KADIR HAS UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINERRING

MESSAGE-PASSING BASED ALGORITHM FOR THE GLOBAL ALIGNMENT OF CLUSTERED PAIRWISE PPI NETWORKS

GRADUATE THESIS

DOĞAN YİĞİT YENİGÜN

December, 2013

MESSAGE-PASSING BASED ALGORITHM FOR THE GLOBAL ALIGNMENT OF CLUSTERED PAIRWISE PPI NETWORKS

DOĞAN YİĞİT YENİGÜN

Submitted to the Gradute School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science
in
COMPUTER ENGINEERING

KADIR HAS UNIVERSITY
December, 2013

KADIR HAS UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

MESSAGE-PASSING BASED ALGORITHM FOR THE GLOBAL ALIGNMENT OF CLUSTERED PAIRWISE PPI NETWORKS

DOĞAN YİĞİT YENİGÜN

APPROVED BY:

(Title and Name) (Advisor) (Affiliation) Cesim Erten

(Title and Name) (Co-advisor) (Affiliation) _____

(Title and Name) Y. Doç. Dr. Tınaz Ekim (Affiliation) Boğaziçi Üniv.

(Title and Name) Y. Doç. Dr. Sebnem Esra Bozkaya (Affiliation) _____

(Title and Name) (Affiliation) _____

APPROVAL DATE: / /

"I, Doğan Yiğit Yenigün, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis."

DOĞAN YİĞİT YENİGÜN

ABSTRACT

MESSAGE-PASSING BASED ALGORITHM FOR THE GLOBAL ALIGNMENT OF CLUSTERED PAIRWISE PPI NETWORKS

Doğan Yiğit Yenigün

Master of Science in Computer Engineering

Advisor: Assoc. Prof. Cesim Erten

December, 2013

Constrained global network alignments on pairwise protein-protein interaction (PPI) networks involve matchings between two organisms where proteins are grouped together in a great number of clusters, produced by algorithms that seek functionally ortholog ones and these organisms are represented as graphs. Unlike balanced global network alignments (GNA), this has not gained much popularity in bioinformatics. Only a few methods have been proposed thus far; by assuming specific structures of networks including the clusters themselves and the density of the PPI networks are not too large, then optimal alignments can be encountered. Here, we introduce a general-purpose algorithm that is able to work on any kind of graph structures while taking advantage of the message-passing method, based on propagation between clusters. When these graphs satisfy conditions like continuous interaction connectivity of proteins across all neighbored clusters, in addition to previous explanations, the optimality of alignments can still be achieved. Convergence of the cluster network can occur at the point where the maximum number of conserved interactions are detected. Many experiments were made with balanced GNA algorithms and our algorithm may find more conservations and more importantly, alignments have higher biological quality than other ones in various instances.

Keywords: network alignment, graphs, message-passing, clustering

ÖZET

KÜMELENMİŞ İKİLİ PROTEİN-PROTEİN ETKİLEŞİM AĞLARININ GLOBAL HİZALANMASI İÇİN MESAJ VERMEYE DAYALI ALGORİTMA

Doğan Yiğit Yenigün

Bilgisayar Mühendisliği, Yüksek Lisans

Danışman: Doç. Dr. Cesim Erten

Aralık, 2013

İkili protein-protein etkileşim ağları üzerinde kısıtlanmış global ağ hizalaması, işlevsel olarak ortak proteinleri arayan algoritmalar tarafından üretilen çok sayıdaki küme içerisinde gruplanmış olan iki organizmanın proteinleri arasında en iyi eşleşmeleri içerir ve bu organizmalar graph yapısı olarak gösterilirler. Dengeli global ağ hizalamanın aksine biyoenformatik alanında fazla popülerlik kazanmamıştır. Şu ana kadar sadece birkaç yöntem önerilmiştir; kümelerin kendileri de dahil özel ağ yapıları ve protein-protein etkileşim ağlarının yoğunluğunun çok büyük olmadığı varsayılırsa, en iyi hizalamalarla karşılaşılabilir. Burada, her tür graph yapısı üzerinde çalışabilen ve kümeler arasında yayılıma dayalı mesaj verme yönteminden faydalanan genel amaçlı bir algoritmayı sunuyoruz. Bu graphlar önceki varsayımlarla beraber birbirine komşu tüm kümeler boyunca proteinlerin devamlı etkileşim bağlantıları olması gibi koşulları sağlarlarsa, hizalamaların en iyisine halen ulaşılabilir. En çok sayıda korunmuş etkileşimlerin bulunduğu noktada küme ağının yakınsaması meydana gelebilir. Dengeli global ağ hizalama algoritmaları ile birçok deney yapılmıştır ve bizim algoritmamız diğerlerinden daha fazla korunmuş etkileşimi bulabilir ve daha da önemlisi, değişik örneklerde hizalamalar daha yüksek biyolojik kaliteye sahip olabilir.

Anahtar Kelimeler: ağ hizalama, graphlar, mesaj verme, kümeleme

Acknowledgements

I, hereby, thank my thesis instructor, Assoc. Prof. Cesim Erten, for the advices of thesis progression, the proposals of methods, approaches in development of the algorithm and the meetings we have made for disscusions. This thesis is in collaboration with TÜBİTAK project, BionetAlign: Global Alignments with the Intention of Functional Orthology Mining in Biochemical Networks, conducted by himself.

Table of Contents

List of Tables

List of Figures

List of Symbols

α	:	The control parameter
ϕ	:	Unit fixed score
ψ	:	Unit edge score
π	:	Cluster permutations
Λ	:	Existence of external conserved interactions

List of Abbreviations

PPI	:	Protein-protein Interaction
GNA	:	Global Network Alignment
LNA	:	Local Network Alignment
MP-CNet	:	Message-Passing on the Clustered Network
SPINAL	:	Scalable Protein Interaction Network Alignment
MI-GRAAL	:	Matching-based Integrative Graph Aligner
DIP	:	Database of Interacting Proteins
HPRD	:	Human Protein Reference Database

Chapter 1

Introduction

Thanks to the fast growth of available biological data for several decades, in parallel, many computational methods and approaches have been produced over time, paving the way for uncovering interactions and recognizing patterns biologically between organisms as they play an important role in bioinformatics. High-throughput techniques like yeast two-hybrid system (Fields and Songs, 1989; Finley and Brent, 1994; Sato et al., 1994; Ito et al., 2001) and co-immunoprecipitation unified with mass spectrometry (Aebersold and Mann, 2003) have contributed to the presence of such data, particularly various species, by a significant proportion.

In recent years, protein-protein interaction (PPI) *network alignments*, which are the most noticeable type of data accepted by many researchers, are studied for observing the similarities in terms of pathways, homologies and functions between pairs of species. The set of instructions running on network data for the analysis are *network alignment algorithms*. The general aim is to create the best alignment as large and accurate as possible based on given two or more PPI networks from different species. For supplying easiness to these algorithms, these kind of networks can be represented as *graphs* in terms of data structure where proteins are *nodes* and interactions between proteins are *edges*. This is the most convenient way to perform

measurements on PPI networks because recent algorithms have been designed to work on graph structures and make one-to-one matchings, hence these are comprehended like *graph matching algorithms*. Additionally, any other materials may be included as input data (e.g. sequence similarity of proteins) for smoother and more accurate alignments.

In general term, they can be categorized in two main groups: *Local* network alignment (LNA) algorithms identify subnetworks of different species that match closely to each other in terms of network topology and/or other variables. The first known algorithm of this type is PathBLAST (Kelley et al., 2003, 2004) that enforces the BLAST algorithm (Altschul et al., 1990) for searching the high-scoring local alignments between PPI networks. Sharan et al. (2005) extends the idea with NetworkBLAST to include multiple species. MaWISh (Koyutürk et al., 2006) adapts to the duplication and elimination models inspired by biological events to perform local alignment. Graemlin (Flannick et al., 2006) takes advantage of conserved functional modules of networks. *Global* network alignment (GNA) algorithms, on the other hand, take PPI networks as a whole and provide one-to-one matches across all proteins. IsoRank (Singh et al., 2007) is known to be the first algorithm to make alignment globally on pairwise PPI networks, using eigenvalue formulation. Later, the algorithm was expanded to work on multiple networks as well (Singh et al., 2008) and so with IsoRankN (Liao et al., 2009). Graemlin was later modified to generate global alignments beyond pairwise networks, examining phylogenetic relationships (Flannick et al., 2008). PATH algorithm (Zaslavskiy et al, 2009a) adapts to convex-concave relaxation approach to find a solution path over the pairwise networks. PISwap (Chindelevitch et al., 2010) first performs the global alignment by sequence data, then necessary changes are made by benefiting from

network topology. MI-GRAAL (Kuchaiev and Przulj, 2011) constructs the alignment by integration and several different sources of protein similarity, and was demonstrated to outperform other variants such as H-GRAAL (Milenkovic et al., 2010) and GRAAL (Kuchaiev et al., 2010). The last known state-of-the-art algorithm of global alignment is SPINAL (Aladag and Erten, 2013) which consists of two stages; first estimating alignment scores, then resolving conflicts to enhance the alignment while also dealing with scalability issues. Additionally, SubMAP has been proposed (Ay et al., 2011) that has the capability of making one-to-many mappings of proteins.

LNA methods are able to expose more than one region of matches between networks, i.e. local matchings provide specific areas of interactions amongst these network of proteins. However, GNA looks for comprehensive matchings by taking into account all proteins and attempts to align them. This gives the best conserved functions as much as possible. By the aspect of computation, GNA is more difficult than LNA because one protein in a network is sought to match with a protein of the other network that achieves the highest optimality, although it is desirable for many algorithms to detect functional orthologs. In addition, most of GNA algorithms are allowed to use weights between the network topology and protein sequence similarities (denoted with α or λ) in order to perform the alignment. This gives rise to number of different alignments and flexibility to the output. Algoritms that accept such a control parameter are generally classified as *balanced* GNA algorithms.

Another issue of network alignment is intractability in terms of computation, when networks are getting too dense. This causes alignments to become distant from exactness. Furthermore, there is no algorithm to work in polynomial time for the problem (Zaslavskiy et al., 2009b) as the current algorithms endavours to make the

best approximate results by impliying heuristic methods but with NP-hardness (Klau, 2009; Aladag and Erten, 2013).

While on the research of identification of protein interactions across species especially before network alignment algorithms were produced widely, some questions have been arisen such as which proteins or genes are in common with those from other species (i.e. orthologs), and which provide shared biological functions against their ancestors or familiar organisms. For this approach, Inparanoid (Remm et al., 2001), HomoloGene and OrthoMCL (Li et al., 2003) have been proposed to disambiguate functionally ortholog proteins by grouping them in clusters after specific methods are implemented based on the algorithms. By using the information of finite number of clusters as well as covered proteins, the whole can be interpreted as a separate network, i.e. the cluster network and necessary connections are carried out between those with specific regulations; for example, conservation of functions in proteins of pairwise PPI networks for any two clusters. This kind of study is usually referred to *constrained* GNA problem, where proteins are restricted to match with the others solely in the same clustered group and attempts to create an alignment in this way. This may help reduce the intractability issue of the networks and perhaps holding functionally ortholog proteins together may bring more potentials for similar properties.

