as the reference domains) from the ELM and MiniMotif database [37, 38]. The reference SLiMs, which are in the form of regular expression, are listed in the supplementary file at <http://www.comp.nus.edu.sg/~hugowill/D-SLIMMER/ReferenceSLiMs.xls>

We collected PPIs from the BIOGRID [53] release 3.1.73 (dated Feb, 2011) and HPRD [4] release July, 2010. Each protein is identified by their UniProt ID [7] and we identify the Pfam domains [76] in each protein based on the mapping provided in the INTERPRO database release 30.0 [77]. We split the PPI data of BIOGRID based on the species in which the protein interaction occurs \*. There are 8 species which have at least 7 interactions involving the 38 reference domains. The species are *H. sapiens* (Human), *M. musculus* (Mouse), *R. norvegicus* (Rat), *S. cerevisiae* (Baker's yeast), *S. pombe* (fission yeast), *D. melanogaster* (fruit fly), *C. elegans* (ring-worm), and *A. thaliana*. For convenience, we name the datasets using the database and species name, e.g BioGRID-*S.cerevisiae*. Note that the HPRD dataset only contains PPIs of Human (HPRD-*H.sapiens*). We also created a combined PPI dataset from multiple species named BioGRID-ALL. This is because some SLiMs can be uti-

---

\*in cross-species cases, we take the source organism of the protein with the reference domain

lized across species and we may be able to better identify them from this pooled data since there it will contain more of the SLiM’s and its interaction instances. In total, we have 10 PPI datasets to mine SLiMs from. Each dataset’s PPI and protein set sizes are listed in the supplementary file at <http://www.comp.nus.edu.sg/~hugowill/D-SLIMMER/ComparisonStatisticBySpecies.xls>.

For evaluation, we check if each of the reference SLiMs is re-discovered in the top 50 SLiMs reported by each method in each of the 10 datasets. A reference SLiM, in the form of a regular expression, is said to occur in a predicted motif when there is a regular expression match between the reference SLiM and the predicted SLiM.<sup>†</sup>

When a reference SLiM is known to occur within the N-terminal or C-terminal of its protein, we only accept a predicted SLiM when at least half of its occurrences are within 20 amino acids (which is roughly the maximum length of the known SLiMs) from the N-terminal or C-terminal, respectively.

### Program parameters

We ran D-SLIMMER with the  $(L, W)$ -motif parameter set to  $(4, 8)$  (other parameters are as described in the Materials and Methods section above). We ran MotifCluster using  $l = 8$ ,  $d = 4$  and  $numSeed = 5000$ , and SLIDER with  $l = 8$ ,  $d = 4$  and a maximum wall-time set to 24 hours. Since both MotifCluster and SLIDER output SLiM pairs, given a motif pair  $(M_1, M_2)$ , if the reference SLiM  $M$  occurs in  $M_1$ , we require that at least 50% of  $M_2$ ’s occurrences are within the corresponding domain of  $M$ . For SLiMFinder, we provide the partner protein set  $P'(D)$  as SLiMFinder’s input and run it for at most 24 hours as well.

### Comparison result

The comparison result is shown in 5.1. D-SLIMMER was able to re-discover significantly more reference SLiMs than the other methods. It found 29 out of the 113 reference

---

<sup>†</sup>In the case of SLiMFinder, since it can report residue groups in its SLiMs, we require that the intersection between a predicted group P and a reference SLiM’s group R are at least half the size of P and R. That is to say,  $\frac{|P \cap R|}{|P|} \geq 0.5$  and  $\frac{|P \cap R|}{|R|} \geq 0.5$ . For example, if the ELM motif contain the amino acid group [LIVM] and SLiMFinder reported [LIA] for the position, we will accept [LIA] as a matching position for [LIVM] since  $\frac{|LI|}{|LIVM|} = \frac{2}{4} = 0.5$  and  $\frac{|LI|}{|LIA|} = 0.67$ .

Table 5.1: Performance comparison between D-SLIMMER, MotifCluster, SLIDER and SLiMFinder. This table shows the best rank of each method’s detected SLiMs containing a reference SLiM for a domain. The best rank is chosen among all different species’ including the combined species dataset. Ties are resolved by reporting the median rank of the motifs sharing the same score. “-” is listed when a method has not detected any SLiM containing the reference SLiM within its top-50 SLiMs.

Domain	Reference SLiM	D-SLIMMER Rank	Motif Cluster Rank	SLIDER Rank	SLiM Finder Rank
14-3-3	R.[ <sup>^</sup> P][ST][ <sup>^</sup> P]P	1	-	1	-
	R[SFYW].S.P	1	-	1	-
	[RHK][STALV].[ST].[PESRDIF]	1	-	1	1
Arm	K[KR].[KR]	22	30.5	-	-
Cyclin_N	[RK].L.{0,1}[FYLVMP]	28	25	10	-
FHA	T.[DE]	4	-	-	-
	T.[ILV]	5	-	-	-
	T.[SA]	14	-	-	-
MATH	[PA][ <sup>^</sup> P][ <sup>^</sup> FYWIL]S[ <sup>^</sup> P]	13	-	25.5	-
MYND_zc_finger	P.L.P	22	-	-	-
PCNA_C	Q.[ <sup>^</sup> FHWY][ILM][ <sup>^</sup> P] [ <sup>^</sup> FHILVWYP][DHFM][FMY]	2	-	-	-
PID	NP.Y	3	-	-	1
	[LF].NP.Y	-	-	-	1
SH2	Y..M	14	-	1	-
	Y..P	16	-	-	-
	Y.N	-	-	-	3
	Y[VI]N	-	-	-	3
SH3.1	P..P	1	1	1	-
	[RKY]..P..P	1	1	-	-
	[KR].LP..P	30	-	-	-
	P...P..P	14	-	-	-
	P..P.[RK]	1	1.5	1	-
	P...PR	8	16	25.5	-
	P...PR	11	-	-	-
	P.[IV]PPR	8	-	-	-
	P.P	1	1	-	1
P.[PA]..R	1	16	-	-	
WW	PP.Y	1	1	-	1
	PPLP	46	-	-	-
	PPR	22	-	-	-
	[ST]P	35	-	-	3
Ubiquitin	[VILMAFP]K.E	46	-	-	-

SLiMs, covering 12 different domains. The performance of the other three methods are not as good—both MotifCluster and SLIDER recovered 9, while SLiMFinder recovered 8.

We noticed that SLiMFinder mostly returned a small number of SLiMs (less than 10). As such, we also checked the number of SLiMs ranked better than or equal to 10 (instead of 50). In this case, D-SLIMMER is still the leader by finding 15 reference SLiMs compared to 8 found by SLiMFinder. In fact, in the extreme case when we only considering the best SLiM (i.e. rank 1 SLiMs), D-SLIMMER found 9 reference SLiMs while SLiMFinder found 5. Both SLIDER and MotifCluster also found similar number of SLiMs as SLiMFinder. The detailed ranking of the reference SLiMs found in each dataset are listed in the supplementary file at <http://www.comp.nus.edu.sg/~hugowill/D-SLIMMER/ComparisonStatisticBySpecies.xls>.<sup>‡</sup>

### 5.3.2 Scoring function analysis: Occurrence frequency vs. interaction density

We have elected to use interaction density as the scoring function for D-SLIMMER instead of occurrence frequencies. Therefore, an interesting question to ask is whether our choice of using interaction density for scoring was one of the reasons for our D-SLIMMER’s superior performance in detecting the reference SLiMs. Some occurrence-based approaches (Gibbs Sampler [122], MEME [121] and SP-STAR [128]) had already been shown to be inferior in detecting SLiMs in the synthetic dataset and two biological datasets (SH3 and TGF $\beta$ ) when compared to D-STAR [47]. In SLIDER’s paper, Boyen *et al.* also found that many of the SLiMs with high interaction support did not have high occurrence frequencies [46]. However, these results were rather anecdotal based on only a few cases. In this section, we provide an in-depth study on the comparative suitability of the two measures in the context of mining SLiMs from the PPI data.

---

<sup>‡</sup>We also checked if the results of D-SLIMMER depended on the separation into species datasets. As suggested by one of the examiner, I ran D-SLIMMER on the same BioGRID and HPRD datasets, but separated by the experimental method used to generate the PPI data (e.g, Yeast Two Hybrid, TAP-MS, etc). The results is similar with 31 reference SLiM covered (two additional SLiMs are found but the difference is not significant). The detailed result of D-SLIMMER run on these datasets are given in <http://www.comp.nus.edu.sg/~hugowill/D-SLIMMER/ComparisonStatisticByMethod.xls>

We use the two scoring functions  $\mathcal{P}$  and  $\chi$  (Equations 1 and 2) to represent the occurrence-based approach and the interaction-based approach, respectively. For each reference domain  $D$ , we rank all statistically over-represented (4,8)-motifs  $M$  (those whose  $\mathcal{P}(M, |C(P'(D))|, |C(P(M|I(D, M))|, \ell) \leq 0.05$ ) using these two scoring functions. We then compare the computed rankings of those (4,8)-motifs containing the reference SLiM to see which of the two scoring functions assigned higher rankings to these reference motifs more frequently. We compare both the best rank and the sum of the ranks of the best 10 (4,8)-motifs. The full listing of the comparison result is given in the supplementary file at <http://www.comp.nus.edu.sg/~hugowill/D-SLIMMER/OccurrenceVsInteractionBasedScoring.xls>.

We only performed our analysis on the BioGRID-ALL and HPRD datasets—to see if our results are consistent in a combined species and a single species dataset, respectively. For the BioGRID dataset, 82 out of 113 reference SLiMs have instances among the statistically over-represented (4,8)-motifs. Among these 82 SLiMs, 63 of them were ranked higher by the  $\chi$ -score. When we consider the rank sum of the best 10 (4,8)-motifs, the  $\chi$ -score function again dominates with better rankings in 63 of them.

There are 76 reference domains which have statistically over-represented (4,8)-motif(s) in HPRD. Again, our interaction density  $\chi$ -score function assigned higher rankings for those with the reference SLiMs in 47 out of 76 cases. In terms of the rank sum of the best 10 (4,8)-motifs,  $\chi$ -score also gave higher rankings for 51 out of 76 reference SLiMs. These results indicate that our use of interaction density as scoring function is able to detect the weak signals from SLiMs in the PPI data much better than using the conventional motif occurrence frequencies.

### 5.3.3 Biologically interesting SLiMs reported by D-SLIMMER

Computational methods such as D-SLIMMER are useful only if the SLiMs that are detected are indeed biologically interesting. In other words, the detected SLiMs should be *bona fide* for the proteins' interactions and hence worthwhile for biologists to attempt further biological validation. Because of our limited resources for biological validation, we shall illustrate the usefulness of our SLiMs through literature validations with several relevant examples.

We selected four example SLiMs for discussion in this section. These four example SLiMs are supported by multiple PPI detection experiments and other supporting data. Two of them partially matched the known ELM SLiMs (for Calmodulin (Pfam ID: PF00036) and RB\_B domain (Pfam ID: PF01857), respectively). The third is a class 2 SLiM for GYF domain (Pfam ID: PF02213) which was reported elsewhere in the literature but has yet to be added into ELM or MiniMotif. The fourth example is a candidate novel SLiM for Sir2 domain (Pfam ID: PF02146).

### **Variant of the Calmodulin’s IQ motif (PF00036)**

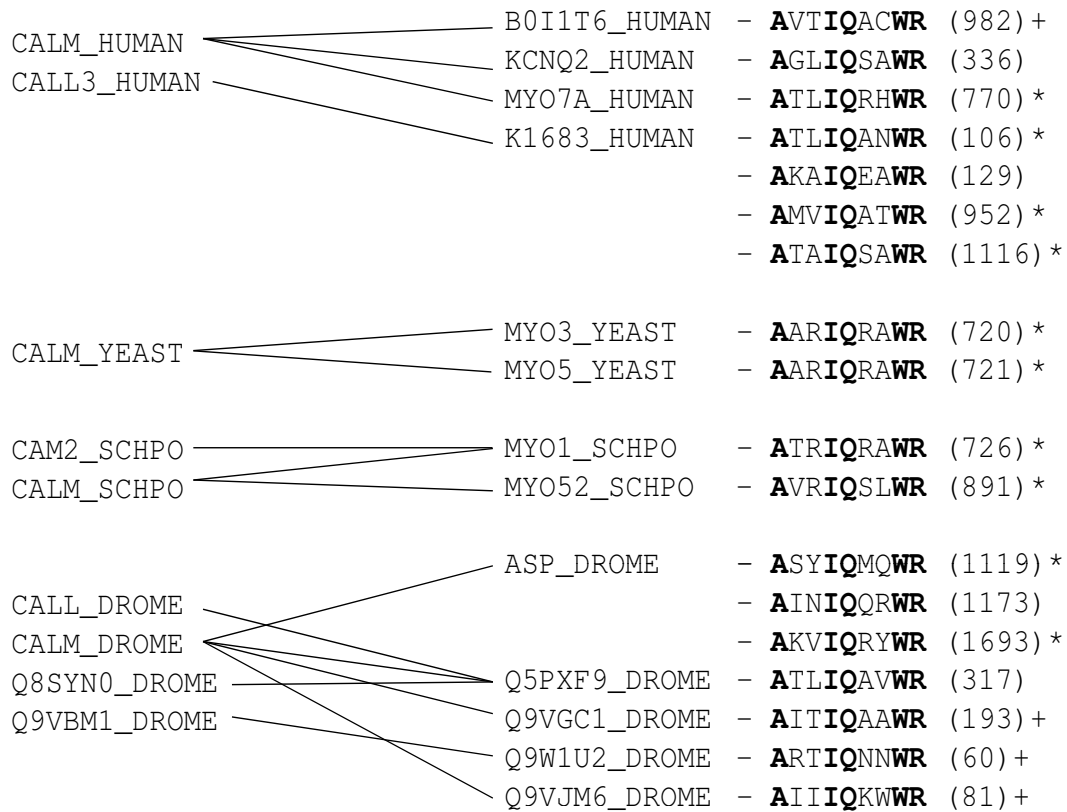
EF hand (Pfam ID: PF00036) is a biologically important family of calcium binding proteins which can be divided roughly into two classes: signaling proteins and buffering/transport proteins. EF hand proteins bind their target proteins in one of two different manners. They bind some of their target proteins in a Calcium ( $\text{Ca}^{2+}$ )-independent manner through binding motifs called the IQ motifs, and their other partners in a  $\text{Ca}^{2+}$ -dependent manner using motifs called the 1-8-14 motif or 1-5-10 motif [141]. The known ELM SLiM for the IQ motif is the SLiM `..[SACLIVTM]..[ILVMFCT]Q...[RK].{4,5}[RKQ]` (ELM ID: LIG\_IQ).

Our D-SLIMMER reported the SLiM `A..IQ..WR`, found in the BioGRID-ALL dataset at rank 14, to be a target of the EF hand domain. The detected SLiM includes 4 out of 5 positions in the ELM’s LIG\_IQ SLiM, and missed the last `[RKQ]` residue. D-SLIMMER also reported a similar SLiM `AT.IQ...R` found in rank 1 from the BioGRID-D.melanogaster dataset. The domain-SLiM interacting proteins are depicted in 5.3.

We note that the position immediately preceding the arginine (R) residue in `AT.IQ...R` is also occupied by hydrophobic aromatic residues phenylalanine, tryptophan or tyrosine (F,W, or Y); similar to the W position in `A..IQ..WR`. This position was also shown to be important for occupying a hydrophobic cleft on the Calmodulin domain and often occupied by residues with large hydrophobic side chain such as phenylalanine(F), tryptophan(W), tyrosine(Y) or methionine(M) [54]. On the other hand, the threonine (T) position in `AT.IQ...R` was generally described as wildcard positions in ELM’s LIG\_IQ and multiple literatures. Thus, we combine the two SLiMs into `A..IQ..[FWY]R`.

We found that 7 out of 14 proteins containing instances of our SLiM have already been annotated by the UNIPROT database to contain IQ motif at the SLiM’s sites [7].

