Algorithms to Identify Functional Orthologs And Functional Modules from High-Throughput Data

Dissertation zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
vorgelegt von

Jialu Hu

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

Berlin 2014

Datum des Disputation:
17.12.2014

Betreuer:
Prof. Dr. Knut Reinert, Freie Universität Berlin, Deutschland
Prof. Dr. Gunnar W. Klau, CWI Amsterdam, Niederlande

For my parents Liu Xiulan and Hu Yanqing

“The recent development of high-throughput, massively parallel technologies has provided biologists with an extensive, although still incomplete, list of these cellular parts. The emerging challenge over the next decade is to systematically assemble these components into functional molecular and cellular networks and then to use these networks to answer fundamental questions about cellular processes and how diseases derail them.”

(Pe’er and Hacohen, 2011)

Abstract

Many studies in the last decade suggest that the biological network topology supplementing the genome is another important source of biological information for understanding the fundamental principle of life processes. A typical approach aiming to gain insights from the network information is *network alignment*. It provides a promising framework to understand the organization, function and evolution of molecular networks. However, current algorithms encounter their bottlenecks in terms of scalability, speed and so forth when applied to analyze multiple networks. Hence, it is desired to develop novel, efficient strategies to cope with the rapidly growing data in this particular field.

In this thesis, we present two new network alignment algorithms, *LocalAli* and *NetCoffee*, and their applications in the analysis of biological data. Both of the two algorithms focus on the problem of multiple network alignment, but they run into different directions: *local* alignment and *global* alignment. *LocalAli* is an evolutionary-based local alignment approach that aims to identify functionally conserved modules from multiple biological networks. In this algorithm, a computational framework is firstly proposed to reconstruct the evolution history of functionally conserved modules. *NetCoffee* is a global alignment approach with a goal to detect function-oriented ortholog groups from multiple biological networks.

The two algorithms have been applied to several real-world datasets. The results show that both *Localali* and *Netcoffee* provide substantial improvements to current algorithms in terms of several criteria such as scalability, coverage and consistency. All the test datasets, binaries and source code used for this thesis are freely available at `https://code.google.com/p/localali/` and `https://code.google.com/p/netcoffee/`.

Zusammenfassung

In der letzten Dekade haben immer mehr Studien gezeigt, dass biologische Netzwerktopologien, zusätzlich zu den bisher verwendeten genomischen Daten, eine wertwolle Ressource darstellen, um die fundamentalen Prozesse und Prinzipien, die in lebenden Organismen involviert sind, zu verstehen. Um Rückschlüsse auf die Organisation, der Funktion und der Evolution solcher Netzwerke zu ziehen werden typischerweise sogenannte Netzwerkalignments berechnet, die Zusammenhänge zwischen zwei oder mehreren Netzwerken identifizieren. Durch den hohen technischen Fortschritt stehen immer mehr Netzwerke und Netzwerkinformationen zur Verfügung. Jedoch zeigt sich, dass die bisherigen Algorithmen schlecht bzw. teilweise gar nicht mit multiplen Netzwerken skalieren.

Während dieser Arbeit wurden zwei neue Alignmentalgorithmen entwickelt, die auf multiple Netzwerke angewendet werden können. Der erste Algorithmus ist *LocalAli*, welcher ein evolutionsbasierter, lokaler Alignmentalgorithmus ist, mit dessen Hilfe funktional konservierte Module zwischen multiplen Netzwerken identifiziert werden können. Dabei wurde eine neue Methode entwickelt um die Evolution von funktionalkonservierten Modulen zu rekonstruieren. Der zweite Algorithmus, namens *NetCoffee*, berechnet globale Alignments um funktionsorientierte orthologe Gruppen zu erkennen.

In der Auswertung konnte gezeigt werden, dass die beiden entwickelten Algorithmen sowohl im Sinne der Skalierbarkeit als auch der Abdeckung und der Konsistent deutlich bessere Ergebnisse liefern als die bisherigen Algorithmen. Die Testdatensätze, sowie die Anwendung und der Quellcode die innerhalb dieser Arbeit entwickelt wurden stehen unter `https://code.google.com/p/localali/` und `https://code.google.com/p/netcoffee/` zur Verfügung.

Preface

This thesis would never become possible without collaboration with other researchers. So, I must here declare my contribution and the contributions I got from others.

Chapter 3 [1] has been published on *Bioinformatics* (Hu and Reinert, 2014). Parts of the validation methods were suggested by Knut Reinert and anonymous reviewers. I implemented the whole algorithm, tested all the involved alignment tools, verified the quality of the results and wrote the paper.

Chpater 4 [2] has been published on *Bioinformatics* (Hu *et al.*, 2014). The idea of triplet extension proposed in this paper grew out of conversations with Knut Reinert. Parts of the validation methods were suggested by Knut Reinert, Gunnar Klau and anonymous reviewers. I implemented the whole algorithm, tested all the involved alignment tools and verified the quality of the results. Birte Kehr and I jointly wrote the published paper.

Sep. 5th, 2014
Jialu Hu

---

[1] Jialu Hu and Knut Reinert, LocalAli: an evolutionary-based local alignment approach to identify functionally conserved modules in multiple networks, Bioinformatics first published online October 4, 2014 doi:10.1093/bioinformatics/btu652 http://bioinformatics.oxfordjournals.org/content/early/2014/10/20/bioinformatics.btu652

[2] Jialu Hu, Birte Kehr, and Knut Reinert, NetCoffee: a fast and accurate global alignment approach to identify functionally conserved proteins in multiple networks, Bioinformatics (2014) 30 (4): 540-548 first published online December 13, 2013 doi:10.1093/bioinformatics/btt715 http://bioinformatics.oxfordjournals.org/content/30/4/540.full

Acknowledgements

A lot of thanks go to my supervisors Knut Reinert and Gunnar Klau for their many important suggestions, their overwhelming passion for scientific problems and their continuous encouragement during my PhD. I am extremely grateful to Knut Reinert for offering me this PhD position that virtually reshaped all my understanding of the west world. I would also like to thank him for giving me the freedom in my research interests, helping me when I get stuck, and encouraging me when my effort goes to a dead end. I am grateful to Gunnar Klau for telling me his valuable insights in the bioinformatics problems during my visit to his group. He is alway ready to kindly accept my invitation for critically reviewing my manuscripts no matter the manuscripts are completed or in-completed.

I am grateful to Birte Kehr for her continuous and considerate help when I firstly arrived Berlin. I would also like to thank Birte Kehr and Enrico Siragusa for lots of fruitful discussions about the research problem. I wish to thank Xiao Liang and René Rahn for agreeing on proofreading my PhD thesis and thank René for helping me write Zusammenfassung. I am grateful to all my past and present office mates Abdul Saboor, Jens Allmer, Canan Has, Xiao Liang, Alexandra Zerck for their kindness and smile which creates a comfortable and relaxed atmosphere in our office. I would like to give my special thanks to the entire algorithmic bioinformatics group in Freie Universität Berlin. They are the most creative and intelligent people I have ever seen. I can, every time, learn something from the discussion with them.

I am greatly indebted to my parents and my three brothers for their love and their continuous support of my research and all my decisions in these years. Finally, please allow me to thank Berlin. I have experienced the most wonderful and unforgettable time in this city because of its beauty and openness.

Sep. 5^{th}, 2014
Jialu Hu

Contents

I Introduction 1

1 Introduction 3
- 1.1 Background 3
- 1.2 Protein-protein interaction networks 5
  - 1.2.1 Protein-protein interactions 6
  - 1.2.2 Experimental methods 6
  - 1.2.3 Scale-free architectures and network models 7
- 1.3 Applications of PPI networks 9
  - 1.3.1 Network motif 9
  - 1.3.2 Functional modules 9
  - 1.3.3 Functional orthologs 11
  - 1.3.4 Protein function prediction 12
- 1.4 An overview of this thesis 13

2 Preliminary Materials 15
- 2.1 Network alignment 15
  - 2.1.1 Definitions and notations 15
  - 2.1.2 Previous algorithms 18
- 2.2 Related problems in graph theory 19
  - 2.2.1 Maximum matching 19
  - 2.2.2 Subgraph search 21

II Methods, Results and Discussion 23

3 An Algorithm for Multiple Local Network Alignment 25
- 3.1 Models of functional module evolution 26

- 3.1.1 Existing evolutionary models . . . 26
- 3.1.2 The evolutionary tree . . . 26
- 3.1.3 Evolutionary events and distances . . . 27
- 3.2 The *LocalAli* algorithm . . . 29
  - 3.2.1 Overview . . . 29
  - 3.2.2 Search for *d-subnets* . . . 31
  - 3.2.3 Reconstruction of ancestral functional modules . . . 33
  - 3.2.4 Scoring function . . . 34
- 3.3 Results and discussion . . . 43
  - 3.3.1 Test datasets . . . 43
  - 3.3.2 Experimental setup . . . 44
  - 3.3.3 Cross-validation . . . 45

4 An Algorithm for Multiple Global Network Alignment 67

- 4.1 The *NetCoffee* algorithm . . . 67
  - 4.1.1 Overview . . . 67
  - 4.1.2 Generating a bipartite graph library . . . 68
  - 4.1.3 Integration of two conservation measures . . . 69
  - 4.1.4 Collection of candidate edges . . . 72
  - 4.1.5 Simulated annealing . . . 73
- 4.2 Complexity analysis . . . 76
- 4.3 Results and discussion . . . 79
  - 4.3.1 Test datasets . . . 79
  - 4.3.2 Experimental setup . . . 80
  - 4.3.3 Performance comparison . . . 80

5 Conclusion and Future Work 91

Bibliography 95

Glossary 111

Index 115

Appendix A Softwares 119

- A.1 LocalAli . . . 119
  - A.1.1 Basic information . . . 119

A.1.2 How to get started 119
A.2 NetCoffee 120
A.2.1 Basic information 120
A.2.2 How to get started 121

Appendix B Curriculum Vitae **123**

Appendix C Declaration **127**

List of Figures

2.1 An example of searching for a *d-subnet* 16

3.1 Illustration of the evolutionary model 27
3.2 A plot of our evolutionary rate 28
3.3 Sketch of reconstructing an evolutionary tree 30
3.4 Sketch of the initial internal nodes 35
3.5 Illustration of the distance distribution 36
3.6 Alignment scores on the worm-fruit fly dataset 36
3.7 Alignment scores on the human-worm-fruit fly dataset 37
3.8 Alignment score on the human-worm-fruit fly-ecoli dataset 37
3.9 10-fold cross validation for protein function prediction 47
3.10 Statistical significance on human and worm 51
3.11 Statistical significance on human and fruit fly 52
3.12 Statistical significance on human and yeast 53
3.13 Statistical significance on human and ecoli 54
3.14 Statistical significance on human and fruit fly 55
3.15 Statistical significance on human and yeast 56
3.16 Statistical significance on human and ecoli 57
3.17 Statistical significance on fruit fly and yeast 58
3.18 Statistical significance on fruit fly and ecoli 59
3.19 Statistical significance on yeast and ecoli 60

4.1 The workflow of the *NetCoffee* algorithm 68
4.2 Empirical distributions of the BLASTP *e-value* and *bitscore* 69
4.3 The workflow of the triplet comparison algorithm 70
4.4 Distributions of sequence-based and topology-based scores 74
4.5 Convergence of *simulated annealing* 78

4.6 Comparison between *NetCoffee* and *IsoRank-N* 84
4.7 Comparison between *NetCoffee* and *SMETANA* 87
4.8 Percentages of functionally related *match-sets* 88

List of Tables

1.1 Public databases for various biological knowledge 4

3.1 PPI networks tested in local network alignment 44
3.2 Parameter used for all possible datasets 50
3.3 Statistics on human and worm 51
3.4 Statistics on human and fruit fly 52
3.5 Statistics on human and yeast 53
3.6 Statistics on human and ecoli 54
3.7 Statistics on worm and fruit fly 55
3.8 Statistics on worm and yeast 56
3.9 Statistics on human and ecoli 57
3.10 Statistics on fruit fly and yeast 58
3.11 Statistics on fruit fly and ecoli 59
3.12 Statistics on yeast and ecoli 60
3.13 Statistics on human, worm and fruit fly 61
3.14 Statistics on human, worm and yeast 61
3.15 Statistics on human, worm and ecoli 62
3.16 Statistics on human, fruit fly and yeast 62
3.17 Statistics on human, fruit fly and ecoli 62
3.18 Statistics on human, yeast and ecoli 63
3.19 Statistics on worm, fruit fly and yeast 63
3.20 Statistics on worm, fruit fly and ecoli 63
3.21 Statistics on worm, yeast and ecoli 64
3.22 Statistics on fruit fly, yeast and ecoli 64
3.23 Statistics on human, worm, fruit fly and yeast 64
3.24 Statistics on human, worm, fruit fly and ecoli 65

3.25 Statistics on human, worm, yeast and ecoli 65
3.26 Statistics on human, worm, yeast and ecoli 65
3.27 Statistics on worm, fruit fly, yeast and ecoli 66
3.28 Statistics on human, worm, fruit fly, yeast and ecoli 66

4.1 PPI networks tested in global network alignment 79
4.2 Performance comparison on four datasets 82
4.3 Match-sets distribution of five algorithms 83
4.4 Running time of *NetCoffee* 89

List of Algorithms

1 Maximum matching with the augmenting path 20
2 Subgraph extension algorithm 21
3 Subgraph enumeration algorithm 21
4 Algorithm of *LocalAli* 38
5 Algorithm of sampling a *k-spine* 39
6 Extend a starting node to a *k-spine* 40
7 Collection of starting nodes 40
8 Seed sampling algorithm 41
9 Search for refined seeds 41
10 Extend refined seeds to *d-subnets* 42
11 Search for optimal internal tree nodes 43
12 Algorithm of triplet comparison 73
13 Algorithm of *NetCoffee* 75
14 Algorithm of updating states 77

Part I

Introduction

Chapter 1

Introduction

1.1 Background

What is the truth about life? This question has continually attracted attentions of scientists in the fields of physiology, pharmacology, psychology, biology, and so on. To answer this question, researchers are trying to understand how the processes of an organism work in such a collaborative and dynamic way and whether there exists an underlying molecular mechanism that steers the activities of molecules within an organism. However, it is still far from explicit due to the lack of deep and fundamental principles (like Newton's laws in physics) that can explain a broad range of processes of lives in biology.

As we know, deoxyribonucleic acid (DNA) is a storage of genetic instructions which involves all kinds of biological processes of organisms in a direct or indirect way. Hence, the knowledge of DNA sequences has become indispensable for many basic biological research fields such as genetics, phylogenetics, and numerous applied fields such as transgenic technology or DNA tests in criminology. To identify and map the total genes of the human genome from both a physical and functional standpoint, researchers in universities and research centers all around the world collaborated in an international project, *Human Genome Project* (HGP), with a cost of 13 years and $3 billion. However, sequencing a general genome was still a challenge for a high cost and a slow speed.

Thanks to the advent of high-throughput technologies, obtaining sequence information is no longer an obstacle to gain insights which can be helpful to uncover the underlying mechanism of biological processes. *High-throughput technologies* massively decreases the cost and accelerates the speed of extracting genomics data, proteomics data and molecular interactions from various species. The most widely used high-throughput technologies include *next-generation sequencing* (NGS), *mass spectrometry* (MS), *microarrays*, *yeast two-hybrid* (Y2H) assays, *RNA sequencing* (RNA-seq) and *ChIP-sequencing* (ChIP-seq). As a consequence, a tremendous amount of the biological information has been generated and becomes

Tab. 1.1: Public databases for various biological knowledge.

Types	Databases	Websites
Nucleotide Sequences	GenBank	`http://www.ncbi.nlm.nih.gov/`
	DDBJ	`http://www.ddbj.nig.ac.jp`
	ENA	`http://www.ebi.ac.uk/ena/`
Protein Sequences	Uniprot	`http://www.uniprot.org/`
	PIR	`http://pir.georgetown.edu/`
	NCBI Protein	`http://www.ncbi.nlm.nih.gov/protein`
Gene Expression	ArrayExpress	`http://www.ebi.ac.uk.arrayexpress/`
	BodyMap	`http://bodymap.ims.u-tokyo.ac.jp/`
	ASDB	`http://cbcg.nersc.gov/asdb/`
Interactions	DIP	`http://dip.doe-mbi.ucla.edu/dip/Main.cgi`
	IntAct	`http://www.ebi.ac.uk/intact/`
	HPID	`http://www.hpid.org/`
Mass spectrometry	GPMdb	`http://gpmdb.thegpm.org/`
	PeptideAtlas	`http://www.peptideatlas.org/`
	PRIDE	`http://www.ebi.ac.uk/pride/archive/`

see more information at `http://www.biologie.uni-hamburg.de/molink.htm`.

available in public databases (see in Tab.1.1). Moreover, these biological data grows at an explosive rate. For instance, the rate of growth of DNA databases such as GenBank and EMA has been following an exponential trend, with a doubling time now estimated to be 9–12 months (Raicu *et al.*, 2012).

With enormous high-throughput data, we are entering the post-genomic era that focuses on understanding the functional roles of various molecular components and how these components work together to affect the biological processes. One of the most remarkable projects is the Encyclopedia of DNA Elements (ENCODE) project, launched by the National Human Genome Research Institute (NHGRI) in September 2003. It is reported that regions of transcription, transcription factor association, chromatin structure and histone modification in the human genome have been systematically mapped (The ENCODE Project Consortium, 2012). As a consequence, hundreds and thousands of biochemical functions have been assigned to their corresponding regions which are in particular outside of the well-studied protein-coding regions. Study reveals that these functional elements are physically associated with one another, as well as expressed genes, forming a regulatory network. Studying such network will be crucial for interpreting personal genome sequences and understanding basic principles of human biology and diseases (Gerstein *et al.*, 2012).

Besides functional annotations, the identification of three-dimensional (3-D) structures of proteins is another central problem in the post-genomic era. Proteins play a major role in a vast array of processes within living organisms, including DNA replication, signal transduction, catalyzing biochemical reactions, transportation of molecules and so on. The structural knowledge of proteins provides visual understanding of how a protein interacts with other molecules, which gives important hints on the protein functions. Moreover, it can also benefit the pharmaceutical research because drug molecules, by binding some target proteins, can inhibit or activate protein functions, then provide the most effective remedy of the disease. For instance, HIV-protease is a protein that makes the replication of human immunodeficiency virus (HIV) possible in an infected patient. It can be inhibited by a known ligand molecule *XK263* from *Dupont Merck* that has a perfect complementarity of the protein's shape (Rarey *et al.*, 2008). With the experimental structure elucidation via *X-ray crystallography* and *nuclear magnetic resonance* (NMR), more than 86,000 biomacromolecules are currently available online in the *Protein Data Bank* (PDB) archive (Rose *et al.*, 2013).

Despite the discovery of enormous biological knowledge, a majority of structures and molecular functions are still unclear. On the other hand, datasets become too large to interpret and use. Therefore, computational thinking of theoretical models and simulations that links the mathematical ideas and biological knowledge together emerges as a crucial strategy to deepen our understanding of the fundamental principles in biology. Hence, numerous computational tools have been developed in the applications of sequence alignment, phylogeny reconstruction, *de novo* structure prediction, protein function prediction and so on.

1.2 Protein-protein interaction networks

Complex biochemical processes that constantly produce and recycle molecules, generate and consume energies in living organisms are organized in a highly coordinated and balanced fashion. Proteins such as kinases, enzymes, signaling molecules, transporting molecules play important roles in this system. They link all biochemical processes as a whole network by interacting with each other. The advent of *high-throughput technologies* allows us to screen all *protein-protein interactions* (PPIs) of a cell in one test. It sheds light on the research of understanding the fundamental biological mechanism by unraveling the encrypted messages encoded in the structure and topology of PPI networks.

1.2.1 Protein-protein interactions

Proteins are biological macromolecules which are formed by linear chains of amino acids connected by covalent (peptide) bonds. They account for more than 50% of the dry weight of cells and play a central role at both cellular and systemic levels, but rarely act alone. *Protein-protein interactions* (PPIs) refer to intentional physical contacts established between two or more proteins as a result of biochemical events or electrostatic forces. PPIs are intrinsic to virtually all biological activities which primarily include DNA replication, transcription, translation, splicing, signal transduction, molecular transportation, intermediary metabolism, muscle contraction. *Stable interactions* involve proteins that interact for a long time, forming permanent complexes as subunits to serve as structural or functional roles. There is a large number of these multisubunit proteins, such as hemoglobin, tryptophan synthetase, aspartate transcarbamylase, core RNA polymerase. In contrast, *transient interactions* involve proteins that interact briefly in only certain cellular context related with cell types, cell cycle stages, *etc.*, as most of these interactions happen in biochemical cascades. All modifications of proteins necessarily involve such transient protein-protein interactions (Phizicky and Fields, 1995). They include the interactions of protein kinases, protein phosphatases, glycosyl transferases, proteases, *etc.*, with their substrate proteins. A comprehensive description of PPIs would contribute considerably to the functional interpretation of fully sequenced genomes.

1.2.2 Experimental methods

There are a multitude of experimental methods (Phizicky and Fields, 1995; Berggård *et al.*, 2007) for detecting protein-protein interactions. The two most widely used methods are the *yeast two-hybrid system* (Fields and Song, 1989; Ito *et al.*, 2001)(Y2H) and *affinity purification* (also called coIP or protein complex purification) coupled to *mass spectrometry* (Mann *et al.*, 2001)(MS). The extensive popularity is due to their abilities of producing large data sets of fairly consistent quality in a high-throughput fashion. For instance, a comprehensive analysis of PPIs in *Saccharomyces cerevisiae* (yeast) was carried out using two large-scale yeast two-hybrid screens that could screen nearly all of the 6,000 predicted yeast proteins (Uetz *et al.*, 2000; Ito *et al.*, 2001).

Y2H is based on the fact that many eukaryotic transcription factors (TFs), such as the yeast enhancer *Gal*4, are composed of two separate fragments, called the binding domain (BD) and activating domain (AD). The binding of these TFs onto an upstream activating sequence (UAS) results in the activation of a downstream reporter gene, such as *LacZ*, whose activity can be detected or measured quantitatively. Y2H offers a sensitive and cost-effective mean to test the direct interaction between two target proteins *in vivo*. Moreover, transient and weak interactions, which are often important in signaling cascades, are more easily detected in Y2H system since the genetic reporter gene strategy results in significant signal amplification (Estojak *et al.*, 1995). However, Y2H system suffers from false negatives and false positives (Goll and Uetz, 2008). False negatives are most likely caused by steric effects that prevent proteins from interacting because of the fused domains. False positives are not reproducible and therefore difficult to explain.

Affinity purification coupled to MS mostly starts by the purification of a tagged protein and its interacting proteins. The most widely used method to purify protein complexes is the *tandem affinity purification* (TAP). Compared with Y2H, affinity-based methods are biased towards proteins that interact with high affinity and with slow kinetic of dissociation (i.e. stable interactions), and may not be optimal for the detection of transient interactions (Aloy and Russell, 2002). Like Y2H, it also suffers from false negatives and false positives. For example, stringent washes may result in the lost of low-affinity targets.

To date, numerous experimentally determined PPIs in various species are detected and available in public databases, such as IntAct (Kerrien *et al.*, 2012), BioGRID (Chatr-aryamontri *et al.*, 2013), STRING (Franceschini *et al.*, 2013), DIP (Salwinski *et al.*, 2004), MINT (Licata *et al.*, 2012), MPPI (Pagel *et al.*, 2005), HPID (Han *et al.*, 2004). In addition, there are also some databases for known protein complexes, such as CORUM (Ruepp *et al.*, 2010) and MPACT (Güldener *et al.*, 2006).

1.2.3 Scale-free architectures and network models

A network consists of many individual vertices and their inner connections, corresponding to a mathematical structure graph. Many complex systems can be modeled in networks whose vertices are the elements of the system and whose edges represent the connections. The functional elements within a cellular system form a large gene regulatory network (Gerstein *et al.*, 2012) (GRN), in which vertices are TFs, genes, miRNAs *etc.*, edges are the biochemical events, such as bindings, activations and inhibitions. Many PPI networks have been also constructed (or reconstructed) for the completely sequenced organisms based on both experimentally determined interactions (Uetz *et al.*, 2000; Giot *et al.*, 2003) and computational methods (Saito *et al.*, 2003; Rhodes, 2005).

A series of studies reveal that a large number of complex networks including genetic regulatory networks (Featherstone and Broadie, 2002), PPI networks (Giot *et al.*, 2003; Li *et al.*, 2004), metabolic networks (Jeong *et al.*, 2000) and various social networks (Barabási and Albert, 1999) are *scale-free* architectures, in which the vertex connectivities follow a scale-free power-law distribution. More mathematically, the probability that a vertex interacts with k other vertices in the complex network follows $P(k) \sim k^{-\gamma}$, where for most cases $\gamma \in [2, 3]$.

This common topological feature is the result of two generic mechanisms hidden behind the complex systems: (i) networks expand continuously by the addition of new vertices, and (ii) new vertices attach preferentially to sites that are already well connected. Based on the two mechanisms, the Barabási-Albert model was designed to simulate the development of the power-law distribution networks. In the model, a scale-free network can be generated by continuously adding a new node with M links to the network, which connects to an already existing node I with probability $\Pi_I = k_I / \sum_J k_J$, where k_I is the degree of node I and J is the index denoting the sum over network nodes.

Two remarkable features of scale-free networks are error tolerance and attack vulnerability (Albert *et al.*, 2000). The scale-free network has only a small number of highly connected vertices that are known as hubs and a large number of sparsely connected vertices. They are surprisingly tolerant against accidental failures: even if 80% of randomly selected nodes fail, the remaining 20% still form a compact cluster with a path connecting any two nodes. the reason is that the removal of randomly selected nodes (i.e. always sparsely connected nodes) does not change the network's integrity. However, the attack of hub nodes can quickly break the scale-free network into many isolated parts. Such hub nodes in cellular networks, as well as in communication systems, might easily become attack targets of various viruses from external organisms or computers.

In PPI networks, gene duplication is most likely the major biological mechanism for generating the scale-free topology (Wagner, 2003). Duplicated genes produce identical proteins that interact with the same protein partners. However, the gene duplications are not enough to explain the power-law degree distribution. Interaction turnover is another serious force. In all, there are three factors collaborating to design the power-law distribution of networks: (i) the rate of interaction addition and deletion must be nearly balanced; (ii) interaction turnover affects preferentially highly connected proteins; (iii) some added interactions add new proteins to the network. Consequently, an evolutionary model based on the hypothesis of evolution by gene duplications and gene divergences was designed to represent the evolution of PPI networks (Vazquez *et al.*, 2003b). A similar evolutionary model (Dutkowski and Tiuryn, 2007) was proposed to reconstruct the phylogenetic history of PPI networks. Besides, another model (Pastor-Satorras *et al.*, 2003) based on gene duplications plus re-wiring of the newly created genes shows that it can reproduce networks with many topological features of their real counterparts.

1.3 Applications of PPI networks

1.3.1 Network motif

Network motifs are patterns of interactions occurring in complex networks at numbers that are significantly higher than those in randomized networks (Milo, 2002). The discovery of *network motifs* may uncover the basic building blocks of most networks.

Network motifs play key roles in processing information in *transcriptional regulatory networks* (TRNs), which control gene expression in cells (Shen-Orr *et al.*, 2002). Three major network motifs have been found in the TRNs, which include *feed forward loop* (FFL), *single input module* (SIM) and *dense overlapping regulons* (DOR). For the case of PPI networks (e.g. the yeast PPI network), it reveals that the participation of network motifs substantially influences the evolutionary conservation of the specific components (Wuchty *et al.*, 2003). There are two evidences for that: (i) orthologs are not randomly distributed in the PPI network but are the building blocks of cohesive motifs, which tend to be evolutionary conserved; (ii) large motifs tend to be conserved as a whole, each of their components having an ortholog. It indicates that network motifs may represent evolutionary conserved topological units of PPI networks. Some other studies suggest that proteins within motifs whose constituents are of the same age class tend to be densely interconnected, co-evolve and share biological functions. And these motifs tend to be within protein complexes (Liu *et al.*, 2011). Conserved network motifs have also been utilized to identify and validate interaction candidates based on the fact of the abundance of conserved network motifs in the PPI networks (Albert and Albert, 2004).

All these findings demonstrate that the concept of network motifs provides a key perspective to understand their structural design principles, protein functions and evolution in PPI networks. Many computational tools for detecting network motifs have been developed in the last decade, such as *mfinder* (Kashtan *et al.*, 2004), *FANMOD* (Wernicke, 2006) and *MAVisto* (Schreiber and Schwbbermeyer, 2005).

1.3.2 Functional modules

A *functional module* is, by definition, a discrete entity whose function is separable from those of other modules (Hartwell *et al.*, 1999). They are usually separated based on spatial localization (e.g. a ribosome) or chemical specificity (e.g. a signal transduction system) and composed of many types of molecules, such as proteins, DNA, RNA and small molecules. There are some experimental evidences that demonstrate the existence of functional modules within organisms. For example, some modules such as those for protein synthesis, DNA replication, glycolysis, and even parts of the mitotic spindle have been successfully reconstituted *in vitro*. A lack of a comprehensive chart of functional modules within organisms becomes an obstacle to understand the general design principles that govern the structure and behavior of modules, and the evolutionary constraints. To solve this problem, many computational tools aiming to identify functional modules in PPI networks have been developed in the last decade. Basically, the existing algorithms aiming to identify functional modules in PPI networks can be grouped into two classes: *clustering* and *network comparison*.

Many clustering algorithms try to detect groups of nodes that are densely connected internally but sparsely interacting with the rest of the network as putative functional modules (Bader and Hogue, 2003; Spirin and Mirny, 2003; Bu *et al.*, 2003; Newman and Girvan, 2004). Several clustering algorithms detect functional modules in PPI networks based on *markov random walk*, such as *Markov CLustering* (MCL) (Dongen, 2000), *Regularized MCL* (RMCL) (Satuluri and Parthasarathy, 2009) . Some other clustering algorithms identify functional modules by grouping proteins that have similar biological functions into a same module (Navlakha *et al.*, 2009). Moreover, several blockmodel module identification algorithms (Royer *et al.*, 2008; Wang and Qian, 2012) have been proposed based on the observation that proteins interacting with similar sets of proteins in a given network tend to have similar functions (Morrison *et al.*, 2006; Pinkert *et al.*, 2010).

Both *in silico* and *in vivo* studies suggest that functional modules are highly conserved across species (Pellegrini *et al.*, 1999; Roguev *et al.*, 2008). Proteins that interact with many other proteins, such as histones, actin and tubulin, are difficult to evolve. Proteins that function together in a function module (e.g. a *metabolic pathway* and *protein complex*) are likely to evolve in a correlated fashion. Interaction data generated in one species can be used to predict interactions in another species by searching for pairs of orthologous proteins. As a consequence, patterns of interactions that are conserved across species are biologically significant and are more likely to correspond to functional modules. Based on this test hypothesis, many computational tools have been developed based on the network comparison (i.e. *local network alignment*) to detect conserved modules in PPI networks across species, which include: *PathBlast* (Kelley *et al.*, 2004), *NetworkBlast* (Kalaev *et al.*, 2008), *MaWISH* (Koyutürk *et al.*, 2006), *Graemlin* (Flannick *et al.*, 2006), Ali's method (Ali and Deane, 2009), *PINALOG* (Phan and Sternberg, 2012) and so on. Most local network alignment algorithms firstly build an alignment graph in which each node represents a set of orthologous proteins and each edge represents a conserved interaction. Then, they carry out a search for high-scoring subnetworks over the alignment graph. In comparison with clustering algorithms, local network alignment offers limited coverage of proteins. However, it allows us to identify conserved modules across species which might improve our understanding of protein functions and the evolutionary mechanism of modules and networks.

1.3.3 Functional orthologs

After ∼3.5 billion years of evolution (Schopf and Packer, 1987), ∼8.7 million eukaryotic species that originated from simple life forms are currently living on the earth under the natural selection pressure (Mora *et al.*, 2011). Each gene in the extant species is a result of a series of evolutionary processes, such as gene conservation, speciation, duplication and deletion. Selection pressure on a specific gene could be so strong and everlasting that the gene could be present in all extant species, or it could be highly transient or specific to certain species.