To the best of our knowledge, there are not much studies so far involving the solution of this problem. Bandyopadhyay et al. (2006) investigated the proteins between *Drosophila melanogaster* and *Saccharomyces cerevisiae* to identify functionally orthologs using Markov random field (MRF) methods while these are constrained to belong to the respective clusters, produced by Inparanoid algorithm. Another noteworthy procedure is the message-passing algorithm by Zaslavskiy et al.

(2009b), that is able to align the networks optimally with meesage-passing on particular clustered structure of proteins assuming if the network is sparse. Here, message-passing is a form of communication where objects send and receive messages to each other. A special variant, belief propagation (BP) has been introduced (Pearl, 1982) to make inferences on graphical models (e.g. Bayeisan networks and Markov random fields) and included in many applications, for instance, artificial intelligence (Pearl, 1988), statistics (Lauritzen, 1996), low-density parity-check codes and any other areas with experimental successes (Horn, 1999; Aji and McEliece, 2000; Yedidia et al., 2000).

In this paper, we propose a new algorithm, called MP-CNet which is the abbreviation of Message-Passing on the Clustered Network, to execute on constrained GNA problem and it attempts to find the best alignment with as many as conservations of proteins on clustered pairwise PPI networks with BP-based message passing method, possessing some ideas of Zaslavskiy et al. (2009b). Here, clusters propagate their best matches to their neighbors in order to increase the overall amount of conserved interactions and the whole cluster network converges at a point when the maximum amount is reached. Our contribution is that we show a general-purpose algorithm with the implementation of message-passing can be used on any kind of cluster network structure including the PPI networks themselves. The methods, in some cases, have resemblances with the implementation of maximum weight bipartite matching, but are not complicated at all. Also, by checking the capabilities with other algorithms, we draw attention to the situations that our algorithm can reveal larger number of conserved interactions if filterings are applied to other alignments and highlight that it can provide higher quality of biological impacts than existing algorithms.

The rest of the paper is organized as follows. In section 2, we highly detail working principles of MP-CNet by explaining the definition of the problem to solve, data processing stages, any procedures, approaches involved in the message-passing method through the subsections. In addition, the algorithm progession and claims of optimal alignments are elaborated. In section 3, we compare our algorithm to other global alignment algorithms that approaches the problem as balanced, by the aspect of number of found conserved interactions, running performances and general biological quality of the alignments. In Conclusion section, important remarks and final discussions are made, in additon to any future plans for further improvements.

Chapter 2

Methods and The Message-Passing Algorithm

2.1. Problem Definition

Our algorithm, MP-CNet, is designed to work on pairwise protein-protein interaction networks. We denote $G = (V_G, E_G)$ and $H = (V_H, E_H)$ as two undirected graphs which are the PPI networks of two different species. $V_G = \{g_1, \dots, g_M\}$ and $V_H = \{h_1, \dots, h_N\}$ are the finite set of nodes of their respective graphs, each representing the proteins of PPI networks. $E_G \subset V_G \times V_G$ and $E_H \subset V_H \times V_H$ are the edges of these graphs which corresponds to the interactions between proteins. We assume there are no any self-loops i.e., $(g_i, g_i) \notin E_G$ and $(h_j, h_j) \notin E_H$. In addition, we are given a set of disjoint clusters $CG = \{cg_1, \dots, cg_r\}$ such that each cg_p consists of a subset of nodes of V_G and a subset of nodes of V_H.

Here, a pair of node mappings (g_i, h_j), (g_k, h_l) provide a conserved interaction if $(g_i, g_k) \in E_G$ and $(h_j, h_l) \in E_H$. Given the two graphs G and H together with the set of clusters CG, we define the *constrained global network alignment* (GNA) problem that of finding a one-to-one mapping that satisfies the constraints; that is, each mapped pair belongs to the same cluster $cg_p \in CG$ and that maximizes number of conserved interactions. Before discussing the algorithm in detail, we highlight the main processes involved in preparing the input data.

2.2. The Cluster Network

The construction of the cluster network is the first major part of processing the input data. This network itself can be thought of as a separate graph but has a correlation with the PPI network graphs in accordance with the clusterization of nodes. For this purpose, we denote CG be the cluster graph (network) which will be substantially used by MP-CNet in many stages. Of course, like the PPI network graphs, we denote $V_{CG} = \{cg_1, \dots, cg_X\}$ to represent the finite number of disjoint clusters and $E_{CG} \subset V_{CG} \times V_{CG}$ are the finite set of edges of the cluster graph. In the initialization, E_{CG} remains empty. We first describe how the cluster set CG is constructed and the corresponding vertex set of V_{CG}.

2.2.1. Cluster Nodes

For the generation of V_{CG}, we benefit from Inparanoid algorithm (Remm et al., 2001) which is known to exhibit a good overall balance by both sensitivity and specificity (Chen et al., 2007). Mentioning briefly, Inparanoid automatically detects orthologs and in-paralogs between any given two species and uses special techniques for revealing the clusters. From the definition, ortholog proteins are that evolve directly from a single species from the last common ancestor and have a high proportion to share function. Paralog ones are homologs that contain uncertainty of functional equivalence between the orthologs which are derived from a single ancestor at the speciation event. It is also noted that paralogs can be arisen from duplication event before speciation is occured. For this reason, paralogs are split into two types: In-paralogs are the ones that are duplicated after the speciation as they are considered to be orthologs. Those preceding the speciation are denoted as out-

paralogs and they are not counted orthologs. In this context, Inparanoid algorithm does not include out-paralogs in the output.

There are many parameters available in the algorithm which can affect the placement of proteins (i.e. the orthologs) into clusters, thus the whole general output. The most notable one is pairwise similarity score and this is where the orthology detection begins by computing all similarity scores between all existing protein sequences of two species. It is measured by another program called BLAST (Altschul et al., 1990) to create E-values for all possible pairs of proteins from pairwise species. These values are helpful to determine the orthologs and clusters afterwards. Furthermore, a score cut-off is required to distinguish the scores from spurious ones, i.e. any cluster whose score is less than the cut-off will not be included in the output. Overlap cut-off is also used to determine the ratio that the matching protein of the longer sequence must surpass its total length. In-paralog confidence values, bootstrapping for ortholog groups and coverage cut-off are the other countable essential parameters. The output consist of cluster number, its score and proteins from the PPI networks in the related cluster.

The rest of the details of the algorithm are out of the scope of this paper (for more details, see Remm et al., 2001). We only focus on the results (i.e. the set of clusters), generated by Inparanoid and mainly, MP-CNet uses it as a guide to produce the clusters and place the appropriate proteins into them correctly. Only the cluster number and its proteins are needed for generating the clusters. It can be seen that clusters with higher scores are likely to be at the top of the clustering information. However, it is not necessary to be in order, though it helps to become organized and understand them better.

When we look at each cluster, three specific kinds can be encountered: One-to-one clusters contain only one node (protein) from both graphs. In other words, the single node from graph G in this cluster can merely match one node from graph H, leaving no other option to match from outside. One-to-many clusters have one node from graph G and more than one nodes from graph H and vice versa. However, there is still one match that can really occur. For many-to-many clusters, more than one nodes from both graphs are involved. The amount of matches in that kind of cluster is the same with whose number of nodes in the respective graph is less. At the time of the creation of each cluster node cg_i, it is recognizable that not every one of them cover the same number of nodes from both graphs. This especially happens for all one-to-many and some many-to-many clusters. To address this issue, we insert dummy nodes (i.e. artificial nodes) to whichever the number of subset of V_G or V_H nodes is smaller in the cluster node cg_i and they have no edges in their respective PPI-network graphs. The main reason for ensuring the equalization to each cluster is to simplify the algorithm description and implementation details, and this allows every nodes to be matched, although those with dummy nodes are not included in the output. Namely, it must not be perceived such that we do not equalize the number of nodes between graphs G and H; we just perform this on clusters which needs to be equalized by getting the number of 'real' nodes contained. Consequently, $\forall cg_i \in CG$ has the same total number of nodes covered from both graphs and we denote $cg_i = \{(g_a, \dots, g_m), (h_b, \dots, h_n)\}$ where $g_a, \dots, g_m \in V_G$ and $h_b, \dots, h_n \in V_H$.

2.2.2. Cluster Edges

After preparing all cluster nodes, we perform a check on each cluster to observe if it can make connections to other clusters and so, these connections will be interpreted as cluster edges (added into E_{CG}) for our cluster graph.

This process is straightforward. We take every possible combination of cluster pairs, that is, $cg_i, cg_j \in V_{CG}$ and $1 \leq i < R,\ i < j \leq R$, every (cg_i, cg_j) pairs are investigated. For any pair, we start from the subset of nodes of V_G of cluster nodes cg_i and cg_j ($\{g_a, \dots, g_m\} \in cg_i$ and $\{g_b, \dots, g_n\} \in cg_j$) to compare them with all available edges from E_G for an existence of at least one edge, involving these nodes. Next, we advance to make the similar comparison with the subset of nodes of V_H of these clusters to the graph edges of E_H. That is, we perform this with $\{h_a, \dots, h_m\} \in cg_i$ and $\{h_b, \dots, h_n\} \in cg_j$ to all edges from E_H. If such edges are found for both sides, then it is safe to add a connection between these cluster nodes and included in E_{CG}. This goes on like this until every combination of cluster node pairs are examined. Note that in Figure 2.1, an edge between cg_i and cg_k is not formed because there is no edge available in E_H that involves one node from the subsets of V_H of cg_i and cg_k. As a result, the construction of our cluster graph CG is finally completed.