A. SLiM A..IQ..WR from combined BIOGRID PPI (Rank 15)



B. SLiM AT.IQ...R from *D.melanogaster's* BIOGRID PPI (Rank 1)

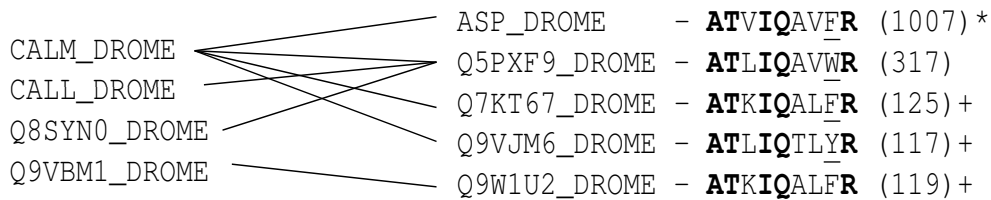


Figure 5.3: (A) The PPI corresponding to EF hand domain and the SLiM A..IQ..WR found from the combined PPI data of BioGRID. The source organism are indicated in the protein names. The instances of A..IQ..WR are listed along with their position in their respective protein sequences. Among the 13 proteins with A..IQ..WR, 7 of them (the IQ motif sites are marked with asterisks (\*)) are annotated to have IQ motif at the site of the SLiM by UNIPROT. Another 4 are annotated to have the Pfam domain regions of IQ motif (Pfam ID: PF00612) which describe EF hand binding sites (marked with +). The remaining two proteins are also annotated to have the IQ motifs at the occurrence site of A..IQ..WR [5]. (B) A similar IQ motif is also found in the BioGRID PPI dataset of *D. melanogaster*. The SLiM is AT.IQ...R which, upon inspection on the position directly before the last R, is actually AT.IQ..[FWY]R. Combining the (A) and (B) gave us the SLiM A..IQ..[FWY]R for Calmodulin.



```

KCNQ1_HUMAN      ..AASLIQTAWRCYAAEN..
KCNQ2_HUMAN      ..AAGLIQSAWRFYATNL..
KCNQ3_HUMAN      ..AAELIQAAWRYYATNP..
KCNQ4_HUMAN      ..AANLIQSAWRLYSTDM..
KCNQ5_HUMAN      ..AANLIQCVWRSYAADE..

Q5PXF9_DROME     ..AATLIQAVWRCYAADE..

```

Figure 5.4: The sequence alignment of 5 human KCNQ along with *D. melanogaster*'s KCNQ protein Q5PXF9 indicates that their IQ motif instances also missed the last position of the ELM's IQ motif ( $..[SACLIVTM]..[ILVMFCT]Q...[RK].\{4,5\}[RKQ]$ )—the matching positions for the [RKQ] residue are underlined.

Another 5 were annotated to contain Pfam's IQ motif domain (Pfam ID: PF00612) and the SLiM A..IQ..WR occurred within the domain's instances (this annotation was also given in UNIPROT [7]). Out of the remaining two, the KCNQ2\_HUMAN protein also has an IQ motif at the occurrence region of our SLiM [5]. We further confirmed that the instance of A..IQ..WR in Q5PXF9\_DROME (an unreviewed KCNQ protein of *D. melanogaster* in the UNIPROT database) is located at a similar position as KCNQ2's.

Since our SLiM missed the last [RKQ] positions in LIG\_IQ, we further investigated the literature for instances of A..IQ..[FWY]R which also missed the position and yet were shown to be *bona fide* IQ motif sites. We found such cases in the KCNQ proteins. Biologically, KCNQ proteins are potassium channel proteins which form pores on the cellular membrane. They consist of three parts, the N-terminal extracellular domain, the trans-membrane domain and the C-terminal intracellular domain. Yus-Najera *et al* [5] showed that the intracellular part interacts with Calmodulin (an EF hand protein) using the IQ-like motif and the 1-5-10 Calmodulin binding motif. In their paper, they reported that all IQ motif instances of human KCNQ proteins. We noted that all of these instances contained the SLiM A..IQ..[FWY]R. The paper also showed that mutation to this site in KCNQ2 will abolish its interaction with Calmodulin. Interestingly, all 5 KCNQ proteins in human also missed the last [RKQ] positions in the ELM's  $..[SACLIVTM]..[ILVMFCT]Q...[RK].\{4,5\}[RKQ]$ ; KCNQ1, KCNQ4 and KCNQ5 even have an acidic residue there instead of the basic (RK) or neutral residues (Q) required by [RKQ]. We also found acidic residues in the same position for Q5PXF9\_DROME (see the alignment of different KCNQ proteins in 5.3—the matching positions of the

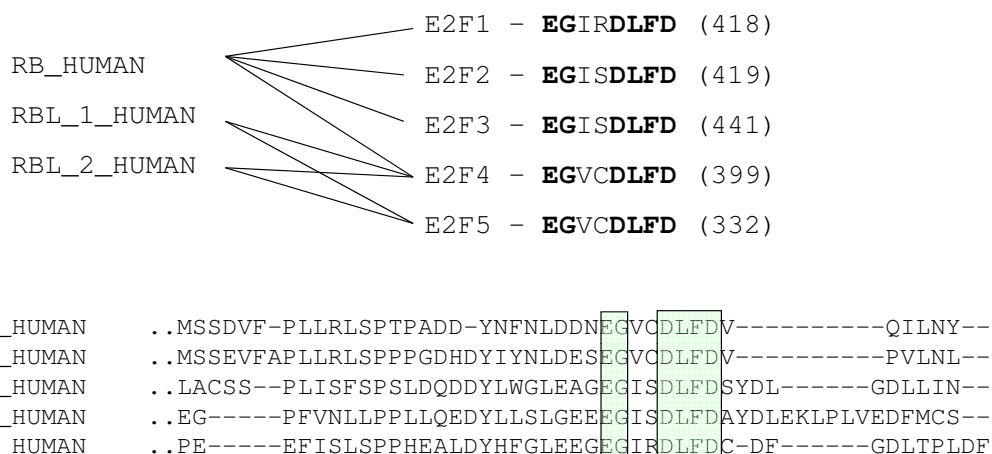


Figure 5.5: (Top) The PPI corresponding to RB\_B domain and the SLiM EG..DLFD. The instances of the SLiM (highlighted in bold) also match correctly against a known ELM SLiM ..[LIMV]..[LM][FY]D which is related to the RB\_B domain (ELM: LIG\_Rb\_pABgroove\_1). (Bottom) The sequence alignment of the C-terminal area of the target E2F proteins indicates that the SLiM region is highly conserved as compared to its neighboring positions.

[RKQ] residue were underlined). The above supporting evidences strongly suggest the existence of a variant of the IQ motif which do not strictly require the presence of [RKQ] at its end.

### SLiM for Retinoblastoma-associated protein B domain (PF01857)

The Retinoblastoma protein (Rb) is a tumor suppressor protein that belongs to a nuclear pocket protein family. Many human cancers are associated with disruptive mutations in it. It is known that E2F family transcription factors bind and recruit Rb to repress transcription from their target promoters.

D-SLIMMER reported the SLiM EG..DLFD as the best SLiM for the RB\_B domain in the HPRD dataset. The suffix part of this SLiM (LFD) is similar to the known ELM SLiM for RB\_B domain, ..[LIMV]..[LM][FY]D (ELM: LIG\_Rb\_pABgroove\_1). By aligning all the instances of EG..DLFD (see 5.5), we can see that the first wildcard position (at position 3 from the left) is occupied by aliphatic residues (I and V) as defined in the ELM SLiM (the first [LIMV] position).

Five E2F proteins containing the SLiM EG..DLFD interact with three RB\_B domain proteins in a total of 8 non-homologous interactions (see 5.5 (Top)). Furthermore, one of

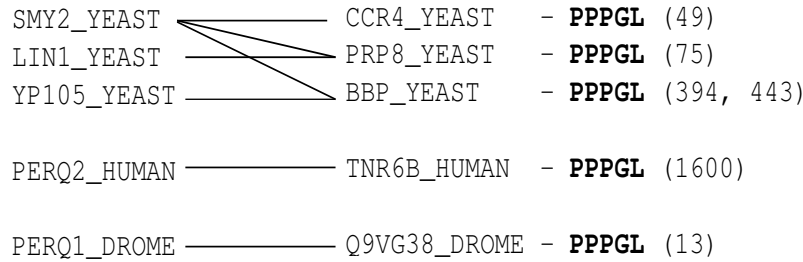


Figure 5.6: The PPI corresponding to GYF domain and the SLiM PPPGL.

our SLiM's instances, the peptide **EGISDLFD** (from E2F2), had been shown to bind RB\_B by X-ray crystallography (PDB ID: 1N4M). 5.5 (Bottom) shows the sequence alignment of the C-terminal regions of E2F proteins containing the SLiM's instances. The sequence alignment also verifies that the SLiM's instances are indeed found in a non-homologous region of the E2F proteins.

Interestingly, it has been reported that the same mode of binding is also utilized by several adenoviruses to hijack the DNA replication machinery of host cells to enable viral protein transcription [55,142]. Adenovirus E1A protein was shown to contain an instance of the SLiM which it used to block the E2F-Rb interaction. This relieves Rb-mediated repression and supports the synthesis of the viral DNA [55]. Based on the known short peptides used to bind RB\_B by different adenoviruses, we observe a consensus viral binding sequence [PM][ST]**L**[HYQ] [ED]**L**[**YF**]**D**[LV]. Notably, the consensus sequence also include an acidic residues just before the conserved [LM][FY]D in the ELM SLiM. This requirement is also reflected in our D-SLIMMER's SLiM EG..DLFD. We can therefore formulate the conjecture that this requirement is specific to E2F family's RB\_B binding site and the adenovirus has evolved to mimic this position to effectively compete with the original E2F sites.

### **Class 2 SLiM for the GYF domain (PF02213)**

GYF domains are small, versatile adaptor domains that recognize proline-rich sequences (PRS). The ELM SLiM related to this domain is [QHR].{0, 1}P[PL]PP[GS]H[RH] (ELM ID: LIG.GYF). None of the programs that we compared are able to find an exact match to LIG.GYF from both BioGRID and HPRD datasets. D-SLIMMER did found a similar SLiM, PPPGL, at rank 9 from the BioGRID dataset. We also found another similar

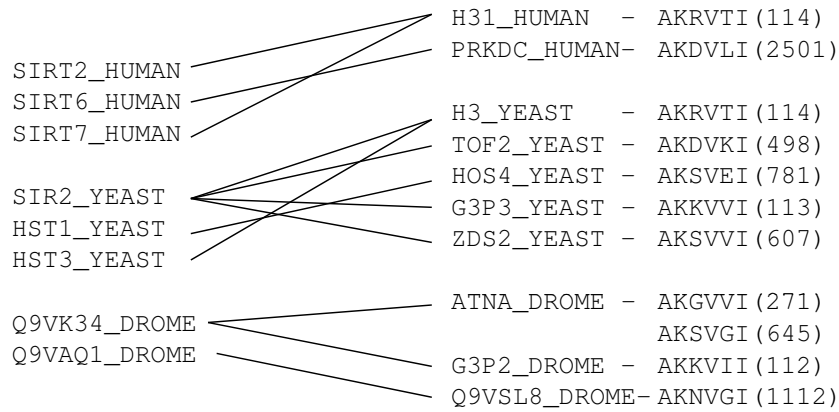


Figure 5.7: The PPI between 8 Sir2 proteins and 10 proteins containing the SLiM AK.V.I. The K is the predicted acetyllysine position. The SLiM AK.V.I fulfils the requirement of having an alphatic residue at position +2 w.r.t the acetyllysine in [6].

SLiM, R..**PPG**, which matches the prefix part of `LIG_GYF` at rank 2 from our HPRD dataset.

We note that the SLiM `PPPGL` is similar to ELM’s `LIG_GYF` SLiM except for one mismatch at the Histidine (H) position. Interestingly, this SLiM matches another recently reported SLiM of `GYF—PPG $\phi$`  motif (where  $\phi$  = hydrophobic amino acid, except for tryptophan) recently described in [56]. In the latter, the GYF domain was mentioned to have two subfamilies which recognizes the SLiM `PPPGL` (the `CD2BP2` subfamily) and `PPPGL` (the `Smy2` subfamily), respectively [56]. We observe that ELM’s `LIG_GYF` is more similar to `PPPGL` (both tryptophan(W) and histidine(H) are aromatic residues) while D-SLIMMER’s `PPPGL` is clearly an instance of `PPPGL` (which has yet to be included in the current reference SLiM databases).

The details on the PPI of the GYF domain and the `PPPGL` SLiM are depicted in 5.6. The domain-SLiM pair involves 7 protein interactions between 5 GYF proteins and 5 proteins with the SLiMs. One interacting protein pair, `SMY2_YEAST` and `BBP_YEAST` has a supporting structural data (PDB ID: 3FMA).

### Candidate novel SLiM for the Sir2 domain (PF02146)

The Sir2 proteins, or Sirtuins, is a family of protein deacetylases which removes an acetyl from an acetyllysine residue. These important proteins are involved in repression of gene transcription in the telomeres, DNA repair, cell cycle progression, chromosomal

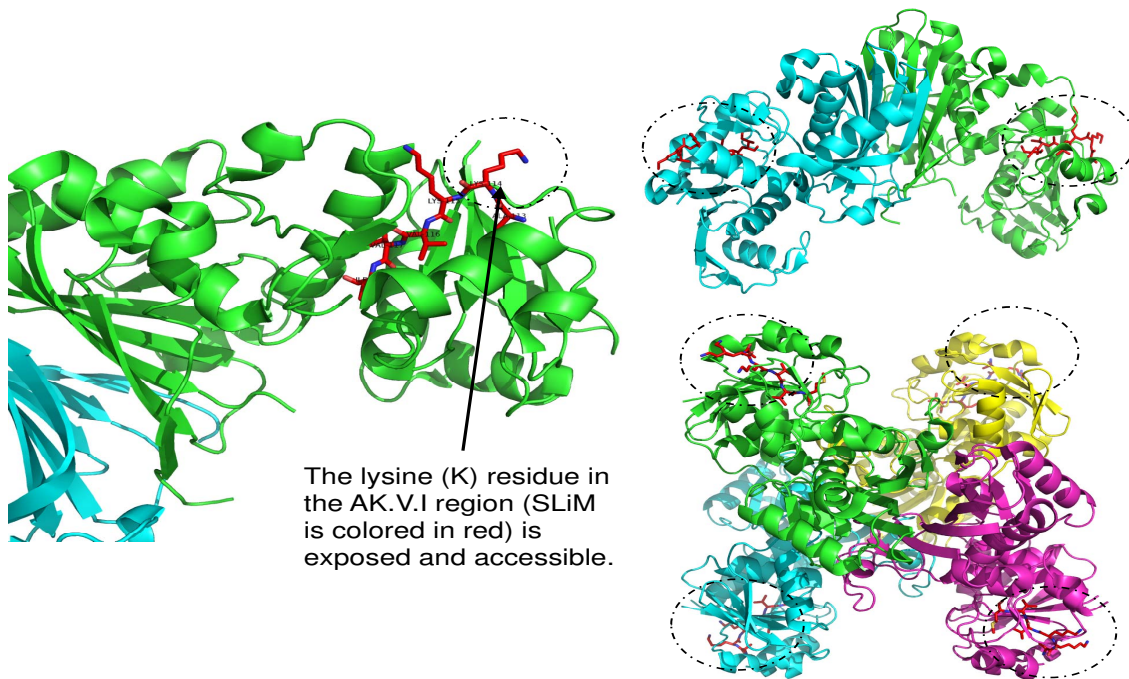


Figure 5.8: Location of the SLiM AK.V.I in Glyceraldehyde-3-phosphate dehydrogenase proteins. The left picture shows that the predicted acetyllysine position is pointing outward of the protein (PDB ID:2I5P). On the right, we show that both the dimeric (PDB ID:2I5P) and tetrameric complexes (PDB ID:2VYN) present the SLiM region (circled) at their outer peripheries. The figures are generated by PyMOL [2].

stability and cell aging [57].