Orthologs are genes/proteins derived from a single ancestral gene in the last common ancestor of the compared species (Koonin, 2005; Park *et al.*, 2011). *Paralogs* are genes/proteins related via duplication. Generally, orthologs are assumed to have the same biological function in different species, and paralogs offer new biological functions for current species. However, orthologs in different genomes may have different functions. A major reason is a large number of duplications and/or deletions along a gene's evolutionary history could indicate neofunctionalization and/or non-orthologous gene displacement which consequently results in different functions for orthologs in different genomes (Fang *et al.*, 2010). *Function-oriented ortholog* groups, also known as *functional orthologs* (FOs), contain orthologs that play functionally equivalent roles in different species and also include recent paralogs with a same biological function (i.e. *inparalogs*) (Remm *et al.*, 2001). In simple words, functional orthologs are genes/proteins that perform functionally equivalent roles in different species (Park *et al.*, 2011).

The identification of functional orthologs is a fundamental task in comparative systems biology (Tatusov *et al.*, 1997; Park *et al.*, 2011), which might benefit researchers in the fields of function annotations and phylogenetics. For example, the function of an uncharacterized protein could be predicted from other characterized proteins in the same FO group through a strategy of *annotation transfer*. This practical use has motivated a lot of work in the identification of functional orthologs.

It is often assumed that two proteins with similar sequences or similar structures have similar functions, and conversely that functionally related proteins have similar sequences. Based on this assumption, a number of approaches that use sequence similarity have been developed, e.g. reciprocal-best-BLAST-hits (RBH), for predicting functional orthologs. This resulted in several orthologs databases, such as the *Clusters of Orthologous Groups* (COGs) (Tatusov *et al.*, 2000), *Inparanoid* (O'Brien *et al.*, 2005) and *OrthoDB* (Waterhouse *et al.*, 2011). However, high sequence similarity does not necessarily indicate functional conservation. Since functional sites of proteins are usually only one or several small parts of the whole sequence, two proteins can have a highly significant overall similarity even though all functional sites are completely different (Brutlag, 2008). An even worse case is that the protein in question may not be functional at all as for pseudo-genes. To overcome this problem, *network alignment* approaches (Bandyopadhyay *et al.*, 2006; Liao *et al.*, 2009; Shih and Parthasarathy, 2012) have been proposed that supplement sequence-based algorithms with information from protein-protein interaction (PPI) networks.

1.3.4 Protein function prediction

Protein function prediction (PFP) is a central problem of computational biology and bioinformatics in the post-genomic era. To know how proteins carry out their functions is a basic requirement for understanding the mechanism of life processes at the molecular level. It can also help us understand the causes of diseases, because alterations of protein function are responsible for many diseases and the function of disease-related proteins might be used for drug design (Radivojac, 2013). However, there is a large gap between experimentally annotated proteins and the vast amount of sequenced genomes. Currently, there are ∼7,000 sequenced genomes and ∼21,000 in progress (Pagani *et al.*, 2012). And it suggests that an estimated number of 10–100 million species exist on the earth in total. Moreover, the available function data is incomplete, biased and noisy because of the misinterpretation of experiments, curation errors, and experimental biases.

All the above reasons place the automated annotation of protein functions at the forefront. Computational function predictions can thus be used to formulate biological hypotheses and guide wet lab experiments through prioritization. Therefore, a number of algorithms were proposed for predicting protein functions and inferring evolutionary relationships from genomic context (Pellegrini *et al.*, 1999; Marcotte *et al.*, 1999), protein-protein interaction networks (Vazquez *et al.*, 2003a; Sharan, 2005), protein structures (Pazos and Sternberg, 2004), and microarrays (Huttenhower *et al.*, 2006).

To evaluate the performance of these methods, a large-scale community-based critical assessment of protein function annotation (CAFA) was carried out on a target set of 866 proteins from 11 organisms (Radivojac *et al.*, 2013). It finds out that there is a pressing demand of developing faster and more efficient tools for predicting protein function, although today's best algorithms substantially outperform widely used first-generation methods.

1.4 An overview of this thesis

In this thesis, we describe two *multiple network alignment algorithms LocalAli* (Hu and Reinert, 2014) and *NetCoffee* (Hu *et al.*, 2014) and their applications in high-throughput data. *LocalAli* is designed for *local network alignment* aiming to identify functionally conserved modules across multiple species. *NetCoffee* is designed for *global network alignment* aiming to detect functional orthologs across multiple species. To evaluate the performance of our algorithms, each algorithm has been tested on several real biological datasets. The results suggest that both of the two algorithms provide substantial improvements to currently existing algorithms.

Chapter 2 describes the preliminary materials such as definitions, notations and a review of previous algorithms. Section 2.1 gives the definitions of *global network alignment* and *local network alignment.* In Section 2.2, two problems in the graph theory that involve in the problem of network alignment are introduced. Chapter 3 describes the *LocalAli* algorithm in detail and its application in 26 real datasets and 1040 random datasets. An evolutionary model and a concept of evolutionary distance are introduced in Section 3.1. The detailed information of the *LocalAli* algorithm is given in Section 3.2. The test data sets and the performance evaluation are included in Section 3.3. Another computational tool *NetCoffee* is introduced in Chapter 4 for solving the problem of *global network alignment.* In this chapter, the *NetCoffee* algorithm is described in Section 4.1. Then, computational complexity is calculated in Section 4.2. Finally, Section 4.3 presents the result part and the performance comparison between *NetCoffee* and other previous algorithms. Chapter 5 gives the conclusion and the future work.

Chapter 2

Preliminary Materials

Owing to recent advancements in high-throughput technologies, PPI networks of more and more species become available in public databases. Subsequently, one of the most interesting questions that scientists are concerned with is how to get biologically meaningful knowledge that hidden behind these data. Analogous to *sequence alignment*, *network alignment* provides a promising framework for understanding biological function, evolution, and disease. In this chapter, we explicitly introduce the *network alignment* problem and some other related graph problems such as *graph matching* and *subgraph searching*.

2.1 Network alignment

2.1.1 Definitions and notations

Network alignment aims to find similarities between the structure or topology of two or more networks which mainly include PPI networks (Kuchaiev *et al.*, 2010; Neyshabur *et al.*, 2013; Singh *et al.*, 2007), metabolic networks (Ay *et al.*, 2012; Ma *et al.*, 2013; Pinter *et al.*, 2005) and gene regulatory networks (Gülsoy *et al.*, 2012). In addition to the network topology, other biological information has often been taken into consideration in the similarity calculation, such as sequence similarity, phylogeny, co-expression, co-inheritance, co-evolution and co-location (Flannick *et al.*, 2006). In this thesis, we are going to focus on the problem of PPI *network alignment.* If there is no special mention, *network alignment* refers to PPI *network alignment* in the following parts of this thesis.

Generally, the result of *network alignment* is a *one-to-one* or *many-to-many* node mapping table for the input networks. Nodes that are grouped into a same cluster in a node-mapping table constitute an equivalence class. Each equivalence class must have at most one node from each species in a *one-to-one* table, whereas it might have more than one node from each species in a many-to-many table. *Network alignment* algorithms have been applied to understand various biological questions, such as protein function, functional or-

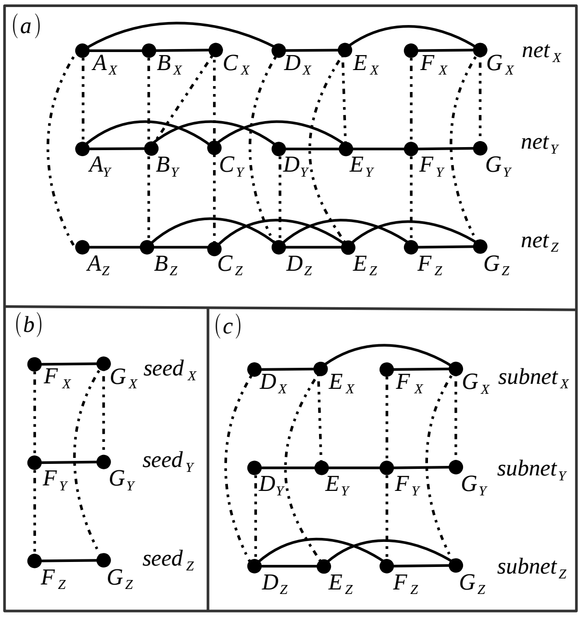

(a)
A_X B_X C_X D_X E_X F_X G_X net_X
A_Y B_Y C_Y D_Y E_Y F_Y G_Y net_Y
A_Z B_Z C_Z D_Z E_Z F_Z G_Z net_Z
(b)
F_X G_X $seed_X$
F_Y G_Y $seed_Y$
F_Z G_Z $seed_Z$
(c)
D_X E_X F_X G_X $subnet_X$
D_Y E_Y F_Y G_Y $subnet_Y$
D_Z E_Z F_Z G_Z $subnet_Z$

Fig. 2.1: An example of searching for a *d-subnet* from PPI networks of three species, X, Y, Z. (a) A 3-layer (*k-layer* in general case) graph consisting of PPI networks and their bipartite graphs of the three species. In the graph, each layer is a PPI network , solid lines are interactions and dashed lines are edges of homologous proteins. (b) One of refined seeds consisting of two *k-spines*. (c) A *d-subnet* extended from the seed in (b).

thologs, functionally conserved modules, molecular evolution and phylogeny.

Network alignment algorithms can be categorized into *pairwise* and *multiple* network alignments according to the number of species, and into *local* and *global* network alignments according to its target regions of interest. Pairwise approaches align two networks and multiple approaches three and more networks.

Local alignment approaches detect node mapping tables for two (pairwise local alignment) or more (multiple local alignment) conserved subnetworks which are usually independent and high-scoring local regions, each implying a putative functional module such as a protein complex (Sharan, 2005) or metabolic pathway (Kelley, 2003; Kelley *et al.*, 2004). Both pairwise and multiple local alignment attempt to find optimal *many-to-many* mapping tables. We use attributed undirected graphs $\{G_1, G_2, \cdots, G_k\}$ to represent protein-protein interaction (PPI) networks of k different species. Each graph $G_i = (V_i, E_i, \mathcal{A}_i)$ corresponds to a species, where V_i represents all the proteins, E_i the collection of interactions and $\mathcal{A}_i : V_i \rightarrow \Sigma^*$ a labeling function that assigns protein sequences to their nodes. Further, a set of $\binom{k}{2}$ bipartite graphs $B_{ij} = (V_i \cup V_j, E_{ij})$ can be constructed by joining pairs of proteins between V_i and V_j ($i < j$ and $i, j \in \{1, 2, \cdots, k\}$) if their sequences are sufficiently similar. To be clear, we refer to elements of E_i and E_{ij} as *interactions* (solid lines in Fig. 2.1(a)) and *edges* (dashed lines in Fig. 2.1(a)), respectively. A set of k proteins, each from one species, which are connected by *edges* is termed as a *k-spine* (Kalaev *et al.*, 2009), such as $\{A_X, A_Y, A_Z\}$ in Fig. 2.1(a). And a set of d *k-spines* connected by *interactions* form a *d-subnet*, such as the four *k-spines* in Fig. 2.1(c). Proteins that participate in a common structural complex or metabolic pathway are called *functionally linked* (Pellegrini *et al.*, 1999). These groups of functionally linked proteins are *functional modules*. Then, we formulate the problem of *local network alignment* as a problem of searching for *d-subnets*.

Definition 1. *Let* $\{G_1, G_2, \cdots, G_k\}$ *be a set of PPI networks,* Ξ *all possible* d-subnets*,* $\varphi : \Xi \rightarrow R$ *a scoring function,* ***local network alignment*** *is a problem of finding a collection of high-scoring* d-subnets *of* $\{G_1, G_2, \cdots, G_k\}$*, in which each* d-subnet *represents a set of k conserved modules.*

Global alignment approaches determine an optimal global node mapping table for the input PPI networks (Huang *et al.*, 2013; Li *et al.*, 2007; Milenković *et al.*, 2013; Singh *et al.*, 2007, 2008), each set of matched nodes (i.e. proteins) implying a putative function-oriented ortholog group. Proteins aligned in an equivalence class are supposed to be descended from the same protein of their common ancestral species according to a series of evolutionary events: protein deletion, protein duplication, protein mutation and paralog mutation (Flannick *et al.*, 2008). Typically, *pairwise global* alignment attempts to provide a one-to-one mapping table between PPI networks and *multiple global* alignment attempts to find a many-to-many mapping table. Let $\{G_1, G_2, \cdots, G_k\}$ represent a set of $k \geq 3$ PPI networks. Each network $G_i = (V_i, E_i)$ is an unweighted graph, where V_i is a set of nodes representing proteins and E_i a set of binary interactions appearing in the networks. Let $V = \cup_{i=1}^{k} V_i$ be the union of all nodes. A *match-set* ϑ is a subset of V. Then, we formulate the problem of global network alignment as one of finding a set of mutually disjoint *match-sets*, which has an optimal overall alignment score.

Definition 2. *Let* $\{G_1, G_2, \cdots, G_k\}$ *be a set of k PPI networks and* $\Phi : \mathbb{A} \rightarrow R$ *a scoring function for global alignments* $\mathbb{A}$*,* ***global network alignment*** *is a problem of finding an optimal solution* $\mathbb{A}^*$ *which is a set of mutually disjoint* match-sets $\mathbb{A}^* = \{\vartheta^1, \vartheta^2, \cdots, \vartheta^m\}$*, where* $\vartheta^i \cap \vartheta^j = \emptyset$*,* $\forall i, j, i \neq j$ *such that* $\Phi(\mathbb{A}^*)$ *is the maximum.*

Each protein in a *global network alignment* belongs to just one *match-set*, whereas each protein in a *local network alignment* or a *d-subnet* might be present in multiple *k-spines*.

In a global alignment, each *match-set* is an equivalence class. In a local alignment, a set of *k-spines* constitute an equivalence class if they share common proteins within them.

2.1.2 Previous algorithms

The network alignment approach provides an effective way of systematically identifying biologically significant patterns or protein groups by comparing the similarity of PPI networks. To date, many network alignment algorithms have been published.

Local network alignment

Both *in silico* and *in vivo* studies suggest that *functional modules* of organisms tend to be conserved during the evolution history (Roguev *et al.*, 2008; Pellegrini *et al.*, 1999). Based on this test hypothesis, *local network alignment* provides a general computational framework which searches for high-scoring conserved subnetworks to detect functionally conserved modules across species.

The development of *local* alignment tools or web servers has become a quite active field in the last decade. The most notable *pairwise local* alignment tools include *PathBlast* (Kelley *et al.*, 2004), *MaWISh* (Koyutürk *et al.*, 2006), *NetworkBlast* (Kalaev *et al.*, 2008), *AlignNemo* (Ciriello *et al.*, 2012) and *NetAligner* (Pache and Aloy, 2012; Pache *et al.*, 2012). Just a few *multiple local* alignment tools have been developed. The currently existing *multiple local* alignment tools include *Graemlin* (Flannick *et al.*, 2006, 2009), *CAPPI* (Dutkowski and Tiuryn, 2007) and *NetworkBlast-M* (Kalaev *et al.*, 2009). In addition, there are also some works trying to detect functionally conserved modules by using a combination of clustering algorithms and global alignment algorithms, such as *PINALOG* (Phan and Sternberg, 2012). An evolutionary-based multiple local network alignment tool *LocalAli* (Hu and Reinert, 2014) is described in chapter 3.

Global network alignment

With the development of high-throughput technologies such as mass spectrometry (Ho, 2002), microarrays (Lashkari *et al.*, 1997), yeast two-hybrid assays (Ito *et al.*, 2001) and next-generation sequencing, a tremendous amount of genomics, proteomics, and protein interaction data has been generated and became available in public databases (Uniprot Consortium, 2007; Szklarczyk *et al.*, 2011). This comprehensive experimental data provides a basis for analyses that aim at discovering conservation of protein function among different species, such as *functional orthologs*. At the very beginning, sequence-based algorithms, such as reciprocal-best-BLAST-hits (RBH), were widely used to predict *functional orthologs*. However, many studies suggest that sequence similarity is not necessary to indicate functional conservation. For an example, sequence-based algorithms usually employ a dynamic programming that permits arbitrary amino acid substitutions. If such substitutions occur within functional sites, then the inference of a common function may be wrong despite a highly significant overall similarity. To address this problem, some pioneering *global network alignment* algorithms (Bandyopadhyay *et al.*, 2006; Liao *et al.*, 2009; Shih and Parthasarathy, 2012) were proposed to predict *functional orthologs* with the integrated information of PPI networks, co-evolution, sequence similarity etc.

Many *pairwise global* alignment tools have been proposed in the last decade, which include *IsoRank* (Singh *et al.*, 2007), MNAligner (Li *et al.*, 2007), Corbi (Huang *et al.*, 2013), GNA and PATH (Zaslavskiy *et al.*, 2009), *PISwap* (Chindelevitch *et al.*, 2010, 2013), *MI-GRAAL* (Kuchaiev and Pržulj, 2011), *Natalie* 2.0 (El-Kebir *et al.*, 2011), *GHOST* (Patro and Kingsford, 2012), GRAAL (Kuchaiev *et al.*, 2010), H-GRAAL (Milenković *et al.*, 2010), SPINAL (Aladağ and Erten, 2013), MAGNA (Saraph and Milenković, 2014), NETAL (Neyshabur *et al.*, 2013) and so forth. With the increasing availability of PPI networks, the demand for global alignment tools of multiple networks has risen. Hence, several *multiple global alignment* tools have been developed, which include *Graemlin* 2.0 (Flannick *et al.*, 2008), *IsoRank-N* (Liao *et al.*, 2009), *SMETANA* (Sahraeian and Yoon, 2013) and *BEAMS* (Alkan and Erten, 2014). In chapter 4, we introduce our new multiple global alignment algorithm *NetCoffee* (Hu *et al.*, 2014) in a full detail.

2.2 Related problems in graph theory

Network alignment is a problem of comparing the similarity of two or more networks (graphs). To resolve this problem, it usually involves solving some sub-problems in graph theory, such as *exact graph matching*, *inexact graph matching*, *subgraph searching*, finding a *maximum matching*. Here, we briefly introduce the graph problem of *maximum matching* and *subgraph searching.*

2.2.1 Maximum matching

Definition 3. *Given a graph $G = (V, E)$, a* ***matching*** *M of G is a subset of the edges E such that no vertex in V is incident to more than one edge in M.*

Definition 4. *Given a graph $G = (V, E)$, a matching M of G is a* ***maximum matching*** *or* ***maximum cardinality matching*** *if for any other matching M' of G, $|M'| \leq |M|$.*

Definition 5. *Given a weighted graph $G = (V, E)$, a matching M of G is a* ***maximum weighted matching*** *if the sum of the values of the edges in M have a maximal value.*

Definition 6. *Given a matching M, an* ***augmenting path*** *is a path with an odd number of edges $\{e_1, e_2, \cdots, e_m\}$ such that $e_{odd} \notin M$ and $e_{even} \in M$.*

Theorem 1. *A matching M is maximum iff it has no augmenting path (Berge, 1957).*

Algorithm 1 Finding a maximum matching by using the augmenting path algorithm

```
1: $M = \emptyset$;
2: while $augmentingPath(G, M, P)$ do        ▷ An augmenting path algorithm.
3:     $M = M \oplus P$;
4: end while
5: return $M$;
```

If a matching M has an augmenting path P, then switching the edges along the path P from in-to-out of M and vice versa. This operation can be defined as $\oplus$. Each $\oplus$ can yield a new matching M' which has one more edge than M. With Berge's theorem, the problem of *maximum matching* is reduced to a problem of finding an augmenting path. As shown in Algorithm 1, the function $augmentingPaht(G, M, P)$ (line 2) represents an algorithm of finding an augmenting path P for a *matching* M of G. It returns false if M has no augmenting path, otherwise true.

Generally, there are four closely related problems of finding a *maximum matching*: *maximum cardinality matching in bipartite graphs* (Problem 1), *maximum cardinality matching in general graphs* (Problem 2), *maximum weighted matching in bipartite graphs* (Problem 3), *maximum weighted matching in general graphs* (Problem 4) (Galil, 1983). They are all special cases of the problem of *maximum weighted matching* in general graphs. However, usually they are considered in increasing order of difficulty. *Maximum matching* in a bipartite graph is the simplest problem, because *augmenting path algorithm* can easily find an augmenting path if it exists. Hopcroft-Karp algorithm (Hopcroft and Karp, 1973) solved this problem in $O(\sqrt{V}E)$ time. Problem 3 is also known as the *assignment problem*. The first polynomial-time algorithm of this problem is the *Hungarian algorithm* (Kuhn, 1955). If the Bellman-Ford algorithm is used for the shortest path search in the augmenting path algorithm, the running time of *Hungarian algorithm* becomes $O(V^2E)$. The first polynomial-time algorithms for Problem 2 and 4 are due to Edmonds (Edmonds, 1965b,a). Edmonds's algorithm solves Problem 4 in the running time of $O(V^4)$. Later, Lawler and Gabow improved Edmonds's algorithm by finding a way to implement it in

Algorithm 2 Extend Subgraph to a desired size (Wernicke, 2006).

```
function ExtendSubgraph($V_{Subgraph}$,$V_{Extension}$,$v$)
    if $|V_{Subgraph}| = k$ then
        output $G[V_{Subgraph}]$; return ;
    end if
    while $V_{Extension} \neq \emptyset$ do
        Remove an arbitarily chosen vertex $w$ from $V_{Extension}$;
        $V'_{Extension} \leftarrow V_{Extension} \cup \{u \in N_{excl}(w, V_{Subgraph}) : u > v\}$;
        ExtendSubgraph($V_{Subgraph} \cup w$,$V'_{Extension}$,$v$);
    end while
end function
```

Algorithm 3 EnumerateSubgraphs(G, k) (ESU) (Wernicke, 2006)

```
for each vertex $v \in V$ do
    $V_{Extension} \leftarrow \{u \in N(v) : u > v\}$;
    ExtendSubgraph($V_{Subgraph}$,$V_{Extension}$,$v$);
end for
return ;
```

$O(V^3)$. An $O(VE \log V)$ algorithm (Galil *et al.*, 1986; Mehlhorn and Schäfer, 2002) based on Edmonds's algorithm was further implemented by using generalized priority queues, which was much better for sparse graphs.

2.2.2 Subgraph search

Given a large network, exhaustively enumerating all subgraphs with a given number of vertices is known to be computationally hard. It is also known as the problem of counting subgraphs. To solve this problem, an exhaustive-enumeration algorithm was developed to count the occurrences of all types of k-node connected subgraphs in a large network (Milo, 2002; Shen-Orr *et al.*, 2002). However, the running time increases dramatically when k increases. To cope with the complexity of subgraph counting in large networks, a probabilistic algorithm which so-called ESA based on a randomly sampling technique was developed to estimate the number of larger subgraphs (Kashtan *et al.*, 2004). Later, a more efficient algorithm known as ESU-RAND was proposed to estimate the frequency of subgraphs in networks (Wernicke, 2006). In contrast to ESA, ESU-RAND is orders of magnitude faster than previous algorithms, thus allowing the search for subgraphs of a larger size. This idea starts with an algorithm termed as *ESU* that efficiently enumerates all size-k subgraphs. In the process of searching subgraphs, *ESU* visits nodes of each subgraph in a sequential order and the node with the smallest id must be its starting node. The *ESU* algorithm is then adapted for an unbiased subgraph sampling algorithm that randomly skips over some of these subgraphs. Because of its efficiency, we adapted *ESU* for a search for *k-spines* in the problem of *local network alignment*.

Part II

Methods, Results and Discussion

Chapter 3

An Algorithm for Multiple Local Network Alignment

Although many efforts have focused on the problem of local network alignment, just a few alignment tools have been developed for multiple networks, including *Graemlin*, *CAPPI* and *NetworkBlast-M*. Basically, currently existing *multiple local* alignment tools are concerned with three major issues. The first one is the scalability. To date, *CAPPI* was only applied to three networks and compatible with particularly designed data. *NetworkBlast-M* is unable to run on networks which have nodes with a large degree (Hu *et al.*, 2014). Thus, the scalability of these tools is at a modest level. Another issue is the evolutionary relevance of the reported hits. To answer the question of how conserved modules of descendants have been evolved from their origin, the scoring schemes shall be more strongly rooted in an evolutionary model (Sharan and Ideker, 2006). But, neither the evolution history nor a probabilistic model of network growth was considered by *Graemlin* and *NetworkBlast-M*. The third issue is speed. The problem of aligning multiple networks is computationally intractable (Kalaev *et al.*, 2009). Parallelization techniques can largely speed up local alignment algorithms because each target of interest can be searched through one single thread. Yet, none of the existing multiple local alignment tools support parallel computing.

To remedy these limitations, we developed a fast and scalable *multiple local* network alignment tool, *LocalAli*, for the identification of functionally conserved modules. In this algorithm, we firstly proposed a new framework to reconstruct the evolution history of conserved modules based on a *maximum-parsimony* evolutionary model. By relying on this model, *LocalAli* facilitates the interpretation of resulting local alignments in terms of *conserved modules* that have been evolved from a common ancestral module through a series of evolutionary events.

In this chapter, we first introduced an evolutionary model of functional modules . Subsequently, the *LocalAli* algorithm was described along with a simple example. Afterward, we applied *LocalAli* and several previous algorithms to 26 real-world datasets and 1040 random datasets. Last, we evaluated the biological quality and statistical significance of our results in terms of a series of criteria.

3.1 Models of functional module evolution

3.1.1 Existing evolutionary models

In PPI networks, gene duplication and divergence are the underlying mechanism that most probably generates the scale-free topological feature (Vazquez *et al.*, 2003b; Wagner, 2003). Among all existing *multiple local* alignment tools, only *CAPPI* uses a network growing model (*i.e.* duplication-divergence) to derive the posterior probabilities of interactions in ancestral PPI networks, whereas other tools are not strongly rooted in an evolutionary model. In addition, there are some other computational models applied to the problem of network history inference, such as *maximum-likelihood* (Zhang and Moret, 2008) and *parsimonious-histories* (Patro *et al.*, 2012; Patro and Kingsford, 2013). Inspired by the latter approaches, we here introduce a similar parsimony-based model that aims to reconstruct the ancestral subnetwork for a set of conserved subnetworks. This model was designed based on a hypothesis that proteins that function together in a pathway or a structural complex are likely to evolve in a correlated fashion (Pellegrini *et al.*, 1999). It means that proteins of functional modules tend to be either preserved or eliminated all together during the evolution of the whole PPI network from their common ancestor. Unlike Dutkowski's algorithm (Dutkowski and Tiuryn, 2007) which gives a global view of the evolution of the whole networks, *LocalAli* provides a new framework to reconstruct the evolution of conserved subnetworks.

3.1.2 The evolutionary tree

To elucidate the phylogenetic relationships of functional modules that are evolved from a common ancestor, we use a binary tree as the evolutionary tree to model the evolutionary process (see in Fig. 3.3(a,b)). In the tree of functional modules (Fig. 3.3(b)), *external nodes*, which are also called *leaves*, represent the observed functional modules. *Internal nodes* represent the corresponding functional modules of the predecessor species. The *root* represents the corresponding functional module of the original species.

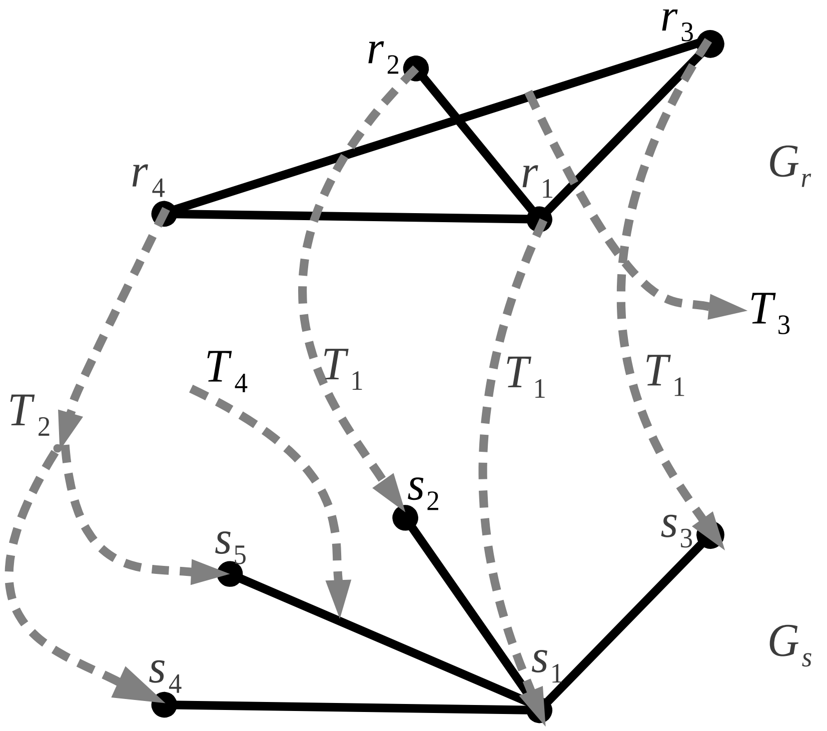


Fig. 3.1: Illustration of the evolutionary model. G_r and G_s are two functional modules. Proteins are represented by nodes, interactions by solid lines, evolutionary events from G_r to G_s by dashed arrows. T_1, T_2, T_3, T_4 refer to evolutionary events *protein mutation*, *protein duplication*, *interaction deletion* and *interaction insertion*, respectively. Suppose $t = 1$, $\alpha = 0.2$ and $\beta = 2$, by definition, the evolutionary distance is calculated as follows: $f(G_r, G_s, \mathcal{M}_{rs}) = f_1 + f_2 + f_3 + f_4 = (e^{-0.2} + e^{-0.2\times 2} + e^{-0.2\times 3}) + e^{-0.2\times 2} + e^{-0.2\times 2} + e^{-0.2\times 2} = 4.049$.

3.1.3 Evolutionary events and distances

Evolutionary events are the basic building blocks of network evolution, and evolutionary distance describes how far a descendant subnetwork goes away from an ancestral subnetwork. To infer the evolution history, it is necessary to introduce the definition of evolutionary event and distance. Pellegrini's investigation and the scale-free topological features show that duplication and divergence are the major driving forces of network evolution (Pellegrini *et al.*, 1999; Wagner, 2003). Taking these evidences into consideration, we attempt to understand the evolution process using the following four types of evolutionary events:

(1) *Protein mutation*: the sequence change of two proteins in two species;

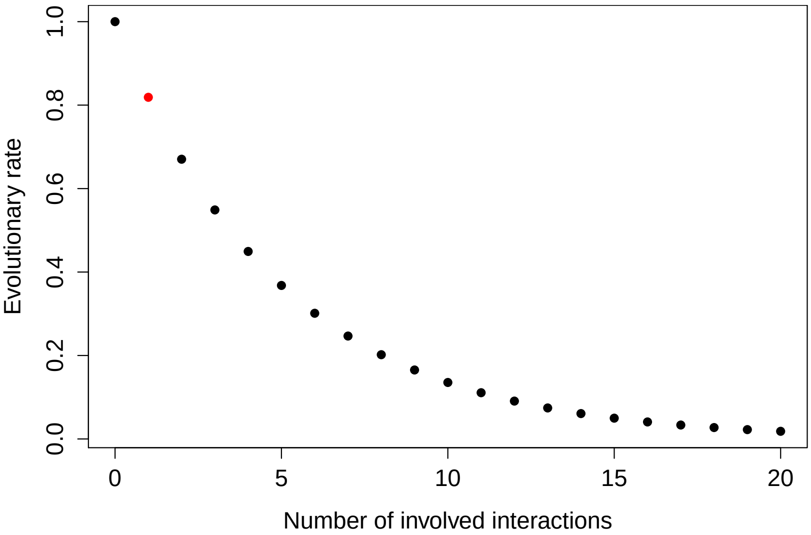


Fig. 3.2: The evolutionary rate of proteins with different involved interactions. The red point represents the evolutionary rate for interaction insertion and deletion. Here $\alpha = 0.2$ and $\beta = 2.0$.

(2) *Protein duplication*: the duplication of a protein in an offspring species;

(3) *Interaction deletion*: the loss of an interaction from one network to another;

(4) *Interaction insertion*: the gain of an interaction from one network to another.