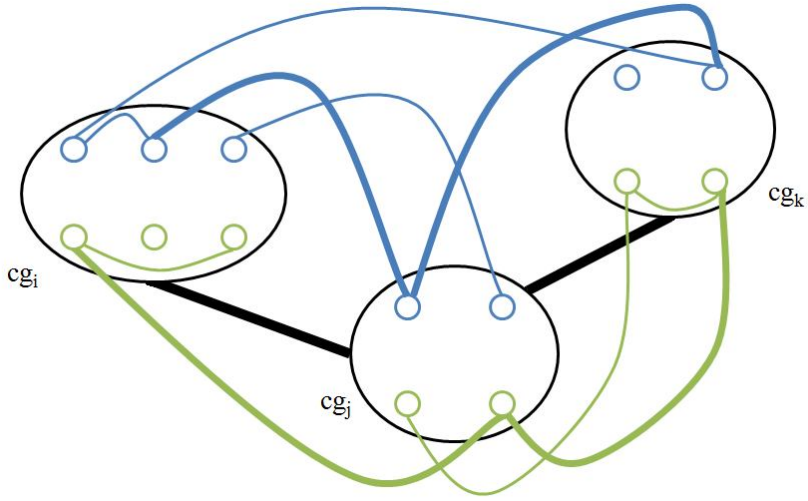

Figure 2.1 An illustration of the addition of cluster edges $\boldsymbol{E_{CG}}$. Blue and green components represent nodes and edges of graph $\boldsymbol{G}$ and $\boldsymbol{H}$, respectively. Here, bold edges satisfy connections between $\boldsymbol{cg_i}$-$\boldsymbol{cg_j}$ and$\boldsymbol{cg_j}$ -$\boldsymbol{cg_k}$.

The structure of CG is vital for MP-CNet because it will be used every time in the oncoming stages and so the finding of conserved interactions between cluster nodes. The structure can be in any form such that there may be many cluster nodes having one or more connections (sometimes called adjacents or neighbors) to any other; lots of small or a large group of joint networks. In the meantime, the number of cluster edges E_{CG} can vary, depending on the number of cluster nodes, number of covered nodes in clusters $\forall cg_i \in CG$ and the edges of E_G and E_H. Note that more connections among all existing cluster nodes allow to get more conserved interactions.

2.2.3. Cluster Permutations

We have presented an easiness to our algorithm by adding dummy nodes to the necessary clusters to make the number of subset of nodes from V_G and V_H equal. Now, we would like to extract how many different varieties of matches could be carried out for all clusters. For this reason, the creation of permutations begins when construction of CG is finished.

Let $\boldsymbol{P}$ be the permutation space, containing all possible permutations of $\forall cg_i \in V_{CG}$, a multi-dimensional variable. Also, let $\pi(cg_i)$ be the permutations of cg_i. Here, in this process, one-to-one and many-to-many clusters are used, since all one-to-many clusters are converted to many-to-many by the insertion of dummy nodes to them. We assume $\forall cg_i$, these permutations are derived from the subset of nodes of V_H, while fixing the subset of V_G nodes. This is not limited to this shape and one can make by using the appropriate V_G nodes, too. However, this requires a comprehensive redesign to our algorithm for further steps to provide full compatibility. Hence, from the rest of the paper on, we comply that the permutations remain with subset of nodes of V_H at all times.

For any cluster cg_i, its subset of nodes of V_H are taken and sent out to a special function to generate the permutations. We follow the permutation-without-repetiton rule, so these permutations are distinctive from each other when generated. There is another variable, p, which holds the current iteration of nodes to check if it is a permutation candidate to be added to not. In the intial step, first node is placed to p, then a second one and so on like a stack. We actualize the examination of duplication when p contains more than one node and less than n, which is the maximum number of nodes can be together, and the checks are happened in every addition. For the group of nodes in p at any time, we first treat them like they form a permutation and continue to add nodes up to n if there is no duplication occured within all nodes inside, otherwise the last added node is removed (i.e. popped back) from p and the next one is placed. When the number of nodes in p is reached to n, we ensure that every node is different from each other, therefore it is secure to mark them as a permutation of cg_i and included in $\pi(cg_i)$, then certainly to $\boldsymbol{P}$ (Figure 2.2). This goes on until all the subset of nodes of V_H from cluster nodes are taken care of.

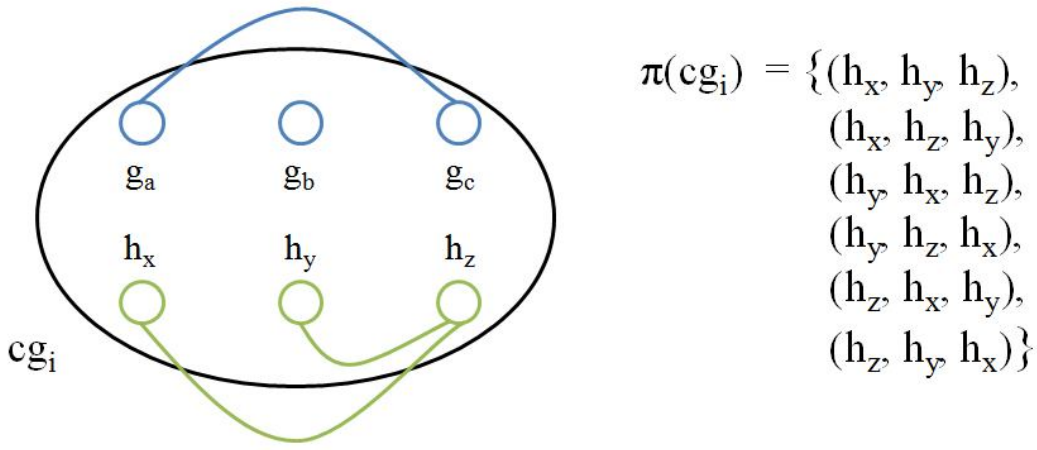

Figure 2.2 An example of all available permutations of a cluster. Note that the cluster $\boldsymbol{cg_i}$ has 3 nodes from $\boldsymbol{V_H}$ and 3! = 6 different permutations were generated.

A cluster cg_i has $n!$ amount of permutations, similarly what we have said previously, where n is the number of subset of V_H nodes covered in cg_i. They are always stored till the completion of MP-CNet. Like the cluster edges of CG, the permutation space $\boldsymbol{P}$ is important. In the next stages and even in the message-passing process for the conserved interactions, we always iterate through permutations.

2.3. Conserved Interactions Within Clusters

The stages from now on until the message-passing derive the second major part of input data processing. Here, we attempt to uncover the conserved interactions for all clusters internally and the first usage of permutation space $\boldsymbol{P}$ take place for this purpose. Any found conserved interactions here can make a small contribution to the optimal alignment; that is, such a permutation of a cluster with internal conserved interactions is likely to be selected as the best although it is not always guaranteed. Because there could be another permutation in that cluster whose, for example, amount of external conserved interactions is higher then others with only internal ones, where this alteration of selection is happened in the message-passing. This will be discussed in Section 2.5. All in all, finding this kind of interactions in clusters is an important property for all permutations as an indicator of their capabilities.

We first denote $\mathcal{FS}$ to keep the number of internal conserved interactions found for all permutations of any cluster $cg_i \in V_{CG}$; along with this, $FS(cg_i)$ represents amount of found internal ones for all its permutations $\pi(cg_i)_x \in \pi(cg_i)$ where $1 \leq x \leq n!$. Mostly, we call this value as *permutation fixed score* or *permutation intra-cluster score*. The main reason why we name them in this shape is because we normally allow the user in MP-CNet to enter a unit score for each internal conserved interactions discovered, denoted with ϕ and more importantly, these values are held

as constant for use in other stages, especially in the message-passing. For easy understanding to the rest of the paper, we always treat them like a fixed score.

For any cluster node cg_i, it is not required to utilize the cluster edges. Only the matches which the permutation $\pi(cg_i)_x$ creates and the edges from E_G and E_H are needed. Note that the edges connected within cg_i could provide fixed scores. As a side note, meanwhile, we demonstrate these matches for any cluster permutation like this: for $\{g_a, g_b, \dots, g_u\} \in cg_i$ and $\{h_i, h_j, \dots, h_z\} \in cg_i$, in first permutation $\pi(cg_i)_1$, for instance, the matches are made as follows: (g_a, h_i), (g_b, h_j),…, (g_u, h_z). The similar production of matches are applied to the forthcoming permutations but with the necessary alterations for which subset of nodes from V_G are matched against the subset of nodes from V_H as the node order of $\pi(cg_i)_x$ states. So, the last permutation is the completely reversed one of the first and its matches go like in this shape: (g_a, h_z), (g_b, h_y),…, (g_u, h_i). From these descriptions we have explained thus far, a permutation can have n amount of matches, which is also the same with the number of subset of nodes covered from V_G and V_H. In addition, $(n(n-1)\ /\ 2)$ number of different pairs of matches can be selected.

The computation of conserved incteractions within clusters is simple. In any cluster $cg_i \in V_{CG}$, we initally choose a pair of matches $M(i,j) = [(g_a, h_c), (g_b, h_d)]$ that are made within the permutation $\pi(cg_i)_x \in \pi(cg_i)$, and $1 \leq i < n$, $i < j \leq n$. Here, g_a and g_b are taken to search for the existence of an edge in E_G. Then, we are ready to move on to search for h_c and h_d by scanning all edges of E_H, if such an edge is available from E_G. Otherwise, we pass on to the next possible pair of matches, as it is certain the previous one does not have the opportunity to make a internal conservation. To sum it up for this process, fixed score of a cluster permutation $\pi(cg_i)_x$ increases by the value of ϕ if and only if

(g_a, g_b) and (h_c, h_d) of selected pair of matches from the subset of nodes of V_G and V_H enclose an identical edge from E_G and E_H, respectively (Figure 2.3). At the end, the score is placed to $FS(cg_i)$ and to $\mathcal{FS}$ when completed $\forall\pi(cg_i)$. It can be also noted that a permutation can have a maximum fixed score of $(n \times \phi)$ which could be feasible whether the subset of nodes of that cluster mold complete subgraphs. Later, we check which permutation(s) possesses the best fixed score for each cluster by creating a list variable $\mathcal{FSB}$, for this purpose. It is necessary to keep the best permutations for comparison with another list variable, which is mentioned in Section 2.4 for getting our algorithm ready for message-passing. Here, in any cluster cg_i, fixed score of every permutation is checked and the best ones are added to $FSB(cg_i) \subset \mathcal{FSB}$.

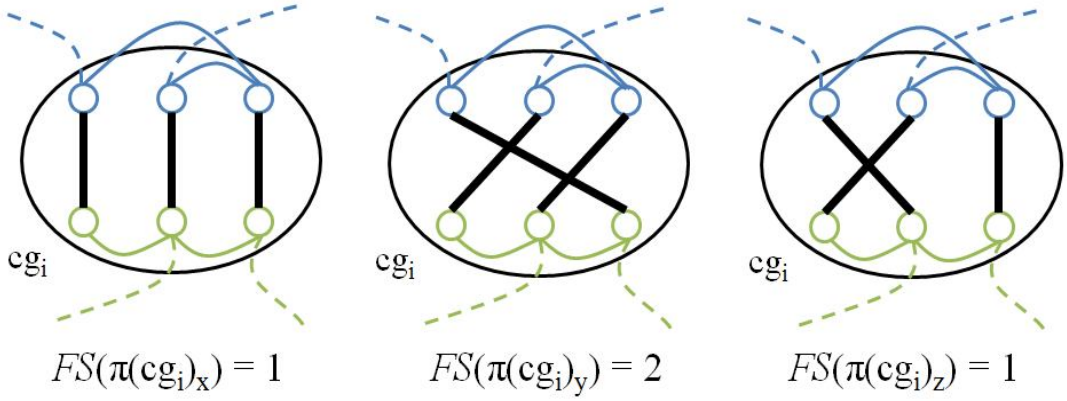

Figure 2.3 A demonstration of internal conserved interactions (fixed scores) for some permutations from $\boldsymbol{\pi(cg_i)}$ for $\boldsymbol{\phi}$ = 1.0.