Our D-SLIMMER discovered a novel SLiM AK.V.I that is associated with the Sir2 domain in our BIOGRID-ALL dataset (at rank 20). The K residue is assumed to be the acetyllysine position. The motif is found in a domain-SLiM pair with 8 Sir2 and 10 partner proteins (see 5.7). Out of a total of 12 interactions, one interaction between Yeast Sir2 and Yeast H3K115ac has been confirmed in-vitro [6]. By homology, we infer that the interaction between Human Sirt2 and Human Histone 3.1 is also based on the same mechanism. We also found that AK.V.I includes all important positions for (one type of) Sir2 recognition, namely the acetyllysine position (occupied by residue K), the -1 (A) and +2 (V) positions [6]. Alanine (A) at the -1 position has been found in several bonafide Sir2 targets such as H4K77ac (peptide sequence: **AK**RRKTV) and H4K16ac (**AK**RHRK. The acetyllysine residues are in bold) [6]. Position +2 is involved in hydrophobic interactions with phenylalanine 162 and valine 193 of the Sir2tm protein [6], and our discovered SLiM correctly required an aliphatic residue valine (V)

```

sp|P00359|G3P3_YEAST      ...DGKKIA--TYQERD-PANLPWGSNNVDIAIDSTGVFKELDTAQKHIDAGAKKVVIIITAPSS...
sp|P07487|G3P2_DROME     ...NGQKIT--VFSERD-PANINWASAGAELYVESTGVFTTIDKASTHLKGGAKKVVIIISAPSA...

1 UniRef50_P15115      ...NGKEII--VKAERN-PENLAWGEIGVDIVVESTGRFTKREDAAKHLEAGAKKVVIIISAPAK...
2 UniRef50_P00358     ...DGHKIA--TFQERD-PANLPWASLNIDIAIDSTGVFKELDTAQKHIDAGAKKVVIIITAPSS...
3 UniRef50_P04406     ...NGNPIT--IFQERD-PSKIKWGDAGAELYVESTGVFTTMEKAGAHLQGGAKRVIIISAPSA...
4 UniRef50_O83816     ...GGHRICKVCGRGLK-PSQLPWKDLGIEVVIEATGIYAN-ESSYGHLEAGAKKVVIIISAPAK...
5 UniRef50_Q9Z518     ...DGKTIK--VLSERN-PADIPWGEIGVDIVIESTGIFTKKADAEKHIAGGAKKVVIIISAPAK...
6 UniRef50_P46713     ...GSEKIK--ALAVREGPAALPWAFGVDVVVESTGLFTNAAKAKGHLEAGAKKVVIIISAPAT...
7 UniRef50_Q9ZKT0     ...GSLEIP--VFNSIK-----DLKGVGVIIECSGKFLPEKPTLENYLLLAGAKKVVIIISAPFM...
8 UniRef50_Q31EG6     ...-----MIEATGKFRTRRESLQAYLDQGVKQVIVIAAPMK...
9 UniRef50_Q8CNY0     ...NGHEIK--LLSDRN-PENLPWNEMDIDVVEATGKFNHGDKAVAHINAGAKKVVIIITGPSK...
10 UniRef50_P29272    ...GRGPIK--VTAIRN-PAELPWA--GVDMAECTGIFTTKEKAAHNLQNGAKKVVIIISAPCD...
11 UniRef50_Q6L125    ...KGTLND-----LMSSDIIVDATPEGMGMENIKIYKKKRVKAIIFQGGEEKS...
12 UniRef50_A1RV79    ...AGTIED-----LIKASDIIIDASPEDVGRENKEKYQRYDKPVIFQGGEE...
13 UniRef50_A4WIW2    ...AGTIED-----LIKASDVIIDASPEDVGAENKEKYKSFDPKPVIFQGGEE...
14 UniRef50_A6UUN9    ...QGNIFD-----IIEEADIVVDCAPGGIGKDNINENIYKKYKKAIVQGGEK...
15 UniRef50_A3CYG8    ...AGDVEA-----MLKAADIVVDATPGGVGEKNRPIYEKLGKKAIFQGGEDH...
16 UniRef50_A7IB57    ...AGSVED-----MCKAADIVVDATPGDIGVTNKPLYEKLGLKALWQGGEDH...
17 UniRef50_Q48335    ...DGTDFEAGIFHETD-PTQLPWDDLVDVAFEATGIFRTKEDASQHLDAGAKKVVIIISAPFK...
18 UniRef50_Q10SA3    ...-----MATHAALAASRIPATARLH...
19 UniRef50_P25857    ...DGKLIK--VVSNRD-PLKLPWAEGLGIDIVIEGTGVFVDGPGAGKHIQAGASKVIIITAPAK...
20 UniRef50_O14556    ...DNHEIS--VYQCKE-PKQIPWRVAVGSPYVVESTGVYLSIQAAADHISAGAKRVIIISAPSP...
21 UniRef50_Q64467    ...DNLEIN--TYQCKD-PKEIPWSSIGNPNVVECTGVYLSIEAASAHISSGARRVVVTAPSP...
22 UniRef50_A4AD74    ...GKKRIR--VLSERD-PSRLPWKALNVDVVECTGVFTARDKAQAHLAAGARKVIVSAPSA...
23 UniRef50_P34918    ...DSTPLS---FSEYGKPEVDVPWEDFVLDVLECSGKFRTPATLDPYFKRGVQKVIIVAAAPVK...
24 UniRef50_A8UN04    ...-----KEGAHHFLLERFKN...
25 UniRef50_Q4D9M5    ...-----MTGQPRD...
26 UniRef50_Q4D3Y9    ...-----
27 UniRef50_B1WNQ3    ...-----SEGID-----
28 UniRef50_B1L717    ...KGFLED--FLEGIDFLMEYDPNELSIKLTFFEGTGIQLSPKD-----

```

Figure 5.9: The conservation of AK.V.I instances in Glyceraldehyde-3-phosphate dehydrogenase (GPDH) proteins from the UniREF50 database [7]. The sequences are at most 50% similar to one another. Our predicted SLiM is conserved in 11 out of 28 GPDH reference proteins and they are all aligned to the AK.V.I instances in the GPDH proteins found by D-SLIMMER (UniProt ID:P07487 and P00359). 5 GPDH proteins have the exact AK.V.I SLiM while another 6 have an approximate match to the SLiM. For approximate matching, position -1’s Alanine (A) can be replaced by a similarly small Valine (V) residue. Position +2’s Valine (V) can be replaced by other aliphatic residues like Leucine (L) and Isoleucine (I). We also allow the same replacement for the position +4’s Isoleucine (I). The protein alignment is generated by MUSCLE [8].

in this position. Interestingly, two other methods using structure based SLiM mining also reported a SLiM for Sir2 domain [143, 144]. Stein and Aloy reported the pattern HKKLM [143] while SLiMDiet predicted the SLiM [HK][ASRK][MK][STRL][VM] [144]. Both reported SLiMs have an aliphatic residue on position +2 with respect to all possible acetyllysine (K) positions.

We also observed that Sir2 vs. Glyceraldehyde-3-phosphate dehydrogenase (GPDH) interaction was reported on both yeast (*S. cerevisiae*) and fruit fly (*D. melanogaster*) using two different PPI detection methods—affinity-capture (Yeast) and two-hybrid (fly) [53]. However, the interaction lacks any further detailed characterizations in the literature. As such, we checked for conservation of the AK.V.I SLiM in other species’ GPDH proteins and also inferred the position of SLiM in the GPDH proteins using existing GPDH 3D structures in PDB (see Fig. 5.8, Fig. 5.9). We found that our proposed SLiM AK.V.I

is indeed conserved in GPDH proteins, and it is located in an accessible location within the protein. These evidence strongly supported AK.V.I as a viable SLiM for binding to the Sir2 domain and it is therefore a novel discovery that is worth further experimental investigations.

## 5.4 Conclusions

In this work, we have adopted a novel domain-SLiM (i.e. non-linear and linear motif pair) interaction density approach instead of the conventional SLiM-SLiM approach for detecting SLiMs from PPI data. As a result, our D-SLIMMER was able to detect approximately twice as many reference SLiMs from the given PPI data as compared to the existing programs. We also highlighted four biologically interesting SLiMs detected by D-SLIMMER to show that D-SLIMMER can provide biologists with novel discoveries and valuable insights into the interaction mechanisms of many important protein domains.

Because of real life SLiMs' inherent degeneracy, the usefulness of our D-SLIMMER could be further improved by the inclusion of the amino acids groups, based on the residues' physico-chemical properties, into the motif model. This would enable more biologically relevant representation of the SLiMs but computational methods are needed to tackle the significantly larger search space of such model. We also observed that not all interaction contained in the PPI data are relevant for SLiM finding. Interactions within a single complexes mostly are obligate domain-domain interaction which are less likely to be domain-SLiM ones. The power graph analysis was proposed by Royer *et al* was shown to be able to differentiate between interactions within and outside protein complexes based on the topology of the PPI network [145]. We may be able to utilize their result to filter the PPI input to improve D-SLIMMER's result.

Another aspect for improvement is the grouping of the domains into different classes to detect specific classes of SLiMs that interact with the different classes of the domains. This is not straightforward because we have observed that the crucial SLiM recognition residues in a domain are often located at non-linear positions. However, possible solutions may be found along the line of the method proposed recently by Mazin *et al* [146].

## 5.5 List of publication

1) Hugo W, Ng S K, Sung W K. On Finding Domain-SLiM Interaction Motif from High-Throughput Protein-Protein Interaction data. Proceedings of RECOMB Satellite Conference on Computational Proteomics, March 11–13, 2011.



## Chapter 6

# Discovering Interaction Motifs from Protein Structural Data: SLiMDiet

### 6.1 Introduction

In the previous two chapters, we have studied the problem on mining SLiMs from high throughput PPI data. We observe that there are several inherent limitations with these approaches. First, as the SLiMs are highly degenerate and domains are mostly homologous, most of these algorithms mask out conserved domain regions (which are assumed not to have many SLiMs) to reduce false positives hits arising from the homology. Recently, it was found that such filtering would cause some true motifs to be missed [42]. Second, the motifs identified via the sequence-based approaches are not guaranteed to occur on the binding interface. Such atomic level of details can only come from high resolution three-dimensional (3D) structures [147]. Third, the algorithms are highly dependent on the accuracy of the interaction identification experiments. However, these interaction data, being dominated by high throughput PPI data, are known to be noisy [148].

The rapid increase of protein structure data in the PDB database [24] offers an excellent opportunity to detect SLiMs directly from 3D structures instead of the proteins' sequences. Some researchers have begun to exploit the structural data by using the

structures as templates to find seed binding motifs which are subsequently enriched using the available PPI data [51, 149]. They therefore suffer from the accuracy and coverage limitations of the PPI data like the previous methods. In this work, we directly find *de-novo* SLiMs on domain interfaces extracted from 3D structures of protein-protein interactions (Domain interface extraction, or Diet). The SLiMs are extracted from structurally clustered domain-SLiM interaction 3D data for all PFAM domains which have available structures in the PDB database.

Our SLiMDiet method comprises two steps: (i) *Domain interface clustering*: interaction interfaces belonging to the same domain are grouped together and classified using structural clustering; and (ii) *SLiM extraction*: interaction interfaces in each domain interface cluster are structurally aligned and the corresponding SLiM is extracted from the alignment. We reported 452 distinct SLiMs found on the domain interaction interfaces where 40 of them are known in the literature, 54 have at least one supporting *domain-short peptide structure* (a PDB structure which shows that a single short peptide instance of the SLiM is sufficient for binding the protein domain) and another 61 SLiMs are found to be over-represented in the PPI data collected from the BioGRID [53].

Our data also revealed that the common assumption that SLiMs occur outside the globular domain regions could be a cause for the lacklustre coverage of current SLiM detection methods [40, 42, 47, 48]. Among the 452 distinct SLiMs that we reported, 198 of them have been detected on domain-domain interaction interfaces (we call these *domain-domain SLiMs*). Current high throughput PPI-based SLiM detection methods are not amenable to mining these domain-domain SLiMs since they rely on a motif’s over-representation over a set of non-homologous protein sequences. It is virtually impossible to detect the over-representation of a domain-domain SLiM using sequence-based methods since the domain’s homology would overwhelm the SLiM’s much weaker similarity.

We compared SLiMDiet against SCOWLP [50] on the task to identify different binding class of three known peptide recognition domains; the SH2, SH3 and 14-3-3 domains. We show that SLiMDiet has better performance in 7 out of 10 different classes that we study. On comparison with D-MIST, SLiMDiet covered 40 known SLiMs that have been validated previously while D-MIST covered 9.

We also conducted a further study on four novel domain-domain SLiMs that we have

found. The first one is a domain-domain SLiM bound by the Tumor Necrosis Factor (TNF, PFAM domain:PF00229) domain on the BAFF proteins that have been implicated in B cell hyperplasia and development of severe autoimmune diseases [150, 151]. A previous experiment reported in the literature has showed that an instance of our predicted SLiM (a short peptide DLLVRHWV) can prevent the pathogenic condition from BAFF overexpression [152]. Another domain-domain SLiM of interest is a novel SLiM found on the dimer interfaces of the Glyceraldehyde-3-phosphate dehydrogenase enzyme which is associated with neurodegenerative disorders such as Huntington's disease, Alzheimer's disease, Parkinson's disease and Machado-Joseph disease [153, 154]. We also discovered two SLiMs that are implicated in amyloid fibril formation implicated in several debilitating human diseases such as Alzheimer's disease, prion based encephalopathies, liver cirrhosis and lung emphysema [155]. The class of domain-domain SLiMs could therefore be particularly useful for designing inhibitors to disrupt the domain-domain interactions which underlie the formation of pathogenic protein complexes.

The fine atomic details offered by structural data made them an attractive data source for discovering SLiMs that are beyond the coverage of existing sequence-based methods. SLiM detection methods designed to directly find SLiMs on 3D interaction interfaces can uncover new SLiMs that were undetected by the existing sequence-based SLiM detection algorithms, in particular, those that occur on domain-domain interaction regions. These domain-domain SLiMs could be good targets for disrupting the formation of pathogenic protein complexes mediated by domain-domain interactions. Currently, roughly half of all available PDB structures contain multi-chain protein complexes (more than 30000 structures); we can mine the interfaces of known protein domain from these large number of structural data. With the number of available protein structures continuing to grow rapidly, we can expect to discover even more biologically significant novel SLiMs in the near future.

## 6.2 Methods

### 6.2.1 SLiMDiet’s workflow

In this study, we devised a method named SLiMDiet, a de-novo Short Linear Motif discovery method by Domain Interface extraction from 3D protein structure data. SLiMDiet consists of two steps: a DIet step, followed by a SLiM step. The DIet step takes a set of protein structures from PDB as input, finds all known domains within the input structures and extracts the domain interfaces associated with each of them. A domain interface comprises two sets of amino acid residues: one found along a domain chain (the set is called *the domain face*) while the other on a partner chain (*partner face*), that are in close vicinity of each other. The interaction interfaces of each domain are then clustered based on structural similarity. The resulting domain interface clusters represent various modes of interactions for the domain. In the SLiM step, we conduct an approximate structural multiple alignment to align the domain faces and the partner faces in each cluster. We then check if the alignment of the partner faces contains any conserved linear region (called a 'block') of length three to twelve residues. To ensure robustness, we require that a block is constructed only from non-homologous partner chains and we require at least four of them. Finally, we construct a (linear) Gapped PSSM from the block to represent the predicted SLiMs. An illustration of SLiMDiet algorithm can be seen in Fig. 6.1.

### 6.2.2 Domain identification

A structural dataset was downloaded from the Protein Data Bank (PDB) on Aug 24th, 2009, containing 57559 structures. We chose structures containing at least one protein chain and whose resolution is 3.0 Å or better, giving a total of 54981 legible structures with 130488 protein chains. PFAM domain annotations on each PDB chain are computed by running the *hmmpfam* program from the HMMER library version 2.3.2 [156] using the latest PFAM 23.0 library [76].

We use PFAM [76] as our choice of protein domain definition, as opposed to the structurally defined domains like SCOP [80] and CATH [81], because of the relatively better coverage of PFAM. PFAM was previously reported to have 57% coverage on SWISSPROT+TREMBL sequences while SCOP covers 31% [157]. PFAM also has

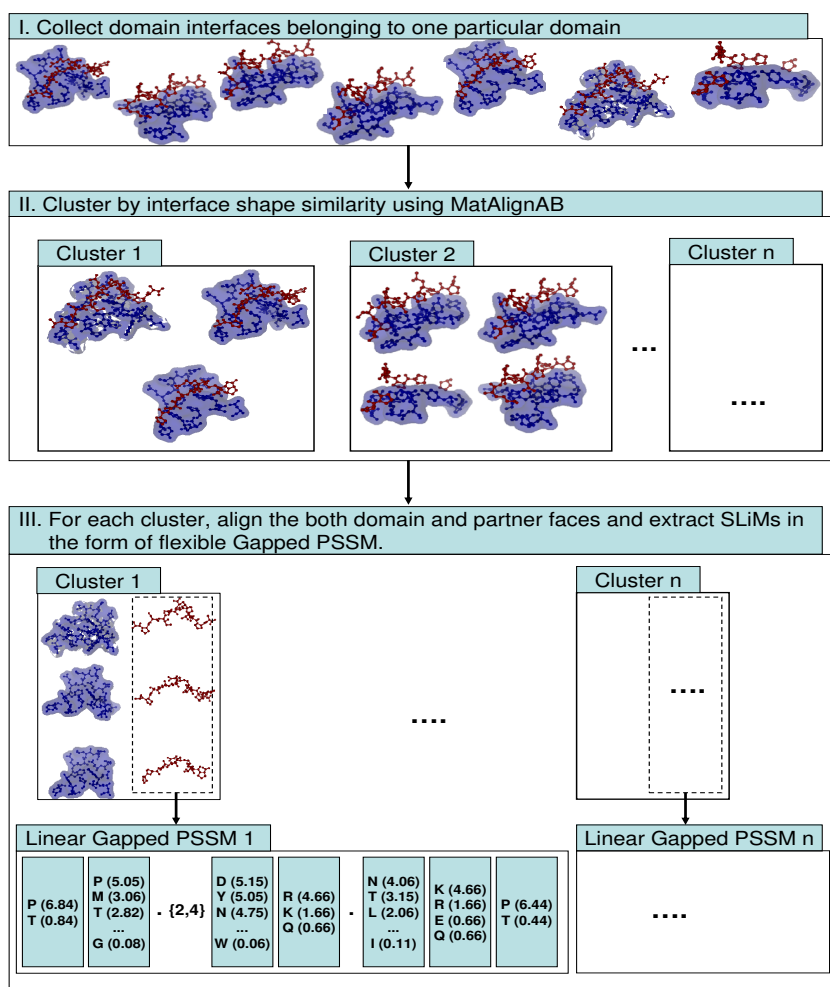


Figure 6.1: SLiMDiet's overview. The domain interfaces of each PFAM domain are clustered by their structural similarity. Next, from each cluster, the domain and partner faces are structurally aligned and we build a Gapped PSSM based on the contacts on the partner faces. The Gapped PSSM has flexible gaps defined by the minimum and maximum gaps observed between two PSSM positions. We define a Gapped PSSM as linear when the total length of its non-gap positions is three to twenty residues with gaps of at most four residues between any consecutive residue positions. To detect domain-SLiM interfaces, we collect domain interface clusters whose partner faces are covered by a linear Gapped PSSM.

higher PDB chain coverage on the current dataset (PFAM version 23.0, released July 2008, covering 112424 chains (86.16% coverage)) as compared to SCOP (version 1.75, dated June 2009, covering 87064 chains (66.72% coverage)) and CATH (version 3.2.0, dated July 2008, covering 86105 chains (65.99% coverage)). However, PFAM domain does have its own limitation. It currently does not define structural domains that are formed by multiple protein chains. Nevertheless, one can always apply SLiMDiet on

SCOP/CATH domain definition without major change on the program.