Let $G_r = (V_r, E_r, \mathcal{A}_r)$ and $G_s = (V_s, E_s, \mathcal{A}_s)$ be two functional modules. As illustrated in Fig. 3.1, G_s descends from G_r according to a correspondence match $\mathcal{M}_{rs} : V_r \to V_s$. We denote as $f_i(G_r, G_s, \mathcal{M}_{rs})$ the evolutionary distance caused by type i events during the evolution from G_r to G_s. An investigation (Fraser *et al.*, 2002) shows that proteins with more interactions (i.e. hub nodes) evolve more slowly because more proteins are directly involved in the functions of these hub nodes. In other words, proteins with different number of interactors have different evolutionary rates. Hence, we choose $e^{-\alpha \cdot deg(v)}$ as the function to calculate the evolutionary rate of a protein v, and $e^{-\alpha \cdot \beta}$ as the evolutionary rate of each interaction in the PPI networks (see an example in Fig. 3.2). Consequently, the evolutionary distance function for each type of event is written as follows:

$$f_i(G_r, G_s, \mathcal{M}_{rs}) = \begin{cases} \Sigma_{v \in T_i} e^{-\alpha \cdot deg(v)} t & i \in \{1, 2\} \\ \Sigma_{e \in T_i} e^{-\alpha \cdot \beta} t & i \in \{3, 4\} \end{cases}$$

where T_i is the collection of type i events, $deg(v)$ is the number of interactions connected with protein $v \in V_r$, α and β are parameters adjusting the evolutionary rates, t is the evolutionary

time from G_r to G_s. The *evolutionary distance* between G_r and G_s, $f(G_r, G_s, \mathcal{M}_{rs})$, is defined as $f(G_r, G_s, \mathcal{M}_{rs}) = \Sigma_{i=1}^{4} f_i(G_r, G_s, \mathcal{M}_{rs})$. We chose proper values for α and β so that evolutionary distances caused by proteins and interactions are in balance. Generally, the distances would be in balance if the following two requirements are fulfilled: (i) the evolutionary rate of interaction is similar with that of protein with 2 interactions; (ii) the evolutionary rate of protein is <0.2 when the protein has more than 10 interactions. If β is too small, type 3 and 4 events will become unwelcome in searching for optimal evolutionary tree because these events will result a larger evolutionary distance. If β is too large, type 3 and 4 events will become popular because these events would not actually make a big effect on the evolutionary distance. For this reason, we tested a series of parameters and chose $\alpha = 0.2$ and $\beta = 2.0$ for all of our tests because it can make interaction distance and protein distance in balance (see more in Fig. 3.2). We measured the evolutionary time t by the branch weight in the evolutionary tree as shown in Fig. 3.3(a). The topology and branch weight of the evolutionary tree was calculated based on the common tree of NCBI Taxonomy database (Federhen, 2012). For example, given three extant species A, B and C, A and B have a common predecessor D, C and D have a common predecessor E. The number of internal nodes in the tree are as follows:
E→C : 2
E→D : 3
D→A : 1
D→B : 3
Then, the branch weight will be normalized by their longest path from root to the leaves as follows:
E→C : 2/6=0.33
E→D : 3/6=0.5
D→A : 1/6=0.17
D→B : 3/6=0.5
See an example of the evolutionary distance between G_r and G_s in Fig. 3.1.

3.2 The *LocalAli* algorithm

3.2.1 Overview

To identify functionally conserved modules from multiple networks, we proposed an evolutionary-based local alignment approach to heuristically search for high-scoring *d-subnets* with the information of interaction, homologous proteins and phylogenetic trees. First, the method

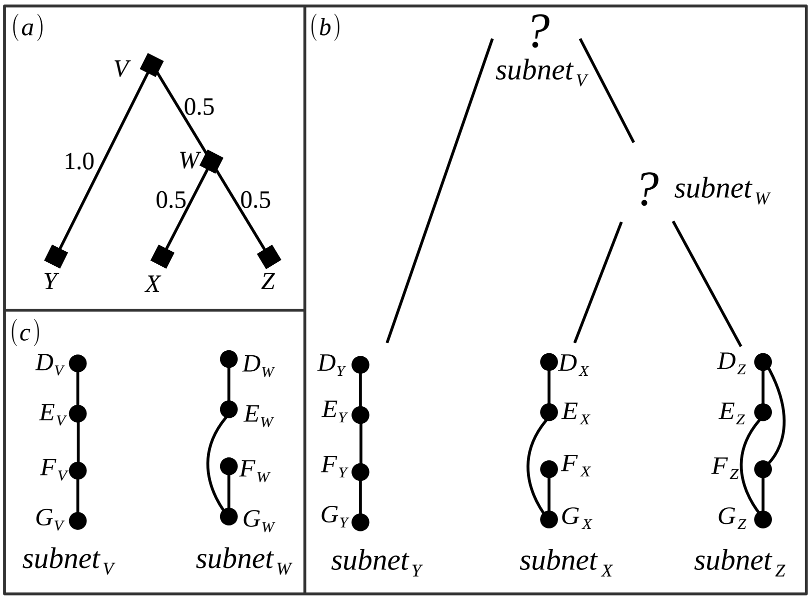


Fig. 3.3: A sketch of reconstructing an evolutionary tree of a *d-subnet*. (a) Give a phylogenetic tree of species X, Y, Z. (b) Set the k induced subnetworks of a *d-subnet* as leaves of its evolutionary tree. This tree has the same topology and branch weight with its species tree. (c) Reconstruct optimal or near-optimal *internal nodes* of $subnet_V$ and $subnet_W$ such that this evolutionary tree has the minimal evolutionary distance. Let $\alpha = 0.2, \beta = 2$. The distance is calculated as follows: $f(V, Y, M_{VY}) + f(V, W, M_{VW}) + f(W, X, M_{WX}) + f(W, Z, M_{WZ}) = (2e^{-2\alpha} + 2e^{-\alpha}) \times 1.0 + (2e^{-2\alpha} + 2e^{-\alpha} + 2e^{-\alpha\beta}) \times 0.5 + (2e^{-2\alpha} + 2e^{-\alpha}) \times 0.5 + (4e^{-2\alpha} + e^{-\alpha\beta}) \times 0.5 = 8.302$.

constructs a set of k PPI networks and bipartite graphs with interactions and homologous proteins. These networks and bipartite graphs are integrated into a *k-layer* graph (Kalaev *et al.*, 2009) as illustrated in Fig.2.1(a). Then, it heuristically searches for a set of refined seeds using a *seed-and-extend* approach (see Fig.2.1(b)) from the *k-layer* graph and extends them with a local search strategy to *d-subnets* (see Fig.2.1(c)), which are in a range of predefined *minimal* and *maximal* size. Afterward, the k-induced subnetworks of each *d-subnet* are set as the leaves of an evolutionary tree (see Fig.3.3(b)), which has the same topology and branch weights as its corresponding phylogenetic tree of the involved species (see Fig.3.3(a)). Under the *maximum parsimony* principle, the optimal or near-optimal internal nodes (e.g. $subnet_v$ and $subnet_w$ in Fig.3.3(b,c)) are found by using *simulated annealing* such that the tree receives a minimal evolutionary distance according to our evolutionary model. Finally, an alignment score of each *d-subnet* is calculated and those with a score less than a threshold are filtered out.

3.2.2 Search for *d-subnets*

As demonstrated in Definition 1, the problem of *local network alignment* is to search for a set of high-scoring *d-subnets* from k input PPI networks. However, the task of enumerating all *d-subnets* is computationally hard (Kalaev *et al.*, 2009), because the complexity of the fastest known algorithm is $O(n^{kd})$. To speedup the search process, we employed a widely used heuristic approach *seed-and-extend* (Kalaev *et al.*, 2009; Sharan, 2005). It practically reduces the computational time by skipping over many *seeds* that are weakly connected by interactions.

Let $\{G_1, G_2, \cdots, G_k\}$ ($G_i = (V_i, E_i)$) represent k PPI networks, where V_i represents all the proteins, and E_i the collection of interactions within the PPI network G_i. Further, a set of $\binom{k}{2}$ bipartite graphs $B_{ij} = (V_i \cup V_j, E_{ij})$ are constructed by joining pairs of proteins between V_i and V_j ($i < j$) if their sequences are sufficiently similar. We construct a k-layer graph G_H=$\{V_H, E_H\}$, where $V_H = \cup_{i=1}^{k} V_i$ and $E_H = \cup_{i<j}(E_{ij} \cup E_i)$ $(i, j \in \{1, 2, \cdots, k\})$. Our aim is to heuristically search for a set of *d-subnets* in the k-layer graph G_H as shown in Fig. 2.1.

Collect the starting nodes

By definition, *k-spine* is a special case of size-k subgraph of the k-layer graph G_H. Therefore, we adapted a subgraph searching algorithm *ESU* (Wernicke, 2006) for sampling *k-spines* from G_H. In this method, all the k nodes of each *k-spine* are visited in order. We called the first visiting node of a *k-spine* the *starting node.* Visiting a non-starting node will never result in a *k-spine* of the k-layer graph. To find *k-spines* in an efficient way, we first collect all these starting nodes in G_H (Algorithm 4). As described in Algorithm 7, *LocalAli* goes through each node in the k-layer graph $G_H = (V_H, E_H)$ and test whether there exists at least one *k-spine* starting from this node. If yes, it is a starting node; otherwise it is not.

Sample a *k-spine*

A set of *k-spines* connected by *interactions* constitute a *d-subnet.* Hence, the problem of sampling a *k-spine* is a subproblem of finding a *d-subnet.* The algorithm used for sampling a *k-spine* from a given starting node is described in Algorithm 5.

For a given graph $G = (V, E)$ and a set of vertices $V' \subset V$, $Adj(V')$ is the set of all vertices from $V \backslash V'$ which are connected to at least one vertex in V'. For a vertex

$v \in V \backslash V'$, we denote $Adj_{excl}(v, V')$ as the set of all vertices neighboring v that do not belong to $V' \cup Adj(V')$. First, we initialize the subgraph `kspine` with a given starting node v (line 3). We add those neighbor vertices of v, $Adj(v) = \{u|uv \in E_H\}$, into `kspine` if two requirements are fulfilled: 1) $uv \notin \cup_i E_i$ (line 7–9); 2) $u > v$ (line 10–12). Then more neighbor vertices are recursively added into `kspine` till k vertices are filled in (see Algorithm 6). During the expansion of `kspine`, we update the set of candidate nodes of `kspine`, `candidates`, using nodes from Adj_{excl}(w,`candidates`) (line 7–12). If it fails to find k vertices for `kspine`, it means that there does not exist a size-k subgraph starting from v. Therefore, to give a starting node is a key for succeeding in sampling a size-k subgraph. We iteratively go through each node in V_H and collect all possible starting nodes that can derive at least one *k-spine* (see Algorithm 7).

Search for refined seeds

A seed is a set of of *k-spines* that are strongly connected through interactions (*seedSize*$\geq$2 in Algorithm 8). To find a refined seed, first, we sample a *k-spine* with an arbitrarily selected starting node v as the initial state of the seed, which was denoted as `subnet` (see line 5, 6, 17). Then a set of starting nodes that connect (through interactions) with at least one vertex of `subnet` are collected in `candidates` (see line 18). Afterward, we repeatedly sample another *k-spine* which starts from one neighbor node in `candidates` until the new *k-spine* is strongly connected to the current seed `subnet` (see line 9-15). We say a new *k-spine* is strongly connected with `subnet` iff three requirements are fulfilled: 1) the *k-spine* does not share any vertex with `subnet`; 2) at least one vertex in the *k-spine* directly connects (through interaction) to `subnet`; 3) other vertices in the *k-spine* connect (through interactions) with at least one vertex of `subnet` by a distance equal to or less than `extdist1`. Here, `extdist1` is a user-specified parameter. Consequently, given a k-layer graph G_H, we are able to sample a random number of seeds by repeatedly calling the procedure of sampling seeds (see Algorithm 9). A higher number of seeds will increase the coverage of reported *d-subnets*, but might also result in more overlapped *d-subnets* and more computational time.

Extend refined seeds to *d-subnets*

Given a refined seed, we use a local search strategy extending it to a *d-subnet* with m another *k-spines*, where m is bounded by the interval $[minExt, maxExt]$. As shown in Algorithm 10, the method of extending a seed to a *d-subnet* is similar with the progress of extending a *k-spine* to a seed. We also repeatedly sample a new *k-spine* that connects (through interactions) with the current seed `subnet` and add it as a new member of `subnet` if two requirements are fulfilled: 1) at least one vertex in the *k-spine* directly connects (through interaction) to `subnet`; 2) other vertices in the *k-spine* connect (through interactions) with at least one vertex of `subnet` within a distance of `extdist2`. If it fails to find at least *minExt* new *k-spines* for a seed according to the above conditions, then this seed would be eliminated from the refined set.

3.2.3 Reconstruction of ancestral functional modules

To develop more understandings of the evolutionary history of the extant functional modules, the reconstruction of ancestral functional modules becomes a central problem. To come up with this problem, we model it as an optimization problem of finding a series of optimal ancestral subnetworks that yield the smallest distance in the evolutionary tree. Subsequently, we use a meta-heuristic method *simulated annealing* (SA) (Kirkpatrick *et al.*, 1983) to find the optimal or near-optimal solution (see in Fig.3.3).

The optimization problem

To explain the descent of the extant functional modules, we estimate their ancestral functional modules (or internal tree nodes) using the *maximum parsimony* principle (Fitch, 1971; Felsenstein, 2003). It means that the generated evolutionary tree requires the optimal internal tree nodes (*i.e.* the optimal ancestral functional modules) such that it yields the smallest evolutionary distance of the tree.

Let T be the evolutionary tree that includes a set of leaves $L = \{P_1, P_2, \cdots, P_k\}$, internal nodes $I = \{P_{k+1}, P_{k+2}, \cdots, P_{k+m}\}$. We refer to $B \subset N \times N$ as all branches of T where $N = I \cup L$, and Γ as the collection of all possible I. We define $\mathcal{M}_{ij}$ as the node correspondence match of P_i and P_j. On the basis of the *maximum parsimony* rule, we reconstruct the set of internal nodes by solving an optimization problem $\min_{I \in \Gamma} \sum_{i,j} f(P_i, P_j, \mathcal{M}_{ij})\delta_{ij}$, where $\delta_{ij} = 1$ iff $(P_i, P_j) \in B$ and $i, j \in \{1, 2, \cdots, m+k\}$.

Search for optimal internal tree nodes

With a tree topology B and its leaves L, the computation of exhaustively searching for the optimal internal tree nodes I^* is numerically intractable. Hence, we use SA to detect optimal or near-optimal answers (see the pseudocode in Algorithm 11).

Let $\mathbf{x} = (e_0, e_1, e_2, \cdots)$ be a series of binary variables which represent the appearance of interactions in the internal nodes. Then, $\mathbf{x}$ can describe the current state of the evolutionary tree. For each observed *d-subnet*, the SA approach starts with a series of non-interaction subnetworks as the initial internal tree nodes (see Fig. 3.4) and specifies the initial temperature to its maximum (see line 1). The initial state is $\mathbf{x} = (0, 0, 0, \cdots)$ since the absence of interactions in all ancestral modules. Then, we use $\Theta(\mathbf{x})$ as our objective function $\sum_{i,j} f(P_i, P_j, \mathcal{M}_{ij})\delta_{ij}$. In the following phase, we diminish the temperature and repeatedly perturb the current state $\mathbf{x}$ with a *Metropolis* scheme using $\pi_i \propto \exp(\Theta(\mathbf{x})/(sT_i))$ as the equilibrium distribution (Kirkpatrick *et al.*, 1983). It is noted that SA allows the alteration of only one interaction from one state $\mathbf{x}_j$ to its neighbor state $\mathbf{x}_{j+1}$ (*i.e.* $|\mathbf{x}_j - \mathbf{x}_{j+1}|$=1) (see line 7). This process continues until the temperature T_i decrease to T_{min}. Eventually, all the internal nodes I^* are reconstructed according to the final solution $\mathbf{x}^*$. By doing so, the topology of ancestral functional modules are reconstructed and the evolution history of the *d-subnets* can be elucidated as a series of evolutionary events in the PPI networks. An example of the reconstruction of ancestral functional modules of a *d-subnet* is illustrated in Fig.3.3.

3.2.4 Scoring function

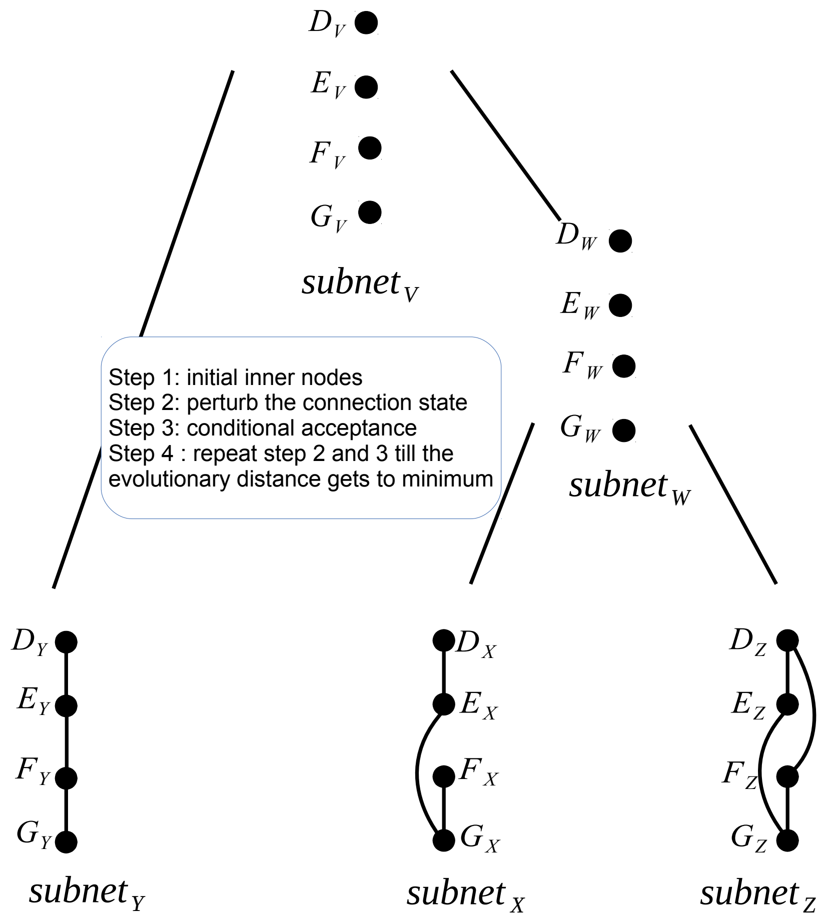
To search for high-scoring local alignments, it is necessary to find a suitable scoring scheme that assigns each *d-subnet* an alignment score. The alignment scores reflect the fit of *d-subnets* to functionally conserved modules.

We introduce a scoring function that can foretell how likely a *d-subnet* could be functionally conserved modules. As mentioned before, each *d-subnet* can be put an evolutionary distance. However, it is not enough to calibrate *d-subnets* of various sizes because the evolutionary distances of *d-subnets* tend to be linearly related to the number of *k-spines* within it. Fig. 3.5 gives the distance of 48,364 *d-subnets* sampled from our datasets. However, it is obvious that *functional modules* are not biased toward the one of a bigger size. So, we assigned each *d-subnet* an alignment score in the following way. Let $\mathcal{U}$ be a *d-subnet*, which includes a set of d *k-spines* and k induced subnetworks of the PPI networks. Regarding the k subnetworks as the leaves L=$\{P_1, P_2, \cdots, P_k\}$ of the evolutionary tree T, we set the scoring function for the *d-subnet* $\mathcal{U}$ as

$$\varphi(\mathcal{U}) = \frac{d}{\min_{I \in \Gamma} \sum_{i,j} f(P_i, P_j, \mathcal{M}_{ij})\delta_{ij}}.$$

Hence, the score of each *d-subnet* is a positive value that indicates the fit of the observed *d-subnet* to a certain conserved functional module. Given two *d-subnet* with a same number of *k-spines*, we intuitively assumed the one with a smaller evolutionary distance is more likely to be a conserved module. The distributions of alignment scores for *d-subnets* sampled from our datasets are in Figures 3.6 to 3.8.

Our algorithm iteratively computes the alignment score of each selected *d-subnet* using this scoring function and filters out those *d-subnets* with a score lower than an user-specified threshold (th). The complete pseudocode of the *LocalAli* algorithm is given in Algorithm 4.



Step 1: initial inner nodes
Step 2: perturb the connection state
Step 3: conditional acceptance
Step 4 : repeat step 2 and 3 till the evolutionary distance gets to minimum

Fig. 3.4: The initial internal nodes of the evolutionary tree.

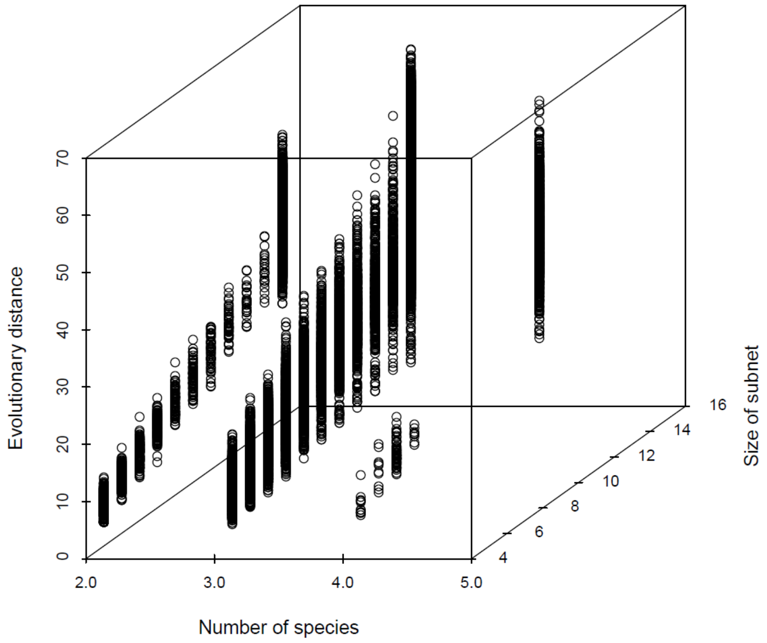


Fig. 3.5: Illustration of the distance distribution on different number of compared species.

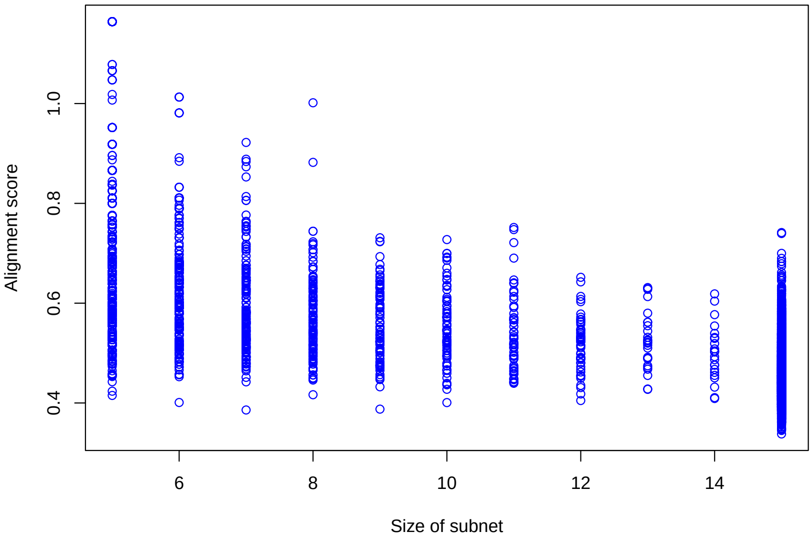


Fig. 3.6: The distribution of alignment score on the cel-dme dataset.

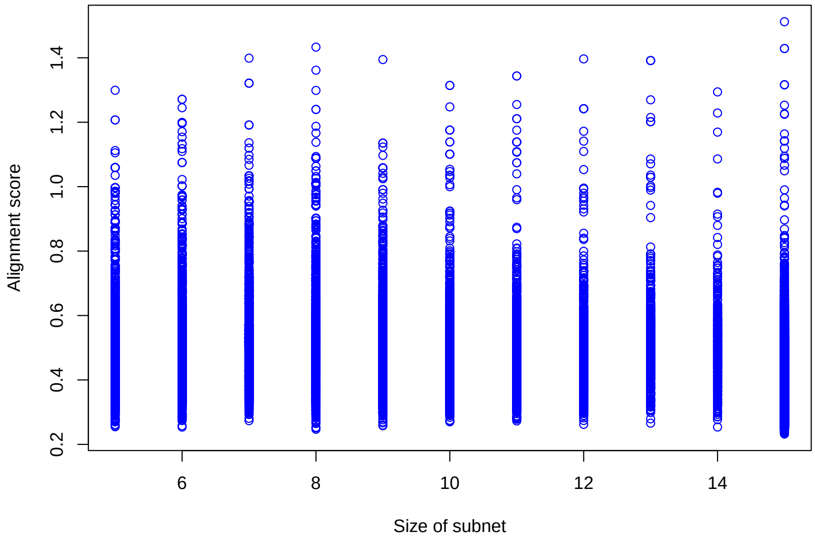


Fig. 3.7: The distribution of alignment score on the hsa-cel-dme dataset.

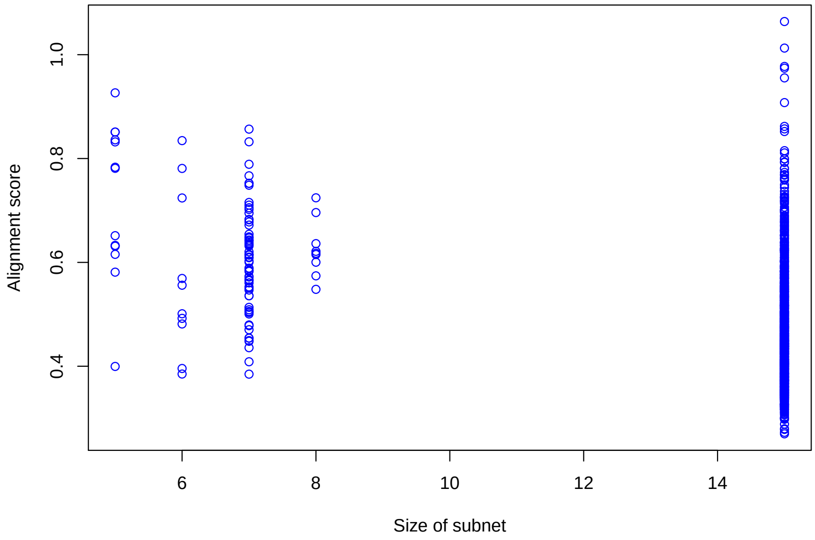


Fig. 3.8: The distribution of alignment score on the hsa-cel-dme-eco dataset.

Algorithm 4 The pseudocode of the *LocalAli* algorithm.

```
$startNodes[] \leftarrow collectStartNodes(G_H)$;
$refinedSeeds[] \leftarrow searchSeeds(G_H, startNodes)$;
$minSize \leftarrow seedSize + minExt$;
$maxSize \leftarrow seedSize + maxExt$;
for $i := 1$ to $|refinedSeeds|$ do
    $subnet \leftarrow expandSeed(G_H, refinedSeeds[i])$;
    if $|subnet| > minSize$ then
        $subnetList \leftarrow subnetList \cup subnet$;
    end if
end for
for $i := 1$ to $|subnetList|$ do
    $\mathbf{x}^* \leftarrow simulatedAnnealing(subnetList[i])$;
    $score \leftarrow |subnetList[i]|/\Theta(\mathbf{x}^*)$;
    if $score > th$ then
        output $subnetList[i]$;
    end if
end for
```

Algorithm 5 Sample a k-spine with a starting node v using a method adapted from the ESU-RAND algorithm.

Input: A given node v, an output parameter *kspine*, G_H

Output: Return false if there is no *k-spine* starting from v, else return true and an output parameter `kspine`.

```
function sampleKSpine($v, kspine, G_H$)
    $host \leftarrow getHost(v, G_H)$;                          ▷ $v \in G_{host}$
    $kspine[host] \leftarrow v$;
    $candidates \leftarrow \emptyset$
    for each vertex $u \in Adj(v)$ do                           ▷ $uv \in E_H$
        $host \leftarrow getHost(u, G_H)$;
        if $isOccupied(kspine, host)$ then                      ▷ $kspine[host]$ has a node.
            continue;
        end if
        if $v < u$ then
            $candidates \leftarrow candidates \cup \{u\}$;
        end if
    end for
    return expandKSpine($kspine, candidates, v, G_H$);
end function
```

Algorithm 6 Extend a starting node to a k-spine in G_H.

```
function EXPANDKSPINE($kspine, candidates, v, G_H$)
    if $isFull(kspine)$ then                                    ▷ $kspine[1..k]$ is full.
        return $true$;                              ▷ Succeed in finding a size-$k$ subgraph.
    end if
    while $candidates \neq \emptyset$ do
        Remove an arbitarily chosen vertex $w$ from $candidates$;
        for each vertex $u \in Adj_{excl}(w, candidates)$ do
            $host \leftarrow getHost(u, G_H)$;
            if $u > v \,\|\, isFree(kspine, host)$ then
                $candidates' \leftarrow candidates \cup \{u\}$;
            end if
        end for
        return EXPANDKSPINE($kspine \cup \{w\}, candidates', v, G_H$);
    end while
    return $false$;                              ▷ No size-$k$ subgraph starting from $v$.
end function
```

Algorithm 7 Collect all the starting nodes.

```
function COLLECTSTARTNODES($G_H$)                                ▷ $G_H = (V_H, E_H)$
    for each vertex $v \in V_H$ do
        if SAMPLEKSPINE($v, kspine, G_H$) then
            $startNodes \leftarrow startNodes \cup \{v\}$;
        end if
    end for
    return $startNodes$;
end function
```

Algorithm 8 Randomly sample a small subnet as a candidate of refined seeds.

```
function sampleSeed($G_H, startNodes$)
    $candidates \leftarrow \emptyset$;
    for $i := 1$ to $seedSize$ do
        if $i = 1$ or $candidates = \emptyset$ then
            $v \leftarrow rand(startNodes)$;        ▷ Select an arbitary starting node.
            sampleKSpine($v, kspine, G_H$);
        else
            $num = 1$;
            do
                if $num++ > numSpinetries$ then        ▷ The $numSpinetries$ is a user-specified parameter.
                    return $false$;
                end if
                $v \leftarrow rand(candidates)$;
                sampleKSpine($v, kspine, G_H$);
            while $(!isStronglyConnected(kspine, subnet))$
        end if
        $subnet \leftarrow subnet \cup kspine$;
        $candidates \leftarrow searchCandidates(G_H, startNodes, subnet)$;
    end for
    return $true$;
end function
```

Algorithm 9 Search for small densely connected subnets as refined seeds.

```
function searchSeeds($G_H$,$startNodes$)
    $i \leftarrow 0$;
    while $i < numseeds$ do        ▷ The $numseeds$ is a user-specified parameter.
        $subnet \leftarrow$ sampleSeed($G_H, startNodes$);
        $i \leftarrow i + 1$;
        $refinedSeeds \leftarrow refinedSeeds \cup subnet$;
    end while
    return $refinedSeeds$;
end function
```

Algorithm 10 Expand a refined seed to a *d-subnet*.

```
function EXPANDSEED($G_H, subnet$)
    $candidates \leftarrow searchCandidates(G_H, subnet)$;
    $i \leftarrow 0$;
    while $i < maxExt$ do
        $i \leftarrow i + 1$;
        if $candidates = \emptyset$ then
            return$false$;
        else
            $num = 1$
            do
                if $num{+}{+} > numSpinetries$ then
                    if $i >= minExt$ then
                        return $true$;
                    else
                        return $false$;
                    end if
                end if
                $v \leftarrow rand(candidates)$;
                SAMPLEKSPINE($v, kspine, G_H$);
            while $(!isStronglyConnected(kspine, subnet))$
        end if
        $subnet \leftarrow subnet \cup kspine$;
        $candidates \leftarrow searchCandidates(G_H, subnet)$;
    end while
    return $true$;
end function
```

Algorithm 11 Search for optimal internal tree nodes.

```
function SIMULATEDANNEALING($subnet, T_{max}, T_{min}, s$)
    $T_0 = T_{max}, i = 1, \mathbf{x} = (0, 0, 0, \cdots)$;
    while $i \leq K$ do
        $n \leftarrow 0$;
        $T_i \leftarrow T_0 - \frac{i \cdot (T_{max} - T_{min})}{K}$;
        while $n < N$ do
            $\mathbf{x}' \leftarrow moveToNeighbor(\mathbf{x})$;    ▷ Insert or delete an interaction of inner Modules.
            $\Delta\Theta \leftarrow \Theta(\mathbf{x}') - \Theta(\mathbf{x})$
            if $\Delta\Theta < 0$ then
                $\mathbf{x} \leftarrow \mathbf{x}'$;
            else if $rand(0, 1) < \exp(-\Delta\Theta/(sT_i))$ then
                $\mathbf{x} \leftarrow \mathbf{x}'$;
            end if
            $n \leftarrow n + 1$;
        end while
        $i \leftarrow i + 1$;
    end while
    return $\mathbf{x}$;
end function
```

3.3 Results and discussion

3.3.1 Test datasets

All experimentally determined interactions of five species were collected from the IntAct database (Kerrien *et al.*, 2012) as the test data of our evaluation (downloaded on February 10, 2014). The five species include *Homo sapiens* (hsa), *Caenorhabditis elegans* (cel), *Drosophila melanogaster* (dme), *Saccharomyces cerevisiae* (sce) and *Escherichia coli* (eco). The protein sequences were downloaded from a reviewed and manually annotated database, UniprotKB/Swiss-Prot (Magrane and Consortium, 2011). All-against-all protein sequence similarity are calculated with the program BLASTP (Altschul *et al.*, 1997), and these with *E-value* $\leq 1.0e^{-7}$ are selected as homologous proteins. The phylogenetic relationship of the five species was obtained from the NCBI taxonomy database (Federhen, 2012). With

Tab. 3.1: Proteins and interactions of our five observed species which are collected from the databases of IntAct and Uniprot/Swiss-Prot.