It is admissible that one-to-one clusters cannot have fixed scores, due to having only one match and a single node from V_G and V_H as its subset. Hence, they are directly skipped by our algorithm. For some many-to-many clusters which were normally one-to-many clusters, all of them, but converted to that type by the addition of dummy nodes, we do not expect any fixed scores again, although they are

permitted to be searched. So, it shows that only the 'real' many-to-many clusters have the capability to include fixed scores for their permutations.

2.4. Plausible Edge Counts

After the preparation of permutation fixed scores for all clusters, additionally we apply a heuristic approach to measure the number of occurences in edges of E_G and E_H for the covered subset of nodes $\forall cg_i \in CG$. This may give a clue for the permutation $\pi(cg_i)_x$, having a greater number of edges that their subset of nodes existed in the edges of E_G and E_H respectively, brings more potential to extract many conserved interactions, so it is likely to be chosen as the best permutation in that cluster. It should be noted that every kind of edges of E_G and E_H are searched, including those within clusters or going from one cluster to another and no matter the cluster cg_i has neighbors or not. We denote $\mathcal{EC}$ on this objective to store the edge counts of all cluster nodes and an extra sub-dimension $EC(cg_i) \subset \mathcal{EC}$ for all permutations of cg_i. These values are merely used in this step and together with the fixed scores $\mathcal{FS}$ we have gathered previously, the best permutations are decided initially for getting ready MP-CNet to the message-passing stage.

In any cluster cg_i, the matches of permutation $\pi(cg_i)_x$ are taken one by one as they are shown with, for instance, $\pi(cg_i(M(z)))_x = (g_a, h_b)$ where $1 \leq z \leq n$. We begin browsing all edges of E_G to count how many times the node g_a was encountered in them and keep the total number in $pls(g_a)$. Then, the same action is applied to h_b for counting the occurence in the set of edges E_H and the value is assigned to $pls(h_b)$. By comparing both $pls(g_a)$ and $pls(h_b)$, we determine the plausible edge count of the match with these rules:

(a) If $pls(g_a) = pls(h_b)$, $pls\big(\pi(cg_i(M(z)))_x\big) = pls(g_a) \vee pls(h_b)$;

(b) Else if $pls(g_a) < pls(h_b)$, $pls\big(\pi(cg_i(M(z)))_x\big) = pls(g_a)$;

(c) Else if $pls(g_a) > pls(h_b)$, $pls\big(\pi(cg_i(M(z)))_x\big) = pls(h_b)$.

Simply, the count total of a match is equal to whichever of $pls(g_a)$ or $pls(h_b)$ is less. It is also comprehensible that matches made with dummy nodes do not yield any counts. When all matches are measured, we sum every value to $pls(\pi(cg_i)_x) = \sum_{z=1}^{n} pls\big(cg_i(M(z))\big)$ to finally specify the plausible edge count of that permutation of cluster cg_i (Figure 2.4).

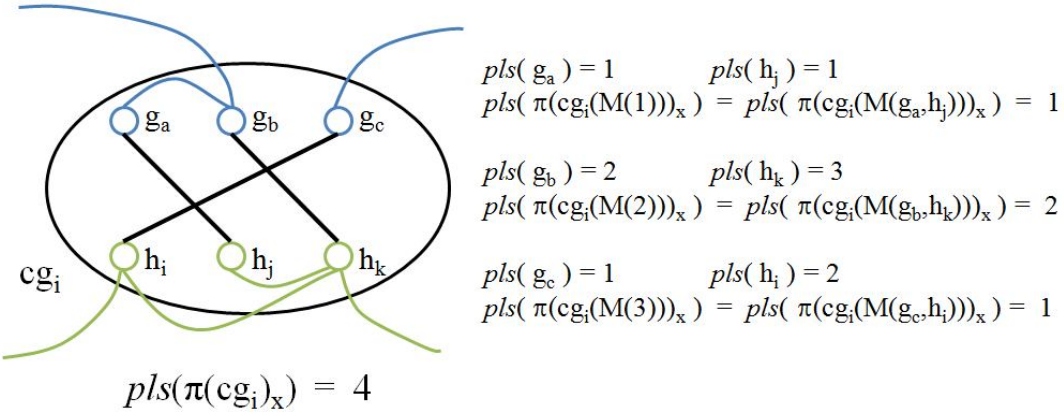

Figure 2.4 Counting plausible edges of a cluster permutation. Here, in the example above, all available matches plausibly carry 1,2 and 1 edges respectively for a total of 4 plausible edges.

The next step is to decide, similarly to fixed scores, which permutation(s) has the best value of edge counts $\forall cg_i$. Each value of edge counts is investigated and permutation(s) achieving the best value is assigned to another list $\mathcal{ECB}$ to hold the indexes of the best permutations $ECB(cg_i) \subset \mathcal{ECB}$.

Now, we are one step closer ahead to select the initial best permutations $\forall cg_i$ to start the message-passing process. All clusters are examined in order, so we bring both best fixed-scoring and edge-counting permutation sublists ($FSB(cg_i)$ and

$ECB(cg_i)$) for the appropriate cluster to be compared with each other. Any permutation which is encountered in both lists is added to $CM(cg_i) \subset \mathcal{CM}$, indicating the common bests. In a situation that more than one permutation are included in $CM(cg_i)$, one permutation is picked randomly to become the best. Another case can happen that no common-best permutation is returned where we give the priority to the best fixed-scoring permutations to supply that such a cluster node as the best one. Finally, we move these best permutations of all clusters to $\mathcal{PB}$, for usage in the message-passing. By reaching this point, we accomplish the input data processing series.

2.5. The Message-Passing Method

Our algorithm, fundamentally, perform the alignment with the message-passing. The importance of this stage is elevated because the output available upon completion of the algorithm is entirely built on passing information between clusters as accurately as possible. For this reason, we have progressed through many steps to make the data consistent in hand to not unexpectedly fall with unusual results and coherently match the definition of constrained GNA problem. Here, we emphasize on the operation of message-passing, evaluations of the progess, getting optimal alignment and convergence in the following sections.

2.5.1. Our Formation and Vision of Message-Passing

Message-passing, in general, is the paradigm of communication where messages are sent from a sender source to one or more recipients. Our approach is very similar to this notion, but with slight modifications; that is, at any iteration, clusters of graph CG look for the neighborhoods of other clusters and those neighbored ones convey their best permutations back to currently investigated ones.

At that moment, clusters use them with their own permutations to make consequences. Then, upon getting the knowledge, a new best permutation is decided for the next iteration, and so progressively constitute the alignment with the highest number of conserved interactions comprised. While performing the message-passing, the most necessary factor is the number of external conserved interactions that permutations of clusters produced. As deciding the new best ones, not only do we take advantage of these numbers, but also their internal conserved interactions (i.e. fixed scores).

At the time of designing our message-passing method, we were highly thrilled from the elucidations of Zaslavskiy et al. (2009b). The authors asserted a method which could be solved exactly and efficiently in some cases and the definition of clusterization technique was what we have expressed in Section 2.2.1. Moreover, this situation was studied beforehand for finding the functionally ortholog proteins (see Bandyopadhyay et al., 2006). The strong assumption the authors claimed is the exact optimization can be obtained if the cluster network is a particular structure such that it has many isolated cluster nodes, no loop throughout all connected clusters, or is just a single connected component like a tree. Their advancement of the message-passing method is as follows: One of the cluster node is selected to be the root to assign any other clusters a distance value, indicating how many connections must be passed over in order to reach the root. They are required for assessing which are the parents and children of each other and traversing each cluster by using breadth first search. The algorithm begins at the highest distance value of clusters as they are children and thus, only the conserved interactions within are measured and a vector is placed to them that carries out the best number of these. While moving up, conserved interactions between child clusters of parents are also calculated, hence

the root should have the highest. This covers the forward step so far. Next, the backward step is implemented to gather those vectors in each cluster to achieve the optimal alignment overall.

Considering the definition of constrained GNA problem and the idea of our proposed algorithm here, the cluster network must not have restrictions to any specific structure as miscellaneous types can be encountered depending mainly on the structure of both PPI network graphs and the generated clusters by a clustering algorithm. In other words, CG can contain a large component of joint clusters or lots of small ones separated from each other and reasonably become cyclic. Therefore, we always obey the generality of these graph structures and our aim, hereby, is to build an algorithm which is able to work on any of them; still matching the description of constrained GNA problem.

Unlike Zaslavskiy et al. (2009b), there does not have to be a root cluster to be chosen and this dismisses the setup of the assignment of distance values to all other clusters. Though every time marking a random cluster as root gives a different result, we are on the side of having a steady one to make the general traceability of the output easier. In addition, the forward and backward steps, that is, the message-passing seems to happen only one time. As we do not want to limit to just one iteration like this, a larger number of iteration scheme is adopted enabling our algorithm to search the conserved interactions on clusters again for probability of having more than previous iterations. These adjustments are applied to make it also more compatible against the form of any cluster networks.

2.5.2. Functions of the Message-Passing, External Conservations and Output

We express how our message-passing method is progressed. Thus far, the best permutations of all clusters, stored in $\mathcal{PB}$, have been acquired along with the

permutation fixed scores $\mathcal{FS}$. First of all, we start with a simple check if $E_{CG} \neq \emptyset$. Otherwise, it is not worth to do the message-passing on CG; the alignment is provided only with the contribution of fixed scores from clusters. Furthermore, there is a maximum limit to the number of iterations, denoted with k_{max} which is user-defined, to ensure our algorithm to cease at a certain point if no convergence is occured.

At any iteration k, each cluster cg_i is traversed to detect its neighbors (i.e. connections) and these are kept in $N(cg_i)$ so the implementation of message-passing to which clusters are determined at the moment. It is noticeable that any cluster with no neighbors are skipped so we leave them with only the best fixed score for the contribution of the final alignment. Certainly, it is not essential to change the best permutation of those clusters at every iteration, holding them fixed across all iterations until k_{max} or convergence. Otherwise, for a cluster cg_i with $N(cg_i) \neq \emptyset$, its all available permutations are examined, because having connections can affect the total number of conserved interactions between other clusters (Zaslavskiy et al., 2009b). In any $\pi(cg_i)_x$, we begin with the addition of fixed score, fetched from $FS(cg_i)$, to $SC(\pi(cg_i)_x)$ which represents the total score of the permutation. Meanwhile, the cluster is made ready by having one-to-one matches of $\pi(cg_i)_x$. Then, the neighbored clusters are called to propagate their best permutations that are available from $\mathcal{PB}$ to the current cluster. All clusters in $N(cg_i)$ are later adjusted with the setting of subset of nodes of V_H, indicated by the propagated permutations to obtain appropriate matches. After that, the finding of external conserved interactions between the currently investigated cluster cg_i and its neighbors begins.