### 6.2.3 Interface extraction

For each PDB structure, we find the PFAM domains in its chains. For each domain, we computed the domain interfaces as follows. First, we define the distance between two amino acid residues to be the nearest distance between any pair of non-hydrogen atoms between the two residues. As done in PSIMAP [158], we also use a contact distance cutoff of 5Å here.

A domain interface comprises two sets of amino acid residues: the domain face and the partner face. Each amino acid on one face must be within the defined contact distance from some amino acid on the other face. The residues on each face must originate from a single protein chain (named domain and partner chain, respectively). However, they need not be located consecutively in their respective chains. For the domain face, the residues must also be within a single protein domain region of the domain chain.

To curb possible non-biological (crystal) interfaces, which are generally of smaller area, we set a threshold of having domain interfaces involving a minimum of eight amino acids on the domain face and four amino acids on the partner face. This lower bound corresponds to a binding area larger than 800 Å<sup>2</sup> – which is roughly the average size of a domain interface [49]. For intrachain domain interfaces, we also require that the residues on the partner face are not within ten residues from the ends of the domain, to avoid recognizing local contacts as interaction interfaces. This resulted in 270739 domain interfaces involving 4780 PFAM domains.

### 6.2.4 Pairwise structural alignment within each domain interface group

To classify similar interfaces that correspond to the same domain interaction class, we define the similarity of two interfaces using the modified\* S-score function from [159] as follows:

$$S_{\text{norm}} = \frac{1}{(1+\Delta)} \cdot \frac{N}{\min(|A|, |B|)}$$

---

\*The function is normalized by the size of the interface and scaled to yield similarity score between 0 to 1

where  $\Delta$  is the root mean square distance (RMSD) between the two structures being aligned,  $N$  is the number of aligned residues between the two interfaces,  $|A|$  and  $|B|$  are the sizes of the aligned interfaces respectively.

Usually, the RMSD between two proteins is approximated by the RMSD of their backbone's  $C_\alpha$  atoms. Since SLiMDiet's domain interfaces only consist of the contact residues (instead of the whole protein or domain), the  $C_\alpha$  representation is rather inadequate. To capture the similarity better, we measure the similarity of two interfaces using the backbone **and** side chain conformation of the residues on each interface. We use the  $C_\beta$  atom position to represent the direction of the side chain with respect to its backbone  $C_\alpha$  (a similar  $C_\beta$  approximation was mentioned in [160]).

When comparing two interfaces, we treat both domain and partner faces of each domain interface as one rigid continuous structure. We designed MatAlignAB for comparing domain interfaces, a modified algorithm of MatAlign [161], which only aligns residues from the same face type (i.e. residues from domain face in one interface can only be aligned to residues in the domain face of the other) and aligns atoms of the same atom type (i.e.  $C_\alpha$  ( $C_\beta$ , resp.) to  $C_\alpha$  ( $C_\beta$ , resp.)). As with the original algorithm, MatAlignAB produces alignments which follow the sequential ordering of the residues within their respective domain and partner sequences. The final results of this step consist of the similarity scores and pairwise alignments among all pairs of domain interfaces of each domain.

### 6.2.5 Hierarchical agglomerative clustering on the domain interfaces

For every domain, we cluster its interfaces into domain interface clusters by following the steps of hierarchical agglomerative clustering algorithm using average linkage, where the similarity of two clusters is defined to be the average pairwise similarity between all the members of the two clusters (as done in [49]). The algorithm starts by setting every domain interface as a cluster with one member. Next, it picks the pair of clusters which has the highest pairwise similarity and combine the pair. Then, it computes the average similarity of the combined cluster with the rest of the cluster. The latter two steps are repeated until the similarity score between every possible pair of the clusters is below a certain threshold. In SLiMDiet, we use the following range of thresholds 0.15, 0.2,

<b>P (6.84)</b> <b>T (5.14)</b> <b>V (3.20)</b> <b>L (1.46)</b> <b>M (0.84)</b> <b>R (0.66)</b> ... <b>W (-2.13)</b>	.{2,3}	<b>P (5.05)</b> <b>M (3.06)</b> <b>T (2.82)</b> <b>S (1.64)</b> <b>I (0.82)</b> <b>G (0.08)</b> ... <b>R (-3.0)</b>	<b>D (5.15)</b> <b>E (3.88)</b> <b>Y (2.05)</b> <b>N (1.75)</b> <b>Q (1.27)</b> <b>T (1.00)</b> ... <b>W (-4.06)</b>	.{1}	<b>N (6.06)</b> <b>T (5.15)</b> <b>L (5.06)</b> <b>I (2.11)</b> <b>E (0.66)</b> <b>D (0.66)</b> ... <b>F (-1.0)</b>
---	--------	--	---	------	--

Figure 6.2: An example of SLiMDiet’s gapped PSSM.

0.25, 0.3 to generate sets of (possibly overlapping) clusters each under the corresponding threshold level. For those clusters which have more than 70% overlap, we group them together and report one of the clusters as the representative.

### 6.2.6 Quantification of the clustering performance

Suppose  $C$  is a cluster of domain interfaces computed by a particular algorithm and  $R$  is the reference cluster (in our case,  $R$  is the set of domain interfaces (manually) grouped according to the literature). We use the *F-score*, which is the harmonic mean of the sensitivity and specificity scores [162], to quantify the similarity of the predicted cluster  $C$  and the reference cluster  $R$ .

$$\text{F-score}_{(C,R)} = \frac{2 \times \text{Spec}_{(C,R)} \times \text{Sens}_{(C,R)}}{(\text{Spec}_{(C,R)} + \text{Sens}_{(C,R)})}$$

where  $\text{Spec}_{(C,R)}$  is the specificity of the cluster  $C$  with respect to a reference cluster  $R$  which is computed by  $\text{Spec}_{(C,R)} = |C \cap R| / |C|$ .  $\text{Sens}_{(C,R)}$  is the sensitivity of the cluster  $C$  with respect to  $R$  which is computed by  $\text{Sens}_{(C,R)} = |C \cap R| / |R|$ .

The F-score of an algorithm for a particular reference cluster  $R$  is the best score among its computed clusters  $C$ . The F-score measure is used to compare the clustering performance of SLiMDiet to SCOWLP’s on the benchmark data.

### 6.2.7 SLiM extraction from the interface clusters

We employ a position specific scoring matrix (PSSM) with flexible gaps, called *gapped PSSM* to define the binding motif on the interaction interfaces. The gaps are defined between any two consecutive positions in the PSSM. An example of our gapped PSSM is depicted in Fig. 6.2.

Given a cluster of domain interfaces, the construction of a gapped PSSM is performed



in two steps. First, the partner faces from the interface cluster are aligned to the cluster center’s partner face. By aligning to the cluster center, which has the best average similarity to the rest of the member of the cluster, we generate an approximate multiple alignment of the partner faces.

Then, we ensure that the alignment contains 4 non-homologous faces. A face  $f_a$  is defined as homologous to  $f_b$  when (1)  $f_a$  and  $f_b$ ’s aligned residues in the alignment are exactly the same and, (2) their full protein chains share more than 50% sequence similarity. This means that two interfaces whose partner chains share high sequence similarity can still be defined as non-homologous as long as their aligned interface residues differs.

For each alignment column, we check if the column has at least 50% occupancy. Specifically, the number of non-empty residues aligned in each column must be greater or equal than half of the number of non-homologous interfaces aligned. Some alignment column has empty residues because the pairwise structural may choose not to align a residue from a particular partner face to the cluster center’s residue when these residues’ 3D positions are too different.

From the alignment of the non-homologous faces, a block is defined as a set of three to twelve consecutive alignment positions with gaps of at most four residues in between them. The SLiM corresponding to an interface alignment is computed from the longest block in it. A linear block is said to be covering a partner face  $f$  if it covers at least half of the contact residues in  $f$ . Given a set of partner faces  $F$  of size  $|F|$ , the number of partner faces  $f \in F$  covered by the block must be at least  $\lfloor \frac{|F|}{2} \rfloor$ . We only extract a SLiM from a block that satisfies the coverage constraints given above. The step-by-step construction of the multiple alignment and the linear block extraction is given in Fig. 6.3.

From a linear block satisfying the coverage constraint, we construct a Gapped PSSM for the SLiM by extrapolating the score of all 20 amino acids against the residues observed in each alignment column based on the BLOSUM62 substitution score [163]. As our multiple alignment is derived from a limited structural data, we refrain from directly scoring a residue with its observed frequency in the alignment. Instead, we define the score of a residue  $X$  on the alignment column  $i$  by

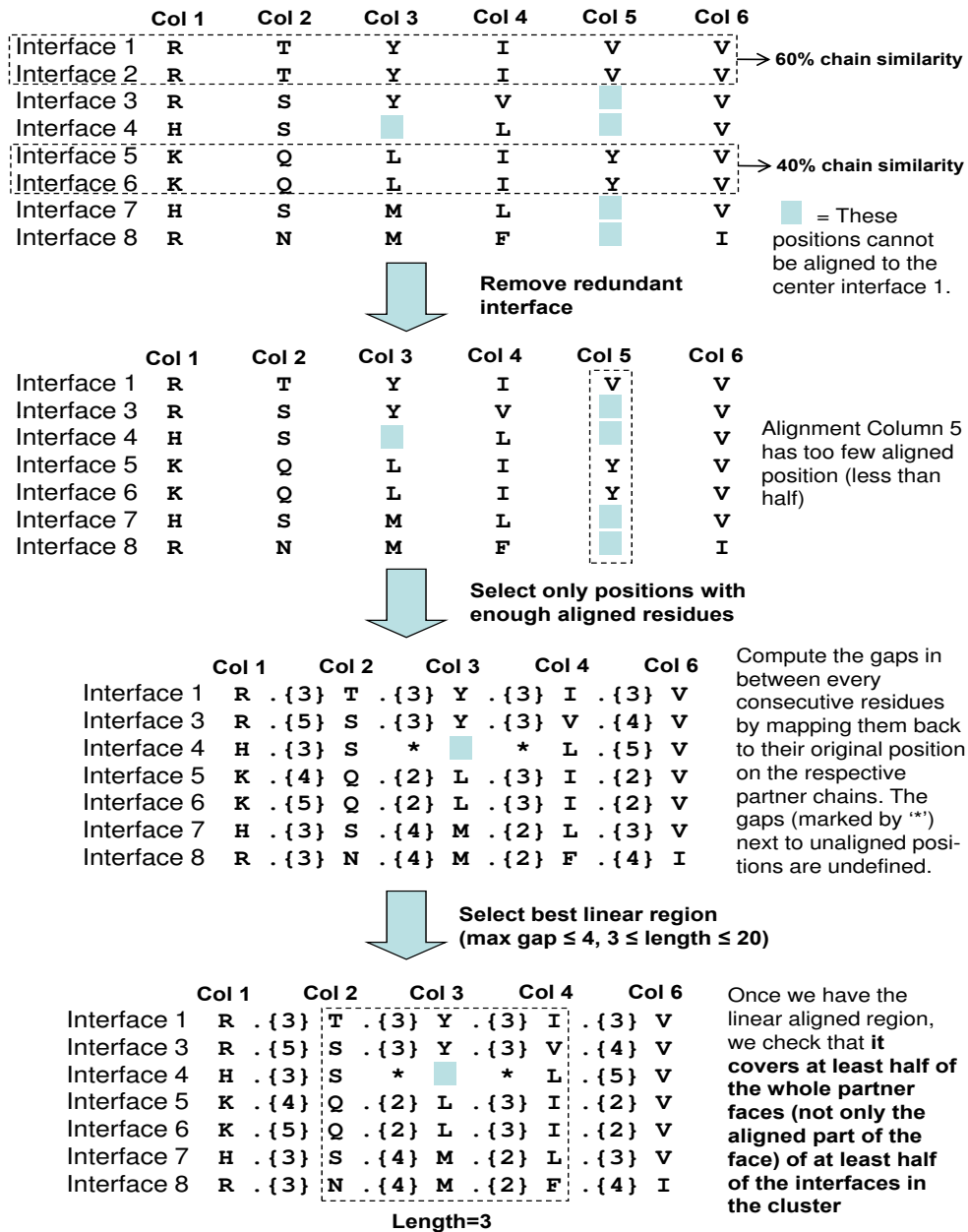


Figure 6.3: Partner face alignment steps for finding the longest linear block. The latter is where we extract the SLiM from.

$$GappedPSSM(i, X) = \ln \left( \sum_{AA \in Res(i)} freq_i(AA) \cdot e^{BLOSUM(X, AA)} \right)$$

where  $Res(i)$  is the set of amino acids seen in the column  $i$  of the alignment and  $freq_i(AA)$  is the frequency of residue  $AA$  in column  $i$ . Basically, the formula computes the weighted combination of the BLOSUM62 substitution score of any residue  $X$  against the residues seen in the column  $i$  ( $Res(i)$ )— it extrapolates the feasibility of having

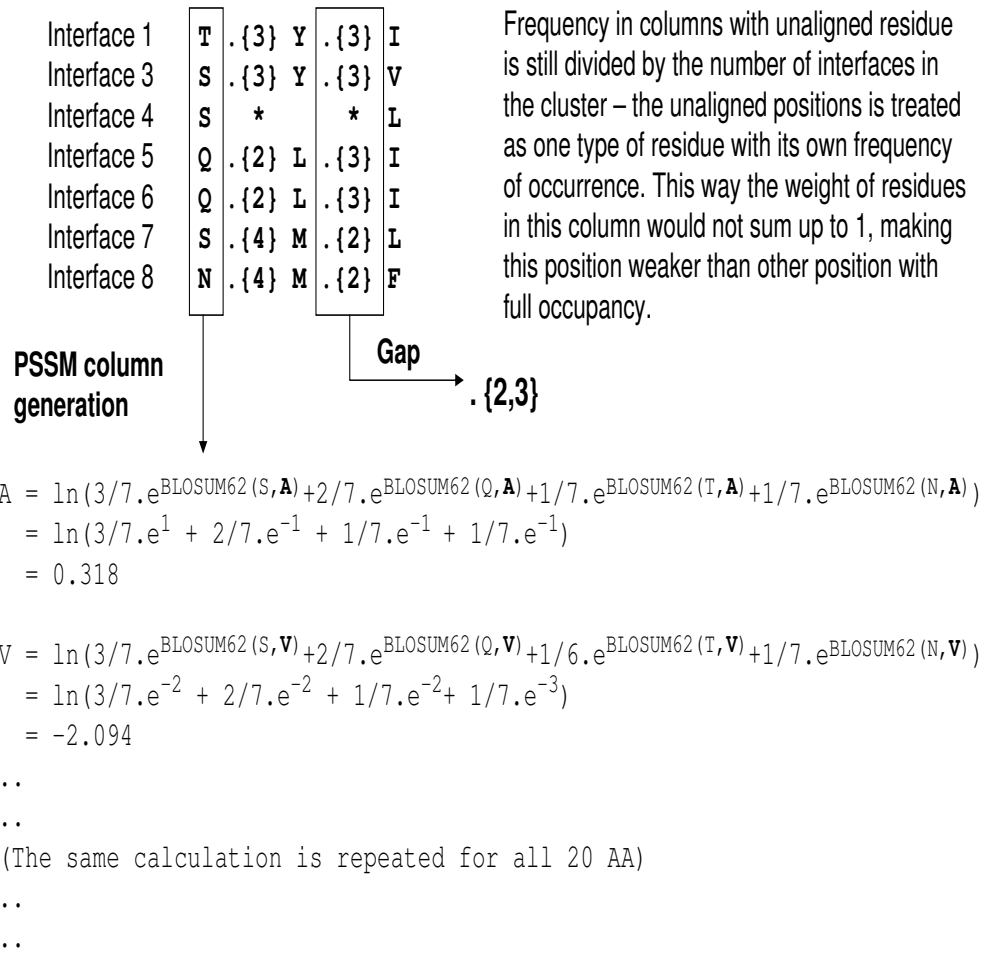


Figure 6.4: An illustration of SLiMDiet’s gapped PSSM generation from a linear block computed from the multiple interface alignment.

other residues in the alignment column based on the BLOSUM62 substitution score. The gap in between each alignment column is simply computed by taking the minimum and maximum gap observed between two residue positions. An illustration of Gapped PSSM construction can be seen in Fig. 6.4.