Species	Proteins	Interactions
H. sapiens	11258	47031
C. elegans	9302	15669
D. melanogaster	8725	27053
S. cerevisiae	5494	54163
E. coli	2985	14467

the real-world knowledge of the five species (see in Tab. 3.1), we performed *LocalAli* and several existing algorithms on 26 real datasets including all possible combinations of the test species. To test the statistical significance of our alignment results, *LocalAli* were also tested on 1040 random datasets (40 random k-layer graphs for each combination). All these random k-layer graphs remain the same number of interactions and edges as the real k-layer graphs. Moreover, high-quality associated gene ontology annotations which were downloaded from the *Uniprot-GOA* database (on March 14, 2014) and a reference dataset CORUM (Ruepp *et al.*, 2010) were used to help assess the biological quality of the results.

3.3.2 Experimental setup

We have implemented *LocalAli* in C++ using the *LEMON Graph Library* (Dezső *et al.*, 2011) version 1.2.3 and OpenMP (Chapman *et al.*, 2007). The implementation supports multicore parallelism in the search for high-scoring *d-subnets*. *LocalAli* provides many user-specified parameters that are used to determine the topological feature of target regions and the scoring scheme, such as $seedSize$, $minExt$, $maxExt$, α and β. The default values are now $seedSize = 2$, $minExt = 3$, $maxExt = 13$, $\alpha = 0.2$ and $\beta = 2$. More elaborate information about the other specific parameters are described in Tab. 3.2. We first performed *LocalAli* 20 times with a single core, and then ran it 20 times again with 16 cores in parallel on each real dataset. And the best, average and worst results were applied to assess the performance. *NetworkBlast-M* were subsequently executed on the same datasets with the extension scheme of *relaxed order*. In addition, three pairwise local alignment tools *NetworkBlast*, *AlignNemo* and *MaWISh* were applied to all of our *2-way* alignments. However, another two multiple local alignment tools *Graemlin* and *CAPPI* were not taken into consideration in our assessment, as *Graemlin* did not compile successfully (the current available version is outdated), and *CAPPI* was only compatible with particularly designed data.

The reported alignments might be overlapped in a quite different degree, because the extension of two refined seeds are completely independent between each other. For a fair comparison of the quality of alignment results from different alignment tools, we filtered out highly redundant solutions (>0.5) from the results. We removed them in two steps: 1) sort all the alignments from the highest to the lowest, according to their alignment scores; 2) iteratively visit the elements and remove all other alignments intersecting it by more than 50%. Given two alignments A and B, the intersection level is calculated as the number of shared proteins $|A \cap B|$ over $\min\{|A|, |B|\}$. All experiments mentioned in the following parts were carried out on an Intel(R) Xeon(R) CPU X5550 with 2.67GHz.

3.3.3 Cross-validation

We assessed the quality of the alignment results in four ways: coverage, consistency, prediction of protein functions and prediction of protein complexs. Coverage indicates the amount of input data the algorithm can explain. Consistency implies the functional coherence of identified *d-subnets*. Our goal is to find a series of *d-subnets* that have a good consistency while reporting as many *d-subnets* as possible (i.e. a high coverage) within reasonable time. Consistency can be well accomplished by sacrificing coverage and vice versa. Further, to determine how much our alignment results agree with known biological knowledge, *LocalAli* was also applied to predict protein functions and protein complexs. Finally, we compared the performance of the alignment tools in terms of scalability and running time.

Coverage and consistency

The coverage was measured in two ways. First, we measured it by the number of reported *d-subnets* (or hits) after the elimination of redundant solutions. Second, the coverage was measured by the *percentage of proteins value* (PPV), which calculated the percentage of proteins covered by the identified hits over all the proteins. We performed functional enrichment analyses based on Gene Ontology annotation data (Ashburner *et al.*, 2000) to assess the functional coherency of each subnetworks in the reported hits. A powerful package *GO-TermFinder* (Boyle *et al.*, 2004) was used to calculate the statistical significance of GO annotations. Those subnetworks that had one or more enriched *GO* terms (i.e. corrected p-value$\leq$0.01) were regarded as *functionally coherent subnetworks* (FCS) and likely to be functional modules. Therefore, we measured consistency by the number of reported FCS and the portion of FCS over all identified subnetworks (i.e. precision). All the results of the

26 real datasets which included 10 two-way alignments, 10 three-way alignments, 5 four-way alignments and 1 five-way alignment were analyzed as shown in Tab. 3.3–3.28.

In comparison with *NetworkBlast* (NB), *NetworkBlast-M* (NBM), *AlignNemo* (AN) and *MaWISh* (MW), *LocalAli* (LA) basically outperforms all existing algorithms in the aspect of coverage. As shown in Tab. 3.3–3.12, for instance, LA reported 477, 408 and 348 hits in the best, average, and worst case in the two-way alignment of hsa-cel, while merely 367, 160 and 252 hits were reported by NB, NBM, and MW. The worst PPV value of LA was also upto 10.8%, which was obviously more advanced than that of other algorithms. This was not a unique instance in the 10 two-way alignments as shown in Tab. 3.5–3.11. However, LA reported less hits than NB, NBM, MW in the 2-way alignment of hsa-dme. It was because that the threshold of the alignment score was too high for this dataset. More than 90% d-subnets were filtered out. Comparing with NBM in multiple alignments, LA also reported more hits and higher PPV in many cases such as the hsa-cel-dme alignment (Tab. 3.13), the hsa-dme-eco (Tab. 3.17), and the four-way alignment of hsa-cel-dme-eco (Tab. 3.24). And NBM failed to report any hit in many other multiple alignments such as the three-way alignment of hsa-dme-sce (Tab. 3.16) and hsa-sce-eco (Tab. 3.18) because of its limited scalability.

In the aspect of consistency, *LocalAli* also identified much more FCS than NB, NBM, AN, MW in both of the pairwise and multiple alignments, meanwhile retained a high precision. For instance, LA found 1628, 1535 and 1402 FCS in the best, average, and worst case in the hsa-eco alignment (Tab. 3.6), whereas only 5, 31, 81 and 79 were found by NB, AN, NBM and MW, respectively. Moreover, the worst success rate of identifying FCS was also upto 99.4%, which was higher than all other algorithms. Similar results could be found in many other pairwise alignments such as the hsa-eco alignment (Tab. 3.6), the cel-dme (Tab. 3.7), and the cel-sce (Tab. 3.8). In the alignment of multiple networks, it shows that LA has a competitive advantage in FCS over NBM, as well as a comparable precision. For example, it found 360 FCS in the worst case of the hsa-dme-eco alignment (Tab. 3.17) which was five time as many as these reported by NBM. At the same time, it got the same average precision as NBM. More importantly, LA successfully aligned many datasets, such as the hsa-cel-sce (Tab. 3.14) and the hsa-dme-sce (Tab. 3.16), in which NBM however reached its limitation. NBM nevertheless got a higher precision than LA in the cel-dme-eco alignment (Tab. 3.20).

Moreover, we executed LA on random datasets of each possible combination of input species to verify the statistical significance of our results. As a result, we found all these data about hits, FCS and precision were non-random and statistically significant. As shown

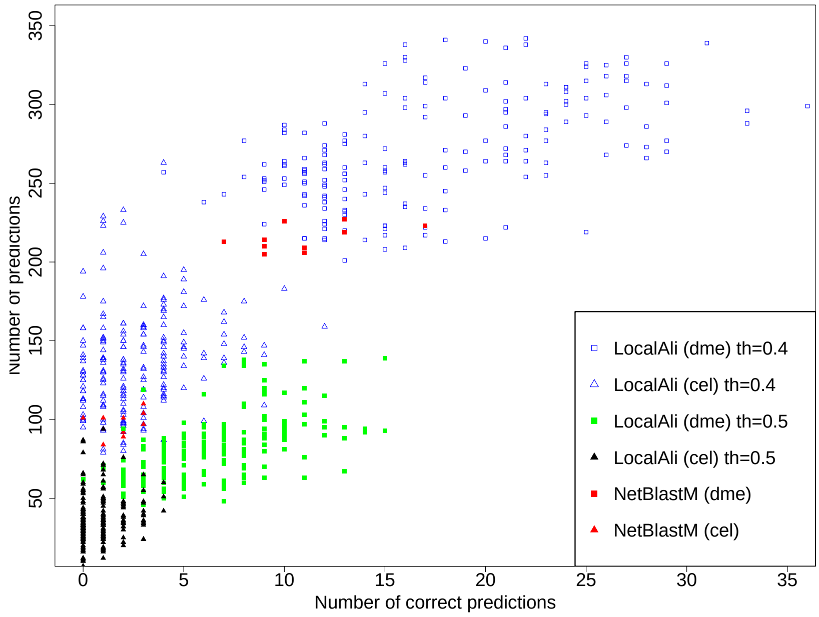


Fig. 3.9: 10-fold cross-validation for function predictions on the cel-dme alignment using *LocalAli* and *NetworkBlast-M* (NetBlastM). The parameter of threshold (th) is used to filter out *d-subnets* with a lower alignment score.

in Fig. 3.10–3.19, the random results (blue triangles) are very far away from the real-world data (red points). It indicates that these results of hits and FCS in real-world data are unlikely to happen in the random data. Further, the results show that most of the red points stand quite close to the oblique line while the blue triangles are far away from the line. This evidence implies that the precision of LA is also statistically significant because the closer the points are, the higher precision they have. There is no figure illustrating multiple alignments of the random datasets, since LA can hardly find any *d-subnet* in the multiple alignments of random datasets.

Prediction of protein functions

Proteins that function in a pathway or structural complex are functionally related. It spontaneously leads us to the tentative functional assignments, which can be called by applying the method of *annotation transfer* (Sharan, 2005). Given a set of proteins, we predicted new protein functions whenever all the following four requirements were fulfilled: (i) the set of proteins was significantly enriched for a particular GO annotation (corrected p-value≤ 0.01); (ii) at least three of the proteins were annotated with the GO annotation;

(iii) the percentage of proteins annotated with this GO annotation over all characterized proteins was >0.5; (iv) the GO annotation was at a GO level of three or higher in the GO tree. All the remaining proteins will be considered to have the annotation if all the four demands are satisfied. If there are several GO annotations fulfilling the four requisites, just the one with the lowest corrected *p*-value will be applied for the prediction. According to the four requirements, all the cel-dme alignments that reported by NB, NBM, AN and LA were analyzed for predicting gene-associated ontology with the aspect of *biological process*. As a result, LA recognized 214.9 predictions of new GO annotations for proteins in *cel*, 286.2 predictions for proteins in *dme* in the average case. In contrast, NB reported 26 predictions in *cel*, 31 predictions in *dme*; AN found 18 in *cel*, 55 in *dme*; NBM found 165 in *cel*, 229 in *dme*.

To validate the quality of the predicted functions, we estimated the success rate of our predictions using a method of 10-fold cross-validation, in which we equally separated the annotation data into 10 parts, iteratively hid one part and used the remaining data to predict the held-out annotations (Sharan, 2005). The prediction will be considered correct if the protein has some true annotation that lies on a path in the gene ontology tree from the root to a leaf that visits the predicted annotation. According to this rule, the number of correct predictions obtained from NBM and LA were illustrated in Fig. 3.9 on the 10-fold cross-validation. The blue points of LA are much more than that of NBM in the figure since all 20 × 10 samples are plotted. Then, we tried it again after increasing the threshold to 0.5 (th=0.5) to verify whether our scoring scheme is indeed closer to the truth of biology. As indicated in the figure, LA was preferable to NBM in predicting the correct protein functions with th=0.4 for both *cel* and *dme*, though it also made some false positive points (i.e. these tended to travel to the left upper corner) for *cel*. In the case of th=0.5, it was more clear to see that LA had similar number of correct functions with NBM by using less number of predictions. The average success rates of NBM were 1.83 and 5.05% for *cel* and *dme*, respectively. They were less than that of LA with th=0.4, which were 1.96 and 6.35%. They increased to 2.26 and 7.67% when th=0.5. To sum up, we can conclude that LA, in comparison with NBM, is more precise in the prediction of functional annotations, and the higher-scoring *d-subnets* are more favorable for the prediction of protein functions.

Validation of predicted functional modules

To validate the predicted functional modules, we collected a benchmark set of protein complexs that belonged to *hsa* as annotated in *CORUM* (Ruepp *et al.*, 2010) (released in February 2012). Overall, there were 1283 protein complexs consisting of three or more proteins in our benchmark set. Then, we compared these identified conserved subnetworks with the benchmark set of complexes. Let S represent proteins of a conserved subnetwork, C be proteins of a known protein complex. We will consider S to be a successful prediction of C if and only if two requirements are fulfilled: (i) $|S \cap C| \geq 3$; (ii) $\frac{|S \cap C|}{\max\{|C|,|S|\}} \geq 0.2$. If S corresponds to a protein complex in *CORUM*, it will be a *pure module*. As a result, NBM successfully recognized 29 *pure modules* from the human PPI network with a success rate of 11.9%. In contrast, LA recognized 55.8 *pure modules* on average with a success rate of 17.4%. It indicates that LA is more accurate than NBM in recognizing biologically meaningful modules.

Scalability

Scalability is a bottleneck problem that limits the applications of existing alignment tools. Many pairwise alignment tools attempting to search for densely connected subgraphs in an alignment graph are difficult to extend to multiple networks because alignment nodes in the graphs will grow exponentially when the number of networks increases. In comparison with other algorithms in our tests, LA demonstrated the best performance in the aspect of scalability. It was the only algorithm that favorably ran on all the 26 datasets. In contrast, NBM encountered its limitation when some network had a protein connected to a large number of other proteins, such as PPI networks of *sce* in Tab. 3.1.

Running time

Parallelization is a key technique that enables LA to speed up. We first performed LA on each real dataset 20 times with a single core, and then ran it 20 times again with 16 cores in parallel. In comparison with NB, NBM, AN and MW, LA was the most favorable alignment tool in the pairwise alignments. As shown in Tab. 3.3–3.12, LA finished all the pairwise alignments within several minutes (≤ 3) using a single core. The parallelism yielded a speedup of LocalAli. Generally, it could be three to six times faster in the pairwise alignments. In contrast, NB spent about 5h on the hsa-cel alignment, 10h on hsa-dme, >24h on hsa-sce and 0.5h on dme-sce. MW spent 15 min on hsa-dme, 26 min on hsa-sce. Although, NB, NBM, AN and MW were faster than LA in some alignment such as hsa-eco and cel-eco, they accomplished the advancement with a serious sacrifice of coverage. In the multiple alignment, NBM was faster than LA in many cases but with a smaller number of reported hits and a limited scalability (Tab. 3.13–3.28).

Tab. 3.2: Parameters used for k-way ($k \in \{2, 3, 4, 5\}$) alignment on all possible datasets.

aligners	*datasets*	*threshold*	*numseeds*	*seedtries*	*seedsize*	*extdist1*	*extdist2*	*minext*	*maxext*
LocalAli	2-way	0.4	1	10	2	1	2	3	13
	3-way	0.2	2	20	2	2	2	3	13
	4-way	0.3	3	10	2	2	3	3	13
	5-way	0.3	3	10	2	2	3	3	13

Tab. 3.3: Coverage, consistency and running time on Human(hsa) and Worm(cel). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
hsa-cel	Hits	477	408.25	348	367	/	160	252
	PPV(%)	11.94	11.3	10.76	5.19	/	7.58	5.3
	FCS	946	807.8	681	729	/	319	454
	Precision(%)	99.6	98.9	97.85	99.4	/	99.7	90
	Time(s)× 1	59.19	62.33	65.7	16260	/	24	276
	Time(s)× 16	15.43	16.4	18.1	/	/	/	/

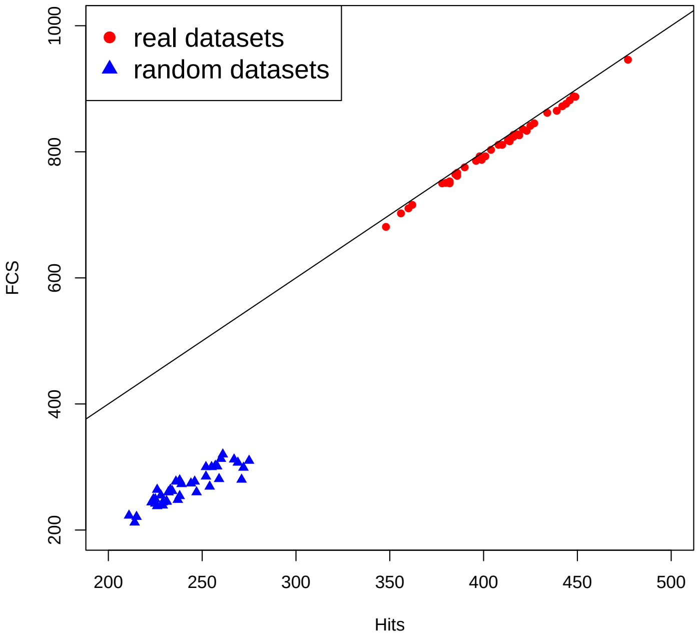


Fig. 3.10: A plot of Hits vs FCS on Human and Worm. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.4: Coverage, consistency and running time on Human(hsa) and Fruit Fly(dme). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
hsa-dme	Hits	77	54.875	38	813	/	323	295
	PPV(%)	5.57	4.25	3.2	9.79	/	13	8.02
	FCS	151	107.2	75	1562	/	646	570
	Precision(%)	100	97.7	93.7	96.1	/	100	96.6
	Time(s)× 1	54.8	56.84	59.06	36750	/	55	889
	Time(s)× 16	16.34	16.97	17.61	/	/	/	/

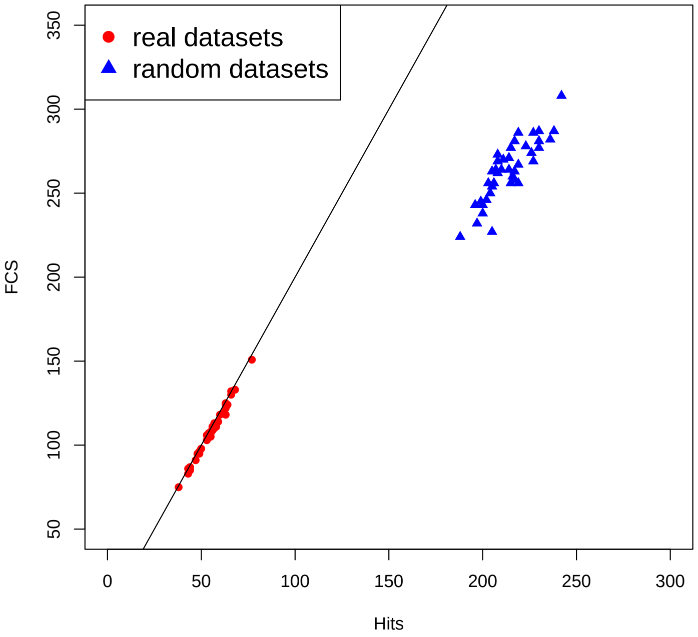


Fig. 3.11: A plot of Hits vs FCS on Human and Fruit Fly. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.5: Coverage, consistency and running time on Human(hsa) and Yeast(sce). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	LA(best)	LA(average)	LA(worst)	NB	AN	NBM	MW
hsa-sce	Hits	889	863.55	836	/	/	/	322
	PPV(%)	30.05	29.6	29.07	/	/	/	13.3
	FCS	1633	1590.4	1529	/	/	/	613
	Precision(%)	93.5	92.1	91	/	/	/	95.2
	Time(s)× 1	126	129.7	135.4	$> 24hours$	/	/	1587
	Time(s)× 16	32.04	33.38	34.61	/	/	/	/

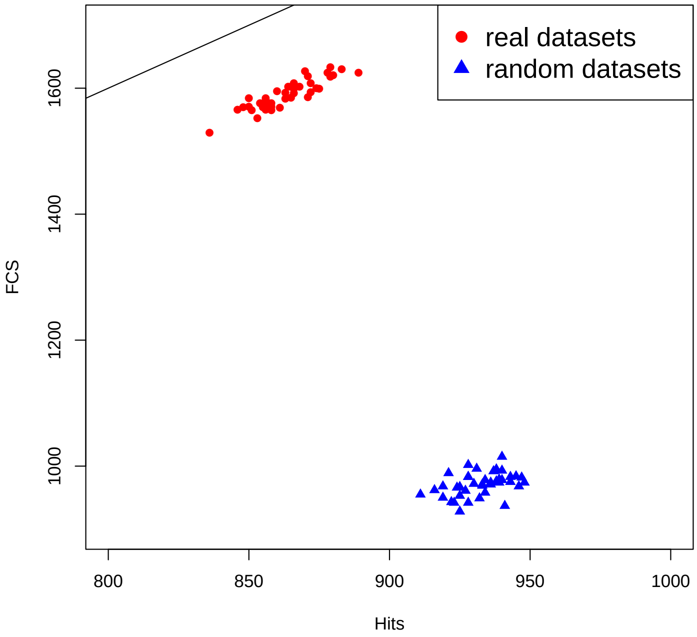


Fig. 3.12: A plot of Hits vs FCS on Human and Yeast. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.6: Coverage, consistency and running time on Human(hsa) and E. coli(eco). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
hsa-eco	Hits	815	769.325	701	3	17	42	49
	PPV(%)	9.6	9.43	9.28	0.197	2	3.17	1.36
	FCS	1628	1534.85	1402	5	31	81	79
	Precision(%)	100	99.8	99.4	83.4	91.2	96.4	80.6
	Time(s)× 1	76.39	81.05	83.93	10	/	6	1
	Time(s)× 16	20.27	21.01	21.82	/	4	/	/

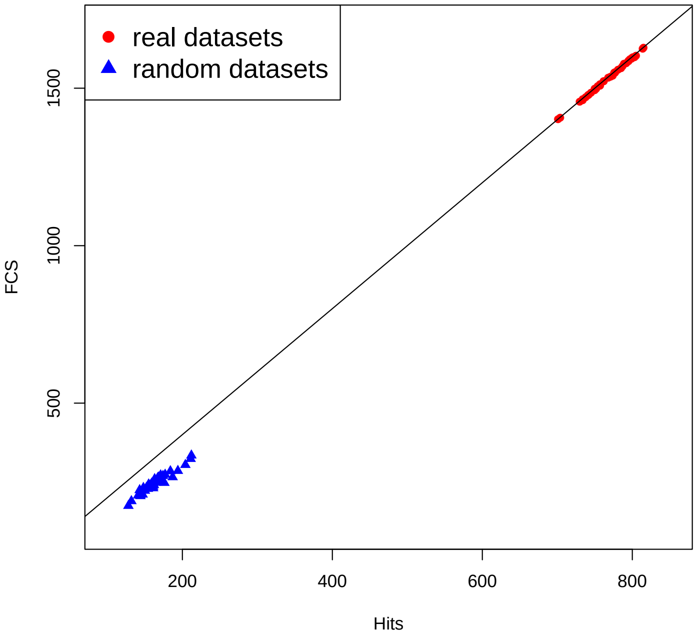


Fig. 3.13: A plot of Hits vs FCS on Human and E. coli. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.7: Coverage, consistency and running time on Worm(cel) and Fruit Fly(dme). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
cel-dme	Hits	402	322.65	276	16	52	60	156
	PPV(%)	6.51	6.26	6	0.871	3.03	3.41	2.15
	FCS	794	632.35	547	28	80	119	230
	Precision(%)	99.1	98	96.15	87.5	76.9	99.2	73.7
	Time(s)× 1	30.57	31.63	32.71	17	/	5	4
	Time(s)× 16	9.652	10.41	11.12	/	11	/	/

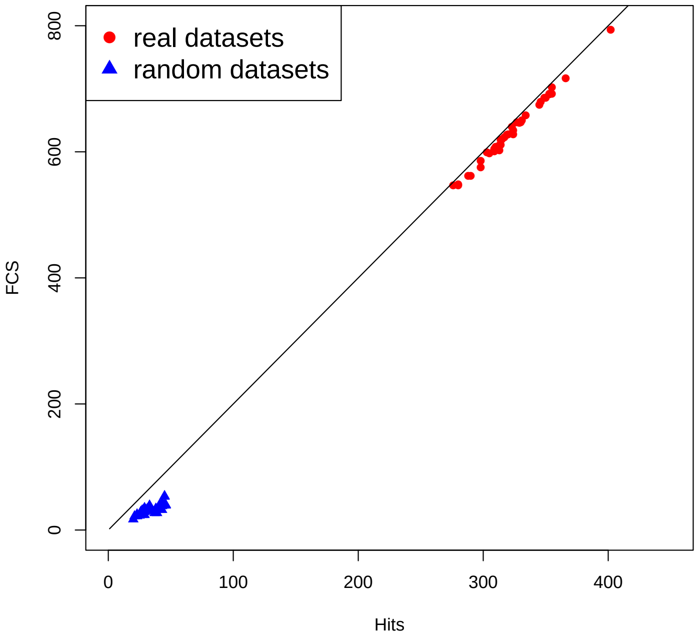


Fig. 3.14: A plot of Hits vs FCS on Worm and Fruit Fly. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.8: Coverage, consistency and running time on Worm(cel) and Yeast(sce). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
cel-sce	Hits	245	217.575	190	3	13	/	146
	PPV(%)	6.83	6.36	5.94	0.378	1.76	/	2.99
	FCS	425	380.725	331	6	19	/	212
	Precision(%)	91.1	87.5	84.4	100	73.1	/	72.6
	Time(s)× 1	116.9	120.3	125	59	/	/	10
	Time(s)× 16	24.42	26.71	28.52	/	7	/	/

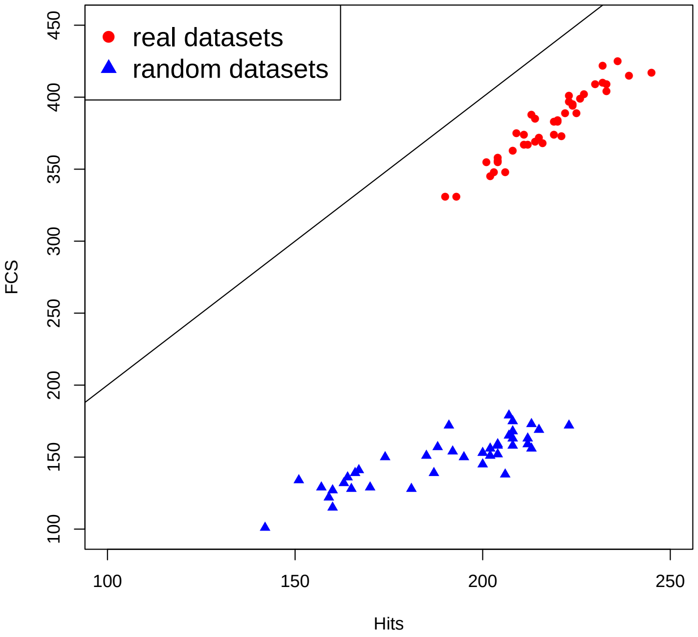


Fig. 3.15: A plot of Hits vs FCS on Worm and Yeast. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.9: Coverage, consistency and running time on Worm(cel) and E. coli(eco). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
cel-eco	Hits	74	64.675	52	0	4	6	5
	PPV(%)	1.98	1.71	1.45	0	0.456	0.749	0.179
	FCS	144	122.15	98	0	8	12	10
	Precision(%)	97.3	94.4	90.35	/	100	100	100
	Time(s)× 1	50.8	53.22	54.67	4	/	1	0
	Time(s)× 16	11.5	11.95	13.44	/	1	/	/

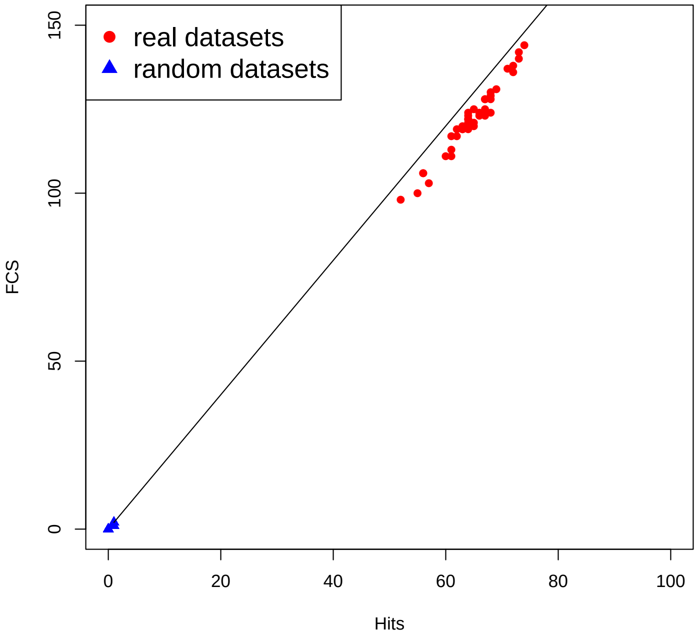


Fig. 3.16: A plot of Hits vs FCS on Worm and E. coli. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.10: Coverage, consistency and running time on Fruit Fly(dme) and Yeast(sce). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
dme-sce	Hits	734	702.525	670	72	39	/	314
	PPV(%)	15.64	15.3	14.98	2.5	3	/	5.86
	FCS	1382	1315.825	1239	139	55	/	438
	Precision(%)	94.85	93.6	92.2	96.5	70.6	/	69.8
	Time(s)× 1	130.3	144.3	201.7	1558	/	/	40
	Time(s)× 16	30.33	32.38	35.69	/	43	/	/

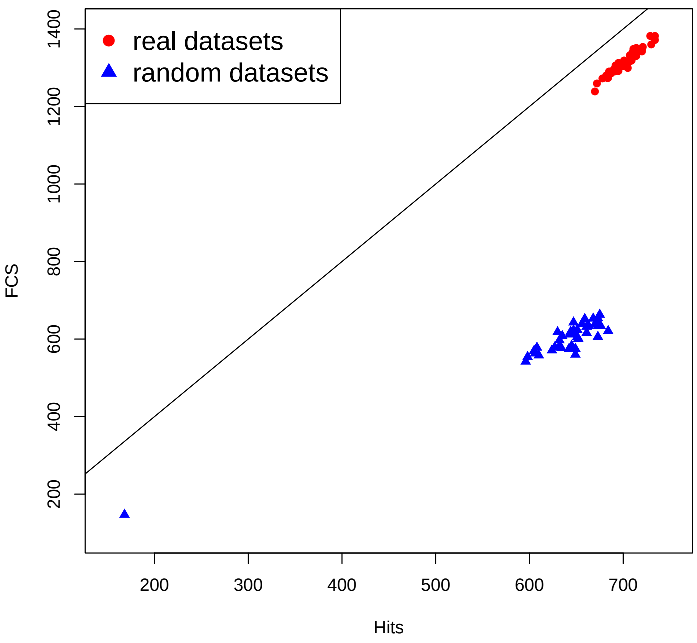


Fig. 3.17: A plot of Hits vs FCS on Fruit Fly and Yeast. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.11: Coverage, consistency and running time on Fruit Fly(dme) and E. coli(eco). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
dme-eco	Hits	256	182.875	136	1	2	9	7
	PPV(%)	3.71	3.48	3.21	0.0854	0.709	1.16	0.256
	FCS	499	350.975	255	2	4	18	8
	Precision(%)	97.75	95.9	93.55	100	100	100	57.1
	Time(s)× 1	58.95	61.5	63.98	6	/	1	0
	Time(s)× 16	14.61	15.28	18.28	/	1	/	/

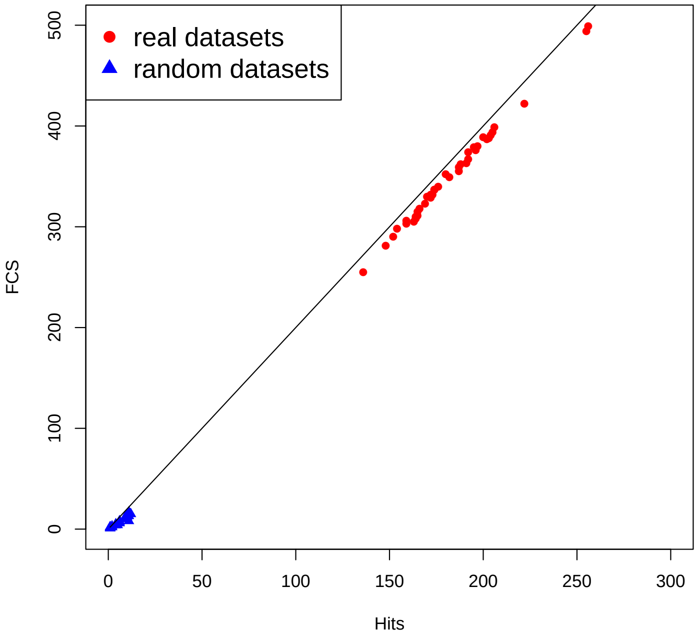


Fig. 3.18: A plot of Hits vs FCS on Fruit Fly and E. coli. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.12: Coverage, consistency and running time on Yeast(sce) and E. coli(eco). The five algorithms of LocalAli, NetworkBlast, AlignNemo, NetworkBlastM and MaWISh are shortly represented by LA, NB, AN, NBM and MW, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NB*	*AN*	*NBM*	*MW*
sce-eco	Hits	134	115.05	90	21	26	/	168
	PPV(%)	14.1	13.2	12.02	2.02	3.72	/	5.78
	FCS	262	218.325	165	37	44	/	174
	Precision(%)	97.75	94.8	91.5	88.1	84.6	/	51.8
	Time(s)× 1	57.74	59.43	60.63	91	/	/	4
	Time(s)× 16	20.63	21.16	21.81	/	20	/	/

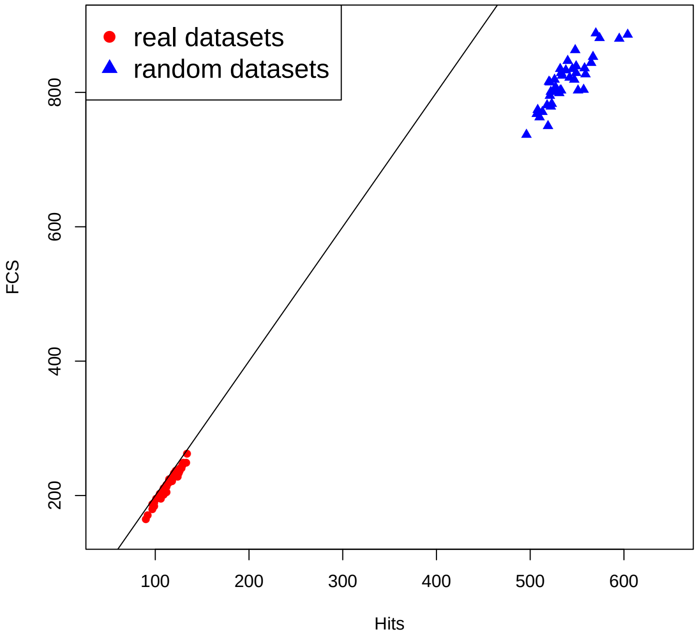


Fig. 3.19: A plot of Hits vs FCS on Yeast and E. coli. In order to test the statistical significance of LocalAli's result, we also tested LocalAli on forty randomly generated datasets based on a null model. Here, red points represent the result on real datasets and blue points represent the result on random datasets. The closer the point to the oblique line is, the higher its precision is.