Like the implementation of fixed score, described in Section 2.3, external conserved interactions can be represented like a score and mostly define them as

permutation edge score or *permutation inter-cluster score*, denoted with ψ and our algorithm allows the user to enter a unit score for that kind. In contrast to getting fixed scores, now two clusters are required for search.

In the message-passing method, we always keep the matches of neighbor clusters in $N(cg_i)$ static until all permutations of $\pi(cg_i)$ are investigated. These neighbors are traversed one by one as we get the external conservations independently. Assuming cluster cg_i and one neighbor in $N(cg_i)$ is fully ready for searching, we take one match from each. Note that if a permutation of $\pi(cg_i)$ contains m matches and the other one from $\pi(cg_j)$ contains n matches, where $cg_j \in N(cg_i)$, then up to $m \times n$ different pairs of matches are investigated for getting edge score. For any matches, e.g. $\pi(cg_i(\,M(y)\,))_x = (g_a, h_c)$ and $\pi(cg_j(\,M(z)\,))_p = (g_b, h_d)$, where $1 \le y \le m$, $1 \le z \le n$; $g_a, g_b \in V_G$, $h_c, h_d \in V_H$, $cg_j \in N(cg_i)$ and p indicates the best permutation among all in $\pi(cg_j)$, we first look into the edge existence in E_G (i.e. protein interaction) between g_a and g_b nodes. If our algorithm comes across such an edge, then the same action is performed for h_c and h_d nodes with E_H. When both conditions are satisfied, the permutation $\pi(cg_i)_x$ gains unit edge score ψ (Figure 2.5). When all match combinations are examined, the accumulated edge score is divided by 2 and then it is added to $SC(\pi(cg_i)_x)$. The total score of any permutation can be universally formulated by the equation below in any iteration k:

$$SC(\pi(cg_i)_x) = FS(\pi(cg_i)_x) + \sum_{\forall cg_j \in N(cg_i)} \frac{\Lambda[\,\pi(cg_i(M(y)))_x, \pi(cg_j(M(z)))_p] \times \psi}{2} \qquad (2.1)$$

In equation (2.1), $\Lambda[-,-]$ indicates the availability of conserved interaction between the matches of those corresponding cluster permutations, given a value of 0 or 1 and is multiplied by ψ. There is an important reason for dividing the unit score

by 2. This is because we expect cg_j, when it becomes the currently investigated cluster by MP-CNet and where cg_j is the neighbor of cg_i at the time, formally $cg_j \in N(cg_i)$, to exhibit the similar satisfaction exactly between the matches $M(z)$ of its such permutation and $M(y)$ of the permutation of its neighbor, those which provided conserved interaction already. In this way, they mutually supplement each other by guaranteeing there really exists a conserved interaction with these selected permutations and matches. The only concern might be that if the permutation in $\pi(cg_i)$ which carries out the same interaction(s) with the one in $\pi(cg_j)$ is not the best, at the time cg_j is being investigated for edge conservation while cg_i becomes its neighbor, there is a situation that another permutation from $\pi(cg_i)$ with the convenient match can compensate this affair.

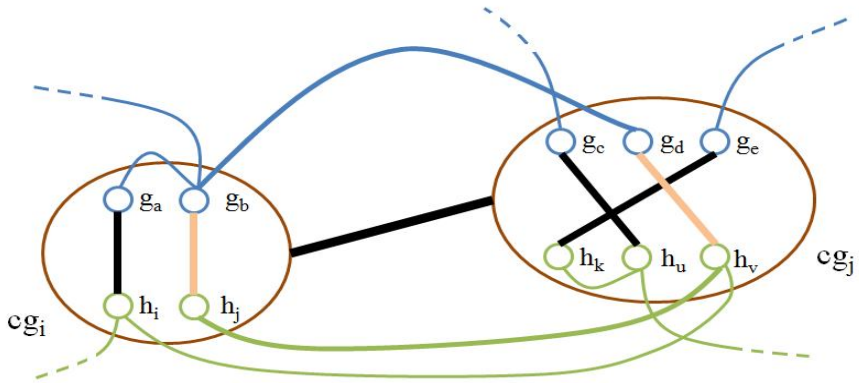

Figure 2.5 An example of external conserved interaction between clusters. Here, $(\boldsymbol{g_b}, \boldsymbol{h_j})$ match of $\boldsymbol{\pi(cg_i)_x}$ and $(\boldsymbol{g_d}, \boldsymbol{h_v})$ match of $\boldsymbol{\pi(cg_j)_p}$ supply a conservation externally.

After the calculation of edge scores of all permutations $\forall cg_i$, we look for the best scorings to be included in $\mathcal{PB}'$ along with the highest score achieved for cg_i. That means every permutation score $SC(\pi(cg_i)_x)$ are compared, just like the previous sections involving the selection of the bests. For those having more than one cluster accomplishing the same highest scores, these are broken randomly by selecting one of them and we put it into $\mathcal{PB}'$ and, meanwhile, the highest score is

accumulated in another variable, denoted with Θ_{CG}, hence the total best scores of all clusters are acquired in every iteration. This can be formed with the formulation in the equation below.

$$\Theta_{CG} = \sum_{cg_i \in V_{CG}} \max\, SC(\pi(cg_i)_x), \qquad \forall x \tag{2.2}$$

Note that every best score from clusters summed to Θ_{CG} may or may not contain the fixed scores as only we check the value of permutation scores. When a higher Θ_{CG} is achieved in iteration k than previously, the value is kept in $\max\Theta_{CG}$, so the highest possible score is determined until either convergence or k_{max} is reached.

It should be pointed out that we do not remove $\mathcal{PB}$ of the current iteration k yet. As every best permutations of all cg_i clusters is placed into $\mathcal{PB}'$, now these values are used together to know the state of convergence of the cluster network. This is an important property that determines the continuity of conserved interaction searching, and it is checked in every iteration when all clusters are traversed for edge scores. This process is basically done by comparing the eqaulity of the best permutations of each cluster such that if every value in $\mathcal{PB}$ is completely the same with $\mathcal{PB}'$, for instance, $PB(cg_1) = PB'(cg_1)$, $PB(cg_2) = PB'(cg_2)$, and so on, then MP-CNet no longer proceeds the computation of edge scores i.e. convergence is achieved and the alignment is prepared for the output. Otherwise, it continues the computation till k_{max} is reached, and $\mathcal{PB} \leftarrow \mathcal{PB}'$ is performed at the end of every iteration.

We also take another action, particularly when edge score calculation is finished thus the best permutations ($\mathcal{PB}'$) are decided, which could be a heuristic approach to create an alternative alignment at the end. For this purpose, $\mathcal{CPB}$ is created at the beginning of the message-passing, holding how many times a

permutation is chosen as the best and the value corresponding to that permutation of cg_i is increased by 1 and stored in $CPB(cg_i) \subset \mathcal{CPB}$. Note that the size of $CPB(cg_i)$ is equal to the number of permutations generated for cluster cg_i. It is implemented in every iteration till the convergence is achieved or k_{max} is reached. Then, the most selected permutations among all ongoing iterations are picked up and placed in $\Delta\mathcal{PB}$. Then, the alternative alignment is made available to the output, which can be supportive to the regular one in terms of number of conserved interactions with slightly different results. It is noted that the convergence check is not performed with $\Delta\mathcal{PB}$; only $\mathcal{PB}$ is required for this objective.

The last portion of the message-passing of our algorithm is to present the alignment output, additionally how many conserved interactions occurred across all available matches of CG, and which pairs of matches provide these interactions by showing their relative nodes from V_G and V_H. Every 'valid' matches are gathered from clusters (i.e. not including matches involving a dummy node), hence denote $\mathcal{M}_{CG}$ to contain all these matches together. Every combination of $\mathcal{M}_{CG}[i]$~$\mathcal{M}_{CG}[j]$ $(1 \leq i < max, i < j \leq max)$ is examined. Similar to what we have done previously, the matched nodes of V_G and V_H are taken for the existence of interactions (edges) between them in E_G and E_H, respectively. Those pair of matches are made available to the output and placed to $\mathcal{CI}_{CG}$ when conditions are met. Then, the same process is applied to the alternative alignment by getting the valid matches and checking the presence of these (to $\mathcal{CI}_{\Delta CG}$). As a result, we create not only one steady alignment but also a different one with our alternative intuition. We expect the number of recognized conserved interactions and those derived from matches should be identical.

2.5.3. Detailed Analysis and Claims

When MP-CNet achieves convergence of the input or k_{max} is reached, total score, alignment, and conserved inteactions actualized by the pair of matches are presented. However, it is challenging make sure if the alignment is really optimal. About networks which are not large and too dense, relating to PPI networks and clusters, it may be somewhat predictable. During our research, we had come up with some remarkable claims that is visible to the most of the input whereas there is still extreme situations that are excluded from them. In this subsection, we would like to explain these by general basic samples, then moving to more complex ones.