SLiMDiet found a total of 39170 domain interface clusters with at least four interfaces, of which, 7473 have at least four non-homologous interfaces. Out of these, only 1592 met the coverage constraint and produced valid linear blocks. We then grouped interface clusters from different similarity cutoffs when they have at least 70% member overlap. The grouping yields 452 distinct Gapped PSSMs involving 280 PFAM domains. The full listing of SLiMDiet’s predicted SLiMs and their Gapped PSSM can be downloaded from <http://www.comp.nus.edu.sg/~hugowill/SLiMDiet/SLiMListing.doc>.

### 6.2.8 Computing the statistical significance of the SLiM using PPI data

When a SLiM is extracted from a particular domain-SLiM interface clusters, we conduct statistical tests to see if the motif occurs significantly more in the interaction partners of the domain as compared to any random interaction.

Given a protein sequence  $S$ , the Gapped PSSM score of one particular position  $j$  in  $S$  is just the maximum sum of the Gapped PSSM's residue scores starting at  $j$  over all possible gap combination in the PSSM. For example, the best score of position 0 in the string  $\underline{F}$ SDTK based on the gapped PSSM<sup>†</sup>  $\begin{bmatrix} L:4.62 \\ F:1.38 \end{bmatrix} \cdot \{1, 2\} \begin{bmatrix} T:2.4 \\ D:-.12 \end{bmatrix}$  would be

$$\max \begin{cases} 1.38 + (-0.12) & (gap=1), \\ 1.38 + 2.4 & (gap=2) \end{cases}$$

We define a position  $j$  in a protein with a gapped PSSM score  $s$  as an *occurrence* of the PSSM if the probability of scoring  $s$  or better by random in a set of random protein sequence is at most equal to  $10^{-4}$ . To this end, we created a set 10000 random protein sequences, each of length 500, whose amino acid distribution follows the one observed in our PPI data from BioGRID [53] release 2.0.58. For each gapped PSSM, we computed its scores on all protein positions in the random dataset (of approximately 5 million positions) and sorted the scores in non-increasing order. The 500<sup>th</sup> score on the sorted score list would have an empirical P-value of  $10^{-4}$  and is chosen as the cutoff score for the occurrence of the gapped PSSM.

Given a SLiM's gapped PSSM, the probability of observing a certain number of occurrences in the partners of a protein domain by random can be computed by the standard hypergeometric distribution function

$$\text{P-value}(I, I_D, I_M, I_{DM}) = \frac{\binom{|I_M|}{|I_{DM}|} \binom{(|I|-|I_M|)}{(|I_D|-|I_{DM}|)}}{\binom{|I|}{|I_D|}}$$

where  $I$  is the whole set of the high throughput PPI data,  $I_M$  is the subset of  $I$  which contain an occurrence of the motif  $M$ ,  $I_D$  is the subset of  $I$  containing the domain  $D$  and  $I_{DM}$  is the subset of  $I_D$  which contain an instance of  $M$  in the interaction partners of

---

<sup>†</sup>This is a mini-version of Gapped PSSM for exemplary purpose, the real gapped PSSM would have scores for all 20 amino acids

the domain  $D$ . To construct  $I$ , we collected a set of 181997 non homologous PPI data by combining all available interactions from all species included in the BioGRID interaction database version 2.0.58 [53] (dated Oct, 2009). We removed genetic (non-physical) interactions (as defined by BioGRID) and those derived directly from structural data (to avoid self-discovery). Non-homology is enforced by keeping only one interaction among those whose both interacting proteins are at least 70% homologous to another pair(s) of interacting protein.

We checked the correctness of our PPI dataset and hypergeometric scoring function by checking the hypergeometric p-values of SLiMs known in the literature (we call them as the literature SLiMs). To this end, we collected 34 ELM and MiniMotif (MnM) SLiMs that are also predicted by SLiMDiet. We expect that the majority of the literature SLiMs are over-represented in the PPI data. We also check if the P-values computed for our Gapped PSSM are consistent with respect to the literature SLiM's p-value. It turns out that 23 out of the 34 literature SLiMs are enriched in our PPI data (hypergeometric P-value  $\leq 0.05$ , matching is done using Regular Expression—since the literature SLiMs are defined using regular expressions). Out of this 23, 16 are enriched for their Gapped PSSM too. The detailed listing of the p-values for both literature SLiMs and our Gapped PSSM is given in Fig. 6.5.

### 6.2.9 Computing the statistical significance of domain-domain SLiM

Some of the SLiMs that are mined by SLiMDiet are domain-domain SLiMs. A domain-domain SLiM is a SLiM found in an interface cluster with at least four non-homologous partner faces which occur within some (not necessarily the same) PFAM domains. These domains are called the SLiM's *host domains*. We want to know if these domain-domain SLiMs are over-represented in the protein sequences of their host domain(s). If they do, we can reasonably expect that the domain-domain SLiMs are conserved and commonly used to mediate the domain-domain interaction.

We again use the hypergeometric P-value to compute the significance of the occurrence of the SLiM within its host domain's sequence as compared to its occurrence across a set of unrelated, non-homologous domain sequences. Formally, let the whole set of non-homologous protein domain sequences that we collected be  $S$  and the subset of  $S$

No	Domain Name	Literature SLiM	Literature SLiM p-value	SLiMDiet p-value
1	14-3-3	[RHK][STALV].[ST].[PESRDIF]	1.72E-04	9.93E-02
2	Alpha_adaptinC2	[DE][DES][DEGAS]F[SGAD][DEAP][LVIMFD]	2.89E-04	2.15E-01
3	Arm	K[KR].[KR]	1.27E-02	1.55E-04
4	BIR	A[VIT]P[FYVI]	3.10E-02	3.93E-02
5	Cyclin_N	[RK].L.{0,1}[FYLVIMP]	3.81E-12	1.00E+00
6	Dynein_light	[^P].[KR].TQT	4.50E-02	6.11E-09
7	efhand	...[SACLIVTM]..[ILVMFCT]Q.{3}[RK].{4,5}[RKQ]..	1.22E-06	7.69E-02
8	FHA	..T..[DE].	3.45E-06	1.00E+00
9	Hormone_recep	[^P]L[^P][^P]LL[^P]	1.00E+00	1.00E+00
10	Hormone_recep	L[^P].{2}[HI][^P].{2}[IAV][IL]	1.00E+00	1.00E+00
11	IRS	[IL]...NP.Y	1.00E+00	1.96E-01
12	MATH	[PSAT].[QE]E	2.76E-05	3.56E-02
13	MATH	[PA][^P][^FYWIL]S[^P]	5.59E-09	8.18E-02
14	PCNA_C	(^{0,3} Q).[^FHWHY][ILM][^P][^FHILVWYP][DHFM][FMY]..	1.36E-04	8.51E-06
15	PDZ	.[ST].[VIL]\$	4.59E-21	3.08E-04
16	PID	NP.Y	4.15E-02	2.28E-01
17	PID	[IL].NP.Y	1.00E+00	1.96E-01
18	Pkinase	[RK]..S[VI]..	2.69E-64	1.69E-03
19	Pkinase	[KR]{0,2}[KR].{0,2}[KR].{2,4}[LVM].[ILVF]	2.97E-50	2.61E-03
20	Pkinase_Tyr	IYE	1.00E+00	1.00E+00
21	Profilin	[GL]PPPPPP	8.41E-02	2.87E-02
22	SH2	Y[QDEVAIL][DENPYHI][IPVGAHS]	1.00E+00	1.00E+00
23	SH2	Y.N.	1.00E+00	1.00E+00
24	SH2	Y[IV].[VILP]	1.00E+00	1.00E+00
25	SH2	Y..M	4.49E-09	4.83E-02
26	SH2	Y..P	1.00E+00	6.08E-02
27	SH3_1	[RKY]..P..P	8.40E-40	2.00E-36
28	SH3_1	P..P.[KR]	9.34E-42	2.69E-41
29	SH3_1	P.[IV][ND]R..KP	6.96E-11	1.46E-09
30	SWIB	F...W..[LIV]	1.43E-01	1.00E+00
31	TPR_1	(.[SAPTC][KRH][LMFI]\$)(([KRH][SAPTC][NTS][LMFI]\$)	5.63E-05	5.94E-04
32	ubiquitin	[WFY]..A...S..[DE]	2.52E-02	8.34E-03
33	WW	PP.Y	1.12E-25	1.20E-12
34	WW	PPLP	2.60E-02	7.19E-10

Figure 6.5: P-value checking on the literature SLiMs and SLiMDiet's Gapped PSSM based SLiMs. The 'motif' column shows the literature's reference SLiM. We can see that 23 out of the 34 known SLiMs in ELM and MnM are enriched in our PPI data based on the hypergeometric  $p\text{-value} \leq 0.05$ . The p-values of 17 of SLiMDiet's Gapped PSSM are also  $\leq 0.05$  with 16 of them overlap with the 23 SLiMs from ELM and MnM with  $p\text{-value} \leq 0.05$ .

containing the occurrence of a SLiM  $P$  (in the form of Gapped PSSM) be  $S_P$ . Next, let the set of domain sequences of  $P$ 's host domains be  $D$ . When there are more than one such domains,  $D$  would be the union of each host domain's set of instances. Finally, let the set of host domain instances which contain the occurrence of  $P$  be  $D_P$ . For multiple host domains,  $D_P$  is also the union of the sequence instances with  $P$ 's occurrence on each host domain's instances. Then the hypergeometric P-value of having  $D_P$  occurrence in  $D$  by random is

$$\text{P-value}(S, D, S_P, D_P) = \frac{\binom{|S_P|}{|D_P|} \binom{(|S|-|S_P|)}{(|D|-|D_P|)}}{\binom{|S|}{|D|}}$$

## 6.3 Results

### 6.3.1 Both known and novel SLiMs are discovered

SLiMDiet detected 452 distinct SLiMs from the whole PDB dataset (dated August 2009). 40 of which are known in the literature. Amongst the remaining 412 candidate novel SLiMs, 54 have at least an instance of a domain-short peptide structure in their respective domain-SLiM clusters. The presence of such a domain-short peptide structure is a strong indicator that the domain is capable of binding a linear peptide defined by the predicted SLiM. Indeed, all of the literature-backed SLiMs have at least one domain-short peptide structure.

From the remaining 358 candidate novel SLiMs, we found 61 are over-represented in the interaction partners of their respective domains within the high-throughput PPI data (p-value  $\leq 0.05$ ). The detailed listing of the total of 155 validated SLiMs is given at <http://www.comp.nus.edu.sg/~hugowill/SLiMDiet/ValidatedSLiM.xls>.

It is important to note that SLiMs with poor p-value are not necessarily erroneous since the PPI data is far from complete. Indeed, as many as 145 of the remaining 297 SLiMs (those with p-value  $> 0.05$ ) have less than 10 distinct interaction data—99 of them have no PPI data support at all. This shows the limitation of SLiM detection methods that relied solely or heavily on PPI data.

### 6.3.2 SLiMs with validations from the literature

We compared our predicted SLiMs with those listed in the ELM [37] and MiniMotif database [38]. SLiMDiet reported 40 SLiMs with strong similarity with the known SLiMs in literature. Since there is a significant overlap in the entries of ELM and MiniMotif, most of our SLiMs correspond to more than one database entry in both databases. In summary, our SLiMs covered 30 out of 136 known ELM SLiMs and 72 of 524 MiniMotif SLiMs (from the publicly available MiniMotif ver. 1). The coverage is significant considering that the SLiMs are solely computed from a more limited structural data source.

As a comparison, we also checked the discovery of these literature-backed SLiMs in the profiles collected by D-MIST [51]. The HMM model by Henschel *et al* generates descriptors for the interface that binds the SLiM instead of characterizing the SLiM [149]— hence it is not directly comparable for our purpose.

D-MIST, like SLiMDiet, constructs binding profiles of different domains based on the structural data. However, it relies on the high-throughput PPI data to refine their predicted motifs. Out of the 40 literature-backed SLiMs found by SLiMDiet, we could only find the corresponding D-MIST profiles for 9 of them. For the missing 31 SLiMs, D-MIST did not have any profile related to the SLiM’s domain for 24 of them and for the remaining 7 SLiMs, D-MIST’s profiles are too divergent from the literature SLiMs. Such poor coverage could be due to the fact that D-MIST was collected from a subset of PDB (10064 structures). However, we observe that even the older, well-studied SLiMs recognized by domains like SH2(Grb2), WW, FHA, PDZ, and PID(PTB) were also missing. We present the detailed listing of matched D-MIST profiles in the supplementary file at [http://www.comp.nus.edu.sg/~hugowill/SLiMDiet/D-MIST\\_comparison.xls](http://www.comp.nus.edu.sg/~hugowill/SLiMDiet/D-MIST_comparison.xls).

## 6.4 Discussion

### 6.4.1 Different SLiM classes have different interface geometries

It has been known that some SLiM-recognizing domains can bind multiple classes of SLiMs. The SH3 domain, for example, is known to recognize two classes of SLiMs; [KRY]..P..P (SH3 class 1 SLiM) and P..P.[KR] (SH3 class 2 SLiM) [37]. We hypothesize



Table 6.1: The benchmark interfaces and their classification based on the literature reference.

Interface Class	Benchmark Size	Reference ID
SH3-class 1	32	PMID:14672668
SH3-class 2	44	PMID:14672668
SH3 P.[VI][DN]R..KP	4	PMID:12773374
SH2-(class 1A)	65	PMID:17956856
SH2-(class 1B)	9	
SH2-(class 1C)	24	
SH2-(class 2A)	16	
SH2-(class 2B)	9	
14-3-3 Class 1	6	PMID:10488331
14-3-3 Class 2	17	PMID:10488331
14-3-3 Class 3	4	PMID:16091624

that the existence of such different classes of SLiM that can bind to the same domain is due to observable differences in their corresponding domain interface geometries. In other words, one can differentiate domain-SLiM interfaces belonging to different classes of SLiMs through geometric comparison.

To verify our conjecture, we hand-curated a benchmark set of 230 domain-SLiM interfaces from three well-studied domains—SH2 (123 interfaces), SH3 (80 interfaces) and 14-3-3 (27 interfaces)—whose interaction classes are well-annotated in the literature. For example, from the reference paper [164], we know that the SH3 domain in the chain C of PDB structure 1oeb recognizes the motif P.[VI][DN]R..KP. The SH3 domain in PDB structure 1uj0 was also reported to recognize the same motif in [165]. Then we check if our program would cluster the interfaces of these two PDB structures under one cluster. The detailed listing of the benchmark interfaces can be found at <http://www.comp.nus.edu.sg/~hugowill/SLiMDiet/BenchmarkInterfaceList.pdf>.

We compare the structural clustering of SLiMDiet with an existing domain interface clustering method SCOWLP [50] on the benchmark clusters. The computed clusters of SLiMDiet and SCOWLP are selected from the clustering output of the two methods over a range of similarity cutoff. The cutoff used for SLiMDiet is 0.15, 0.2, 0.25, and 0.3

while SCOWLP uses 0.1, 0.2, 0.3, and 0.4 (as provided by SCOWLP’s authors as the standard range of cutoff values). This way, both SCOWLP and SLiMDiet have four sets of clusters originating from each method’s preferred cutoff value. For each domain-LM class in the benchmark data, the best scoring cluster from any of the four set from each method is reported where the clustering score is computed based on the F-Score function defined in Section 6.2.6. As SCOWLP might not include all interfaces in our benchmark datasets (it is based on an older release of PDB), its sensitivity is computed based on the subset benchmark interfaces that are already existent when SCOWLP was built.