Tab. 3.13: Coverage, consistency and running time on Human(hsa),Worm(cel) and Fruit Fly(dme). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-dme	Hits	395	328.2	283	243
	PPV(%)	7.79	7.318	6.85	7.42
	FCS	1156	949.1	820	726
	Precision(%)	97.8	96.4	94.4	99.6
	Time(s)× 1	3143.79	3292.036	3390.03	227
	Time(s)× 16	265.702	284.66345	304.984	／

Tab. 3.14: Coverage, consistency and running time on Human(hsa),Worm(cel) and Yeast(sce). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-sce	Hits	2304	2129	1962	／
	PPV(%)	14.13	13.89	13.61	／
	FCS	6481	5994	5509	／
	Precision(%)	94.8	93.9	93	／
	Time(s)× 1	1029.38	1055.189	1073.14	／
	Time(s)× 16	247.493	257.9092	270.716	／

Tab. 3.15: Coverage, consistency and running time on Human(hsa),Worm(cel) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-eco	Hits	14	11.55	9	14
	PPV(%)	0.72	0.5588	0.41	1.18
	FCS	42	32.95	24	38
	Precision(%)	100	95	86.7	90.5
	Time(s)× 1	1047.39	1161.1855	1624	5
	Time(s)× 16	865.333	1119.06825	1272.58	⁄

Tab. 3.16: Coverage, consistency and running time on Human(hsa),Fruit Fly(dme) and Yeast(sce). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-dme-sce	Hits	3403	3268	2930	⁄
	PPV(%)	22.15	21.96	21.7	⁄
	FCS	9289	8902	8022	⁄
	Precision(%)	91.4	90.8	90.2	⁄
	Time(s)× 1	2200.25	2281.7855	2383.87	⁄
	Time(s)× 16	204.366	211.43815	219.9	⁄

Tab. 3.17: Coverage, consistency and running time on Human(hsa),Fruit Fly(dme) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-dme-eco	Hits	194	159	126	25
	PPV(%)	3.62	3.352	3.11	1.83
	FCS	561	458.4	360	72
	Precision(%)	97.5	96	94.7	96
	Time(s)× 1	2405.05	2593.7125	2816.24	16
	Time(s)× 16	1350.6	1746.3	1958.26	⁄

Tab. 3.18: Coverage, consistency and running time on Human(hsa),Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-sce-eco	Hits	3947	3924	3894	/
	PPV(%)	14.26	14.07	13.88	/
	FCS	11517	11430	11353	/
	Precision(%)	97.4	97.1	96.5	/
	Time(s)× 1	1677.12	1800.295	1924.56	/
	Time(s)× 16	271.142	295.1718	316.262	/

Tab. 3.19: Coverage, consistency and running time on Worm(cel),Fruit Fly(dme) and Yeast(sce). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
cel-dme-sce	Hits	1136	1011	871	/
	PPV(%)	7.2	7.049	6.87	/
	FCS	3271	2910	2508	/
	Precision(%)	96.9	95.9	94.9	/
	Time(s)× 1	931.271	1852.4552	2588.25	/
	Time(s)× 16	192.827	199.7882	210.676	/

Tab. 3.20: Coverage, consistency and running time on Worm(cel),Fruit Fly(dme) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
cel-dme-eco	Hits	6	4.525	3	6
	PPV(%)	0.41	0.2235	0.16	0.62
	FCS	16	10.88	6	17
	Precision(%)	93.3	79.5	58.3	94.4
	Time(s)× 1	1087.89	1225.4395	1339.38	1
	Time(s)× 16	226.042	272.53595	303.416	/

Tab. 3.21: Coverage, consistency and running time on Worm(cel),Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
cel-sce-eco	Hits	505	460.9	133	/
	PPV(%)	4.09	3.937	3.09	/
	FCS	1464	1332	384	/
	Precision(%)	97.5	96.3	94.8	/
	Time(s)× 1	3277.73	3482.7515	3655.78	/
	Time(s)× 16	250.955	252.15815	275.12	/

Tab. 3.22: Coverage, consistency and running time on Fruit Fly(dme),Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
dme-sce-eco	Hits	3020	2774	2406	/
	PPV(%)	7.48	7.352	7.25	/
	FCS	8986	8252	7169	/
	Precision(%)	99.4	99.2	98.8	/
	Time(s)× 1	2316.76	2450.2115	2534.97	/
	Time(s)× 16	204.455	211.4621	218.722	/

Tab. 3.23: Coverage, consistency and running time on Human(hsa),Worm(cel),Fruit Fly(dme) and Yeast(sce). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-dme-sce	Hits	250	204.5	163	/
	PPV(%)	7.33	6.806	6.06	/
	FCS	981	789.9	634	/
	Precision(%)	98.1	96.5	95	/
	Time(s)× 1	1389.4	1473.8075	1575.09	/
	Time(s)× 16	363.119	445.35695	514.922	/

Tab. 3.24: Coverage, consistency and running time on Human(hsa),Worm(cel),Fruit Fly(dme) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-dme-eco	Hits	100	83.22	56	31
	PPV(%)	2.85	2.589	2.23	1.79
	FCS	386	318.4	211	113
	Precision(%)	97.9	95.6	93.4	91.1
	Time(s)× 1	2576.42	2919.1695	3333.32	152
	Time(s)× 16	1721.29	2137.6375	2924.53	/

Tab. 3.25: Coverage, consistency and running time on Human(hsa),Worm(cel),Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-sce-eco	Hits	135	112.4	82	/
	PPV(%)	3.65	3.342	2.87	/
	FCS	500	419.5	301	/
	Precision(%)	95.6	93.4	90.3	/
	Time(s)× 1	506.98	579.72535	642.928	/
	Time(s)× 16	157.4	197.66365	259.277	/

Tab. 3.26: Coverage, consistency and running time on Human(hsa),Fruit Fly(dme),Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-dme-sce-eco	Hits	245	217.9	190	/
	PPV(%)	7.02	6.637	0	/
	FCS	887	792.1	689	/
	Precision(%)	92.6	90.9	88.4	/
	Time(s)× 1	706.675	850.19905	954.841	/
	Time(s)× 16	261.356	375.35105	544.023	/

Tab. 3.27: Coverage, consistency and running time on Worm(cel),Fruit Fly(dme),Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
cel-dme-sce-eco	Hits	112	88.82	60	⁄
	PPV(%)	2.49	2.3	2.05	⁄
	FCS	447	350.1	234	⁄
	Precision(%)	99.8	98.5	95.8	⁄
	Time(s)× 1	760.734	884.5283	968.757	⁄
	Time(s)× 16	125.965	160.45365	204.812	⁄

Tab. 3.28: Coverage, consistency and running time on Human(hsa), Worm(cel), Fruit Fly(dme), Yeast(sce) and E. coli(eco). The algorithms of LocalAli and NetworkBlastM are shortly represented by LA and NBM, respectively.

Dataset	Measure	*LA*(best)	*LA*(average)	*LA*(worst)	*NBM*
hsa-cel-dme-sce-eco	Hits	78	64	34	⁄
	PPV(%)	3.17	2.848	2.09	⁄
	FCS	381	311.5	164	⁄
	Precision(%)	98.9	97.3	95.5	⁄
	Time(s)× 1	8457.52	10344.525	12424.8	⁄
	Time(s)× 16	3867.05	6535.725	9886.51	⁄

Chapter 4

An Algorithm for Multiple Global Network Alignment

As what we mentioned before, most of previous algorithms for *multiple global network alignment* encountered some limitations. For example, *Graemlin 2.0*[1] (Flannick *et al.*, 2008) requires additional training data of known alignments (i.e. Orthology Groups from KEGG database) to learn its many network dependent parameters and a phylogenetic relationship of involved species, which means it can not be applied to species without known alignments or a phylogenetic tree. As a remedy for these limitations, we present a fast and accurate tool *NetCoffee*, which addresses the problem of global alignment of multiple networks. In this chapter, we first introduce the *NetCoffee* algorithm with a specific example. Then we analyze the computational complexity of this algorithm. Finally, our attention focuses on the detail of test data sets, experimental setup and performance evaluation in comparison with previous algorithms.

4.1 The *NetCoffee* algorithm

4.1.1 Overview

The algorithm implemented in *NetCoffee* has four main steps. First, we construct k PPI networks with experimentally determined interactions which are downloaded from public databases. Additionally, we build a library of bipartite graphs for each pair of involved species by joining homologous proteins which have sufficient sequence similarity. Then, an integrated score is calculated for each edge in the bipartite graphs with two conservation measures *topology similarity* and *sequence similarity*. Topology-based score and sequence-based score are calculated by *triplet comparison* and *log-ratio model*, respectively. Third,

---

[1] Graemlin 2.0 are used for both *multiple local alignment* and *multiple global alignment*. Here, we refer it to the latter one. See more information at `http://graemlin.stanford.edu/download.php`.

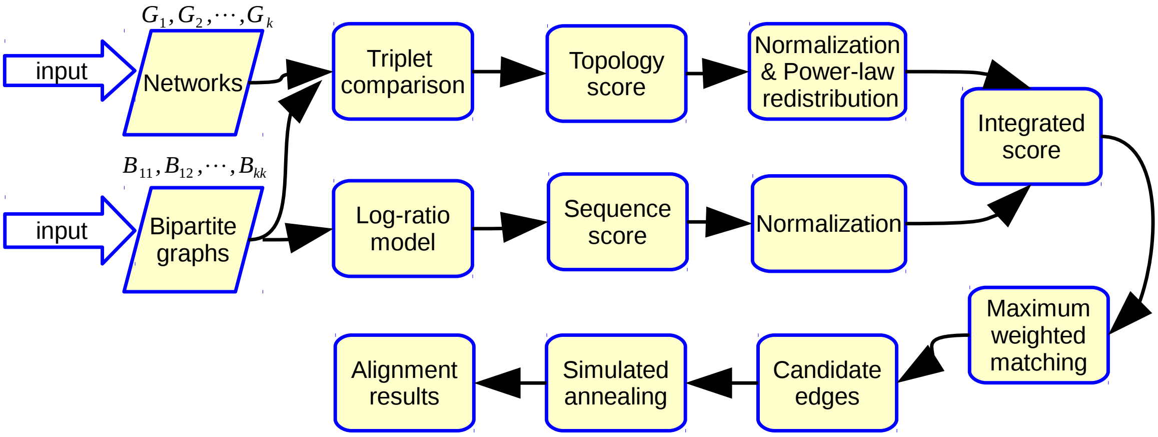


Fig. 4.1: An overview of the workflow of *NetCoffee* in *multiple global network alignment*.

a search space of candidate protein pairs which are likely to match with each other is collected via the method of *maximum weighted matching*. Last, we use a *simulated annealing* (SA) approach (Kirkpatrick *et al.*, 1983) to search for an optimal or near-optimal global alignment. A workflow of our algorithm is illustrated in Fig.4.1.

4.1.2 Generating a bipartite graph library

Let $\{G_1, G_2, \cdots, G_k\}$ represent a set of k PPI networks. Each network $G_i = (V_i, E_i)$ is an unweighted graph, where V_i represents a set of proteins and E_i a set of interactions. We build a bipartite graph library that contains graphs $B_{ij} = (V_i \cup V_j, E_{ij}), i \leq j, i, j \in \{1, 2, \cdots, k\}$. We use the term *edges* to refer to elements in E_{ij}, and the term *interactions* to refer to elements in E_i. To determine the sets E_{ij}, we perform an all-against-all sequence comparison with the program BLASTP (Altschul *et al.*, 1997) for each pair of species, including pairs of the same species like human-human. Then, the set of $\binom{k+1}{2}$ bipartite graphs can be constructed by simply joining protein pairs $v_1 \in V_i$, $v_2 \in V_j$ that have an e-value $\leq 10^{-7}$ by edges $(v_1, v_2) \in E_{ij}$. In the bipartite graph B_{ii} of the same species, we add only edges for pairs of two distinct proteins $v_1 \neq v_2$ to E_{ii}. This allows us to construct match-sets that might reflect duplication events within a species and hence exhibit the functional relation within a species.

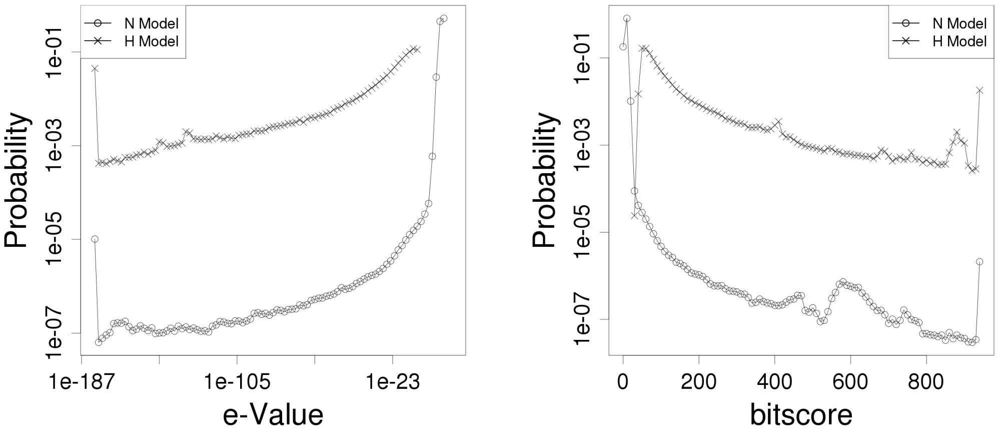


Fig. 4.2: Empirical distributions of the BLASTP e-value and bitscore for estimating sequence scores of edges in the bipartite graph library. Data of the homology model and the null model was sampled from five eukaryotic species: human, mouse, fruit fly, nematode, and yeast.

4.1.3 Integration of two conservation measures

With the information of network topology and protein sequences, we develop a linear scoring model that assigns a weight to each edge of the bipartite graphs. The development of the scoring model is intuitively guided by two basic assumptions: 1) functionally conserved proteins are likely to have sequence similarity; and 2) interactions among orthologous proteins are likely to be conserved across species. Likewise, our scoring model consists of two independent parts for sequence and topology similarity. Given an edge $e = (v_1, v_2)$, we use $S_r(v_1, v_2)$ to denote a normalized sequence score and $S_t(v_1, v_2)$ to denote a normalized topology score for proteins v_1 and v_2. A combined score for the edge e is calculated with $S(v_1, v_2) = (1 - \alpha)S_r(v_1, v_2) + \alpha S_t(v_1, v_2)$ where α is a user-defined parameter controlling how much of the topology score contributes to $S(v_1, v_2)$.

To compute the sequence-based score $S_r(v_1, v_2)$ for a pair of proteins v_1 and v_2, we adopt a previously introduced log-ratio scoring function that uses distributions of e-values in two models, the homology model H and the null model N (Flannick *et al.*, 2006). The null model includes all pairs of proteins from the input networks, whereas the homology model includes only pairs of proteins with e-value $\leq 10^{-7}$. Given the distributions of e-values in these two models, we calculate the probabilities to observe the e-value $x_{v_1 v_2}$ of

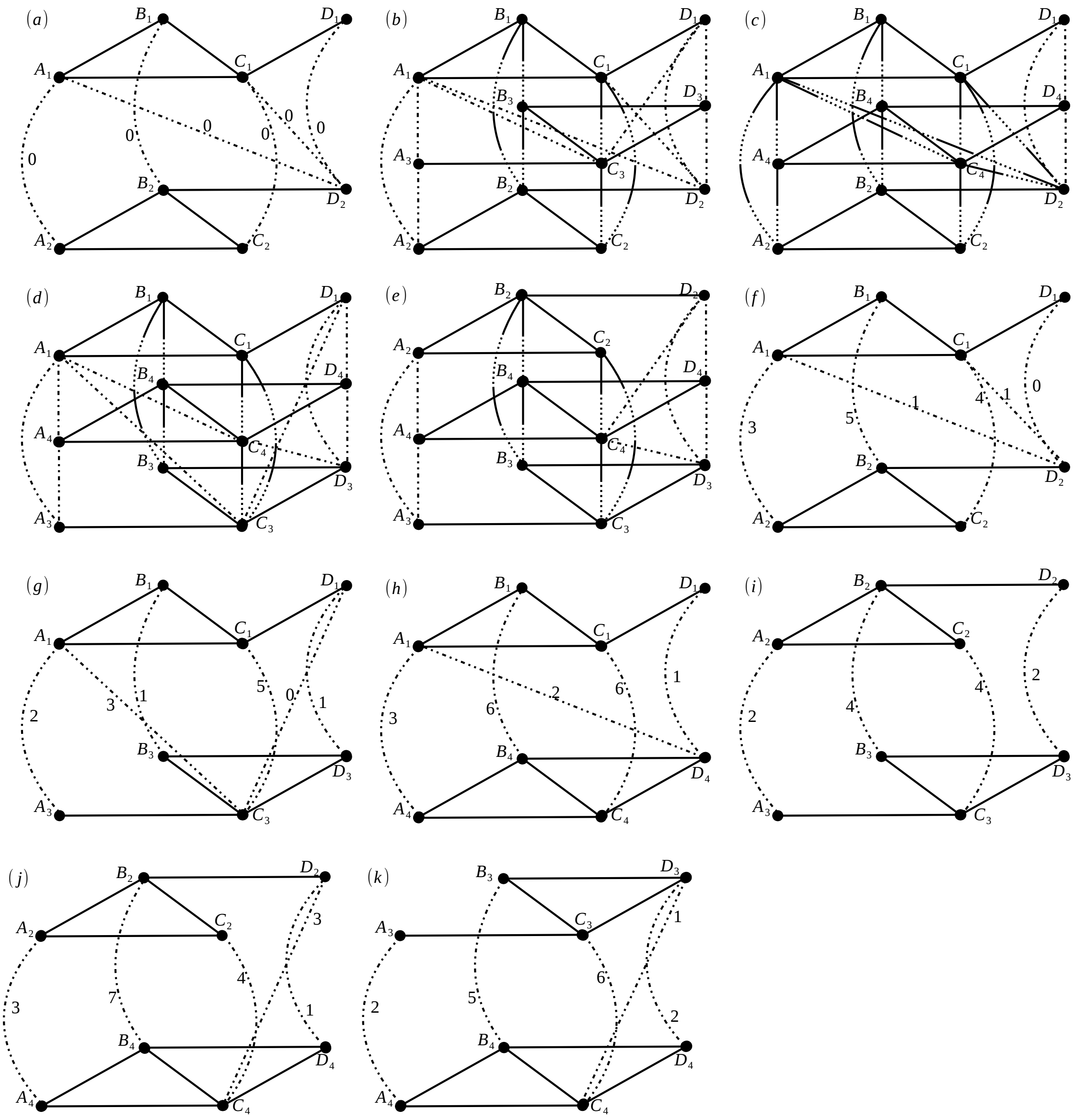


Fig. 4.3: The workflow of our triplet comparison approach on an example with four species. Proteins are represented by nodes, PPIs by solid lines and edges of bipartite graphs by dashed lines. (a) Initialization of the bipartite graph for species 1 and 2. (b-e) Comparison of triplets. Increasing edge scores of pairs of triplet matches whose proteins share three PPIs, e. g. the red bipartite edges. (f-k) The final topology scores of edges in the six bipartite graphs.

the two proteins v_1 and v_2 in the two models, $Pr(x_{v_1v_2}|H)$ and $Pr(x_{v_1v_2}|N)$. To calculate the probability of $x_{v_1v_2}$ (i.e. e-value or bitscore) in both of the two models, the probability distributions of H and N are required to know. Thus, we sample e-value and bitscore for all pairs of proteins in five eukaryotic species: human, mouse, fruit fly, nematode, and yeast. The empirical distributions are illustrated in Fig. 4.2. Our normalized sequence score is the log-ratio

$$y_{v_1v_2} = \log \frac{Pr(x_{v_1v_2}|H)}{Pr(x_{v_1v_2}|N)}$$

of these probabilities scaled to the range from 0 to 1 with the minimal observed log-ratio y_{min} and maximal observed log-ratio y_{max} of all protein pairs in the H model:

$$S_r(v_1, v_2) = \frac{y_{v_1v_2} - y_{\min}}{y_{\max} - y_{\min}} .$$

To compute the topology-based score $S_t(v_1, v_2)$ for each edge, we use a *triplet comparison approach* that bears similarities to the concept of overlapping weights (Morgenstern, 1999) and T-Coffee's consistency approach (Notredame *et al.*, 2000) in multiple sequence alignment. Our approach is an incremental process with the final score reflecting the likelihood of a pair of proteins being topologically conserved. Initially, we set the topology-based scores of all edges in the $\binom{k}{2}$ bipartite graphs of two different species to zero. After this initialization, each of the edges has an equal right to be a part of the global alignment with regard to the topology similarity. Fig. 4.3(a) illustrates an example of species that are numbered 1 and 2. A *triplet* is a set of three PPI networks and the three involved bipartite graphs. We can construct a series of triplets by combining any three different PPI networks. A set of three nodes that are mutually connected by edges is a *triplet match*, e. g. $\{A_1, A_2, A_3\}$ in Fig. 4.3(b). Next, we do a series of triplet comparisons for each *triplet* as described in Algorithm 12. Given a triplet of three networks $G_i = (V_i, E_i), i \in \{1, 2, 3\}$ and their corresponding bipartite graphs $B_{ij} = (V_i \cup V_j, E_{ij}), i, j \in \{1, 2, 3\}$ (see in line 1), our method exhaustively searches all pairs of triplet matches that are connected by three interactions in the triplets (lines 3–9), then increases the score of all the six edges of the two triplet matches by one (see line 10). We denote the neighbors of nodes v in graph G as $\mathcal{N}(v, G)$. Finally, the scores of all edges are divided by two because each pair of match-sets has been counted twice in the for-loop procedure.

In the process of reweighing, we consider all pairs of triplet matches that are connected by conserved interactions in all three networks, such as the edges in line with fine dots in Fig. 4.3(b-e). All edge scores of each bipartite graphs are illustrated in Fig. 4.3(f-k). As an example, the overall topology-based score for the two proteins B_1 and B_2 in Fig. 4.3(f) is five which is explained as follows: In Fig. 4.3(b) the conserved interaction between $\{B_1, C_1\}$ and $\{B_2, C_2\}$ is confirmed by $\{B_3, C_3\}$, and hence the triplet matches $\{B_1, B_2, B_3\}$ and $\{C_1, C_2, C_3\}$ are completely connected by interaction edges contributing one to the score. Note that in Fig. 4.3(b) the triplet matches $\{A_1, A_2, A_3\}$ and $\{B_1, B_2, B_3\}$ do not contribute because of the missing interaction edge $\{A_3, B_3\}$. In Fig. 4.3(c) the four combinations of triplet matches $\{\{B_1, B_2, B_4\}, \{A_1, A_2, A_4\}\}$, $\{\{B_1, B_2, B_4\}, \{C_1, C_2, C_4\}\}$, $\{\{B_1, B_2, B_4\}, \{A_1, D_2, C_4\}\}$, and $\{\{B_1, B_2, B_4\}, \{C_1, D_2, C_4\}\}$ contribute four to the score.

After this process, each edge of the bipartite graphs has been assigned a topology-based score, which we normalize to the range between 0 and 1. However, the distribution of the topology-based score is extremely non-uniform. For example, approximately 90% of the protein pairs have a normalized topology score between 0 and 0.1 in our Dataset-2 (see in Fig. 4.4(a)). The detail of our test datasets are described in the latter part. In contrast, 95% of the protein pairs have a normalized sequence score between 0.6 and 1 (see in Fig. 4.4(b)). Therefore, the topology score of most edges have a very small impact on the alignment in the integrated score.

The non-uniform distribution of the topology score can be explained by the fact that biological networks are scale-free networks whose connectivity follows a power-law distribution (Barabási and Albert, 1999). In these networks, there are just a few hub nodes and a large number of nodes sparsely connecting other nodes. Only the edges between hub nodes obtain large topology scores, whereas all other edges have topology scores close to 0. For example, one edge in Dataset-2 is supported by 16228 triplet matches leading to scores below 0.1 for edges that are supported by less than 1622 triplet matches. However, edges that are supported by thousands of triplet matches also indicate a high probability of functional relatedness and shall be assigned topology scores that are significantly different from zero.

Therefore, it is necessary to lift these small scores up to make sure the topology scores play a real impact role for the optimal alignment when α=0.5. To solve this problem, we redistribute the topology scores using a power-law function $S_t(a, b) = (\frac{t_{ab} - t_{min}}{t_{max} - t_{min}})^\beta$ as shown in Fig. 4.4(c). Here, (a, b) is an edge, t_{ab} the topology score and $\beta = 0.1$. This concludes the computation of the edge scores $S(v_1, v_2)$ where each score now reflects sequence similarity and topology conservation.

4.1.4 Collection of candidate edges

After the process of score integration, we have given $\binom{k+1}{2}$ weighted bipartite graphs, $\binom{k}{2}$ of which formed by proteins from two different species. The weight of each edge in $B_{ij}, i < j$, reflects the likelihood of the edge to be a true match of the global alignment, including infor-

Algorithm 12 The Triplet Comparison Approach

```
function TripletComparison($G_1, G_2, G_3, B_{12}, B_{13}, B_{23}$)
    Initialize the topology score of all edges with 0;
    for each edge $e = (a_1, a_3) \in E_{13}$ do                ▷ $a_1 \in V_1, a_3 \in V_3$
        for each node $a_2 \in \mathcal{N}(a_1, B_{12})$ do                ▷ $a_2 \in V_2$
            if $a_2 \in \mathcal{N}(a_3, B_{23})$ then        ▷ If true, $(a_1, a_2, a_3)$ is a triplet match.
                for each node $b_1 \in \mathcal{N}(a_1, G_1)$ do
                    for each node $b_2 \in \mathcal{N}(a_2, G_2)$ do
                        for each node $b_3 \in \mathcal{N}(a_3, G_3)$ do
                            if $(b_1, b_2) \in E_{12}$ && $(b_1, b_3) \in E_{13}$ && $(b_2, b_3) \in E_{23}$ then
                                Increase the score for all six edges by 1;
                            end if
                        end for
                    end for
                end for
            end if
        end for
    end for
    Divide each score by 2;        ▷ Each pair of match-sets is counted twice.
end function
```

mation about sequence and topology conservation. We use a *maximum weighted matching* algorithm based on an improvement of *Edmond's Algorithm* (Galil, 1983; Mehlhorn and Schäfer, 2002), to find a one-to-one node mapping table in each of the $\binom{k}{2}$ bipartite graphs and collect the matching edges as candidate edges. Furthermore, we collect protein pairs of the same species with scores higher than a threshold $\sigma = \eta(1-\alpha)$. The parameter η is user-defined and enables our method to identify match-sets formed by proteins of one species. The term $(1 - \alpha)$ accounts for the fact that the topology score of these edges is always 0. We obtain a collection of candidate edges, denoted as Ω. The collection of candidate edges reduces the computational complexity while retaining the sensitivity and specificity of the algorithm in praxis.