The smallest cluster network CG we are able to reduce at most is only two cluster nodes, call cg_1 and cg_2, along with only one edge available in E_G and E_H, connected between these clusters respectively and no edge internally. The number of subset of nodes from V_G and V_H in clusters can be at least one, but for comfortable analysis, let cg_1 have m and cg_2 have n subset of nodes, no matter if there exists dummy nodes. For any iteration k, $\mathcal{PB}$ was already made ready in the previous iteration, $k-1$. As we traverse each cluster node, and so initially for cg_1, there is only one neighbor available, say $N(cg_1) = \{cg_2\}$. Then, cg_2 is set to contain matches for its best permutation, obtained from $\mathcal{PB}$. It is noticable in this simple example that there should exist a match from $\pi(cg_2(M(x)))_p$ that connects the nodes g_u and h_v that already have an edge in E_G and E_H, going to other appropriate node in cluster cg_1. Likewise, cg_1 should have a match between g_a and h_b, for some permutations. To know which of them satisfies, all matches of each permutation of cg_1 are examined to check for existence, as their matches are taken one by one against the matches of the best permutation of cg_2. Suppose $\pi(cg_1(M(y)))_t$ is selected and these matches between two clusters achieve an external conserved

interaction, if the matched nodes of g_a-g_u and h_b-h_v are the edges themselves in E_G and E_H. In this manner, permutation $\pi(cg_1)_t$ gets a score by the value of $\psi/2$. However, as we continue examining the permutations, there could be another one, $\pi(cg_1)_w$, that has the same match, making connection between the same nodes g_a and h_b, thus gaining $\psi/2$ score. Here, we assert that by fixing the match of such permutation of cg_1, and there are $m-1$ other matches remaining, maximum of $(m-1)!$ different forms of matches can be attained. This constitutes a significant claim such that for any cluster, $(m-1)!$ different permutations gain edge score by $\psi/2$ if there goes one interaction from E_G and E_H to its neighbor cluster in $N(cg_i)$, for $m \geq 3$[1]. Likewise, all of these explanations are valid for cg_2, as its neighbor becomes cg_1, where the matches are necessarily set for the best permutation. Here, $(n-1)!$ permutations can get $\psi/2$, so obviously more than one of them reach the same achievement of having the best score. After these two cluster node traverses are made, however, there are $(m-1)!$ and $(n-1)!$ choices of best permutations of respective clusters that MP-CNet should select. As discussed in Section 2.5.2, a greedy approach is implemented to represent the permutations to become the best for the next iteration by first assigning to $\mathcal{PB}'$. When no convergence is materialized then these are overwritten to $\mathcal{PB}$, so these actions are taken again. On the other hand, due to the design of the algorithm, when there are no more iterations for message-passing by reaching k_{max}, the alignment output is represented optimally. Meanwhile, here comes the better understanding of why ψ edge score is divided by 2. As the best permutations of cg_1 and cg_2 together generate the same external conservation even

[1] Here, (3-1)! = 2! = 2, so there is not only one permutation getting the edge score by having the same matches; also always valid for much higher number of pairs of nodes in clusters. For others, like one or two node pairs, 1! = 2! = 1 permutation can obtain the score.

though asynchronously, summing these scores guarantee a such external conservation truly exists.

Now, we extend this situation by inserting new edges to E_G (or E_H) that goes from one cluster to another while the other remains at one edge. It should be approached with two main types: Whether these edges are connected with the other one or disconnected to observe the effects to the whole cluster network, external conservation finding and convergence. First, assume it sustains the connectivity with the existed edge. "Connectivity", hereby, means there is a common node between them, for instance, considering an edge g_a-g_b and the new edge g_a-g_c both source and target nodes are covered in different clusters, the common node g_a provides the connectivity of these two edges. We also agree that a single edge between clusters is a connection, too. Imagine the current cluster network like this, two different nodes of V_G in cg_1 have one plausible edge and one node of V_G from cg_2 have two plausible edges while the other graph remains unaffected. Here, cg_2 still possesses the probability for $(n-1)!$ permutations to get $\psi/2$ because there is only one node (as well as match) in cg_2 that can give the conserved interaction externally. If we take a look at the possibilities for cg_1, then the number of permutations that are able to obtain edge score become $2 \times (m-1)!$ as there are now two-to-one possible such matches for some permutations in cg_1 satisfying connections between nodes with available edges in their networks. Therefore, at the end of iteration k, MP-CNet now has $2 \times (m-1)!$ and $(n-1)!$ permutations to select as their bests for the next one, respectively. Total edge score, however, is still the same. We claim from this current sample that the number of available edges from one graph, travelling through two clusters and maintaining the other with one edge merely allows a cluster the possibility for up to $n \times (n-1)! = n!$ permutations have edge score, but because of

one edge existence of the other PPI-network graph, there is no way to multiply the total score of these two connected clusters as only one match of their such permutations can perform it. The convergence and optimal alignment is still achieved by the algorithm, but due to the increased number of permutation choices, this also raises the finite number of iterations k to make it happen. Considering the edge disconnection of edges of E_G and/or E_H, we have conjectured that this, unfortunately, often causes no convergence for CG no matter how k_{max} is as much as high. This is such a bizarre condition that by examining the possible best permutations $SCB(cg_i)$ for each cluster in every iteration until the maximum, the best of one or more are completely different than the previous iteration that leave MP-CNet no opportunity to make convergence, though the highest total score can be reached. The discrepancy of dissimilar best permutations of clusters repeat in at least two-iteration cycles, and by checking $\mathcal{PB}$ at k_{max}, these permutation indexes cannot form the optimal alignment but partially i.e. an approximate one is encountered.

A little more complex structure between these two clusters could be that both E_G and E_H contain more than one edge and still going from cg_1 to cg_2. This affects some permutations that achieve conserved interactions externally with more than one matches. Suppose the edges in their respective PPI-network graphs remain the connectivity for having convergence and alignment optimally, now such a permutation from $\pi(cg_i)$ possesses a likelihood that its matches against the ones of the best permutation of the neighbor clusters together produce $c \times (\psi/2)$ edge scores, where c can take any value, indicating the times of happenings of conservation with these permutation matches, here $c > 1$ is preferred for this purpose (Figure 2.6). There can be other permutations that have edge scores but not higher than the bests when all are investigated. Therefore, this shape of two clusters

allow MP-CNet to make the best permutation selection of each cluster with fewer options for determining $\mathcal{PB}$ for the next iteration and can help convergence occurence by a bit less iteration made. Additionally, another important claim is obtained, thanks to our analysis, that the edge score of a cluster permutation can get at most is $c \times (\psi/2)$. In other words, the maximum edge score of a permutation is dependent on the number of edges between two clusters whichever is lower. If there are three edges from E_G and two edges from E_H, going from one cluster to another, for example, then a permutation cluster edge score can be up to $2 \times (\psi/2)$.

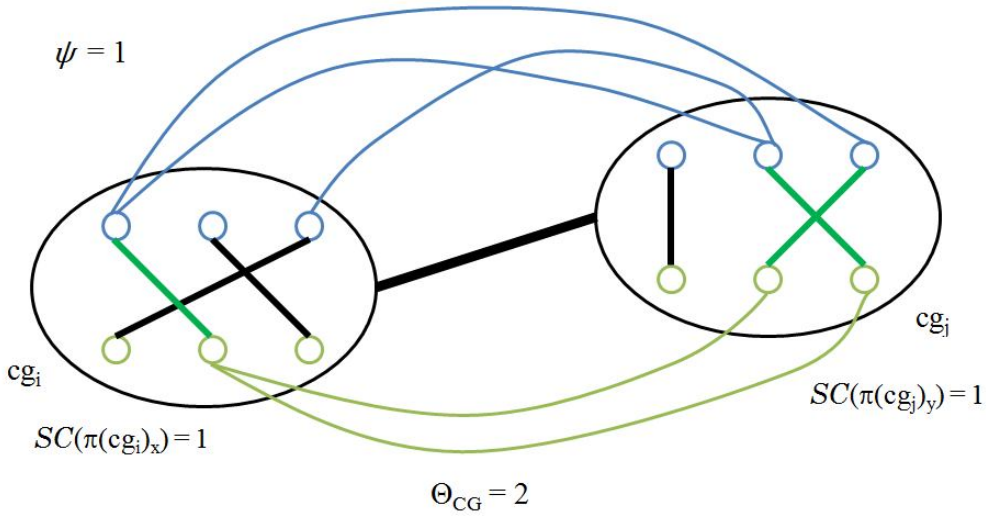

Figure 2.6 Alignment optimality between two clusters with multiple edges

The more complicated cluster network input for the algorithm is surely enough multiple clusters as real examples should contain hundreds of clusters, along with lots of possible cluster edges in terms of the clustering algorithm output. For making necessary interpretations of this form, consider simple CG consists of (i) three clusters where cg_1 is connected to cg_2, and cg_2 is connected to cg_3, (ii) four clusters where the connections are made between cg_1-cg_3 and cg_2-cg_4, so two separate sub-cluster networks and, of course, edge connectivity of proteins is still maintained for

those. About (i), the algorithm always starts at cg_1 with $N(cg_1) = \{cg_2\}$ and definitely, all permutations in $\pi(cg_1)$ are compared with the best permutation $\pi(cg_2)_p$, hence there will be best-scoring permutations to choose from and similarly for cg_3, $N(cg_3) = \{cg_2\}$. For cg_2, however, there are more neighbors detected, say $N(cg_2) = \{cg_1, cg_3\}$ and $N(cg_2) > 1$. MP-CNet traverses these neighbors one by one for permutations to make external conservations with the bests of its neighbors. Here, more important matter is the maximal edge score of a permutation of a cluster being examined can go up to $(c + d) \times (\psi/2)$, where c and d denote the highest times of conservation occurences from neighbor clusters and reflect the lesser number of edges of E_G or E_H moving from this cluster to the related neighbor. From this vision, it is discernable for a cluster that having many neighbor connections can also bring more scores to its permutations and it will become easier for MP-CNet to select the best one if the matches of the permutation are capable of performing the most conserved interactions with the neighbor bests externally, so contributing to convergence of CG in a way that it can always become the best one through iterations. About (ii), the investigation of sub-network clusters are applied independently from each other such that only the neighbors can affect the others with regard to conserved interactions and determination of best permutations. On the one hand, a sub-cluster network has no influence to other familiar structure, even if the convergence happens between these clusters at iteration k, they must wait for the other in order to cary out a full convergence, otherwise they are continued for searching. That is because our algorithm is designed for globally aligning the clusters (and not for local search) that all best permutations must be exactly the same with the previous selections, as indicated in Section 2.5.2 like $\mathcal{PB} = \mathcal{PB}'$. This points out a

disadvantage that causes more iterations to be done overall. Despite under this setting, the optimal alignment is still returned.

So far, we have elucidated a lot for necessities of alignment optimality of the cluster network by also adapting to the methods of MP-CNet. To sum the things up, we propose both E_G and E_H must have sustainable connectivity between each connected clusters in order to converge at a finite number of iterations and at that point the best possible alignment is made to the output, particularly achievable for small or medium sized graphs, otherwise the algorithm is progressed until k_{max} for the best approximate one. Edge disconnections of protein networks across some or many clusters can cause the impossibility of convergence at all, and again an approximate alignment. In addition, we never discussed the situation of the inclusion of fixed scores (internal conserved interactions) for clusters. Here, our researches on many different structures showed that by testing the same instances with fixed scores ($\mathcal{FS}$) do not make any changes to the general optimality of alignment and convergence. There is a relationship between ϕ and ψ unit scores that we always recommend the scoring schema of $\phi \geq \psi$. This is another criterion for getting optimal alignments. Initially, permutations with best fixed scores and with the maximum plausible edge counts become the best of their related clusters before message-passing begins and have more probability to remain it for the most iterations unless another permutation has higher score. In this way, we do not see a loss of total conserved interactions that prevents the optimality but other distinct variants may be encountered like some external conservations are not presented at the end. In contrast where $\phi < \psi$, the algorithm sometimes returns less total score and significantly decreases the likelihood of the alignment to become optimal. This is not the desired situation as keeping both values equal is most convenient way. We

adopt the value of 1.0 for the experiments in Section 3 to see the exact number of total conserved interactions. All in all, these further explanations were made for MP-CNet to show how it works at a smooth rate of performance against the given input.