Table 6.2 shows that SLiMDiet’s clustering has a better overall average specificity, sensitivity and F-score in the benchmark. On the classification of SH3, we observe that SH3’s Class 1 and Class 2 have different peptide orientations but they make use of essentially the same domain face on the SH3 domain. This causes SCOWLP to have difficulties to distinguish the two main classes (and, to a certain extent, the P.[VI][DN]R..KP class which has a Class-2 like conformation). SLiMDiet can easily recognize the difference when it aligns the peptides as well. We can also see that the SH3 class 2 and the P.[VI][DN]R..KP class are separated well, thanks to the  $C\alpha$  and  $C\beta$  structural comparison done by MatAlignAB.

On 14-3-3, SLiMDiet somehow have a less than expected performance on distinguishing class 1 from class 3 in the dataset. A further check on our benchmark interfaces reveals that some 14-3-3 class 1 interfaces only shows an incomplete peptide which only covers a portion of the known Class 1 motif (for example, the structure PDB ID: 1ywt has two interfaces with only 2 residues before the phosphoserine). These small interfaces, most probably results from a poor resolution of the crystal, become indistinguishable by SLiMDiet when compared to similarly small interfaces of 14-3-3 class 3.

The performance of SLiMDiet on SH2 is more mixed. It is better than SCOWLP on class 1A and 1C; have similar performance (less than 10 percentage point difference in their F-score) on class 1B and 2B; and is worse than SCOWLP on class 2A. These cases are mainly caused by the SH2 SLiM classification being based more on the chemical properties of the SH2 interface rather than its shape. Class 2A in particular, contains both the (hydrophobic) two-pronged and the (hydrophobic) extended confirmation that were separated in the earlier classification of SH2 [166]. On the other hand, SH2 class 1A and 1C have distinctive shapes (class 1A contains the two-pronged, polar, SH2-peptide

Table 6.2: Clustering performance comparison of SLiMDiet and SCOWLP. We collected the interfaces of the SH2, SH3 and 14-3-3 domains whose domain-SLiM interaction class is defined in their respective reference papers. The grouping from the literature constitutes the reference clusters, against which the accuracy of both SLiMDiet and SCOWLP are computed. The cases where one method outperforms the other are printed in bold.

Interaction Class	SLiMDiet			SCOWLP		
	Sens	Spec	F-Scr	Sens	Spec	F-Scr
SH3-class 1	0.97	1.00	<b>0.98</b>	0.71	0.55	0.62
SH3-class 2	0.98	0.92	<b>0.95</b>	0.88	0.54	0.67
SH3 P.[VI][DN]R..KP	1.00	1.00	<b>1.00</b>	0.25	1.00	0.40
SH2-(class 1A)	0.75	0.86	<b>0.80</b>	0.62	0.67	0.65
SH2-(class 1B)	0.67	1.00	0.80	0.75	1.00	<b>0.86</b>
SH2-(class 1C)	1.00	1.00	<b>1.00</b>	0.83	0.59	0.69
SH2-(class 2A)	0.25	1.00	0.40	0.50	1.00	<b>0.67</b>
SH2-(class 2B)	0.67	0.86	0.75	0.67	1.00	<b>0.80</b>
14-3-3 Class 1	1.00	0.50	0.67	0.50	1.00	0.67
14-3-3 Class 2	1.00	1.00	<b>1.00</b>	0.67	1.00	0.80
14-3-3 Class 3	0.50	1.00	<b>0.67</b>	1.00	0.33	0.50

binding while the peptides bound by SH2 Class 1C have a  $\beta$ -turn conformation). Because of the distinct shapes, SLiMDiet performed reasonably well on both cases.

Nevertheless, the overall higher correspondence of SLiMDiet’s structural clusters with the literature reference clusters indicates that different classes of domain-SLiM interfaces indeed are associated with different domain interface geometries. We also note that SCOWLP was not designed specifically for clustering domain-SLiM interfaces— but it was the only existing method we found to be able to cluster domain-SLiM interfaces.

#### 6.4.2 Known and Novel SLiMs are found on domain-domain interfaces

Interestingly, we observed that 198 of the total 452 predicted SLiMs are domain-domain SLiMs. We found 2 of our 198 reported domain-domain SLiMs have literature support. They are SLiMs found within SH3.1 and Ubiquitin domain. All of the instances of our predicted SLiM for Ubiquitin are found within a domain called Ubiquitin Interacting

Motif (UIM, ID: PF02809), which shows that the SLiM is genuine. We also found another 6 domain-domain SLiMs with supporting domain-short peptide structures and 35 domain-domain SLiMs with over-representation in the PPI data.

### **Domain-domain SLiMs are over-represented in their host domains**

We also checked the over representation of these 198 domain-domain SLiMs within the domain they occur in (their host domains). To this end, we listed the set of 228 host domains of the 198 SLiMs (some of the domain-domain SLiMs have multiple host domains). For each domain, we use INTERPRO version 23.1 [77] and UNIPROT sequence data version 15.10 [7] to generate the set of domain sequence instances. To save time on computation, we just sample at most 50 sequences with less than 50% homology to one another to generate each host domain's instance set. When any domain has less than 50 non-homologous sequence instances, we use all that are available. The 50% homology cutoff was also applied on the sequences across different domain's instances to ensure that the overall occurrence of our SLiM is not due to homology. Using this procedure, we generated a total of 9283 non-homologous domain instances (denoted by the set  $S$  in the equation in Section 6.2.9) for the 228 host domains. We then computed the set  $S_{match}$ ,  $D$  and  $D_{match}$  for every domain-domain SLiM and proceed to compute their P-values. 143 out of 198 domain-domain SLiMs that we found are over-represented in their respective PFAM domains (P-value  $\leq 0.05$ ). The list of these domain-domain SLiMs along with their P-values are listed at <http://www.comp.nus.edu.sg/~hugowill/SLiMDiet/DomainDomainSLiM.doc>.

### **Candidates for Novel Domain-domain SLiMs**

Finding domain-domain SLiMs is an important discovery since it is commonly believed that SLiMs occur outside the globular domain regions [37]. In fact, most of the current SLiM detection methods remove domain regions from the search space [40,42] because of this belief. The discovery of such domain-domain SLiMs also indicates that many of the apparent domain-domain interactions could be mediated by domain-SLiM interactions. Indeed, a recent study had actually found genuine occurrences of ELM SLiMs on the accessible parts of a globular domain [167].

One particularly interesting novel domain-domain SLiM found by SLiMDiet is a

SLiM that is bound by the Glyceraldehyde 3-phosphate dehydrogenase, C-terminal (Gp\_dh\_C) domain (ID: PF02800). The Gp\_dh\_C domain is the C-terminal domain of Glyceraldehyde 3-phosphate dehydrogenase (GAPDH) enzyme. The enzyme exists as a tetramer of identical chains, each containing two conserved functional domains, the Gp\_dh\_N (ID: PF00044) and Gp\_dh\_C (ID: PF02800) domain. Fig. 6.6 (A) shows the structure of half of the tetramer, comprising of two chains of GAPDH (one chain on the left and one on the right). Fig. 6.6 (B, C) illustrates only the Gp\_dh\_C domain surfaces with the linear peptide regions of Gp\_dh\_N on them.

Glyceraldehyde 3-phosphate dehydrogenase has an important role in glycolysis and gluconeogenesis, and it is also involved in the signaling mechanism for programmed cell death (apoptosis) ([153]). Several studies associated the enzyme with neurodegenerative disorders such as Huntington's disease, Alzheimer's disease, Parkinson's disease and Machado-Joseph disease ([153,154]). The SLiM computed by SLiMDiet for Gp\_dh\_C is [YH]..[KRQ][YH]D[ST] which is found within the Gp\_dh\_N domain. The predicted SLiM is found within 9 non-homologous GAPDH dimers. It was reported in an earlier study that inhibition on the formation of GAPDH tetramer protects against neuronal induced cell-death ([168]), a phenomenon frequently seen in many neurodegenerative diseases. Interestingly, the dimeric and monomeric form of the enzyme retain its glycolysis and gluconeogenesis functionality and research had shown that they have higher catalytic activity ([169]). We suggest that our domain-domain SLiM could be used as a template for designing inhibitors to disrupt the enzyme's complex formation and keep it in its monomeric form.

Another notable example of domain-domain SLiMs is a SLiM interacting with the Tumor Necrosis Factor domain (ID: PF00229) of BAFF proteins. SLiMDiet predicted that it binds a SLiM D[LHS]L[LV][RH]..[IV] on its domain partners (BaffR-Tall\_bind (ID: PF09256), BCMA-Tall\_bind (ID: PF09257), TACI-CRD2 (ID: PF09305)). BAFF protein overexpression was previously shown to result in B cell hyperplasia and development of severe autoimmune diseases ([170,171]). In fact, it has already been reported that an instance of the SLiM can confer BAFF binding and block the signaling pathway leading to the pathogenic condition from BAFF overexpression ([152]). However, there were no TNF binding SLiM for BAFF reported in the literature and SLiMDiet managed to predict one. The predicted SLiM could provide further insights for designing more

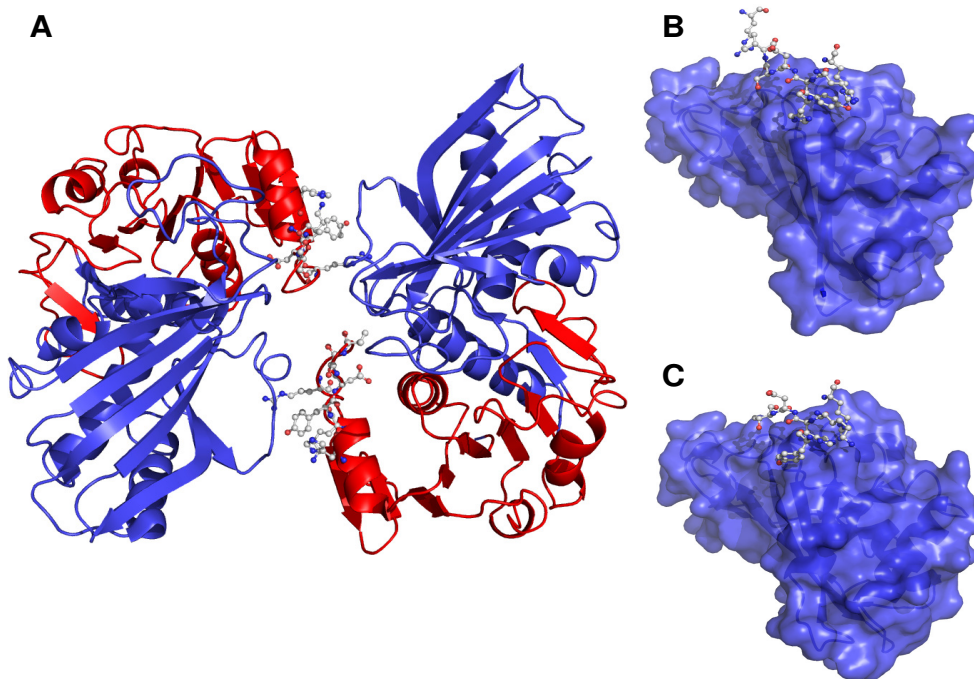


Figure 6.6: Domain-SLiM interface between Glyceraldehyde 3-phosphate dehydrogenase, C-terminal (Gp\_dh\_C, ID: PF02800) and Glyceraldehyde 3-phosphate dehydrogenase, N-terminal (Gp\_dh\_N, ID: PF00044). (A). The dimer of the Glyceraldehyde 3-phosphate dehydrogenase complex (PDB ID:1gd1). The blue part is the C-terminal domain and the red part mark the N-terminal domain. The C-terminal domain binds to a linear region on the N-terminal domain of the opposite chain (highlighted in ball-and-stick mode). SLiMDiet's predicted SLiM for this region is [YH].[KRQ][YH]D[ST] (B). The surface representation of the Gp\_dh\_C domain of Holo-glyceraldehyde-3-phosphate dehydrogenase from *Bacillus stearothermophilus* (PDB ID:1gd1). The linear region **HLLKYDSVHGR** of the opposite N-terminal domain bound to the domain is shown in ball-and-stick representation. (C). The structure of linear sequence **YQMKHDTVHGR** bound to the Gp\_dh\_C domain of *Leishmania mexicana*'s glycosomal glyceraldehyde-3-phosphate dehydrogenase (PDB ID:1a7k). This figure is generated by PyMOL [2].

effective treatments. Fig. 6.7 shows two PDB structures in which two TNF domains are binding a short peptide and a full partner domain, respectively; both containing our predicted SLiMs.

A third domain-domain SLiM is found on the dimer interface of RnaseA domains (ID: PF00074) of Ribonuclease protein. The protein is known to form dimers using two modes of domain swapping. The major mode swaps the C-terminal beta sheets

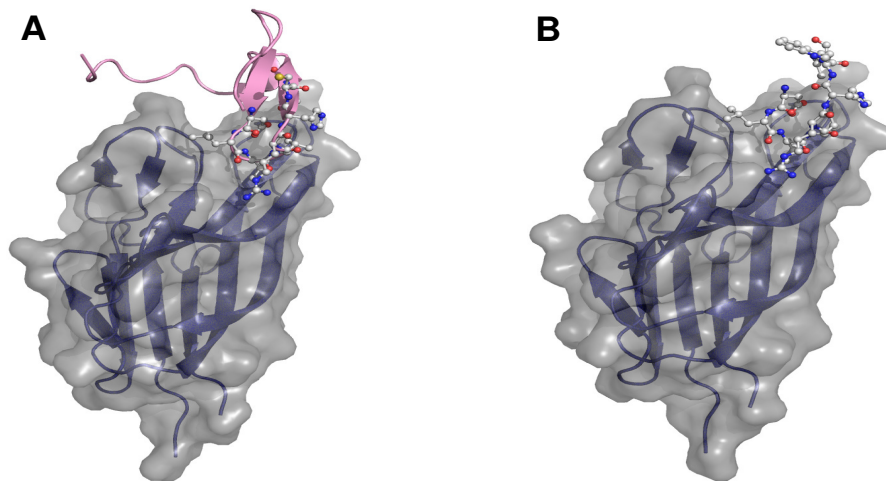


Figure 6.7: Domain-SLiM interfaces of TNF domain of BAFF proteins recognizing the SLiM D[LHS]L[LV][RH].[IV]. (A). The TNF interface from BAFF with a part of BAFF receptor protein (PDB ID:1oqe). The linear region is shown in ball-and-stick display, comprising the residues DLLVRHCV. (B). The structure between the TNF domain of BAFF complexed with only the minimal peptide DLLVRHWV (shown in ball-and-stick, PDB ID:1osg). This figure is generated by PyMOL [2].

( [172]) while the minor mode swaps the N-terminal helix ( [173]). Previous experiments have shown that a peptide instance of the N-terminal helix could compete with the minor mode of the domain swapping and disrupt dimer formation ( [173]). It has also been reported that domain swapping is one possible mechanism of amyloid fibril formation ( [155,172]) and based on the domain swapping observed in Rnase, Liu *et al.* proposed a model of amyloid fibril formation which is stabilized by the swapped domain binding ( [172]). The formation of amyloid is associated with a variety of neurodegenerative diseases such as Alzheimer’s disease, Huntington’s disease and the new variant Creutzfeldt-Jakob disease (nvCJD). It is also implicated in other diseases such as the sickle cell anemia,  $\alpha$ -antitrypsin related liver cirrhosis and emphysema ( [155]). In such a model, knowing the SLiM bound by the domain would enable one to design an inhibitor to destabilize and prevent the amyloid formation. SLiMDiet predicted two distinct novel SLiMs that correspond to the two swapping modes of the Rnase domain, namely YVPVH[FYL][DAN]AS (major mode) and AA.[FAM]ERQH.DS (minor mode).

## 6.5 Conclusion

SLiMs are important mediators of protein-protein interactions but they are difficult to detect experimentally and computationally. In this work, we showed that it is possible to systematically detect *de novo* SLiMs on domain interaction interfaces extracted directly from structural data. The atomic level of details available in the high resolution 3D structures provide a rich source of data for discovering SLiMs that are guaranteed to occur on the binding surfaces. In fact, by mining the different domain-SLiM interaction classes from the PDB database, our SLiMDiet method detected many novel SLiMs, including the domain-domain SLiMs.

The discovery of domain-domain SLiMs uncovered a limitation in the current SLiM detection approaches. These SLiMs are located in regions that are routinely masked out by the current SLiM detection methods. They cannot be detected simply by turning off the masking step—the strong similarity of the domain regions would bury the weak signal of the degenerate SLiM(s) in them. This class of SLiM is therefore currently under-represented in the known databases and literature, and they present real opportunities for domain-domain interaction inhibitor design.