4.1.5 Simulated annealing

To find a multiple global alignment $\mathbb{A} \subseteq \Omega$, we define the scoring function $\Phi(\mathbb{A}) = \sum_{\vartheta \in \mathbb{A}} f(\vartheta)$, where $f(\vartheta)$ is the score of a match-set $\vartheta = \{v_1, v_2, \cdots, v_{|\vartheta|}\}$. The score of ϑ is calculated

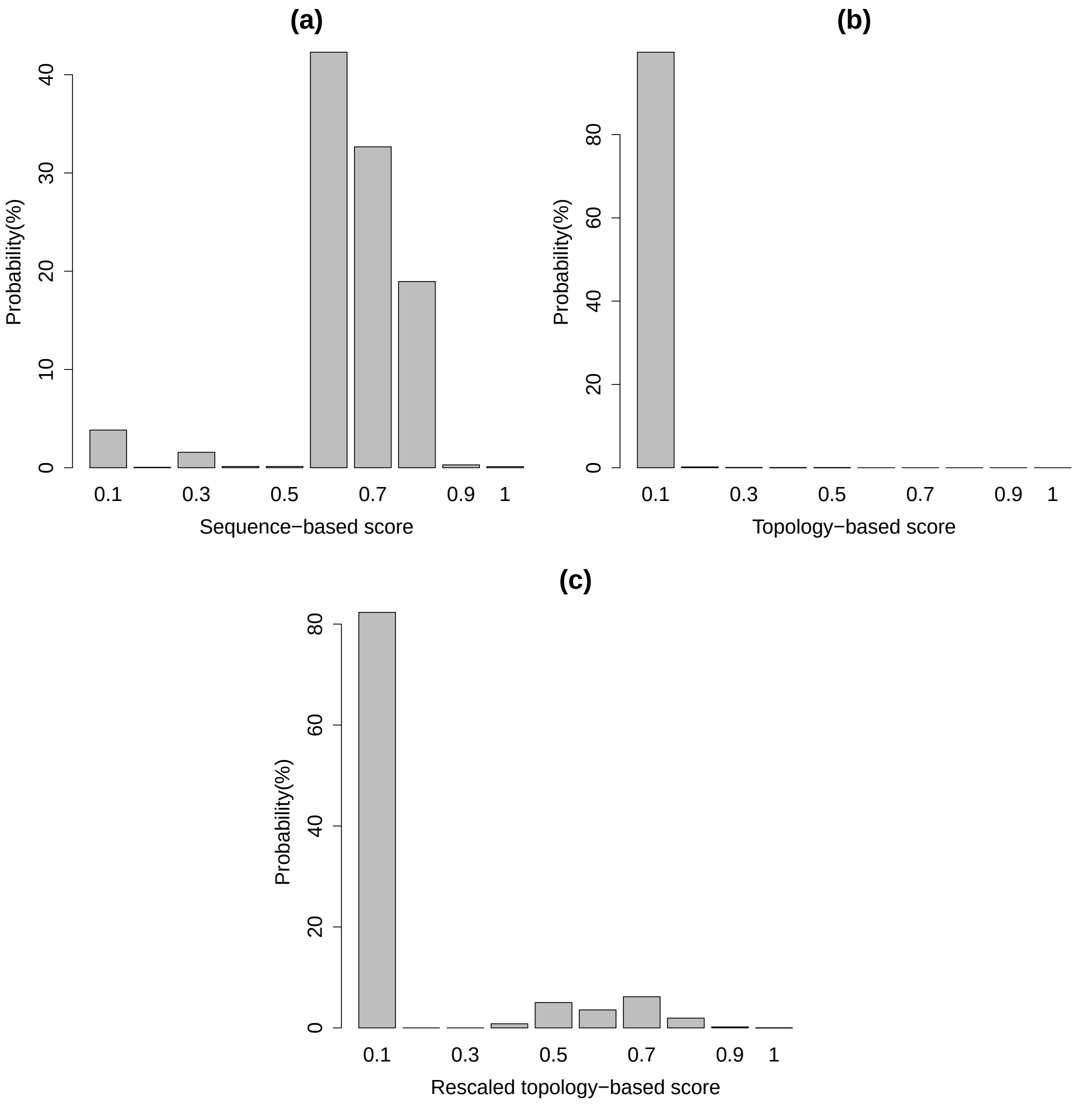


Fig. 4.4: Distributions of normalized sequence scores and normalized topology scores on Dataset-2. (a) The distribution of sequence scores normalized by the linear function $\frac{x_{ab}-x_{min}}{x_{max}-x_{min}}$; (b) The distribution of topological scores normalized by the linear function $(\frac{t_{ab}-t_{min}}{t_{max}-tmin})$; (c) The distribution of normalized topological scores rescaled by the power-law function $(\frac{t_{ab}-t_{min}}{t_{max}-t_{min}})^{\beta}, \beta = 0.1$.

Algorithm 13 Simulated annealing in *NetCoffee*

```
Input: Matching edges $\Omega$, $K$, $T_{\min}$, $T_{\max}$, $s$
Output: A solution $\mathbf{x}^*$ with a set of mutually disjoint match-sets
$\mathbf{x} = \emptyset, T_0 = T_{\max}, i = 1$;
while $i \leq K$ do
    $n = 0$;
    $T_i = T_0 - \frac{i \cdot (T_{\max} - T_{\min})}{K}$;
    while $n < N$ do
        draw arbitrary sample $\xi \in \Omega$ from uniform distribution;
        $\mathbf{x}' = updateState(\mathbf{x}, \xi)$;
        $\Delta\Phi = \Phi(\mathbf{x}') - \Phi(\mathbf{x})$;
        if $\Delta\Phi > 0$ then
            $\mathbf{x} = \mathbf{x}'$;
        else $rand(0, 1) < \exp\{\Delta\Phi/(sT_i)\}$
            $\mathbf{x} = \mathbf{x}'$;
        end if
        $n = n + 1$;
    end while
    $i = i + 1$;
end while
$\mathbf{x}^* = \mathbf{x}$;
return $\mathbf{x}^*$;
```

with the function $f(\vartheta) = \sum_{i,j} S(v_i, v_j)\delta_{ij}$, where $\delta_{ij} = 1$ iff $\{v_i, v_j\} \in \Omega$, otherwise $\delta_{ij} = 0$.

Let I be the collection of all possible global alignments. Then, the problem of multiple global alignment can be modeled as an optimization problem $\max_{\mathbb{A} \in I} \Phi(\mathbb{A})$. We use the SA approach to approximate the highest-scoring alignment. Annealing is known as a thermal process for obtaining a minimum energy state of solid in a heat bath, which includes two major steps: i) raising the temperature to melt the solid metal; ii) decreasing the temperature in a proper strategy so that the inner particles arrange themselves in a state of lower energy. The SA phase is a crucial process in our method. Unlike the strategy of progressive alignment (Flannick *et al.*, 2006), which successively aligns closest pairs of networks and constructs a new network alignment, the SA approach starts with an empty alignment of all networks and runs a large number of iterations of a *Metropolis* scheme (Metropolis *et al.*, 1953) to maximize $\Phi(\mathbb{A})$. It enables our computational tool to gradually promote our alignment to a best result by repeatedly perturbing the current state. The pseudocode of SA is described in Algorithm 13.

Let $\mathbf{x} \in I$ be a feasible solution (a set of mutually disjoint match-sets) for the problem and $\Phi(\mathbf{x})$ the alignment score of $\mathbf{x}$. At the beginning of the algorithm, we initialize our alignment $\mathbf{x}$ with $\emptyset$ and set a temperature parameter T_0 to its maximum. In the following annealing phase, we decrease the temperature and repeatedly perturb the current solution $\mathbf{x}$ with a *Metropolis* scheme using $\pi_i \propto \exp\left(-\Phi(\mathbf{x})/(sT_i)\right)$ as the equilibrium distribution (Kirkpatrick *et al.*, 1983) (see lines 5–15). Parameters $s, K, N, T_{\min}$ and $T_{\max}$ control the SA. The $\mathit{updateState}(\mathbf{x}, \xi)$ updates the current alignment with an arbitrary sample $\xi = \{u, v\} \in \Omega$. Now, given an arbitrary sample $\xi = (u, v) \in \Omega$, we are faced by a question, how to perturb the current alignment $\mathbf{x}$ with ξ. Basically, the *updateState* process runs into four possible scenarios (see in Algorithm 14) : i) $u \notin \mathbf{x}$ and $v \notin \mathbf{x}$; ii) $u \notin \mathbf{x}$ and $v \in \mathbf{x}$; iii) $u \in \mathbf{x}$ and $v \notin \mathbf{x}$; and iv) $u \in \mathbf{x}$ and $v \in \mathbf{x}$, but u and v are not in the same match-set. In the first scenario, a new match-set would be added to the current alignment (see lines 3–4). Then, we have two possible operations for each of the next two scenarios to update the alignment $\mathbf{x}$. Suppose there is $\xi = (u, v) \in \Omega$, $\exists \zeta \in \mathbf{x},\ s.t.\ v \in \zeta$. We update $\mathbf{x}$ by $\mathit{combine}(\mathbf{x}, u, v)$ if it satisfies one of the following conditions: c1) $\forall w \in \zeta$, $\{w, u\} \notin \cup_{i=1}^{k}(V_i \times V_i)$; c2) $\exists w \in \zeta$, $\{w, u\} \in \cup_{i=1}^{k}(V_i \times V_i)$, $(w, u) \in \Omega$. Otherwise, we update $\mathbf{x}$ by $\mathit{Substitute}(\mathbf{x}, u, v)$. For simplicity, we say $F(\mathbf{x}, u, v)$ is true if one of the two conditions is satisfied (see lines 5–16). $\mathit{Combine}(\mathbf{x}, u, v)$ means combine u to the match-set containing v in $\mathbf{x}$. Certainly, if $u \in \mathbf{x}$, the other u must be erased from the match-set. And $\mathit{Substitute}(\mathbf{x}, u, v)$ means substitute one node $w \in \zeta$, which satisfies $\{w, u\} \in \cup_{i=1}^{k}(V_i \times V_i), (w, u) \notin \Omega$. In the fourth scenario, we choose to use a new match-set which yields a higher score to replace the one overlapped by ξ (see lines 17–33). We continue this process until the “temperature” T_i decrease to $T_{\min}$.

4.2 Complexity analysis

We assume that there are n proteins in the largest PPI network, k input PPI networks, m edges in the largest bipartite graph. As shown in Algorithm 12, the *triplet comparison approach* has a complexity of $\binom{k}{3}O(n^6)$. Suppose there is a general graph, $B_s = (V_s, E_s)$, the runtime complexity of the improved *Edmond's Algorithm* on B_s is $O(|V_\mathrm{s}||E_\mathrm{s}|\log|V_\mathrm{s}|)$. Therefore, the collection of candidate edges costs $\binom{k}{2}O(nm\log n)$ time.

The convergence time of SA has been a widely studied question in the last two decades. We assume $\Delta = \max\{\Phi(\mathbf{x}') - \Phi(\mathbf{x})\}$, where $\mathbf{x}'$ is a neighbor state of state $\mathbf{x}$. As shown by the proof in (Rajasekaran, 1990), SA converges within time $2\beta[d\exp\{\Delta/(sT)\}]^D$ where D is the diameter, d is the degree of the underlying Markov chain, and β is defined by the convergence probability $\geq (1-2^{-\beta})$. Theoretically, D and d are hard to calculate. However, in practice, the complexity of SA only depends on two parameters of the cooling scheme, K

Algorithm 14 Algorithm of updating states

```
function UPDATESTATE($(\xi, \mathbf{x})$)
    $sce \leftarrow scenario(\xi, \mathbf{x})$;
    if $sce == 1$ then
        $\mathbf{y} \leftarrow \mathbf{x} \cup \xi$;
    else if $sce == 2$ then
        if $F(\mathbf{x}, u, v)$ then
            $\mathbf{y} \leftarrow Combine(\mathbf{x}, u, v)$;
        else
            $\mathbf{y} \leftarrow Substitute(\mathbf{x}, u, v)$;
        end if
    else if $sce == 3$ then
        if $F(\mathbf{x}, v, u)$ then
            $\mathbf{y} \leftarrow Combine(\mathbf{x}, v, u)$;
        else
            $\mathbf{y} \leftarrow Substitute(\mathbf{x}, v, u)$;
        end if
    else if $sce == 4$ then
        if $F(\mathbf{x}, u, v)$ then
            $\mathbf{y1} \leftarrow Combine(\mathbf{x}, u, v)$;
        else
            $\mathbf{y1} \leftarrow Substitute(\mathbf{x}, u, v)$;
        end if
        if $F(\mathbf{x}, v, u)$ then
            $\mathbf{y2} \leftarrow Combine(\mathbf{x}, v, u)$;
        else
            $\mathbf{y2} \leftarrow Substitute(\mathbf{x}, v, u)$;
        end if
        if $\Phi(\mathbf{y1}) > \Phi(\mathbf{y2})$ then
            $\mathbf{y} \leftarrow \mathbf{y1}$;
        else
            $\mathbf{y} \leftarrow \mathbf{y2}$;
        end if
    end if
    return $\mathbf{y}$;
end function
```

and N. From Algorithm 13, we can easily find out that the complexity is $\Omega(K \cdot N)$, which is independent of the number of compared species k. To sum up, practically, our algorithm is able to deal with multiple networks and has a very favorable time complexity. Our results show that the alignment score indeed converges rapidly in our experiments (see in Fig. 4.5).

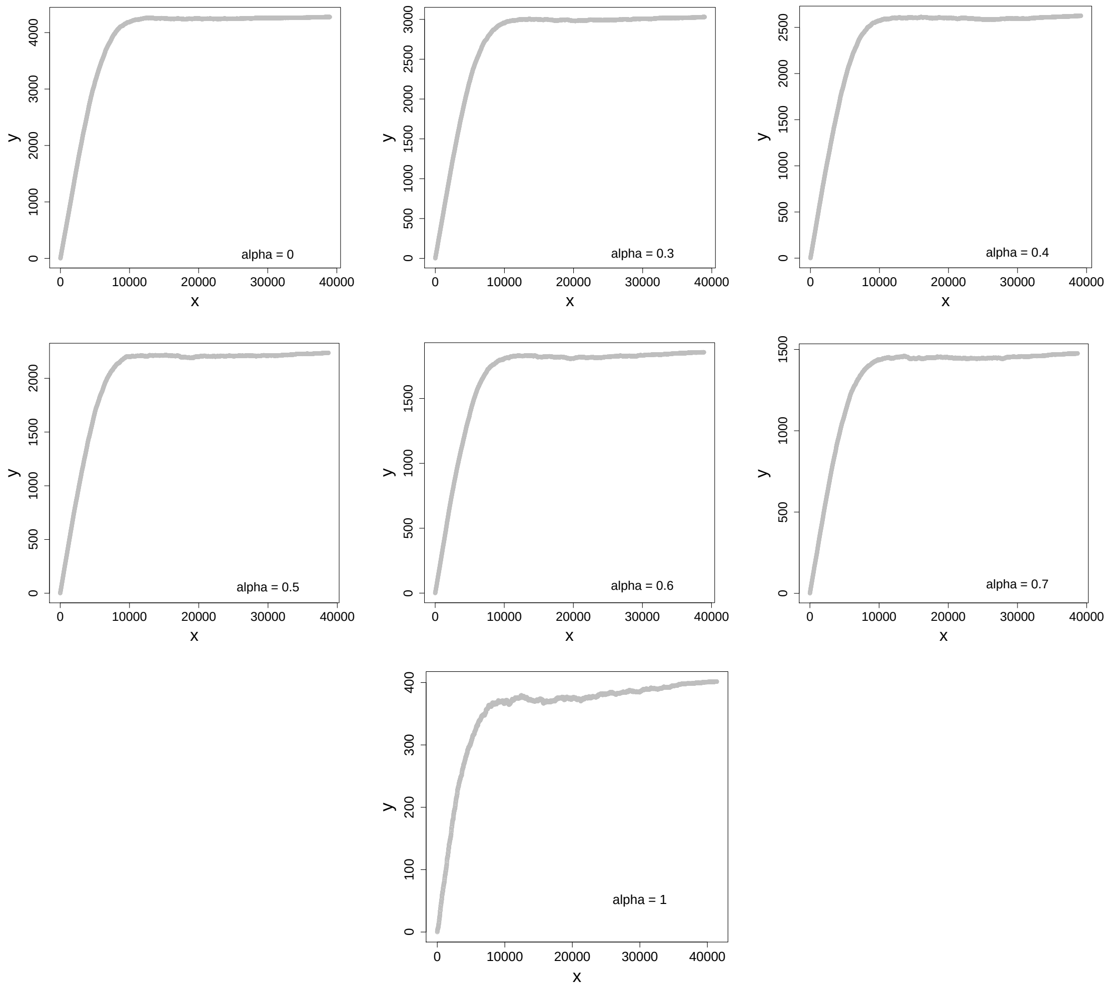


Fig. 4.5: Convergence of the alignment score for $\alpha = \{0, 0.3, 0.4, 0.5, 0.6, 0.7, 1\}$. The vertical axis y represents the alignment score, and the horizontal axis x the number of adjusting steps to optimize the alignment from the initial configuration.

Tab. 4.1: The number of proteins and protein-protein interactions of four datasets which consist of the PPI networks from eleven species.

Species	Proteins	Interactions	Dataset-0	Dataset-1	Dataset-2	Dataset-3
H. sapiens	8777	28366		√	√	
M. musculus	1531	1626		√	√	
D. melanogaster	1534	2664	√	√	√	
C. elegans	767	915	√	√	√	
S. cerevisiae	5739	36226	√		√	
E. coli	4179	169636				√
V. cholerae	3044	76341				√
C. jejuni 11168	1424	76913				√
H. pylori 26695	1206	48430				√
C. crescentus	3022	52302				√
S. typhimurium	4326	151118				√

4.3 Results and discussion

4.3.1 Test datasets

We have evaluated our alignment tool on three datasets of up to five eukaryotic species and one dataset of six microbes as shown in Tab. 4.1. The five eukaryotic species include *Homo sapiens* (human), *Mus musculus* (mouse), *Dorsophila melanogaster* (fruit fly), *Caenorhabditis elegans* (nematode) and *Saccharomyces cerevisiae* (yeast). The six microbes include *Escherichia coli*, *Salmonella typhimurium*, *Vibrio cholerae*, *Campylobacter jejuni NCTC 11168*, *Helicobacter pylori 26695*, and *Caulobacter crescentus.*

To build the five eukaryotic networks of Dataset-0, -1 and -2, we collected all experimentally determined interactions from the public database IntAct (Kerrien *et al.*, 2012). In addition, we collected the reference proteome sets of the five species from UniProtKB/Swiss-Prot release 2012_07 (Uniprot Consortium, 2007), which are used for all-against-all sequence comparisons. To make sure the proteins in our networks are non-redundant and well-annotated, we discarded interactions between proteins that were not in the reference proteome sets. The number of proteins and interactions of these PPI networks are given in Tab. 4.1. Dataset-3 is the same dataset used in the original publication of *Graemlin* 2.0 (Flannick *et al.*, 2008).

For analyzing the biological quality of the alignments, gene ontology (GO) information was collected from UniProt-GOA (Camon *et al.*, 2004) (downloaded on Jan. 8, 2013) to annotate proteins with the three basic types of ontologies: *biological process* (BP), *molecular function* (MF), and *cellular component* (CC). To exclude unreliable function annotations, GO annotations with evidence codes IEA (inferred from electronic annotation) and ISS (inferred from sequence or structural similarity) were discarded.

4.3.2 Experimental setup

We have implemented *NetCoffee* in C++ using the *LEMON Graph Library* (Dezső *et al.*, 2011) version 1.2.3. The implementation supports multicore parallelism for the triplet comparison. We ran *NetCoffee* on all four datasets and tuned its SA parameters such that the SA process converged to a stable score (see in Fig. 4.5). The default values are now $s = 0.005$, $K = 100$, $N = 2000$, $T_{min} = 10$, $T_{max} = 100$, and $\eta = 1.0$.

To compare *NetCoffee* with the state-of-the-art algorithm *IsoRank-N*, we executed *IsoRank-N* on the same datasets with recommended parameters: $K = 20$, thresh $= 10^{-4}$, maxveclen $= 10^6$. Additionally, *NetworkBlast-M*, *Graemlin* 2.0 and SMETANA were included in our assessment. However, *NetworkBlast-M* was unable to work on Dataset-0, -2, and -3 for two reasons. Firstly, the yeast network contained some protein with up to 3276 interactions, which was prohibitive for *NetworkBlast-M*. Secondly, *NetworkBlast-M* required e-values as a protein similarity measure, but Dataset-3 provided only bitscores. Furthermore, we did not run *Graemlin* 2.0 on Dataset-0, -1, -2, because *Graemlin* 2.0 required additional training data (i. e. known alignments for the compared species) to learn its parameters. Since *Graemlin* 2.0 also identified match-sets whose proteins were from a single species, we set $\eta = 0.7$ for Dataset-3 to make sure a fair comparison with *Graemlin* 2.0.

We inputted the networks of the species in the same order for all programs, namely the order from Table 4.1. Note that only the results of *IsoRank-N* depended on the order of input species. All experiments mentioned in the following parts were carried out on the same machine, an Intel(R) Xeon(R) CPU X5550 with 2.67GHz.

4.3.3 Performance comparison

We demonstrate the quality of our alignments in terms of coverage and consistency, and assess the performance of our method by measuring running times. Coverage, which serves as a proxy for sensitivity, indicates the amount of input data the algorithm can explain. Consistency, which serves as a proxy for specificity, measures the functional similarity of proteins in each match-set. Coverage can be easily achieved by sacrificing consistency, and vice versa. The running time demonstrates the ability of *NetCoffee* to deal with large data sets. Intuitively, the goal is to find a global alignment that has a good consistency while explaining as many proteins as possible (i.e. high coverage) in reasonably short time. We first look at differences the programs exhibit in coverage and then investigate the consistency of the match-sets with three measures. Next, we compare running times and, finally, demonstrate how much *NetCoffee* benefits from the integration of similarity and topology score by addressing the influence of the parameter α.

Coverage

For each program, we calculated the percentage of proteins value (PPV), which is the proteins in the alignment over the whole set of proteins, as the coverage (see in Tab. 4.2). In comparison with *IsoRank-N* and *NetworkBlast-M*, the coverage of *NetCoffee* is significantly higher. For instance, the PPV of *NetCoffee* is up to 41.8% for Dataset-1, whereas it is only 31.1% for *IsoRank-N*, and 16.1% for *NetworkBlast-M*. The lower coverage of these two alignment tools can be explained by the facts that *NetworkBlast-M* is a local aligner and, thus, considers only conserved modules; *IsoRank-N* aligns proteins of at least three species into match-sets and does not report match-sets of proteins from only two species[1] (see an example in Tab. 4.3). In comparison with *Graemlin 2.0*, *NetCoffee* also has a slightly higher PPV value except for the extreme case of $\alpha = 1$. When $\alpha = 1$, sequence scores of all pairs of proteins are set to 0 in *NetCoffee*. As a result, all protein pairs from a single species are excluded from the collection of candidate edges and consequently from the alignment. Hence, the coverage drops to 69.7% for Dataset-3. In comparison with *SMETANA*, the coverage of *NetCoffee* is similar. *NetCoffee* achieves a lower PPV for Dataset-0, -1, and -2, but a higher PPV for Dataset-3. Concerning the number of match-sets, *IsoRank-N* identifies more match-sets formed by proteins from three of the compared species, and both *Graemlin* 2.0 and *SMETANA* find more match-sets for Dataset-3 than *NetCoffee* except for $\alpha = 1$ (see in Tab. 4.3).

Consistency

An alignment tool that achieves a high coverage is not necessarily better than others. For example, a random global alignment may cover all proteins, but aligns many unrelated proteins. Hence we now address the performance of the alignment tools in terms of consistency. Consistency demonstrates the biological significance of predicted match-sets.

As a first consistency measure, we computed the mean entropy and the mean normalized entropy of the predicted match-sets in the alignments of each algorithm. We calculated

---

[1] These match-sets can be recognized by running the pairwise aligner *IsoRank* on each pair of species.

Tab. 4.2: Coverage, entropy and speed comparison. The five algorithms *NetCoffee*, *IsoRank-N*, *NetworkBlast-M* (NBM), *Graemlin* 2.0 (Gr. 2.0) and *SMETANA* (SME) were tested on the four datasets. The rows list the PPV, mean entropy (ME), mean normalized entropy (MNE) and the running time. D-x in the first column represents the test dataset, Dataset-x. S,m and h in the row of time represent seconds, minutes and hours. Boldface numbers represent the best performance with respect to each row. Note that the parameter of α in both *NetCoffee* and *IsoRank-N* demonstrates the percentage of the topology score contributing to the whole alignment score.

		NetCoffee							*IsoRank-N*							NBM	Gr. 2.0	SME
		$\alpha = 0.0$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 1.0$	$\alpha = 0.0$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 1.0$	-	-	-
D-0	PPV (%)	28.3	28.2	28.4	28.3	28.1	28.3	27.6	6.75	16.1	16.1	15.7	15.1	14.2	2.14	*	*	**35.2**
	ME	1.525	**1.504**	1.511	1.519	1.522	1.528	1.523	3.562	2.927	2.941	2.952	3.083	3.057	3.235	*	*	2.393
	MNE	0.6081	**0.6026**	0.6031	0.6054	0.6052	0.6059	0.6091	1.093	0.9627	0.967	0.9705	0.9993	0.9942	0.9951	*	*	0.8374
	Time	**2s**	**2s**	**2s**	**2s**	**2s**	**2s**	**2s**	9.9m	20.5m	21.7m	25.3m	32.5m	41m	70.5m	*	*	53s
D-1	PPV (%)	41.8	41.2	41	41	40.5	40.2	41	19.4	31.1	30.9	30.4	30.2	29.7	10.8	16.1	*	**52.6**
	ME	2.603	2.645	2.6	2.615	2.608	2.614	**2.589**	4.366	3.927	3.896	3.986	3.992	4.043	4.863	4.130	*	3.054
	MNE	0.8642	0.8721	0.8638	0.8669	0.8651	0.8652	**0.8601**	1.258	1.173	1.167	1.186	1.185	1.194	1.336	1.227	*	0.9616
	Time	22s	**21s**	**21s**	**21s**	22s	**21s**	22s	14.6m	29.6m	31.4m	37.7m	46.9m	64.2m	101m	13.5m	*	3.3m
D-2	PPV (%)	49.5	49.1	48.8	48.6	48.3	48.1	47.1	21.9	33.8	33.3	32.7	32.1	31.8	9.67	*	*	**58.1**
	ME	2.257	2.288	2.265	2.286	2.293	2.277	**2.238**	3.942	3.597	3.629	3.645	3.681	3.686	4.403	*	*	2.592
	MNE	0.7918	0.7988	0.7931	0.7979	0.7995	0.7954	**0.7883**	1.167	1.103	1.109	1.113	1.12	1.12	1.244	*	*	0.8656
	Time	48.2s	47.3	51.9s	55.6s	**46.6s**	47.8s	48.0s	1.70h	3.01h	3.77h	4.18h	4.81h	5.86h	9.05h	*	*	6.2m
D-3	PPV (%)	**84.8**	84.3	84.3	84.1	83.9	83.6	69.7	*	*	*	*	*	*	*	*	82.4	79.3
	ME	0.267	0.266	0.267	0.268	0.266	0.267	0.249	*	*	*	*	*	*	*	*	0.263	**0.248**
	MNE	0.203	0.203	0.204	0.205	0.204	0.205	0.201	*	*	*	*	*	*	*	*	0.205	**0.199**
	Time	4.9m	5.9m	5.3m	5.8m	5.7m	7.3m	**3.9m**	> 72h	>72h	>72h	>72h	>72h	>72h	>72h	*	6.7h	7.8m

Tab. 4.3: The distribution of match-sets in the alignment of *NetCoffee*, *IsoRank-N*, *NetworkBlast-M*(NBM), *Graemlin 2.0*(Gr.2.0) and *SMETANA*(SME) on our four datasets. Here, i in the second column represents the number of species, and other elements in each row are the number of match-sets conserved in i species.

		NetCoffee							*IsoRank-N*							*NBM*	*Gr.2.0*	*SME*
		$\alpha = 0.0$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 1.0$	$\alpha = 0.0$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 1.0$			
D-0	i=2	739	753	755	749	749	745	726	0	0	0	0	0	0	0	*	*	648
	i=3	265	255	257	259	255	263	256	140	366	360	351	337	321	31	*	*	274
D-1	i=2	1425	1390	1403	1410	1395	1391	1431	0	0	0	0	0	0	0	0	*	1322
	i=3	569	563	534	539	526	502	559	607	1045	1040	1017	1005	989	259	0	*	502
	i=4	179	183	190	182	185	194	158	90	124	122	125	124	125	35	4738	*	195
D-2	i=2	2173	2144	2171	2113	2119	2125	2204	0	0	0	0	0	0	0	*	*	1862
	i=3	852	842	816	856	816	831	870	829	1462	1415	1415	1374	1355	318	*	*	783
	i=4	365	365	373	350	372	353	305	186	219	234	218	216	224	54	*	*	316
	i=5	143	146	136	146	136	133	80	38	42	40	40	42	38	9	*	*	155
D-3	i=2	977	949	950	950	950	940	1486	*	*	*	*	*	*	*	*	1041	1387
	i=3	681	672	691	679	662	662	991	*	*	*	*	*	*	*	*	785	909
	i=4	401	402	382	391	391	386	562	*	*	*	*	*	*	*	*	417	520
	i=5	216	211	216	223	224	221	297	*	*	*	*	*	*	*	*	240	273
	i=6	468	476	476	472	480	476	383	*	*	*	*	*	*	*	*	499	504

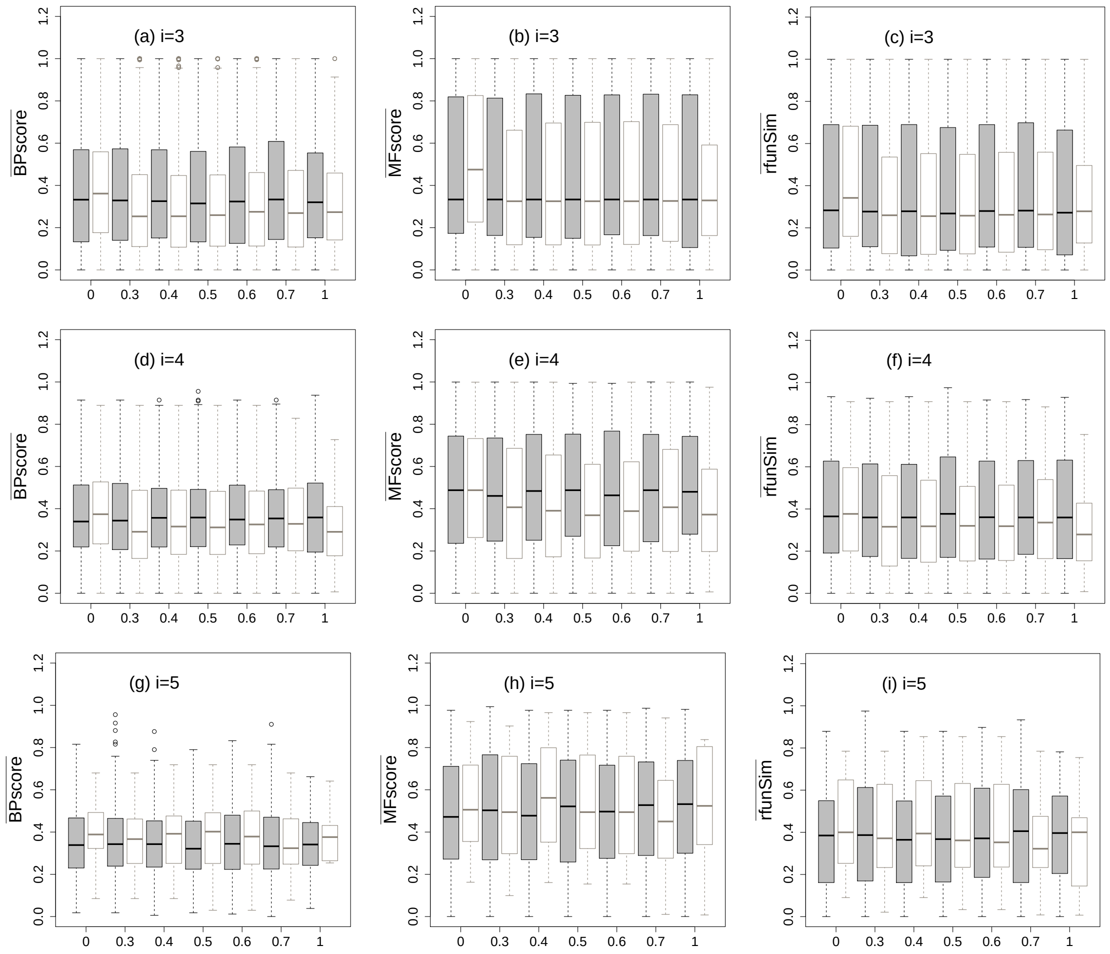


Fig. 4.6: Consistency comparison on Dataset-2 between *NetCoffee* (gray boxes) and *IsoRank-N* (white boxes). Box-plots for the semantic similarity measures $\overline{BPscore}$, $\overline{MFscore}$, and $\overline{rfunSim}$ of match-sets conserved by $i \in \{3, 4, 5\}$ species, with respect to the parameter α (the horizontal axis).

the entropy of a match-set with the same method as in *IsoRank-N* according to its GO annotations. A match-set will have lower entropy if its GO annotations are more functionally coherent. From Tab. 4.2, it showed that the entropy of *NetCoffee* was considerably lower than that of *IsoRank-N* and *NetworkBlast-M* no matter which α was used, whereas at the same time having a high coverage. Additionally, the entropy of *NetCoffee* was lower than that of *SMETANA* on all datasets except for Dataset-3. In comparison with *Graemlin* 2.0, *NetCoffee* achieved nearly identical entropy results for Dataset-3 whereas being considerably faster. The results for $\alpha = 0$ and $\alpha = 1$ demonstrated that both of our two conservation measures could favorably predict the functional relatedness between protein pairs.