2.5.4. Discussions for Convergence

Convergence of CG is a crucial feature for MP-CNet as it makes the decision of if we should go on computing the edge scores of all clusters with the updated best permutations or stop immediately and progress to prepare the alignment. Here, convergence, on one hand, ensures no more conserved interactions (i.e. no higher scores) can be produced by the algorithm and also states that continuing after this point causes loss of time. When this is materialized, we always inquire the output alignment and our algorithm with these questions: (a) Does the algorithm always converge? (b) If it converges, does the alignment become optimum? (c) What is the maximum number of iterations until convergence is achieved?

Depending on the structure of both PPI networks and the cluster network which is made in the data processing stage, it is hard to come up with common and proper answers for these questions, due to the spantaneousness of constrained GNA problem and the design of our algorithm to this. Assuming simple and sparse kinds of these networks each, then the questions of our inquiry can be answered without not much difficulties. On the other hand, the matters are getting to impossibility for more complicated structures of both PPI and cluster networks. Despite the hardness, here we attempt to give general responses to these questions and mostly attribute to the claims and analysis that have been obtained during the research and development of MP-CNet, together with the results that have been encountered by many different inputs as they were explained in the previous section.

Starting from question (a), the cluster network can be converged if CG does not have a complicated joint structure among cluster nodes either having lots of small subnetworks or a large one. Hence, the alignment output can be seen without reaching the maximum iteration limit k_{max} if it is high enough. It should not be forgotten that as we have said in the previous section, the edge connectivity of subset of V_G and V_H nodes between any connected clusters must be sustained and again having disconnections of edges can lead to disallow the algorithm to achieve convergence. This is also valid for graphs with more complexity overall and if it is hoped to ensure full edge connectivities across all clusters, then there is still a probability to achieve convergence during the message-passing but after hundreds or even thousands of number of iterations, leaving the only choice to increase k_{max} to be able to encounter.

For question (b), this always causes anxiety such that it is pretty ambiguous that the optimality of alignment or not should be predicated to either the simplicity of PPIs and the cluster network graphs or the performance of our algorithm against the given input. To make it clear, one way is to run the algorithm several times over the same input and see whether convergence is achieved and the same total score (or total conserved interactions) is obtained every time at the point of convergence. If it does not happen, only the total score is observed. Then, we can interpret the alignment has a likelihood to be optimal, otherwise approximate. Another way is, particularly on cluster nodes with totally less than 10 or 15, an exhaustive search (e.g. brute force) is implemented hereafter the completion of the algorithm. Therefore, the optimality of the alignment can be decided by comparing the outcome of the exhaustive search (as it has usually more than one alignment options) with the

one generated by MP-CNet in terms of total score, number of conserved interactions and the best permutations of clusters.

For question (c), we have already given answers in (a) by a large proportion. In addition to this, it is hard to predict the exact range of number of iterations at first for the given input. Of course, for those where there is always convergence, the minimum and maximum number of k iterations can be observed by running several times again or if not, raising k_{max} is the only relief; and in the worst case, it is not observable at all due to the claims we counted above.

2.6. The Whole Algorithm Pseudocode

The general pseudocode of all the stages of MP-CNet together up to and including the message-passing and the alignment output are given below. Please note that some formulations or equations may slightly differ from what was described in the text but they still do the same matter.

MP-CNet(G, H, CG, ϕ, ψ) {

1 BEGIN

2 Construct PPI-network graphs G and H

3 Begin constructing the cluster network graph CG by placing subset of nodes of V_G and V_H into appropriate clusters and necessarily add dummy nodes for equalization

4 Add connections between any cg_i and cg_j **if** edge connectivities of their V_G and V_H nodes are satisfied

5 Generate permutations $\forall cg_i$, stored in $\mathcal{P}$.

6 Calculate fixed scores $\forall \pi(cg_i)$ and $\forall cg_i$, stored in $\mathcal{FS}$
- **if** an edge from E_G and E_H exists between any pair of matches in $\pi(cg_i)_x$, **then** $FS(\pi(cg_i)_x)$ += ϕ

Decide best permutations for each cg_i, hold them in $\mathcal{FSB}$

Count maximum plausible edges $\forall\pi(cg_i)$ and $\forall cg_i$, stored in $\mathcal{EC}$
- Applied rules for any match
  if $pls(g_a) = pls(h_b)$, $pls\big(\pi(cg_i(M(z)))_x\big) \mathrel{+}= pls(g_a) \vee pls(h_b)$
  else if $pls(g_a) < pls(h_b)$, $pls\big(\pi(cg_i(M(z)))_x\big) \mathrel{+}= pls(g_a)$
  else if $pls(g_a) > pls(h_b)$, $pls\big(\pi(cg_i(M(z)))_x\big) \mathrel{+}= pls(h_b)$

Decide best permutations for each cg_i, hold them in $\mathcal{ECB}$

Get common best permutation by $FSB(cg_i) \cap ECB(cg_i)$ and append to $\mathcal{PB}$
- For more than one permutation, randomly pick one of them
- **if** no common best available, **then** select one from $FSB(cg_i)$

// The message-passing method

$k \leftarrow 1$;

if $E_{CG} = \emptyset$, then do not start the method and prepare the output
- Algorithm exits here

while (**not** converged and $k < k_{max}$)

for $\forall cg_i$:
  Get the neighbors $N(cg_i)$ and their best perm.
  if $N(cg_i) \neq \emptyset$:
    for $\forall\pi(cg_i)_x \in \pi(cg_i)$:
      Add fixed score $FS(\pi(cg_i)_x)$ to $SC(\pi(cg_i)_x)$
      for $\forall cg_j \in N(cg_i)$:
        Set the permutation $\pi(cg_j)_p$
        Determine the total score of $\pi(cg_i)_x$ by comparing all possible pair of matches with $\pi(cg_j)_p$
        - if $\Lambda\big[\pi(cg_i(M(y)))_x, \pi(cg_j(M(z)))_p\big] = 1$, then $SC(\pi(cg_i)_x) \mathrel{+}= ((\Lambda \times \psi) \,/\, 2)$
  Assess the best-scoring perm. and append to $\mathcal{PB}'$
  - For more than one permutation, randomly pick one of them
  Aggregate the best score of cg_i to Θ_{CG}
  $CPB(\pi(cg_i)_x) \mathrel{+}= 1$

```
 14 if Θ_CG > maxΘ_CG: maxΘ_CG = Θ_CG

 15 if PB' = PB: mark CG converged

 16 PB ← PB';
    goto step 12

 17 Decide the most selected permutations
    ∀CPB(cg_i) ⊂ CPB and append to ΔPB

 18 Output the alignments from PB and ΔPB by
    showing the total score maxΘ_CG of PB, matches and
    CI_CG, CI_ΔCG

 19 END
}
```

MP-CNet is written in C++ and LEDA (Mehlhorn and Naher, 1999). The reason for using LEDA libraries is to handle graph structures easily and store necessary values in single or multi-dimensional list variables as the data structure allows them to become flexible (i.e. not static) throughout the algorithm.

Chapter 3

Experiments and Results

MP-CNet algorithm works smoothly over the clustered network as well as maximum amount of conserved interactions possible from pairwise PPI networks and the whole alignment are delivered to the output. To estimate how robust and fast our algorithm is, we have experimented with other algorithms on various biological sources. Here, the experiments are mainly done with balanced GNA algorithms, although their approaches of the problem solution has differences with ours. They are primarily quantified on three important factors: (i) Assess the total number of conserved interactions, (ii) observe the time elapsed while perpetrating the data and (iii) a special experiment to measure the biological quality of the alignments of algorithms in terms of Gene Ontology (GO) annotations that proteins possess their biological features (Ashburner et al., 2000).

3.1. Comparisons with Algorithms

We chose SPINAL[2] (Aladag and Erten, 2013), IsoRank[3] (Singh et al., 2008) and MI-GRAAL[4] (Kuchaiev and Przulj, 2011) graph matching algorithms which are balanced GNA algorithms to do tests with MP-CNet. The basis of these experiments

[2] SPINAL is available to download at http://hacivat.khas.edu.tr/~cesim/spinal.html

[3] IsoRank and other variants as well as IsoBase database are available at http://groups.csail.mit.edu/cb/mna/isobase/index.html

[4] MI-GRAAL is available to download at http://bio-nets.doc.ic.ac.uk/MI-GRAAL/

is to infer if our algorithm, designed to work on constrained global networks, performs as good as against the balanced ones. For data sources, we used the database of IsoBase (Park et al., 2011) and acquired the networks of four species: *Caenorhabditis elegans* (worm), *Drosophila melanogaster* (fruit fly), *Homo sapiens* (human) and *Saccharomyces cerevisiae* (yeast). Moreover, we benefited from Database of Interacting Proteins-DIP[5] (Xenarios et al., 2002) to identify which proteins nestle IDs of DIP from their respective species in IsoBase. Although this causes reduction to total number of proteins as well as interactions, it becomes more tractable to observe the outcome of alignments generated by the algorithms. By sorting out the proteins of each species, total of 2621 proteins of *C. elegans*, 7017 proteins of *D. melanogaster*, 942 proteins of *H. sapiens* and 4894 proteins of *S. cerevisiae* contain IDs from DIP database (format shape as DIP:xxxxxN). The next step is to fetch the protein sequences that are available in FASTA format and they were downloaded from DIP website. With careful analysis, we identified that sequences of a few proteins from each species do not exist, so they are taken off from the input.

Having protein sequences in FASTA format are crucial for Inparanoid algorithm (Remm et al., 2001); the only accepted type of input, for generating clusters as well as ortholog and in-paralog proteins in terms of sequence similarities with BLAST scores and, of course, it composes a part of the input for MP-CNet. Additionally, the algorithm allows third group as an out-group which is useful for eliminating false ortholog assignments but also a risk for losing gracious ones. For this reason, we never used out-groups while generating the clusters and these parameter values are entered: score cut-off of 20 bits, confidence cut-off of 0.05,

[5] DIP database is available at dip.doe-mbi.ucla.edu/dip/Main.cgi

sequence overlap cut-off of 0.5, group merging cut-off of 0.5, no grey zone and BLOSUM62 scoring matrix i.e. the one for comparison of eukaryotes. All of these except score cut-off are default values. Furthermore, we let Inparanoid to run BLAST with two-pass strategy between protein sequences of two species in order to perform clusterization easily due to no knowledge about how much the similarity of all involved proteins are at the beginning. As a result, clusters of every combination of pairwise species are attained, but not all proteins are covered in accordance with the defined values and naturalness of the mechanisms of Inparanoid. Here, *C. elegans-D. melanogaster*, *C. elegans-H. sapiens*, *C. elegans-S. cerevisiae*, *D. melanogaster-H. sapiens*, *D. melanogaster-S. cerevisiae* and *H. sapiens-S. cerevisiae* pairs generate 976, 171, 509, 321, 1300 and 187 clusters respectively. Roughly three fourth of them are one-to-one clusters while the rest are one-to-many and many-to-many. Refer to Table 3.1 for more details.