Current SLiM detection methods also rely heavily on PPI data and are thus affected by its accuracy. An earlier study ([40]) has reported that some of the known SLiMs were not detected in the PPI due to noisy and incomplete interaction data. In our study, we also observed a similar problem where as many as 111 SLiMs do not have any PPI data containing their binding domains. Among them, two are known in the literature, namely the Toxin.1 ([174]) and fn1 domain ([175]) and 10 have domain-short peptide evidences.

As the structural genomic initiatives continue to make more and more high quality structural data available, we can have a viable chance of detecting the SLiMs that mediate many of our important protein-protein interactions directly from 3D structural data. As future work, we plan to continue to improve SLiMDiet’s capability by refining the notion of interface similarity to take into account the interface residues’ chemical properties and their connectivity within the domain interfaces.



## 6.6 List of publication

1) Hugo W, Song F, Aung Z, Ng S K, Sung W K. SLiM on Diet: finding short linear motifs on domain interaction interfaces in Protein Data Bank. *Bioinformatics*, 26(8): 1036–1042, 2010.

## Chapter 7

# Conclusion

This thesis had presented several contributions in the problem of finding interaction motifs from the biomolecular data. We proposed an improved algorithm to infer RNA secondary structure of an RNA sequence given a template RNA structure. The improvement in both time and space complexity is important to enable current existing programs to handle longer RNA sequences and also efficiently solve the secondary structures of more RNA sequences.

We also introduced the correlated motif concept to mine interaction motifs in protein interaction data. We specifically focused on the problem of mining short linear motifs, SLiMs, which are recently receiving considerable attention. Our programs D-STAR and D-SLIMMER have been shown to be able to mine biologically meaningful SLiMs and our comparative study indicates that D-SLIMMER gives the highest accuracy as compared to the existing SLiM mining programs.

From the protein structural data, we devised SLiMDiet to take advantage of the detailed interaction information in the protein structural data to mine SLiMs. SLiMDiet is based on a structural clustering approach on the interaction interfaces of known protein domains. We reported a list of 452 SLiMs; 155 of them are either experimentally verified or significantly enriched in known PPI data. Almost half of the reported 452 SLiMs are found on a domain-domain interaction interface. These SLiMs are virtually undetectable when mining SLiM from the sequence data because the SLiMs conservation signal would be eclipsed by the conservation of the whole domain they occur in. Hence, we propose that SLiMDiet, and SLiM mining from the structural data in general, is an important

direction to complement the current computational prediction of SLiMs.

The methods we presented are made more relevant with the rapid increase in their supporting data. The number of raw RNA sequences is expected to rise rapidly as the use of the second generation sequencing technology is getting more common nowadays. Today, whole transcriptome RNA sequencing have become quite routine and we will have a huge amount of RNA sequence data in the near future. The number of resolved RNA structures, albeit currently being a rather small part in the PDB structures (currently  $\sim 1744$  resolved RNA structures), should increase substantially in the coming years as more resources is put into the studies of yet-uncharacterized non-coding RNA.

On the protein side, both PPI and protein structural data have also increased steadily in the recent years. We also witnessed the same rapid growth trend for protein structural data fueled by the Structural Genomic initiatives. By the time of this thesis' writing, there are 64353 structures in the PDB (roughly half of them contains multiple protein chains from which we can extract domain interfaces from); a significant addition of approximately 7000 structures since SLiMDiet was written in August 2009.

## 7.1 Possible future works

For our RNA algorithms, there are two directions we plan to pursue. Firstly, we plan to apply our program to predict the secondary structure (and hence annotate) of the large number of RNA sequences produced by the new high throughput sequencing technology. Secondly, we wish to look into the possibility of combining both subclasses of the comparative approach. We note that the first subclass has a limitation on having secondary structure models that is unable to detect remote homologs. The second subclass depends on the existence of a known secondary structure. One possible way to combine both subclasses is to start with the first subclass method and come up with a secondary structure. Then, we convert the secondary structure into one or more arc-annotated sequence(s) and try identify more remote homolog of the secondary structures. This way, it is possible to do secondary structure prediction purely on sequence data.

We plan to use D-SLIMMER and SLiMDiet to generate a database of our predicted and validated SLiMs. We also plan to improve the pairwise interface comparison algorithm, currently based on the MatAlignAB algorithm, to include all non-hydrogen atoms

in the side chain. This would allow more fine-grained similarity measures between the domain-SLiM interfaces and allow SLiMDiet to produce even better clustering performance.

Last but not least, we wish to work in collaboration with the experimental biologists to confirm our SLiM predictions. We believe that computational approaches are very useful in filtering out noise in the biological data and proposing statistically significant answers to a biological problem but these may not be necessary and sufficient conditions for actual biological significance. Thus, we need to continually assess our working assumptions by validating our predictions and use the results to enhance our understanding and further improve our methods.

# Bibliography

- [1] P P Gardner et al. Rfam: updates to the RNA families database. *Nucleic Acids Res.*, 37(Database issue):D136–D140, 2009.
- [2] W L DeLano. The pyMOL molecular graphics system., 2002.
- [3] K Zhang. Computing similarity between RNA secondary structures. In *IEEE International Joint Symposia on Intelligence and Systems*, pages 126–132, 1998.
- [4] S Peri et al. Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Res.*, 13:2363–2371, 2003.
- [5] E Yus-Najera, I Santana-Castro, and A Villarroel. The identification and characterization of a noncontinuous calmodulin-binding site in noninactivating voltage-dependent KCNQ potassium channels. *J Biol Chem.*, 277(32):28545–28553, 2002.
- [6] M S Cosgrove et al. The structural basis of sirtuin substrate affinity. *Biochemistry*, 45(24):7511–7521, 2006.
- [7] The UniProt Consortium. The universal protein resource (UNIPROT). *Nucleic Acids Res.*, 36(Database issue):D190–195, 2008.
- [8] R C Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, 32(5):1792–1797, 2004.
- [9] F Crick. On protein synthesis. *Symp. Soc. Exp. Biol.*, 12:139–163, 1958.
- [10] F Crick. Central dogma of molecular biology. *Nature*, 227:561–563, 1970.

- [11] S Washietl et al. Mapping of conserved RNA secondary structures predicts thousands of functional noncoding RNAs in the human genome. *Nat Biotechnol.*, 23(11):1383–1390, 2005.
- [12] F Crick. Codonanticodon pairing: the wobble hypothesis. *J Mol Biol*, 19(2):548–555, 1966.
- [13] P Carninci et al. The transcriptional landscape of the mammalian genome. *Science*, 309(5740):1559–1563, 2005.
- [14] M Zuker and P Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acid Res.*, 9:133–148, 1981.
- [15] R B Lyngsø, M Zuker, and C N Pedersen. Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics*, 15(6):440–445, 1999.
- [16] M Zuker. MFOLD web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res.*, 31(13):3406–3415, 2003.
- [17] J S McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29(6-7):1105–1119, 1990.
- [18] R B Carey and G D Stormo. Graph-theoretic approach to RNA modeling using comparative data. In *Annual International Conference on Intelligent Systems for Molecular Biology*, pages 75–80, 1995.
- [19] J E Tabaska, R B Cary, H N Gabow, and G D Stormo. An RNA folding method capable of identifying pseudoknots and base triples. *Bioinformatics*, 14(8):691–699, 1998.
- [20] J Ruan, G D Stormo, and W Zhang. An iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics*, 20(1):58–66, 2004.
- [21] Y Sakakibara, M Brown, R Hughey, I S Mian, K Sjölander, R C Underwood, and D Haussler. Recent methods for RNA modeling using stochastic context-free grammars. In *Proc. of the Asilomar Conference on Combinatorial Pattern Matching*, 1994.

- [22] L Grate. Automatic RNA secondary structure determination with stochastic context-free grammars. In *Annual International Conference on Intelligent Systems for Molecular Biology*, pages 136–144, 1995.
- [23] B Knudsen and J Hein. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Res.*, 31(13):3423–3428, 2003.
- [24] H M Berman et al. The Protein Data Bank. *Nucleic Acids Res.*, 28:235–242, 2000.
- [25] V Bafna, S Muthukrishnan, and R Ravi. Computing similarity between RNA strings. In *Annual Symposium on Combinatorial Pattern Matching*, volume 937, pages 1–16, 1995.
- [26] S Zhang et al. Searching genomes for noncoding RNA using fastR. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(4):366–379, 2005.
- [27] S Zhang et al. A sequence-based filtering method for ncRNA identification and its application to searching for riboswitch elements. *Bioinformatics*, 22(14):557–565, 2006.
- [28] D DeBlasio et al. PMFastR: A new approach to multiple RNA structure alignment. In *Proceedings of the 9th International Conference on Algorithms in Bioinformatics*, pages 49–61, 2009.
- [29] E Fischer. Einfluss der configuration auf die wirkung der enzyme. *Berichte der deutschen chemischen Gesellschaft*, 27(3):2985–2993, 1894.
- [30] D E Koshland. Application of a theory of enzyme specificity to protein synthesis. *Proc. Natl. Acad. Sci. U S A*, 44(2):98–104, 1958.
- [31] T Pawson and J D Scott. Signaling through scaffold, anchoring, and adaptor proteins. *Science*, 278(5346):2075–2080, 1997.
- [32] M Sudol. From Src Homology domains to other signaling modules: Proposal of the protein recognition code. *Oncogene*, 17:1469–1474, 1998.
- [33] V Neduva and R B Russell. Linear motifs: evolutionary interaction switches. *FEBS Lett.*, 579(15):3342–3345, 2005.

- [34] V Neduva and R B Russell. Peptides mediating interaction networks: new leads at last. *Curr. Opin. Biotechnol.*, 17(5):465–471, 2006.
- [35] F Diella et al. Understanding eukaryotic linear motifs and their role in cell signaling and regulation. *Front Biosci.*, 13:6580–6603, 2008.
- [36] S Fox-Erlich, M R Schiller, and M R Gryk. Structural conservation of a short, functional, peptide-sequence motif. *Front. Biosci.*, 14:1143–1151, 2009.
- [37] P Puntervoll et al. ELM server: A new resource for investigating short functional sites in modular eukaryotic proteins. *Nucleic Acids Res*, 31(13):3625–3630, Jul 2003.
- [38] S Balla et al. Minimoto Miner: a tool for investigating protein function. *Nat. Methods*, 3(3):D175–177, 2006.
- [39] S Rajasekaran et al. Minimoto miner 2nd release: a database and web system for motif search. *Nucleic Acids Res.*, 37(Database issue):D185–190, 2009.
- [40] V Neduva et al. Systematic discovery of new recognition peptides mediating protein interaction networks. *PLoS Biol.*, 3(12):e405, 2005.
- [41] N E Davey et al. SLiMDisc: short, linear motif discovery, correcting for common evolutionary descent. *Nucleic Acids Res.*, 34(12):3546–3554, 2006.
- [42] R J Edwards et al. SlimFinder: a probabilistic method for identifying over-represented, convergently evolved, short linear motifs in proteins. *PLoS ONE*, 2(10):e(967), 2007.
- [43] N E Davey et al. Masking residues using context-specific evolutionary conservation significantly improves short linear motif discovery. *PLoS ONE*, 2(10):e967, 2007.
- [44] Li Haiquan, Li Jinyan, and Wong Limsoon. Discovering motif pairs at interaction sites from protein sequences on a proteome-wide scale. *Bioinformatics*, 22(8):314–324, 2006.
- [45] K Sim et al. Mining maximal quasi-bicliques: Novel algorithm and applications in the stock market and protein networks. *Statistical Analysis and Data Mining*, 2(4):255–273, 2009.



- [46] P Boyen et al. SLIDER: Mining correlated motifs in protein-protein interaction networks. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pages 716–721, 2009.
- [47] S H Tan et al. A correlated motif approach for finding short linear motifs from protein interaction networks. *BMC Bioinformatics*, 7:502, 2006.
- [48] H C Leung et al. Clustering-based approach for predicting motif pairs from protein interaction data. *J Bioinform Comput Biol.*, 7(4):701–716, 2009.
- [49] W K Kim et al. The many faces of protein-protein interactions: A compendium of interface geometry. *PLoS Comput. Biol.*, 2(9):e124, 2006.
- [50] J Teyra et al. SCOWLP classification: Structural comparison and analysis of protein binding regions. *BMC Bioinformatics*, 9:9, 2008.
- [51] D Betel et al. Structure-templated predictions of novel protein interactions from sequence information. *PLoS Comput. Biol.*, 3(9):e182, 2007.
- [52] A D J van Dijk et al. Predicting and understanding transcription factor interactions based on sequence level determinants of combinatorial control. *Bioinformatics*, 24(1):26–33, 2008.
- [53] B J Breitkreutz et al. The bioGRID Interaction Database: 2011 update. *Nucleic Acids Res.*, Epub ahead of print, 2010.
- [54] A Houdusse, J F Gaucher, E Kremtsova, M Suet, K M Trybus, and C Cohen. Crystal structure of apo-calmodulin bound to the first two iq motifs of myosin v reveals essential recognition features. *Proc Natl Acad Sci U S A.*, 103(51):19326–19331, 2006.
- [55] X Liu and R Marmorstein. Structure of the retinoblastoma protein bound to adenovirus E1A reveals the molecular basis for viral oncoprotein inactivation of a tumor suppressor. *Genes & Dev.*, 21(21):2711–2716, 2007.
- [56] M R Ash, K Faelber, D Kosslick, G I Albert, Y Roske, M Kofler, M Schuemann, E Krause, and C Freund. Conserved beta-hairpin recognition by the gyf domains of

- smy2 and gigyf2 in mrna surveillance and vesicular transport complexes. *Structure*, 18(8):944–954, 2010.
- [57] B J North and E Verdin. Sirtuins: Sir2-related NAD-dependent protein deacetylases. *Genome Biol.*, 5(5):224, 2004.
- [58] P G Higgs. Rna secondary structure: physical and computational aspects. *Quarterly Reviews of Biophysics*, 33:199–253, 2000.
- [59] S R Eddy. Non-coding RNA genes and the modern RNA world. *Nat Rev Genet.*, 2(12):919–929, 2001.
- [60] P A Sharp. RNA interference. *Genes & Dev.*, 15:485–490, 2001.
- [61] S R Eddy. Computational genomics of noncoding RNA genes. *Cell*, 109:137–140, 2002.
- [62] H Grosshans and F J Slack. Micro-RNAs: Small is plentiful. *J. Cell. Biol.*, 156:17–21, 2002.
- [63] T H Tang et al. Identification of 86 candidates for small non-messenger RNAs from the archaeon *archaeoglobus fulgidus*. *Proc. Natl. Acad. Sci.*, 99:7536–7541, 2002.
- [64] K M Wassarman. Small RNAs in bacteria: Diverse regulators of gene expression in response to environmental changes. *Cell*, 109:141–144, 2002.
- [65] K Numata et al. Identification of putative noncoding RNAs among the RIKEN mouse full-length cDNA collection. *Genome Res.*, 13(6B):1301–1306, 2003.
- [66] T Ravasi et al. Experimental validation of the regulated expression of large numbers of non-coding RNAs from the mouse genome. *Genome Res.*, 16(1):11–19, 2006.
- [67] J S Mattick. Noncoding RNAs: the architects of eukaryotic complexity. *EMBO Reports*, 11:986–991, 2001.
- [68] J S Mattick and I V Makunin. Non-coding RNA. *Hum. Mol. Genet.*, 15:R17–29, 2006.

- [69] E Szathmáry. The origin of the genetic code: amino acids as cofactors in an RNA world. *Trends Genet.*, 15(6):223–229, 1999.
- [70] Y I Wolf and E V Koonin. On the origin of the translation system and the genetic code in the RNA world by means of natural selection, exaptation, and subfunctionalization. *Biol Direct.*, 2:14, 2007.
- [71] J Shine and L Dalgarno. Determinant of cistron specificity in bacterial ribosomes. *Nature*, 254:34–38, 1975.
- [72] M Kozak. Point mutations close to the AUG initiator codon affect the efficiency of translation of rat preproinsulin in vivo. *Nature*, 308:241–246, 1984.
- [73] B A Lewis et al. PRIDB: a protein-RNA interface database. *Nucleic Acids Res.*, Epub ahead of print, 2010.
- [74] M Terribilini et al. RNABindR: a server for analyzing and predicting RNA-binding sites in proteins. *Nucleic Acids Res.*, 35(Web Server issue):W578–W584, 2007.
- [75] J G Voet. *Biochemistry*, volume 1. Wiley: Hoboken, N J, 3rd edition, 2004.
- [76] R D Finn et al. The Pfam protein families database. *Nucleic Acids Res.*, 36(Database issue):D281–288, 2008.
- [77] S Hunter et al. INTERPRO: the integrative protein signature database. *Nucleic Acids Res.*, 37(Database issue):D211–D215, 2009.
- [78] N Hulo et al. The PROSITE database. *Nucleic Acids Res.*, 34(Database issue):D227–D230, 2006.
- [79] F Corpet et al. ProDom and proDom-CG: tools for protein domain analysis and whole genome comparisons. *Nucleic Acids Res.*, 28(1):267–269, 2000.
- [80] A P Andreeva et al. Data growth and its impact on the SCOP database: new developments. *Nucleic Acids Res.*, 36(Database issue):D419–D425, 2008.
- [81] A L Cuff et al. The CATH classification revisited—architectures reviewed and new ways to characterize structural divergence in superfamilies. *Nucleic Acids Res.*, 37(Database issue):D310–D314, 2009.