Dataset-3 exhibits an interesting trade-off using the α parameter in terms of coverage and consistency. For $\alpha = 1$, *NetCoffee* has the lowest entropy, however at the cost of much lower coverage. Decreasing α improves the coverage while deteriorating the entropy measure. This behavior is less pronounced for the other datasets. However, it shows that the α parameter can be used for having a specificity versus sensitivity trade-off.

Secondly, we assessed consistency by three elaborate semantic similarity measures introduced in (Schlicker *et al.*, 2006, 2007): *BPscore*, *MFscore* and *rfunSim*. Unlike many existing approaches (Kuchaiev and Pržulj, 2011; El-Kebir *et al.*, 2011) that simply evaluate functional similarity by counting the number of common GO terms of involved proteins, *BPscore* and *MFscore* assess the functional similarity of two proteins by exploiting BP and MF annotations with the GO hierarchy tree. The measure *rfunSim* is a combination of *BPscore* and *MFscore*.

The Gene Ontology Consortium (Camon *et al.*, 2004) provides a dynamic and controlled vocabulary describing the function of genes and gene products of organisms. For comparing sets of GO terms and for assessing functional similarity of gene products, semantic similarity measures have been proposed. Three such measures that use information about the lowest common ancestor of two compared GO terms have been proposed by Resnik (Resnik, 1995), Lin (Lin, 1998), and Schlicker (Schlicker *et al.*, 2006). Given two gene ontologies c_1 and c_2, they are defined as follows,

$$\begin{aligned} sim_{\text{Resnik}}(c_1, c_2) &= \max_{c \in S(c_1,c_2)} -\log p(c) \\ sim_{\text{Lin}}(c_1, c_2) &= \max_{c \in S(c_1,c_2)} \frac{2 \log p(c)}{\log p(c_1) + \log p(c_2)} \\ sim_{\text{Rel}}(c_1, c_2) &= \max_{c \in S(c_1,c_2)} \frac{2 \log p(c)}{\log p(c_1) + \log p(c_2)} \cdot (1 - p(c)) \end{aligned}$$

where $S(c_1, c_2)$ is the set of common ancestors of terms c_1 and c_2, and $p(c)$ is the relative frequency, i.e. $freq(c)/freq(root)$, of a term c in a database. Note that $p(c)$ monotonically increases when c moves up to the root of the gene ontology tree.

To evaluate the functional relationship of two gene products, Schlicker (Schlicker *et al.*, 2006, 2007; Schlicker and Albrecht, 2008) designed several measures , such as $rfunSim$ and $funSim$. The definition of these measures uses two ontology scores, $BPscore$ and $MFscore$, which in turn are based on the semantic similarity described above. Suppose two proteins p and q are annotated with two sets of GO terms c^p and c^q, then we can compute a similarity matrix $S = (s_{ij})$ with

$$s_{ij} = sim(c_i^p, c_j^q), \forall i \in \{1, \ldots, N\}, \forall j \in \{1, \ldots, M\}.$$

From this similarity matrix, we can compute a *rowScore* as the average over all row maxima and a *columnScore* as the average over all column maxima. Schlicker (Schlicker *et al.*, 2006) defines the *GOscore*, where *GO* can stand for *MF*, *BP*, or *CC* depending on the set of GO terms used to calculate the similarity matrix, as

$$GOscore = \max\{rowScore, columnScore\}.$$

We report the arithmetic mean of the similarity scores of all involved protein pairs as the functional consistency of a match-set. For instance, given a match-set $\vartheta = (v_1, v_2, \cdots, v_{|\vartheta|})$, the functional consistency of ϑ with respect to the BP annotation is defined as

$$\overline{BPscore}(\vartheta) = \frac{\sum_{i \neq j} BPscore(v_i, v_j)}{\binom{|\vartheta|}{2}}, \quad i, j \in \{1, 2, \cdots, |\vartheta|\}.$$

Analogously, we can calculate $\overline{MFscore}$ and $\overline{rfunSim}$. All three scores range from 0 to 1 which translate into an increasing degree of functional similarity. We calculated the scores using the FSST (Schlicker *et al.*, 2007) package. To avoid skipping too many meaningful match-sets, match-sets that contained less than 40% uncharacterized proteins were also taken into consideration. We separately compared match-sets that contained proteins from 3, 4, and 5 species. And the distribution of match-sets in each category can be seen in Tab. 4.3.

We compared the consistency of *NetCoffee* with that of *IsoRank-N* (see Fig. 4.6) and *SMETANA* (see Fig. 4.3.3) on their alignments of Dataset-2. As shown in Fig. 4.6 (a-c), when $\alpha > 0$, the $\overline{BPscore}$ of *NetCoffee* is higher than that of *IsoRank-N*, and the $\overline{MFscore}$ and $\overline{rfunSim}$ are roughly the same. More importantly, the advantage of *NetCoffee* expands when i (i.e. the number of species) increases to 4, as shown in Fig. 4.6 (d-f) although it identifies more match-sets. *NetCoffee* shows significant improvements with regard to the $\overline{BPscore}$, $\overline{MFscore}$ and $\overline{rfunSim}$ except for the case of $\alpha = 0$. When $\alpha = 0$, *IsoRank-N*

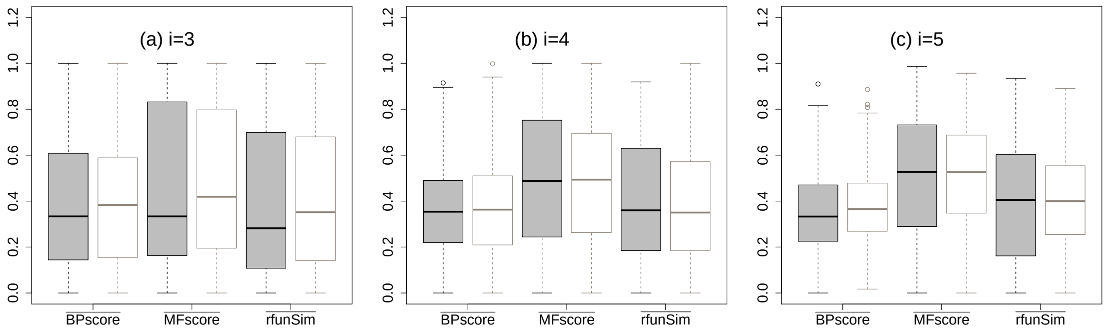


Fig. 4.7: Consistency comparison on Dataset-2 between *NetCoffee* (gray boxes) and *SMETANA* (white boxes). Box-plots for the semantic similarity measures $\overline{BPscore}, \overline{MFscore}$, and $\overline{rfunSim}$ of match-sets conserved by $i \in \{3, 4, 5\}$ species. *NetCoffee* runs with $\alpha = 0.7$.

reaches its highest point. However, we do not recommend to use $\alpha = 0$ for *IsoRank-N* as its coverage drops to only 21.9%. For $i = 5$ illustrated in Fig. 4.6 (g-i), *IsoRank-N* improves the quality of match-sets in terms of $\overline{BPscore}$. The two algorithms are comparable in terms of $\overline{MFscore}$ and $\overline{rfunSim}$. However, *NetCoffee* identifies 3–8 times more match-sets than *IsoRank-N* (see Tab. 4.3). Compared with the alignment of *SMETANA*, match-sets identified by *NetCoffee* have lower semantic scores for $i = 3$ but roughly the same scores for $i = 4$ and $i = 5$ (see Fig. 4.3.3).

Finally, we measured the consistency by computing the percentage of qualified match-sets the algorithms identified. As demonstrated in (Schlicker *et al.*, 2006), 60% of protein pairs in the IO dataset (ontology according to Inparanoid) had *MFscore* >0.8, and 65% had *BPscore* >0.6. Therefore, we regarded those match-sets that had $\overline{MFscore}$ >0.8 or $\overline{BPscore}$ >0.6 as *qualified match-sets*, i. e. functionally related proteins. With these thresholds, 45% of the match-sets recognized by *NetCoffee* were qualified match-sets (see Fig. 4.8), which was significantly more than those identified by *IsoRank-N* (about 25 %) and more than those identified by *SMETANA* (42 %). Visualizations of the GO trees for each qualified match-set (drawn using the package *GO::TermFinder* (Boyle *et al.*, 2004)) and more information about the alignment with $\alpha = 0.3$ are available for download from `https://code.google.com/p/netcoffee/downloads/list`.

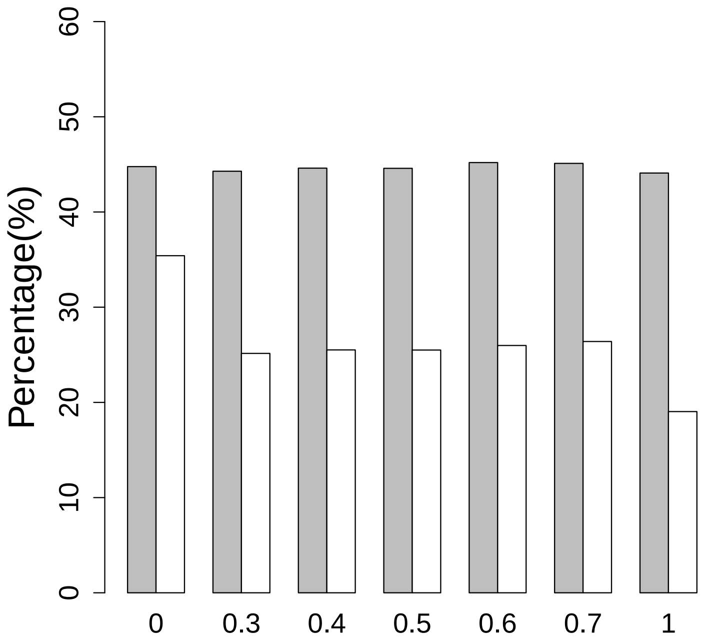


Fig. 4.8: Percentages of functionally related match-sets of *NetCoffee* (gray) and *IsoRank-N* (white) for different values of the parameter α on Dataset-2. Match-sets with an $\overline{MFscore} >$ 0.8 or $\overline{BPscore} > 0.6$ are regarded as functionally related match-sets.

Running time

Tab. 4.2 demonstrates that our method is quite robust to the parameter α in terms of running time. The running time of *IsoRank-N*, however, increases dramatically when α grows. Specifically, *NetCoffee* is 1–3 orders of magnitude faster than *IsoRank-N*, 37 times faster than *NetworkBlast-M*, 82 times faster than *Graemlin* 2.0 (including training time), and 2–26 times faster than *SMETANA*; We choose to report the results achieved with multiple cores (i.e. eight cores), because they are the real running time for *NetCoffee.* Note that *NetCoffee* is still faster than its competitors even on a single core except for *SMETANA* (see Tab. 4.4).

Influence of the parameter α

To figure out how much the alignment tools benefit from the topology and sequence score, we ran both *NetCoffee* and *IsoRank-N* with various α values. If $\alpha = 0$, the global alignment is constructed only based on sequence score, and if $\alpha = 1$, only based on topology score.

Tab. 4.2 and Fig. 4.6 demonstrate that *NetCoffee* is robust to the parameter α in terms

Tab. 4.4: The running time of *NetCoffee* on our four test datasets with a single core.

	NetCoffee(1× core)						
	$\alpha = 0.0$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 0.7$	$\alpha = 1.0$
Dataset-0	2s	2s	2s	2s	2s	2s	2s
Dataset-1	34s	33s	33s	32s	33s	32s	33s
Dataset-2	101s	98s	98s	100s	98s	98s	98s
Dataset-3	11.4m	11.2m	11.4m	11.6m	11.2m	11.3m	11.2m

of coverage, consistency and speed, and that the α parameter can be used for having a specificity versus sensitivity trade-off. Both the topology and the sequence score favorably predict functional relatedness between protein pairs.

However, using either sequence score or topology score alone is not favorable for the coverage of *IsoRank-N* as shown in Tab. 4.2. Furthermore, the alignment quality and the computing time depend on α. Tab. 4.2 suggests that the performance of *IsoRank-N* tends to be the best when $\alpha = 0.3$.

Chapter 5

Conclusion and Future Work

In this thesis, we made a concentrated effort to design algorithms and models with applications in the analysis of biological data. Two efficient network alignment tools *LocalAli* and *NetCoffee* were developed to integrate diverse high-throughput data, and unravel protein function, evolution history and functional modularity in a systematic way. The comparison of biological networks is expected to provide more insights to understand the underlying mechanism of complicate and dynamic life processes in organisms.

LocalAli is a fast and scalable local alignment tool to identify functionally conserved modules across multiple species. It overcomes several limitations of existing algorithms by using a scoring scheme strongly rooted in a *maximum-parsimony* evolutionary model, scaling to multiple networks with tens of thousands of proteins and interactions and parallel computing. By relying on this model, *LocalAli* facilitates interpretation of alignment results in terms of conserved modules that have evolved from an ancestral module through a series of evolutionary events. With a rigorously designed scoring function, we reduced the problem of multiple local network alignment problem to a problem of searching for high-scoring *d-subnets*. *LocalAli* solves the problem in three steps as follows: (i) it searches for a set of *d-subnets* with a heuristic approach *seed-and-extend*; (ii) it reconstructs the evolution history of each *d-subnets* and calculates its alignment score; (iii) these *d-subnets* with an alignment score below a threshold are filtered out.

To evaluate the biological quality and the statistical significance of our results, we applied *LocalAli* to 26 real-world datasets and 1040 random datasets. To compare the performance, several existing algorithms were also performed on the 26 real-world datasets. All the results were analyzed in terms of several criteria. In a short conclusion, *LocalAli* had a superiority of coverage, consistency and scalability over *NetworkBlast-M*, *NetworkBlast*, *AlignNemo* and *MaWISh*, meanwhile retained a high precision in identifying *functional coherent subnetworks*. Furthermore, it predicted >500 new functional annotations for proteins of *worm* and *fruit fly*, and identified 55 *pure modules* which were known protein complexes that belonged to *human* as annotated in *CORUM*. It reported many significant functional modules that were missed by other alignment tools. The results demonstrate that *LocalAli* provides substantial improvements to multiple local network alignment, and might give helpful suggestions to the research community that attempts to determine phylogeny, function annotations and functional modules.

NetCoffee is a fast and accurate algorithm for global alignment of multiple networks. It overcomes several limitations of existing tools by aligning multiple networks without additional training data, finding a global alignment of six species within several minutes, and scaling to networks with tens of thousands of proteins and interactions. Further, it is the first alignment tool that can run with multiple cores in parallel. In this algorithm, we rigorously combine protein sequence similarity and network topology similarity into a suitable scoring scheme for multiple networks, adapting a successful technique from multiple sequence alignment. This allows us to model the problem of multiple global network alignment as a combinatorial optimization problem, which we solve with *simulated annealing*. On PPI networks of five eukaryotic species, human, mouse, fruit fly, nematode and yeast, our implementation *NetCoffee* successfully finds a global alignment covering approximately 50% of the proteins; and about 45% of the match-sets are qualified.

We compared *NetCoffee* to four existing tools, three of which failed to run on at least one of the test datasets in our benchmark. The results indicate that *NetCoffee* outperforms the state-of-the-art algorithm *IsoRank-N* in terms of coverage and consistency, and at the same time is about 1–3 orders of magnitude faster. Compared to *NetworkBlast-M*, *Graemlin* 2.0 and *SMETANA*, *NetCoffee* not only overcomes their limitations, but also retains the quality of alignments in terms of both coverage and consistency. The results show that *NetCoffee* provides substantial improvements to global network alignment and that the research community working on function annotation and phylogenetic analysis can benefit from it. Further, its application is not restricted to PPI networks. It could also be extended to other types of complex networks, such as Scientific Collaboration Networks (SCN) and World Wide Web Networks (WWWN).

Although many computational tools and web servers have been developed and applied in the analyses of various biological data, there are still many problems that block our view of understanding the general biological principles that drive the evolution and regulates various dynamic life processes in organisms. The task of advancing the understanding of living systems through computation will still be a big challenge for a long time. In our future work, we will endeavor to understand the function, evolution, modularity of molecular networks by designing mathematical models, data structure and efficient algorithms for genomics and proteomics data. In the aspect of molecular medicine, the living systems are regarded as complex molecular networks, such as PPI networks, metabolic networks, and gene regulatory networks. One of ultimate goal in the analysis of molecular networks is to elicit a causal connection between the physiological dysfunctions such as cancer, diabetes and sickle cell diseases and their involved regions in the networks. It might suggest promising target proteins or molecular basis for the pharmaceutical research to develop effective drug therapies for these diseases.

Bibliography

Aladağ, A. E. and Erten, C. (2013). Spinal: scalable protein interaction network alignment. *Bioinformatics*, **29**(7), 917–924.

Albert, I. and Albert, R. (2004). Conserved network motifs allow protein-protein interaction prediction. *Bioinformatics*, **20**(18), 3346–3352.

Albert, R., Jeong, H., and Barabási, A.-L. (2000). Error and attack tolerance of complex networks. *Nature*, **406**(6794), 378–382.

Ali, W. and Deane, C. M. (2009). Functionally guided alignment of protein interaction networks for module detection. *Bioinformatics*, **25**(23), 3166–3173.

Alkan, F. and Erten, C. (2014). Beams: backbone extraction and merge strategy for the global many-to-many alignment of multiple ppi networks. *Bioinformatics*, **30**(4), 531–539.

Aloy, P. and Russell, R. B. (2002). The third dimension for protein interactions and complexes. *Trends in Biochemical Sciences*, **27**(12), 633 – 638.

Altschul, S. F., Madden, T. L., Schffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, **25**(17), 3389–3402.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000). Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature genetics*, **25**(1), 25–29.

Ay, F., Dang, M., and Kahveci, T. (2012). Metabolic network alignment in large scale by network compression. *BMC Bioinformatics*, **13**(Suppl 3), S2.

Bader, G. and Hogue, C. (2003). An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics*, **4**(1), 2.

Bandyopadhyay, S., Sharan, R., and Ideker, T. (2006). Systematic identification of functional orthologs based on protein network comparison. *Genome Res.*, **16**, 428–435.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, **286**(5439), 509–512.

Berge, C. (1957). Two theorems in graph theory. *Proceedings of the National Academy of Sciences*, **43**(9), 842–844.

Berggård, T., Linse, S., and James, P. (2007). Methods for the detection and analysis of proteinprotein interactions. *PROTEOMICS*, **7**(16), 2833–2842.

Boyle, E. I., Weng, S., Gollub, J., Jin, H., Botstein, D., Cherry, J. M., and Sherlock, G. (2004). Go::termfinder—open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. *Bioinformatics*, **20**(18), 3710–3715.

Brutlag, D. L. (2008). *Inferring Protein Function from Sequence*, volume 3, chapter 30, pages 1087–1119. Wiley-VCH Verlag GmbH.

Bu, D., Zhao, Y., Cai, L., Xue, H., Zhu, X., Lu, H., Zhang, J., Sun, S., Ling, L., Zhang, N., Li, G., and Chen, R. (2003). Topological structure analysis of the proteinprotein interaction network in budding yeast. *Nucleic Acids Research*, **31**(9), 2443–2450.

Camon, E., Magrane, M., Barrell, D., Lee, V., Dimmer, E., Maslen, J., Binns, D., Harte, N., Lopez, R., and Apweiler, R. (2004). The gene ontology annotation (GOA) database: sharing knowledge in Uniprot with gene ontology. *Nucleic Acids Research*, **32**(suppl 1), D262–D266.

Chapman, B., Jost, G., and Pas, R. v. d. (2007). *Using OpenMP: Portable Shared Memory Parallel Programming (Scientific and Engineering Computation)*. The MIT Press, Cambridge, Massachusetts, London, England.

Chatr-aryamontri, A., Breitkreutz, B.-J., Heinicke, S., Boucher, L., Winter, A., Stark, C., Nixon, J., Ramage, L., Kolas, N., ODonnell, L., Reguly, T., Breitkreutz, A., Sellam, A., Chen, D., Chang, C., Rust, J., Livstone, M., Oughtred, R., Dolinski, K., and Tyers, M. (2013). The biogrid interaction database: 2013 update. *Nucleic Acids Research*, **41**(D1), D816–D823.

Chindelevitch, L., Liao, C.-S., and Berger, B. (2010). Local optimization for global alignment of protein interaction networks. In *Pacific Symposium on Biocomputing'10*, pages 123–132.

Chindelevitch, L., Ma, C.-Y., Liao, C.-S., and Berger, B. (2013). Optimizing a global alignment of protein interaction networks. *Bioinformatics*, **29**(21), 2765–2773.

Ciriello, G., Mina, M., Guzzi, P. H., Cannataro, M., and Guerra, C. (2012). Alignnemo: A local network alignment method to integrate homology and topology. *PLoS ONE*, **7**(6), e38107.

Dezső, B., Jüttner, A., and Kovács, P. (2011). LEMON – an open source C++ graph template library. *Electronic Notes in Theoretical Computer Science*, **264**(5), 23 – 45. Proceedings of the Second Workshop on Generative Technologies (WGT) 2010.

Dongen, S. (2000). A cluster algorithm for graphs. Technical report, Amsterdam, The Netherlands, The Netherlands.

Dutkowski, J. and Tiuryn, J. (2007). Identification of functional modules from conserved ancestral protein-protein interactions. *Bioinformatics*, **23**(13), i149–i158.

Edmonds, J. (1965a). Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards B*, **69**, 125–130.

Edmonds, J. (1965b). Paths, trees, and flowers. *Canadian Journal of Mathematics*, **17**, 449–467.

El-Kebir, M., Heringa, J., and Klau, G. (2011). Lagrangian relaxation applied to sparse global network alignment. In M. Loog, L. Wessels, M. Reinders, and D. Ridder, editors, *Pattern Recognition in Bioinformatics*, volume 7036 of *Lecture Notes in Computer Science*, pages 225–236. Springer Berlin Heidelberg.

Estojak, J., Brent, R., and Golemis, E. A. (1995). Correlation of two-hybrid affinity data with in vitro measurements. *Molecular and Cellular Biology*, **15**(10), 5820–9.

Fang, G., Bhardwaj, N., Robilotto, R., and Gerstein, M. B. (2010). Getting Started in Gene Orthology and Functional Analysis. *PLoS Comput Biol*, **6**(3), e1000703+.

Featherstone, D. E. and Broadie, K. (2002). Wrestling with pleiotropy: Genomic and topological analysis of the yeast gene expression network. *BioEssays*, **24**(3), 267–274.

Federhen, S. (2012). The ncbi taxonomy database. *Nucleic Acids Research*, **40**(D1), D136–D143.

Felsenstein, J. (2003). *Inferring Phylogenies*. Sinauer Associates, 2 edition.

Fields, S. and Song, O. (1989). A novel genetic system to detect protein-protein interactions. *Nature*, **340**, 245–246.

Fitch, W. M. (1971). Toward defining the course of evolution: Minimum change for a specific tree topology. *Systematic Biology*, **20**(4), 406–416.

Flannick, J., Novak, A., Srinivasan, B. S., McAdams, H. H., and Batzoglou, S. (2006). Graemlin: general and robust alignment of multiple large interaction networks. *Genome Research*, **16**(9), 1169–1181.

Flannick, J., Novak, A., Do, C., Srinivasan, B., and Batzoglou, S. (2008). Automatic parameter learning for multiple network alignment. In M. Vingron and L. Wong, editors, *Research in Computational Molecular Biology*, volume 4955 of *Lecture Notes in Computer Science*, pages 214–231. Springer Berlin Heidelberg.

Flannick, J., Novak, A. F., Do, C. B., Srinivasan, B. S., and Batzoglou, S. (2009). Automatic parameter learning for multiple local network alignment. *Journal of Computational Biology*, **16**(8), 1001–1022.

Franceschini, A., Szklarczyk, D., Frankild, S., Kuhn, M., Simonovic, M., Roth, A., Lin, J., Minguez, P., Bork, P., von Mering, C., and Jensen, L. J. (2013). String v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Research*, **41**(D1), D808–D815.

Fraser, H. B., Hirsh, A. E., Steinmetz, L. M., Scharfe, C., and Feldman, M. W. (2002). Evolutionary rate in the protein interaction network. *Science*, **296**(5568), 750–752.

Galil, Z. (1983). Efficient algorithms for finding maximal matching in graphs. In *Proceedings of the 8th Colloquium on Trees in Algebra and Programming*, CAAP '83, pages 90–113, London, UK, UK. Springer-Verlag.

Galil, Z., Micali, S., and Gabow, H. (1986). An $o(ev \log v)$ algorithm for finding a maximal weighted matching in general graphs. *SIAM Journal on Computing*, **15**, 120–130.

Gerstein, M. B., Kundaje, A., Hariharan, M., Landt, S. G., Yan, K.-K., Cheng, C., Mu, X. J., Khurana, E., Rozowsky, J., Alexander, R., Min, R., Alves, P., Abyzov, A., Addleman, N., Bhardwaj, N., Boyle, A. P., Cayting, P., Charos, A., Chen, D. Z., Cheng, Y., Clarke, D., Eastman, C., Euskirchen, G., Frietze, S., Fu, Y., Gertz, J., Grubert, F., Harmanci, A., Jain, P., Kasowski, M., Lacroute, P., Leng, J., Lian, J., Monahan, H., O'Geen, H., Ouyang, Z., Partridge, E. C., Patacsil, D., Pauli, F., Raha, D., Ramirez, L., Reddy, T. E., Reed, B., Shi, M., Slifer, T., Wang, J., Wu, L., Yang, X., Yip, K. Y., Zilberman-Schapira, G., Batzoglou, S., Sidow, A., Farnham, P. J., Myers, R. M., Weissman, S. M., and Snyder, M. (2012). Architecture of the human regulatory network derived from ENCODE data. *Nature*, **489**(7414), 91–100. PMID: 22955619.

Giot, L., Bader, J. S., Brouwer, C., Chaudhuri, A., Kuang, B., Li, Y., Hao, Y. L., Ooi, C. E., Godwin, B., Vitols, E., Vijayadamodar, G., Pochart, P., Machineni, H., Welsh, M., Kong, Y., Zerhusen, B., Malcolm, R., Varrone, Z., Collis, A., Minto, M., Burgess, S., McDaniel, L., Stimpson, E., Spriggs, F., Williams, J., Neurath, K., Ioime, N., Agee, M., Voss, E., Furtak, K., Renzulli, R., Aanensen, N., Carrolla, S., Bickelhaupt, E., Lazovatsky, Y., DaSilva, A., Zhong, J., Stanyon, C. A., Finley, R. L., White, K. P., Braverman, M., Jarvie, T., Gold, S., Leach, M., Knight, J., Shimkets, R. A., McKenna, M. P., Chant, J., and Rothberg, J. M. (2003). A protein interaction map of drosophila melanogaster. *Science*, **302**(5651), 1727–1736.

Goll, J. and Uetz, P. (2008). *Analyzing Protein Interaction Networks*, chapter 31, pages 1121–1177. Wiley-VCH Verlag GmbH.

Güldener, U., Münsterkötter, M., Oesterheld, M., Pagel, P., Ruepp, A., Mewes, H.-W., and Stümpflen, V. (2006). Mpact: the mips protein interaction resource on yeast. *Nucleic Acids Research*, **34**(suppl 1), D436–D441.

Gülsoy, G., Gandhi, B., and Kahveci, T. (2012). Topac: Alignment of gene regulatory networks using topology-aware coloring. *Journal of Bioinformatics and Computational Biology*, **10**(01), 1240001. PMID: 22809302.

Han, K., Park, B., Kim, H., Hong, J., and Park, J. (2004). Hpid: The human protein interaction database. *Bioinformatics*, **20**(15), 2466–2470.

Hartwell, L. H., Hopfield, J. J., Leibler, S., and Murray, A. W. (1999). From molecular to modular cell biology. *Nature*, **402**(6761 Suppl), C47–C52.

Ho, Y. (2002). Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry. *Nature*, **415**, 180–183.

Hopcroft, J. E. and Karp, R. M. (1973). An n5/2 algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, **2**, 225–231.

Hu, J. and Reinert, K. (2014). Localali: An evolutionary-based local alignment approach to identify functionally conserved modules in multiple networks. *Bioinformatics*.

Hu, J., Kehr, B., and Reinert, K. (2014). Netcoffee: a fast and accurate global alignment approach to identify functionally conserved proteins in multiple networks. *Bioinformatics*, **30**(4), 540–548.

Huang, Q., Wu, L.-Y., and Zhang, X.-S. (2013). Corbi: a new r package for biological network alignment and querying. *BMC Systems Biology*, **7**(Suppl 2), S6.

Huttenhower, C., Hibbs, M., Myers, C., and Troyanskaya, O. G. (2006). A scalable method for integration and functional analysis of multiple microarray datasets. *Bioinformatics*, **22**(23), 2890–2897.

Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M., and Sakaki, Y. (2001). A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proceedings of the National Academy of Sciences*, **98**(8), 4569–4574.

Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., and Barabasi, A. L. (2000). The large-scale organization of metabolic networks. *Nature*, **407**(6804), 651–654.

Kalaev, M., Smoot, M., Ideker, T., and Sharan, R. (2008). Networkblast: comparative analysis of protein networks. *Bioinformatics*, **24**(4), 594–596.

Kalaev, M., Bafna, V., and Sharan, R. (2009). Fast and Accurate Alignment of Multiple Protein Networks. *Journal of Computational Biology*, **16**(8), 989–999.

Kashtan, N., Itzkovitz, S., Milo, R., and Alon, U. (2004). Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, **20**(11), 1746–1758.

Kelley, B. P. (2003). Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc. Natl. Acad. Sci. USA*, **100**, 11394–11399.

Kelley, B. P., Yuan, B., Lewitter, F., Sharan, R., Stockwell, B. R., and Ideker, T. (2004). PathBLAST: a tool for alignment of protein interaction networks. *Nucleic Acids Research*, **32**(suppl 2), W83–W88.

Kerrien, S., Aranda, B., Breuza, L., Bridge, A., Broackes-Carter, F., Chen, C., Duesbury, M., Dumousseau, M., Feuermann, M., Hinz, U., Jandrasits, C., Jimenez, R. C., Khadake, J., Mahadevan, U., Masson, P., Pedruzzi, I., Pfeiffenberger, E., Porras, P., Raghunath, A., Roechert, B., Orchard, S., and Hermjakob, H. (2012). The intact molecular interaction database in 2012. *Nucleic Acids Research*, **40**(D1), D841–D846.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, **220**(4598), 671–680.

Koonin, E. V. (2005). Orthologs, paralogs, and evolutionary genomics1. *Annual Review of Genetics*, **39**(1), 309–338.

Koyutürk, M., Kim, Y., Topkara, U., Shankar, S., Szpankowski, W., and Ananth, G. (2006). Pairwise Local Alignment of Protein Interaction Networks Guided by Models of Evolution. *Journal of Computational Biology*, **13**(2).

Kuchaiev, O. and Pržulj, N. (2011). Integrative network alignment reveals large regions of global network similarity in yeast and human. *Bioinformatics*, **27**(10), 1390–1396.

Kuchaiev, O., Milenković, T., Memišević, V., Hayes, W., and Pržulj, N. (2010). Topological network alignment uncovers biological function and phylogeny. *Journal of The Royal Society Interface*, **7**(50), 1341–1354.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterty*, **2**, 83–97.

Lashkari, D. A., DeRisi, J. L., McCusker, J. H., Namath, A. F., Gentile, C., Hwang, S. Y., Brown, P. O., and Davis, R. W. (1997). Yeast microarrays for genome wide parallel genetic and gene expression analysis. *Proceedings of the National Academy of Sciences*, **94**(24), 13057–13062.