- [82] S Jones and J M Thornton. Principles of protein-protein interactions. *Proceedings of the National Academy of Sciences U S A*, 93(1):13–20, 1996.
- [83] Y Ofran and B Rost. Analysing six types of protein-protein interfaces. *J Mol Biol*, 325(2):377–387, 2003.
- [84] P M Kim et al. Relating three-dimensional structures to protein networks provides evolutionary insights. *Science*, 314(5807):1938–1941, 2006.
- [85] Z Itzhaki et al. Evolutionary conservation of domain-domain interactions. *Genome Biol.*, 7(12):R125, 2006.
- [86] E Sprinzak and H Margalit. Correlated sequence-signatures as markers of protein-protein interaction. *J. Mol. Biol.*, 311(4):681–692, 2001.
- [87] X L Li, S H Tan, and S K Ng. Improving domain-based protein interaction prediction using biologically-significant negative dataset. *Int. J. Data Min. Bioinform.*, 1:138–149, 2006.
- [88] I Kim, Y Liu, and H Zhao. Bayesian methods for predicting interacting protein pairs using domain information. *Biometrics*, 63:824–833, 2007.
- [89] E Sprinzak, Y Altuvia, and Margalit H. Characterization and prediction of protein-protein interactions within and between complexes. *Proc. Natl. Acad. Sci. U S A*, 103(40):14718–14723, 2006.
- [90] S K Ng et al. InterDom: a database of putative interacting protein domains for validating predicted protein interactions and complexes. *Nucleic Acids Res.*, 31:251–254, 2003.
- [91] R D Finn, M Marshall, and A Bateman. ipfam: visualization of protein-protein interactions in PDB at domain and amino acid resolutions. *Bioinformatics*, 21(3):410–412, 2005.
- [92] A Stein, A Céol, and P Aloy. 3did: identification and classification of domain-based interactions of known three-dimensional structure. *Nucleic Acids Res.*, Epub ahead of print, 2010.

- [93] T Pawson and P Nash. Assembly of cell regulatory systems through protein interaction domains. *Science*, 300:445–452, 2003.
- [94] H Hu et al. A map of ww domain family interactions. *Proteomics*, 4(3):643–655, Mar 2004.
- [95] H Goehler et al. A protein interaction network links git1, an enhancer of huntingtin aggregation, to huntington’s disease. *Mol Cell*, 15(6):853–865, Sep 2004.
- [96] M Marti et al. Targeting malaria virulence and remodeling proteins to the host erythrocyte. *Science*, 306(5703):1930–1933, Dec 2004.
- [97] N L Hiller et al. A host-targeting signal in virulence proteins reveals a secretome in malarial infection. *Science*, 306(5703):1934–1937, Dec 2004.
- [98] L T Vassilev et al. In vivo activation of the p53 pathway by small-molecule antagonists of MDM2. *Science*, 303(5659):844–848, 2004.
- [99] C Tovar et al. Small-molecule MDM2 antagonists reveal aberrant p53 signaling in cancer: implications for therapy. *Proc Natl Acad Sci U S A*, 103(6):1888–1893, 2006.
- [100] J Vagner, H Qu, and V J Hruby. Peptidomimetics, a synthetic tool of drug discovery. *Curr. Opin. Chem. Biol.*, 12:1–5, 2008.
- [101] B Aranda et al. The IntAct molecular interaction database in 2010. *Nucleic Acids Res.*, 38(Database issue):D525–D531, 2010.
- [102] D H Mathews. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol.*, 288:911–940, 1999.
- [103] R Nussinov and A B Jacobson. Fast algorithm for predicting the secondary structure of single stranded RNA. In *Proc. Natl. Acad. Sci. U S A*, volume 77(11), pages 6309–6313, 1980.
- [104] M Zuker. Prediction of RNA secondary structure by energy minimization. In *Methods in Molecular Biology*, volume 25, pages 267–94, 1994.

- [105] I L Hofacker, W Fontana, P F Stadler, S L Bonhoeffer, M Tacker, and P Schuster. Fast folding and comparison of rna secondary structures. *Monatsh. Chem.*, 125:167–188, 1994.
- [106] S Wuchty, W Fontana, I L Hofacker, and P Schuster. Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers*, 49(2):145–165, 1999.
- [107] R.R. Gutell, N. Larsen, and C.R. Woese. Lessons from an evolving rRNA: 16S and 23S rRNA structures from a comparative perspective. *Microbiological Reviews*, 58(1):10–26, 1994.
- [108] D A M Konings and R R Gutell. A comparison of thermodynamic foldings with comparatively derived structures of 16s and 16s-like rRNAs. *RNA*, 1:559–574, 1995.
- [109] S F Altschul et al. Basic local alignment search tool. *J. Mol. Biol.*, 215:403–410, 1990.
- [110] D S Hirschberg. Algorithms for the longest common subsequence problem. *J. Association of Computing Machinery*, 24(4):664–675, 1977.
- [111] P A Evans. *Algorithms and Complexity for Annotated Sequence Analysis*. PhD Thesis, University of Victoria, 1999.
- [112] J Alber, Gramm J, J Guo, and R Niedermeier. Computing the similarity of two sequences with nested arc annotations. *Theoretical Computer Science*, 312(2–3):337–358, 2004.
- [113] P A Evans. Finding common subsequences with arcs and pseudoknots. In *Annual Symposium on Combinatorial Pattern Matching*, volume 1645, pages 270–280, 1999.
- [114] J Gramm, J Guo, and R Niedermeier. Pattern matching for arc-annotated sequences. In *ACM Transactions on Algorithms*, volume 2556, pages 182–193, 2002.
- [115] T Jiang, G Lin, B Ma, and K Zhang. The longest common subsequence problem for arc-annotated sequences. *Journal of Discrete Algorithms*, 2(2):257–270, 2004.

- [116] G H Lin, Z Z Chen, T Jiang, and J Wen. The longest common subsequence problem for sequences with nested arc annotation. *Journal of Computer and System Sciences*, 65:465–480, 2002.
- [117] G H Lin, B Ma, and K Zhang. Edit distance between two RNA structures. In *Annual International Conference on Research in Computational Molecular Biology*, pages 211–200, 2001.
- [118] W Fu, W K Hon, and W K Sung. On all-substrings alignment problems. In *Annual International Computing and Combinatorics Conference*, volume 2697, pages 80–89, 2003.
- [119] A H Tong et al. A combined experimental and computational strategy to define protein interaction networks for peptide recognition modules. *Science*, 295:321–324, 2002.
- [120] G Cesareni et al. Can we infer peptide recognition specificity mediated by SH3 domains? *FEBS Lett*, 513(1):38–44, Feb 2002.
- [121] T L Bailey and C Elkan. Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *ISMB*, 2:28–36, 1994.
- [122] C E Lawrence et al. Detecting subtle sequence signals: a gibbs sampling strategy for multiple alignment. *Science*, 262(5131):208–214, Oct 1993.
- [123] I Jonassen. Efficient discovery of conserved patterns using a pattern graph. *Comput Appl Biosci*, 13(5):509–522, Oct 1997.
- [124] I Rigoutsos and A Floratos. Combinatorial pattern discovery in biological sequences: The TEIRESIAS algorithm. *Bioinformatics*, 14(1):55–67, 1998.
- [125] K I Goh et al. Classification of scale-free networks. *Proc Natl Acad Sci U S A.*, 99(20):12583–12588, 2002.
- [126] E Sprinzak et al. How reliable are experimental protein-protein interaction data? *J. Mol. Biol.*, 327(5):919–923, 2003.
- [127] D J Reiss and B Schwikowski. Predicting protein-peptide interactions via a network-based motif sampler. *Bioinformatics*, 20 Suppl 1:I274–I282, Aug 2004.

- [128] P A Pevzner and S H Sze. Combinatorial approaches to finding subtle signals in DNA sequences. In *ISMB*, pages 269–278, 2000.
- [129] J Buhler and M Tompa. Finding motifs using random projections. In *RECOMB*, pages 69–76, 2001.
- [130] G Pavesi, G Mauri, and G Pesole. An algorithm for finding signals of unknown length in dna sequences. *Bioinformatics*, 17(Suppl. 1):S207–S214, 2001.
- [131] E Eskin and P A Pevzner. Finding composite regulatory patterns in dna sequences. *Bioinformatics*, 1(1):1–9, 2002.
- [132] U Keich and P A Pevzner. Finding motifs in the twilight zone. *Bioinformatics*, 18(10):1374–1381, 2002.
- [133] A Price, S Ramabhadran, and P A Pevzner. Finding subtle motifs by branching from sample strings. *Bioinformatics*, 19(Suppl. 2):II149–II155, 2003.
- [134] M Barrios-Rodiles et al. High-throughput mapping of a dynamic signaling network in mammalian cells. *Science*, 307(5715):1621–1625, 2005.
- [135] M Deng et al. Inferring domain-domain interactions from protein-protein interactions. *Genome Res.*, 12(10):1540–1548, 2002.
- [136] S K Ng, Z Zhang, and S H Tan. Integrative approach for computationally inferring protein domain interactions. *Bioinformatics*, 19(8):923–929, 2003.
- [137] H D Wang et al. Identifying protein-protein interaction sites on a genome-wide scale. *NIPS*, pages 1465–1472, 2004.
- [138] B K Kay, M P Williamson, and M Sudol. The importance of being proline: the interaction of proline-rich motifs in signaling proteins with their cognate domains. *FASEB J.*, 14(2):231–241, 2000.
- [139] L Salwinski et al. The database of interacting proteins: 2004 update. *NAR(Database issue)*, 32:D449–451, 2004.
- [140] W Z Li and A Godzik. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22:1658–1659, 2006.



- [141] A R Rhoads and F Friedberg. Sequence motifs for calmodulin recognition. *FASEB J.*, 11(5):331–340, 1997.
- [142] X Liu and R Marmorstein. When viral oncoprotein meets tumor suppressor: A structural view. *Genes & Dev.*, 20:2332–2337, 2006.
- [143] A Stein and P Aloy. Novel peptide-mediated interactions derived from high-resolution 3-dimensional structures. *PLoS Comput Biol*, 6(5):e1000789, 2010.
- [144] W Hugo et al. SLiM on Diet: finding short linear motifs on domain interaction interfaces in Protein Data Bank. *Bioinformatics*, 26(8):1036–1042, 2010.
- [145] L Royer, M Reimann, B Andreopoulos, and M Schroeder. Unraveling protein networks with power graph analysis. *PLoS Comput Biol.*, 4(7):e1000108, 2008.
- [146] P V Mazin, M S Gelfand, A A Mironov, A B Rakhmaninova, A R Rubinov, R B Russell, and O V Kalinina. An automated stochastic approach to the identification of the protein specificity determinants and functional subfamilies. *Algorithms Mol Biol.*, 5:29, 2010.
- [147] P Aloy and R B Russell. Structural systems biology: modelling protein interactions. *Nat. Rev. Mol. Cell. Biol.*, 7:188–197, 2006.
- [148] C von Mering et al. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417:399–403, 2002.
- [149] A Henschel, C Winter, WK Kim, and M Schroeder. Using structural motif descriptors for sequence-based binding site prediction. *BMC Bioinformatics.*, 8(Supp. 4):S5, 2007.
- [150] S D Khare et al. Severe B cell hyperplasia and autoimmune disease in TALL-1 transgenic mice. *Proc. Natl. Acad. Sci. USA*, 97(7):3370–3375, 2000.
- [151] J A Gross et al. TACI and BCMA are receptors for a TNF homologue implicated in B-cell autoimmune disease. *Nature*, 404(6781):949–950, 2000.
- [152] N C Gordon et al. BAFF/BLyS receptor 3 comprises a minimal TNF receptor-like module that encodes a highly focused ligand-binding site. *Biochemistry*, 42(20):5977–5983, 2003.

- [153] M D Berry and A A Boulton. Glyceraldehyde-3-phosphate dehydrogenase and apoptosis. *J. Neurosci. Res.*, 60(2):150–154, 2000.
- [154] W Tatton, R Chalmers-Redman, and N Tatton. Neuroprotection by deprenyl and other propargylamines: glyceraldehyde-3-phosphate dehydrogenase rather than monoamine oxidase B. *J. Neural Transm.*, 110(5):509–515, 2003.
- [155] R W Carrell and B Gooptu. Conformational changes and disease-serpins, prions and Alzheimer’s. *Curr. Opin. Struct. Biol.*, 8(6):799–809, 1998.
- [156] S R Eddy. Profile hidden markov models. *Bioinformatics*, 14:755–763, 1998.
- [157] A Elofsson and E L Sonnhammer. A comparison of sequence and structure protein domain families as a basis for structural genomics. *Bioinformatics*, 15(6):480–500, 1999.
- [158] P Dafas et al. Using convex hulls to extract interaction interfaces from known structures. *Bioinformatics*, 20(10):1486–1490, 2004.
- [159] N N Alexandrov and D Fischer. Analysis of topological and nontopological structural similarities in the PDB: new examples with old structures. *Proteins*, 25(3):354–365, 1996.
- [160] J W Torrance et al. Using a library of structural templates to recognise catalytic sites and explore their evolution in homologous families. *J Mol. Biol.*, 347(3):565–581, 2005.
- [161] Z Aung and K L Tan. Matalign: Precise protein structure comparison by matrix alignment. *J. Bioinform. Comput. Biol.*, 4(6):1197–1216, 2006.
- [162] C J Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 1979.
- [163] S Henikoff and J G Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*, 89(22):10915–10919, 2005.
- [164] M Harkiolaki et al. Structural basis for SH3 domain-mediated high-affinity binding between Mona/Gads and SLP-76. *EMBO J.*, 22(11):2571–2582, 2003.

- [165] T Kaneko et al. Structural insight into modest binding of a non-PXXP ligand to the signal transducing adaptor molecule-2 Src homology 3 domain. *J Biol Chem.*, 278(48):48162–48168, 2003.
- [166] J Kuriyan and D Cowburn. Modular peptide recognition domains in eukaryotic signaling. *Annu Rev Biophys Biomol Struct.*, 26:259–288, 1997.
- [167] A Via et al. A structure filter for the eukaryotic linear motif resources. *BMC Bioinformatics*, 10:351, 2009.
- [168] Fukuhara Y et al. GAPDH knockdown rescues mesencephalic dopaminergic neurons from MPP+ induced apoptosis. *Neuroreport*, 42:2049–2052, 2001.
- [169] A P Minton and J Wilf. Effect of macromolecular crowding upon the structure and function of an enzyme: glyceraldehyde-3-phosphate dehydrogenase. *Biochemistry*, 20(17):4821–4826, 1981.
- [170] S D Khare et al. Severe B cell hyperplasia and autoimmune disease in TALL-1 transgenic mice. *Proc Natl Acad Sci U S A*, 97(7):3370–3375, 2000.
- [171] J A Gross et al. TACI and BCMA are receptors for a TNF homologue implicated in B-cell autoimmune disease. *Nature*, 404(6781):949–950, 2000.
- [172] Y Liu, G Gotte, M Libonati, and D Eisenberg. A domain-swapped RNase A dimer with implications for amyloid formation. *Nat Struct Biol.*, 8(3):989–996, 2001.
- [173] Y Liu, P J Hart, M P Schlunegger, and D Eisenberg. The crystal structure of a 3D domain-swapped dimer of RNase A at a 2.1 Å resolution. *Proc. Natl. Acad. Sci. U S A*, 95(7):3437–3442, 1998.
- [174] T Scherf et al. Three-dimensional solution structure of the complex of alpha-bungarotoxin with a library-derived peptide. *Proc Natl Acad Sci U S A*, 94(12):6059–6064, 1997.
- [175] R J Bingham et al. Crystal structures of fibronectin-binding sites from staphylococcus aureus FnBPA in complex with fibronectin domains. *Proc Natl Acad Sci U S A*, 107(34):12254–12258, 2008.