Li, S., Armstrong, C. M., Bertin, N., Ge, H., Milstein, S., Boxem, M., Vidalain, P.-O., Han, J.-D. J., Chesneau, A., Hao, T., Goldberg, D. S., Li, N., Martinez, M., Rual, J.-F., Lamesch, P., Xu, L., Tewari, M., Wong, S. L., Zhang, L. V., Berriz, G. F., Jacotot, L., Vaglio, P., Reboul, J., Hirozane-Kishikawa, T., Li, Q., Gabel, H. W., Elewa, A., Baumgartner, B., Rose, D. J., Yu, H., Bosak, S., Sequerra, R., Fraser, A., Mango, S. E., Saxton, W. M., Strome, S., van den Heuvel, S., Piano, F., Vandenhaute, J., Sardet, C., Gerstein, M., Doucette-Stamm, L., Gunsalus, K. C., Harper, J. W., Cusick, M. E., Roth, F. P., Hill, D. E., and Vidal, M. (2004). A map of the interactome network of the metazoan c. elegans. *Science*, **303**(5657), 540–543.

Li, Z., Zhang, S., Wang, Y., Zhang, X.-S., and Chen, L. (2007). Alignment of molecular networks by integer quadratic programming. *Bioinformatics*, **23**(13), 1631–1639.

Liao, C.-S., Lu, K., Baym, M., Singh, R., and Berger, B. (2009). IsoRankN: spectral methods for global alignment of multiple protein networks. *Bioinformatics*, **25**(12), i253–i258.

Licata, L., Briganti, L., Peluso, D., Perfetto, L., Iannuccelli, M., Galeota, E., Sacco, F., Palma, A., Nardozza, A. P., Santonico, E., Castagnoli, L., and Cesareni, G. (2012). MINT, the molecular interaction database: 2012 update. *Nucleic Acids Research*, **40**(D1), D857–D861.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Liu, Z., Liu, Q., Sun, H., Hou, L., Guo, H., Zhu, Y., Li, D., and He, F. (2011). Evidence for the additions of clustered interacting nodes during the evolution of protein interaction networks from network motifs. *BMC Evolutionary Biology*, **11**(1), 133.

Ma, C.-Y., Lin, S.-H., Lee, C.-C., Tang, C. Y., Berger, B., and Liao, C.-S. (2013). Reconstruction of phyletic trees by global alignment of multiple metabolic networks. *BMC Bioinformatics*, **14**(Suppl 2), S12.

Magrane, M. and Consortium, U. (2011). Uniprot knowledgebase: a hub of integrated protein data. *Database*, **2011**.

Mann, M., Hendrickson, R. C., and Pandey, A. (2001). Protein-protein interactions: methods for detection and analysis. *Annual Review of Biochemistry*, **70**, 437–473.

Marcotte, E. M., Pellegrini, M., Ng, H.-L., Rice, D. W., Yeates, T. O., and Eisenberg, D. (1999). Detecting protein function and protein-protein interactions from genome sequences. *Science*, **285**(5428), 751–753.

Mehlhorn, K. and Schäfer, G. (2002). Implementation of o(nmlogn) weighted matchings in general graphs: The power of data structures. *J. Exp. Algorithmics*, **7**, 4–.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, **21**(6), 1087–1092.

Milenković, T., Ng, W., Hayes, W., and Pržulj, N. (2010). Optimal network alignment with graphlet degree vectors. *Cancer informatics*, **9**, 121–37.

Milenković, T., Zhao, H., and Faisal, F. E. (2013). Global network alignment in the context of aging. In *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, BCB'13, pages 23:23–23:32, New York, NY, USA. ACM.

Milo, R. (2002). Network motifs: simple building blocks of complex networks. *Science*, **298**, 824–827.

Mora, C., Tittensor, D. P., Adl, S., Simpson, A. G. B., and Worm, B. (2011). How many species are there on earth and in the ocean? *PLoS Biol*, **9**(8), e1001127.

Morgenstern, B. (1999). Dialign 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, **15**(3), 211–218.

Morrison, J. L., Breitling, R., Higham, D. J., and Gilbert, D. R. (2006). A lock-and-key model for proteinprotein interactions. *Bioinformatics*, **22**(16), 2012–2019.

Navlakha, S., Schatz, M. C., and Kingsford, C. (2009). Revealing biological modules via graph summarization. *Journal of Computational Biology*, **16**(2), 253–264.

Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys. Rev. E*, **69**, 026113.

Neyshabur, B., Khadem, A., Hashemifar, S., and Arab, S. S. (2013). Netal: a new graph-based method for global alignment of proteinprotein interaction networks. *Bioinformatics*, **29**(13), 1654–1662.

Notredame, C., Higgins, D. G., and Heringa, J. (2000). T-coffee: a novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, **302**(1), 205 – 217.

O'Brien, K. P., Remm, M., and Sonnhammer, E. L. L. (2005). Inparanoid: a comprehensive database of eukaryotic orthologs. *Nucleic Acids Research*, **33**(suppl 1), D476–D480.

Pache, R. A. and Aloy, P. (2012). A novel framework for the comparative analysis of biological networks. *PLoS ONE*, **7**(2), e31220.

Pache, R. A., Céol, A., and Aloy, P. (2012). Netaligner—a network alignment server to compare complexes, pathways and whole interactomes. *Nucleic Acids Research*, **40**(W1), W157–W161.

Pagani, I., Liolios, K., Jansson, J., Chen, I.-M. A., Smirnova, T., Nosrat, B., Markowitz, V. M., and Kyrpides, N. C. (2012). The genomes online database (gold) v.4: status of genomic and metagenomic projects and their associated metadata. *Nucleic Acids Research*, **40**(D1), D571–D579.

Pagel, P., Kovac, S., Oesterheld, M., Brauner, B., Dunger-Kaltenbach, I., Frishman, G., Montrone, C., Mark, P., Stmpflen, V., Mewes, H.-W., Ruepp, A., and Frishman, D. (2005). The mips mammalian proteinprotein interaction database. *Bioinformatics*, **21**(6), 832–834.

Park, D., Singh, R., Baym, M., Liao, C.-S., and Berger, B. (2011). IsoBase: a database of functionally related proteins across PPI networks. *Nucleic Acids Research*, **39**(suppl 1), D295–D300.

Pastor-Satorras, R., Smith, E., and Solé, R. V. (2003). Evolving protein interaction networks through gene duplication. *Journal of Theoretical Biology*, **222**(2), 199 – 210.

Patro, R. and Kingsford, C. (2012). Global network alignment using multiscale spectral signatures. *Bioinformatics*, **28**(23), 3105–3114.

Patro, R. and Kingsford, C. (2013). Predicting protein interactions via parsimonious network history inference. *Bioinformatics*, **29**(13), i237–i246.

Patro, R., Sefer, E., Malin, J., Marcais, G., Navlakha, S., and Kingsford, C. (2012). Parsimonious reconstruction of network evolution. *Algorithms for Molecular Biology*, **7**(1), 25.

Pazos, F. and Sternberg, M. J. E. (2004). Automated prediction of protein function and detection of functional sites from structure. *Proceedings of the National Academy of Sciences of the United States of America*, **101**(41), 14754–14759.

Pe'er, D. and Hacohen, N. (2011). Principles and strategies for developing network models in cancer. *Cell*, **144**(6), 864 – 873.

Pellegrini, M., Marcotte, E. M., Thompson, M. J., Eisenberg, D., and Yeates, T. O. (1999). Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles. *Proceedings of the National Academy of Sciences*, **96**(8), 4285–4288.

Phan, H. T. T. and Sternberg, M. J. E. (2012). Pinalog: a novel approach to align protein interaction networks implications for complex detection and function prediction. *Bioinformatics*, **28**(9), 1239–1245.

Phizicky, E. M. and Fields, S. (1995). Protein-protein interactions: methods for detection and analysis. *Microbiological Reviews*, **59**(1), 94–123.

Pinkert, S., Schultz, J., and Reichardt, J. (2010). Protein interaction networksmore than mere modules. *PLoS Comput Biol*, **6**(1), e1000659.

Pinter, R. Y., Rokhlenko, O., Yeger-Lotem, E., and Ziv-Ukelson, M. (2005). Alignment of metabolic pathways. *Bioinformatics*, **21**(16), 3401–3408.

Radivojac, P. (2013). A (not so) quick introduction to protein function prediction.

Radivojac, P., Clark, W. T., Oron, T. R., Schnoes, A. M., Wittkop, T., Sokolov, A., Graim, K., Funk, C., Verspoor, K., Ben-Hur, A., Pandey, G., Yunes, J. M., Talwalkar, A. S., Repo, S., Souza, M. L., Piovesan, D., Casadio, R., Wang, Z., Cheng, J., Fang, H., Gough, J., Koskinen, P., Toronen, P., Nokso-Koivisto, J., Holm, L., Cozzetto, D., Buchan, D. W., Bryson, K., Jones, D. T., Limaye, B., and and, H. I. (2013). A large-scale evaluation of computational protein function prediction. *Nat Methods*, **10**(3), 221–7+.

Raicu, I., Foster, I., Zhao, Y., and Szalay, A. (2012). Towards data intensive many-task computing, under review as a book chapter. In *in Data Intensive Distributed Computing: Challenges and Solutions for Large-Scale Information Management, IGI Global Publishers, 2009*.

Rajasekaran, S. (1990). On the convergence time of simulated annealing. Technical report, University of Pennsylvania.

Rarey, M., Degen, J., and Reulecke, I. (2008). *Docking and Scoring for Structure-based Drug Design*, chapter 16, pages 541–599. Wiley-VCH Verlag GmbH.

Remm, M., Storm, C. E., and Sonnhammer, E. L. (2001). Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *Journal of Molecular Biology*, **314**(5), 1041 – 1052.

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, IJCAI'95, pages 448–453, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Rhodes, D. R. (2005). Probabilistic model of the human protein-protein interaction network. *Nat. Biotechnol.*, **23**, 951–959.

Roguev, A., Bandyopadhyay, S., Zofall, M., Zhang, K., Fischer, T., Collins, S. R., Qu, H., Shales, M., Park, H.-O., Hayles, J., Hoe, K.-L., Kim, D.-U., Ideker, T., Grewal, S. I., Weissman, J. S., and Krogan, N. J. (2008). Conservation and rewiring of functional modules revealed by an epistasis map in fission yeast. *Science*, **322**(5900), 405–410.

Rose, P. W., Bi, C., Bluhm, W. F., Christie, C. H., Dimitropoulos, D., Dutta, S., Green, R. K., Goodsell, D. S., Prli, A., Quesada, M., Quinn, G. B., Ramos, A. G., Westbrook, J. D., Young, J., Zardecki, C., Berman, H. M., and Bourne, P. E. (2013). The rcsb protein data bank: new resources for research and education. *Nucleic Acids Research*, **41**(D1), D475–D482.

Royer, L., Reimann, M., Andreopoulos, B., and Schroeder, M. (2008). Unraveling protein networks with power graph analysis. *PLoS Comput Biol*, **4**(7), e1000108.

Ruepp, A., Waegele, B., Lechner, M., Brauner, B., Dunger-Kaltenbach, I., Fobo, G., Frishman, G., Montrone, C., and Mewes, H.-W. (2010). Corum: the comprehensive resource of mammalian protein complexes-2009. *Nucleic Acids Research*, **38**(suppl 1), D497–D501.

Sahraeian, S. M. E. and Yoon, B.-J. (2013). Smetana: Accurate and scalable algorithm for probabilistic alignment of large-scale biological networks. *PLoS ONE*, **8**(7), e67995.

Saito, R., Suzuki, H., and Hayashizaki, Y. (2003). Construction of reliable proteinprotein interaction networks with a new interaction generality measure. *Bioinformatics*, **19**(6), 756–763.

Salwinski, L., Miller, C. S., Smith, A. J., Pettit, F. K., Bowie, J. U., and Eisenberg, D. (2004). The database of interacting proteins: 2004 update. *Nucleic Acids Research*, **32**(suppl 1), D449–D451.

Saraph, V. and Milenković, T. (2014). Magna: Maximizing accuracy in global network alignment. *Bioinformatics*.

Satuluri, V. and Parthasarathy, S. (2009). Scalable graph clustering using stochastic flows: Applications to community discovery. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 737–746, New York, NY, USA. ACM.

Schlicker, A. and Albrecht, M. (2008). FunSimMat: a comprehensive functional similarity database. *Nucleic Acids Research*, **36**(suppl 1), D434–D439.

Schlicker, A., Domingues, F., Rahnenfuhrer, J., and Lengauer, T. (2006). A new measure for functional similarity of gene products based on gene ontology. *BMC Bioinformatics*, **7**(1), 302.

Schlicker, A., Rahnenfuhrer, J., Albrecht, M., Lengauer, T., and Domingues, F. (2007). GOTax: investigating biological processes and biochemical activities along the taxonomic tree. *Genome Biology*, **8**(3), R33.

Schopf, J. W. and Packer, B. M. (1987). Early Archean (3.3-billion to 3.5-billion-year-old) microfossils from Warrawoona Group, Australia. *Science*, **237**(4810), 70–73.

Schreiber, F. and Schwbbermeyer, H. (2005). Mavisto: a tool for the exploration of network motifs. *Bioinformatics*, **21**(17), 3572–3574.

Sharan, R. (2005). Conserved patterns of protein interaction in multiple species. *Proc. Natl. Acad. Sci. USA*, **102**, 1974–1979.

Sharan, R. and Ideker, T. (2006). Modeling cellular machinery through biological network comparison. *Nat Biotech*, **24**(4), 427–433.

Shen-Orr, S. S., Milo, R., Mangan, S., and Alon, U. (2002). Network motifs in the transcriptional regulation network of escherichia coli. *Nature Genetics*, **31**, 1061–4036.

Shih, Y.-K. and Parthasarathy, S. (2012). Scalable global alignment for multiple biological networks. *BMC Bioinformatics*, **13**(Suppl 3), S11.

Singh, R., Xu, J., and Berger, B. (2007). Pairwise global alignment of protein interaction networks by matching neighborhood topology. In *Proceedings of the 11th annual international conference on Research in computational molecular biology*, RECOMB'07, pages 16–31, Berlin, Heidelberg. Springer-Verlag.

Singh, R., Xu, J., and Berger, B. (2008). Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences*, **105**(35), 12763–12768.

Spirin, V. and Mirny, L. A. (2003). Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences*, **100**(21), 12123–12128.

Szklarczyk, D., Franceschini, A., Kuhn, M., Simonovic, M., Roth, A., Minguez, P., Doerks, T., Stark, M., Muller, J., Bork, P., Jensen, L. J., and Mering, C. v. (2011). The string database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Research*, **39**(suppl 1), D561–D568.

Tatusov, R. L., Koonin, E. V., and Lipman, D. J. (1997). A genomic perspective on protein families. *Science*, **278**(5338), 631–637.

Tatusov, R. L., Galperin, M. Y., Natale, D. A., and Koonin, E. V. (2000). The COG database: a tool for genome-scale analysis of protein functions and evolution. *Nucleic Acids Research*, **28**(1), 33–36.

The ENCODE Project Consortium (2012). An integrated encyclopedia of DNA elements in the human genome. *Nature*, **489**(7414), 57–74. PMID: 22955616.

Uetz, P., Giot, L., Cagney, G., Mansfield, T. A., Judson, R. S., Knight, J. R., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleisch, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S., and Rothberg, J. M. (2000). A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. *Nature*, **403**(6770), 623–627.

Uniprot Consortium (2007). The universal protein resource (UniProt). *Nucleic Acids Research*, **35**(suppl 1), D193–D197.

Vazquez, A., Flammini, A., Maritan, A., and Vespignani, A. (2003a). Global protein function prediction from protein-protein interaction networks. *Nature Biotechnology*, **21**(6), 697–700.

Vazquez, A., Flammini, A., A., M., and A., V. (2003b). Modeling of protein interaction networks. *ComPlexUs*, **1**, 38–44.

Wagner, A. (2003). How the global structure of protein interaction networks evolves. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, **270**(1514), 457–466.

Wang, Y. and Qian, X. (2012). Functional module identification by block modeling using simulated annealing with path relinking. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, BCB '12, pages 123–130, New York, NY, USA. ACM.

Waterhouse, R. M., Zdobnov, E. M., Tegenfeldt, F., Li, J., and Kriventseva, E. V. (2011). OrthoDB: the hierarchical catalog of eukaryotic orthologs in 2011. *Nucleic Acids Research*, **39**(suppl 1), D283–D288.

Wernicke, S. (2006). Efficient detection of network motifs. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, **3**(4), 347–359.

Wuchty, S., Oltvai, Z. N., and Barabasi, A. L. (2003). Evolutionary conservation of motif constituents in the yeast protein interaction network. *Nat Genet*, **35**(2), 176–179.

Zaslavskiy, M., Bach, F., and Vert, J.-P. (2009). Global alignment of proteinprotein interaction networks by graph matching methods. *Bioinformatics*, **25**(12), i259–1267.

Zhang, X. and Moret, B. (2008). Boosting the performance of inference algorithms for transcriptional regulatory networks using a phylogenetic approach. In K. Crandall and J. Lagergren, editors, *Algorithms in Bioinformatics*, volume 5251 of *Lecture Notes in Computer Science*, pages 245–258. Springer Berlin Heidelberg.

Glossary

assignment problem

Given two sets, A and T, of equal size, together with a weight function $C : A \times T \to R$. Find a bijection $f : A \to T$ such that the cost function: $\sum_{a \in A} C(a, f(a))$ is minimized. 20

d-subnet

A set of d *k-spines* connected by interactions form a *d-subnet*.. 29–32, 34, 35, 44–48

de novo

De novo may be a term used to define methods for making predictions about biological features using only a computational model without extrinsic comparison to existing data. 5

DNA

Deoxyribonucleic acid (DNA) is a molecule that encodes the genetic instructions used in the development and functioning of all known living organisms and many viruses. DNA consists of two long chains of nucleotides twisted into a double helix and joined by hydrogen bonds between the complementary bases adenine and thymine or cytosine and guanine. 3, 10

Dupont Merck

Dupont Merck is a pharmaceutical company.. 5

functional module

A functional module is, by definition, a discrete entity whose function is separable from those of other modules (Hartwell *et al.*, 1999). They are usually separated based on spatial localization (e.g. a ribosome) or chemical specificity (e.g. a signal transduction system) and, composed of many types of molecule, such as proteins, DNA, RNA and small molecules. 10, 25, 26, 28, 33, 34, 45, 48

functional ortholog

Functional ortholog (FO), also known as function-oriented ortholog group, consists of orthologs that play the functionally equivalent roles in different species and also include recent paralogs with the same biological function (i.e. *inparalogs*) (Remm *et al.*, 2001). 11

HGP

Human Genome Project. 3

in silico

Studies that are in silico is performed on computer or via computer simulation. 10, 18

in vitro

Studies that are in vitro are performed with cells or biological molecules studied outside their normal biological context. 10

in vivo

Studies that are in vivo are those in which the effects of various biological entities are tested on whole, living organisms usually animals including humans, and plants. 7, 10, 18

k-spine

A set of k proteins, each from one species, which are connected by *edges* is termed as a *k-spine*.. 17, 22, 31–34

maximum parsimony

In the application of computational phylogenetics, maximum parsimony describes a particular non-parametric statistical method for constructing phylogenies. In this application, the preferred phylogenetic tree is the tree that supposes the least evolutionary change to explain observed data. 25, 30, 33

metabolic pathway

Metabolic pathways are series of chemical reactions occurring within a cell. 10, 16

natural selection

Natural selection is the gradual process by which biological traits become either more or less common in a population as a function of the effect of inherited traits on the differential reproductive success of organisms interacting with their environment. It is a key mechanism of evolution. 11

NGS

Next-generation sequencing, also called high-throughput sequencing technologies, can parallelize the sequencing process, producing thousands or millions of sequence concurrently. 3

ortholog

Orthologs are genes/proteins derived from a single ancestral gene in the last common ancestor of the compared species (Koonin, 2005; Park *et al.*, 2011). 11

paralog

Paralogs are genes/proteins related via duplication. 11

PPI

Protein-protein interactions (PPIs) refer to intentional physical contacts established between two or more proteins as a result of biochemical events or electrostatic forces. 6

protein

Proteins are biological macromolecules which are formed by linear chains of amino acids connected by covalent (peptide) bonds. 6, 10

protein complex

A protein complex (or multiprotein complex) is a group of two or more associated polypeptide chains. Proteins in a protein complex are linked by non-covalent protein-protein interactions, and different protein complexes have different degrees of stability over time. 6, 7, 9, 10, 16, 45, 48

RNA

Ribonucleic acid (RNA) is a ubiquitous family of large biological molecules that perform multiple vital roles in the coding, decoding, regulation and expression of genes. 10

scale-free network

A scale-free network is a network whose degree distribution follows a power law, $P(k) \sim k^{-\gamma}$. 8, 72

Y2H

Yeast two-hybrid (Y2H) system (also known as two-hybrid screening) is a molecular biology technique used to discover protein-protein interactions and protein-DNA interactions by testing for physical interactions (such as binding) between two proteins or a single protein and a DNA molecule, respectively. 6

Index

A

activating domain, 6
affinity purification, 6, 7
alignment graph, 11
amino acids, 6
assignment problem, 20
attack vulnerability, 8
augmenting path, **20**, 20

B

binding domain, 6
BioGRID, 7
bipartite graphs, 67–73

C

clustering algorithms, 10, 11
co-evolution, 15, 19
co-expression, 15
co-inheritance, 15
co-location, 15
COGs, 12
CORUM, 7

D

d-subnets, 17, 29–31, 34, 35, 91
DIP, 7
DNA, **3**, 3–6, 10
drug design, 12

E

ENCODE, **4**
equivalence class, 15, 17
error tolerance, 8
ESA, 21
ESU, 22
ESU-RAND, 21, 39
evolutionary
  distance, 27–30, 33, 34
  events, 25, 27, 34, 91
  model, 8, 25–27, 30
  rates, 28
  tree, 26, 29, 30, 33, 34

F

false
  negatives, 7
  positives, 7
functional
  annotations, 91
  modularity, 91
  modules, **9**, 10, 17, 18, 25–28, 33, 34, 92
  orthologs, 11, 12, 16, 18

G

gene duplications, 8
global network alignment, **17**, 19, 67, 68

graph matching, 15, 19

H

high-throughput technologies, **3**, 3, 15, 18
hits, 25
HIV, 5
HIV-protease, 5
homology model, 69
HPID, 7
hub nodes, 8
Human Genome Project, 3

I

inparalogs, 11, 112
Inparanoid, 12
IntAct, 7
interaction
  deletion, 27, 28
  insertion, 27, 28

K

k-layer graph, 16, **30**
k-spines, 16, **17**, 17, 31–34

L

local network alignment, 10, 11, **17**, 17, 18, 25, 31
log-ratio, 69, 71

M

many-to-many, 15, 17
mass spectrometry, 3, **6**, 18
match-sets, 17, 68, 71, 73, 75, 80, 81, 83, 84, 86–88
maximum matching, 19, 20
maximum parsimony, 30, 33
maximum weighted matching, 68, 73
metabolic networks, 7
metropolis scheme, 34
microarrays, 3, 12, 18
MINT, 7
miRNAs, 7
MPACT, 7
MPPI, 7
multiple local alignment, 16

N

network alignment, **15**, 15, 16, **19**
network motif, 9
next-generation sequencing, 3, 18
nuclear magnetic resonance, 5
null model, 69

O

one-to-one, 15, 17
OrthoDB, 12
orthologous proteins, 10, 11, 69
orthologs, 11

P

pairwise local alignment, 16
parallel computing, 25, 91
paralogs, 11
pathway, 26
phylogenetic tree, 67
phylogeny reconstruction, 5
post-genomic era, 4, 12
proteases, 6
protein
  complexes, 7, 9, 91
  duplication, 27, 28
  kinases, 6
  mutation, 27
  phosphatases, 6
Protein Data Bank, 5

protein function prediction, 5, **12**
protein-protein interactions, **6**, 6, 79
proteins, 4, 5, **6**, 6–13
pseudo-genes, 12
pure modules, 91

R
reciprocal-best-BLAST-hits, 12, 19
regulatory networks, 4, 7, 9
reporter genes, 6, 7
RNA, 10

S
scalability, 25, 45, 46, 49
scale-free
  architectures, 8
  networks, **8**
scientific collaboration networks, 92
seed-and-extend, 30, 31
sequence alignment, 5, 15
sequenced genomes, 6, 12
signal transduction system, 9
simulated annealing, 33, 68, 92
stable interactions, 6, 7
STRING, 7
structural complex, 26
subgraph searching, 15, 19

T
tandem affinity purification, 7
threshold, 31, 35
transcription factor, 4, **6**
transient interactions, 6, 7
triplet matches, 70–72
triplets, 70, 71

U
upstream activating sequence, 6

W
world wide web networks, 92

Y
yeast enhancer, 6
yeast two-hybrid, 3, **6**, 18

Appendix A

Softwares

The two network alginment algorithms described in chapter 3 and chapter 4 of this thesis have been implemented and freely available. Parts of the graph data structures and some graph algorithms such as *maximum weighted graph matching* completely depend on the *LEMON Graph Library* (Dezső *et al.*, 2011), and parallelization techniques largely depends on the OpenMP (Chapman *et al.*, 2007). Both of the two packages support our algorithms as well as the analyzing function which are used to evaluate the alignment results. A guide tour of how to quickly start using our tools on the test datasets is described in the following sections.

A.1 LocalAli

A.1.1 Basic information

Package: *LocalAli*
Language: C++
Support: Windows, Mac OS, Linux
Compilors: g++-4.6, Visual C++ 11.0 or higher
Dependencies: LEMON, OpenMP
Additional dependencies: GO-TermFinder, and some shell scripts
Availability: `https://code.google.com/p/localali/`

A.1.2 How to get started

For easy to explain, we denote $LOCALALI as the path to the LocalAli folder.
Download
First checkout the source code from the subversion repository using the following command: *svn checkout http://localali.googlecode.com/svn/trunk/* $LOCALALI Then, download the

dataset and the lemon library from the following two links, uncompress dataset.tar.gz into the folder $LOCALALI, and uncompress lemon-1.2.3.tar.gz into the folder $LOCALALI/include/:
`http://ftp.mi.fu-berlin.de/jhu/LocalAli/dataset.tar.gz`
`http://ftp.mi.fu-berlin.de/jhu/LocalAli/lemon-1.2.3.tar.gz`

Compile the LEMON GRAPH LIBRARY

```
cd $LOCALALI/include/lemon-1.2.3/
./configure
./make
./make check (optional)
./make install (optional)
```

Compile LocalAli

```
cd $LOCALALI
./make MODE=Release
```

If you want to compile it in Debug mode, run command:

```
./make (MODE=Debug)
```

The binary code will be in the folder $LOCALALI after compilation. If you want to compile it with other compilers such as g++-4.7, do it like this:

```
./make MODE=Release CXX=g++-4.7
```

Run an example

```
cd $LOCALALI
mkdir ./result
mkdir ./result/3-way
mkdir ./result/3-way/alignments
```

set the directory of the dataset in ./profile/profile_test.txt.

```
./localali -alignment -parallel -numthreads 16 -numspinetries 20 -numspecies 3 -numseeds 2000 -score_threshold 0.3 -resultfolder ./result/3-way -seedtries 2 -minext 3 -maxext 13 -profile ./profile/profile_test.txt -verbose 1 -extdist1 2 -extdist2 2 -seedrep 1
```

A.2 NetCoffee

A.2.1 Basic information

Package: *NetCoffee*
Language: C++
Support: Windows, Mac OS, Linux
Compilors: g++-4.6, Visual C++ 11.0 or higher

Dependencies: LEMON, OpenMP
Additional dependencies: GO-TermFinder, FSST and some shell scripts
Availability: `https://code.google.com/p/netcoffee/`

A.2.2 How to get started

For easy to explain, we denote $NETCOFFEE as the path to the NetCoffee folder.

Download

First checkout the source code from the subversion repository using the following command:

```
svn checkout http://netcoffee.googlecode.com/svn/trunk/ $NETCOFFEE
```

Then, download the dataset and the lemon library from the following two links, uncompress dataset.tar.gz into the folder $NETCOFFEE, and uncompress lemon-1.2.3.tar.gz into the folder $NETCOFFEE/include/:

`https://netcoffee.googlecode.com/files/dataset.tar.gz`
`https://netcoffee.googlecode.com/files/lemon-1.2.3.tar.gz`

Compile the LEMON GRAPH LIBRARY

```
cd $NETCOFFEE/include/lemon-1.2.3/
./configure
./make
```

Compile NetCoffee

```
cd $NETCOFFEE
./make MODE=Release
```

If you want to compile it in Debug mode, run command:

```
./make (MODE=Debug)
```

Run an example

```
./bin/netcoffee -alignment -task 1 -out -alpha $ALPHA -alignmentfile ./result/alignment_netcoffee.data -resultfolder ./result/
```

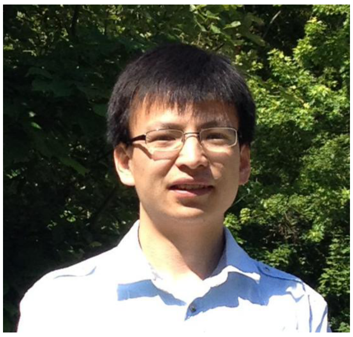
$ALPHA is the parameter you want to specify for α.

Appendix B

Curriculum Vitae

Jialu Hu

Department of mathematics and computer science,
Freie Universität Berlin
Takustraße 9, Room 009
14195, Berlin
Email: *jialu.hu@fu-berlin.de* or
hujialu.xd@gmail.com



Academic History

B.Sc. (2004–2008) in Computer Science And **M.Sc.** (2008-2010) in Bioinformatics, *Department of Computer Science, Xidian University, Xi'an, P.R. China*
Ph.D. (2010–2014) in Bioinformatics, *Department of Mathematics and Computer Science, Freie Universität Berlin, Berlin, Germany*

Research Interests

i) Analyses of biological networks
  Network alignment[1], *Network motif, Clustering*
ii) Theory of computation
  Complexity of graph isomorphism
iii) Analyses of *next-generation sequencing* data

---

[1] Major contributions of my Ph.D. work

Genome comparison, variant detection

Research Activity/Association Membership

Dec. 2011, visit Gunnar W. Klau's Lab at *Centrum Wiskunde* & *Informatica* (CWI).
July 2013, attend the international conference of *ISMB/ECCB* 2013 in Berlin.
2010–2014, attend the SeqAn-BioStore Workshop every year.
2013–2015, *International Society for Computational Biology* (ISCB) membership.
2014, reviewer for *Plos Computational Biology* and *Bioinformatics.*

Thesis

Master thesis, 2010, *An algorithm of graph isomorphism and its application in the identification of network motif*, supervised by *Prof. Lin Gao*
Ph.D. thesis, 2014, *Algorithms to identify functional orthologs and functional modules from high-throughput data*, supervised by *Prof. Knut Reinert* and *Prof. Gunnar Klau*

Poster Presentations in Peer Reviewed Conference

Jialu Hu, Birte Kehr, and Knut Reinert, *M-NetAligner: a novel global alignment approach to identify functional orthologs in multiple networks, 17th Annual International Conference on Research in Computational Molecular Biology* (RECOMB) 2013, Beijing, 7 - 10 Apr 2013, P207

Publications in Peer Reviewed Journals

Jialu Hu, Birte Kehr, and Knut Reinert, *NetCoffee: a fast and accurate global alignment approach to identify functionally conserved proteins in multiple networks, Bioinformatics* (2014) 30 (4): 540-548
Jialu Hu, and Knut Reinert, *LocalAli: an evolutionary-based local alignment approach to identify functionally conserved modules in multiple networks*, Bioinformatics (2014) doi: 10.1093/bioinformatics/btu652

Unpublished Papers

Jialu Hu, and Knut Reinert, *MNetAli: a web server for multiple alignment of protein-protein networks*, (unpublished)
Jialu Hu, Enrico Siragusa and Knut Reinert, *An improved algorithm on graph canonization problem*, (unpublished)

Development of Software Tools And Web Servers

NetCoffee (CLI)
    `https://code.google.com/p/netcoffee/`
LocalAli
    `https://code.google.com/p/localali/`
MNetAli (Web Server)
    `http://page.mi.fu-berlin.de/jhu/index.php`
SGIP (SeqAn application)
    `https://github.com/seqan/seqan/tree/master/extras/apps/sgip`

Honours/Awards/Scholarship

4-year CSC scholarship
5-year school scholarship in Xidian University
Second prize in Shaanxi Province (China) in *the 8th National Undergraduate Electronic Design Contest*
First prize in *the twentieth "Xin Huo Cup" Science and Technology Contest*
2007 Annual Excellent Learning Model in Xidian University

Personal Interests

Favorite sports
    Running, Pingpong, Basketball, Soccer
Favorite books
    ≪Three Kingdoms≫, ≪Phaedo≫

Appendix C

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Sep. 9th, 2014
Jialu Hu