Table 3.1 Results of the clusterization made by Inparanoid

Pairs	Inparanoid Clusters	Proteins Covered	One-to-one Clusters	One-to-many Clusters	Many-to-many Clusters
C.eleg D.mela	976	1045 / 1219	817	144	15
C.eleg H.sapi	171	194 / 235	124	40	7
C.eleg S.cere	509	597 / 627	397	98	14
D.mela H.sapi	321	392 / 434	227	85	9
D.mela S.cere	1300	1577 / 1524	1011	257	32
H.sapi S.cere	187	232 / 227	132	51	4

Besides, we have configured these four species to give them shape as graph structures (i.e. PPI networks) and taken appropriate interactions for full coherence; fulfilling our intention that they now become the input properly for use in MP-CNet

and other balanced GNA algorithms as well with further necessary comparisons. The final input of all species are as follows: 2538 proteins and 3801 interactions in *C. elegans*, 6983 proteins and 22367 interactions in *D. melanogaster*, 895 proteins and 2459 interactions in *H. sapiens* and 4874 proteins and 27712 interactions of *S. cerevisiae*.

In these experiments, SPINAL and IsoRank are executed with various values of the control parameter ($\alpha \in [0,1]$), which is needed to determine usage ratio between network topology and sequence similarity. Starting from α = 0.3, and progressively increasing by 0.1 up to 0.7, different alignment outputs were produced. Additionally, two versions of SPINAL are used as it consists of two main parts; one with the usage of coarse and fine-grained phases (call SPINAL-2 in this experiment) and the other with the usage of only coarse-grained phase (call SPINAL-1) (Aladag and Erten, 2013). For MP-CNet and MI-GRAAL, they do not include a control parameter, hence only one alignment output is used for comparison. MP-CNet is run on available pairs of species several times and the best one yielding the most conservations is acquired, preferably with the results of $\mathcal{PB}$ if the numbers are equal to that of $\Delta\mathcal{PB}$ (see Section 2.5.2). For the alignment of MI-GRAAL algorithm, signatures, degrees, clustering coefficients and BLAST e-value scores for sequence similarity are used together, which composes the *Alignment3* version and gives the most conservations under this setting (Kuchaiev and Przulj, 2011). These experiments were run on a computer with Linux-based operating system, 3.3 GHz x86-64 processor and 3 GBs of RAM.

3.1.1. Evaluation of Conserved Interactions

Counting conservations are based on the matches of alignments that balanced GNA algorithms generated upon completion, including MP-CNet. Firstly, we take all

of them even their numbers are larger than ours highly. Later, for a fair comparison, filterings are applied to all alignments of balanced GNAs to detect which proteins of all matches are available in ones covered in clusters by Inparanoid, regardless of where they belong to. For example, about two organisms between *C. elegans* and *D. melanogaster*, for any match, the protein of *C. elegans* is explored to see if the same one is covered in a cluster, then the other one of *D. melanogaster*, thus it becomes available in the filtered matches of the alignment. Total number of matches of all algorithms are available in Table 3.2, where for those with α are averaged. Note that SPINAL-1 could not complete successfully for *D. melanogaster-S. cerevisiae* pairs, so it is marked with (X) including this and the following tables.

Table 3.2 Number of matches of alignments generated by algorithms (unfiltered / filtered); only unfiltered numbers in MP-CNet column

Pairs	MP-CNet	SPINAL-1	SPINAL-2	IsoRank	MI-GRAAL
C.eleg D.mela	995	2495 / 818	2495 / 350	2523 / 389	2538 / 214
C.eleg H.sapi	180	837 / 133	840 / 61	874 / 70	895 / 28
C.eleg S.cere	524	2354 / 432	2403 / 175	2523 / 290	2538 / 92
D.mela H.sapi	330	844 / 256	856 / 65	874 / 51	895 / 43
D.mela S.cere	1339	X	4871 / 441	4872 / 742	4874 / 311
H.sapi S.cere	191	774 / 160	831 / 52	874 / 65	895 / 11

Before the interpretations of results, there is an important factor that must be conceded at the beginning. Since balanced GNA algorithms perform one-to-one matching of proteins as a whole, and our algorithm is adapted to match them on the basis of restriction in clusters, we do not expect a greater number of conservations than others. It is possible for these algorithms to find hundreds of conserved interactions overall. Therefore, we compare by only taking the filtered alignments,

better for more equitable. These experiments are also made to learn how competitive an algorithm for the solution of constrained GNA problem could be and the general position where it could be placed between those with balanced ones.

Numbers of conserved interactions are determined by picking up two pairs of matches and traversing all possible combinations, say $M(i), M(j) \in \mathcal{M}; 1 \leq i < m$ and $i < j \leq m$. All instances of compared algorithms are investigated with all edges of two PPI networks, E_G and E_H. By cautious implementation, we have extracted these numbers of filtered alignments that are presented in Table 3.3.

Table 3.3 Number of total conserved interactions extracted by algorithms (filtered); while MP-CNet remains unfiltered. Note that numbers ranging from 0.3 to 0.7 indicates the control parameter value (α).

Pairs	MP-CNet	SPINAL-1 / SPINAL-2					IsoRank					MI-GRAAL
		0.3	0.4	0.5	0.6	0.7	0.3	0.4	0.5	0.6	0.7	
C.eleg D.mela	35	28/48	28/32	29/34	31/33	30/45	13	13	12	13	12	32
C.eleg H.sapi	6	5/3	5/5	4/4	4/5	4/0	2	2	2	2	2	2
C.eleg S.cere	25	24/20	24/21	21/23	22/22	23/27	13	11	10	11	12	1
D.mela H.sapi	27	14/2	14/6	15/4	14/5	13/2	3	3	3	3	3	9
D.mela S.cere	99	X/50	X/60	X/56	X/70	X/66	53	53	55	56	51	26
H.sapi S.cere	62	49/7	47/9	47/9	47/8	46/6	16	16	18	17	16	1

From what we have ascertained from the table above, MP-CNet generally performs better than balanced GNA algorithms in terms of total found conserved interactions, mainly due to the included number of matches to filtered alignments that they are not much. Here, excluding our algorithm, SPINAL performs the best nearly in all instances, then sometimes MI-GRAAL comes after and sometimes IsoRank. On the other hand, it is inevitable for these balanced GNA algorithms to have a greater number of convservations when alignments are not undergone for filtering, so this makes MP-CNet become behind these algorithms. In *C. elegans-D.*

melanogaster, for example, alignments of SPINAL-1, SPINAL-2 and IsoRank averagely contain 649, 1972 and 292 conserved interactions and the number for MI-GRAAL is 2052 as only one alignment is available.

During the experiments, we have discovered a correlation for MP-CNet; first, observe the total number of connections (i.e. cluster edges) made between clusters in the following order of pairs of species in Table 3.3: 35, 6, 25, 26, 99 and 62. It remarks that these numbers are equal or very close to the total conservation numbers, definitely indicating there is only one (but not two or more for almost all) external interaction of every connected pairs of clusters and no fixed scores[6]. Herewith, it could be implied that clustering results of Inparanoid and all available interactions in E_G and E_H have direct influences on the success of MP-CNet with an elevated percentage. The general shape they constituted may take the algorithm to the worse situation than balanced GNAs or better, depending on the alignments in unfiltered or filtered form. That is, for a pair of species, MP-CNet can be an alternative tool for uncovering the conservation of proteins by a higher amount than what other algorithms achieved so far if matches of proteins are checked to be covered in cluster, i.e. alignments are filtered with regard to the results of clustering algorithms.

3.1.2. Runtime Performances

Total elapsed time is another considerable criterion for all algorithms while finding the conservations. It is chosen to become preferable if such one can make it faster than others naturally. For this reason, we also measured the time for all algorithms against the instances in this category of our experiments (Table 3.4).

It can be smoothly realized that runtime of any balanced GNA algorithm is changed steadily such that if PPI networks become denser, then the time increases

[6] Only 1 internal conserved interaction (fixed score) is available for *D. melanogaster-S. cerevisiae*.

accordingly. Our analysis showed that MI-GRAAL by far is the slowest algorithm for process of the alignment although given number of conserved interactions could be high when unfiltered (see Aladag and Erten, 2013). Runtime of IsoRank is fairly

Table 3.4 Total runtime of the algorithms against all instances

Pairs	MP-CNet	SPINAL	IsoRank	MI-GRAAL
C.eleg-D.mela	35 mins	6 mins	2-6 mins	2 hours
C.eleg-H.sapi	31 secs	1 min	19-36 secs	5,5 mins
C.eleg-S.cere	13 mins	6 mins	1,5-6 mins	1,4 hours
D.mela-H.sapi	49 mins	3 mins	1-2 mins	23 mins
D.mela-S.cere	2,9 hours	15 mins	7-33 mins	4,1 hours
H.sapi-S.cere	91 secs	5 mins	1-2 mins	22 mins

good and comparable to SPINAL, but low amount of conservations is its weakness on the way. We gave a range of runtimes to that algorithm because changing the control parameter affects the total duration, so the number of iterations it had made for its alignment optimality. There is no doubt for SPINAL in general so that it is a fast algorithm while the alignment output it constituted is the best among other balanced ones if we do not count MP-CNet for filtered ones, thanks to the scalability feature it possesses for what makes it the state-of-the-art algorithm of today (Aladag and Erten, 2013).

For MP-CNet, the runtimes are not regular regardless of the size of the PPI networks. Due to this reason, it is ambiguous where we put the algorithm next to the others. According to the table above, what we learnt that its computation is faster than MI-GRAAL usually. Sometimes, the fastest algorithm especially for *C. elegans-H. sapiens* and *H.sapiens-S. cerevisiae* and if some samples of IsoRank with lower value of α are not counted. For *D. melanogaster-H. sapiens*, the algorithm becomes the slowest among all others. Though the estimation of this situation is pretty hard, it

could be answered partially with the claims we have gotten in the previous test branch: The success rate and total time of our algorithm is not only dependent on the implementation of the message-passing method and heuristic approaches but also the shape of PPI network inputs they form, the set of clusters, proteins covered in clusters and edge connections across all clusters.

3.1.3. Discussion of Our Algorithm Runtime Determinants

The discrepancy of runtime of MP-CNet is concatenated to lots of elements, such as number of proteins and interactions in PPI networks, set of clusters, proteins comprised in them; total number of connections and generated permutations across all clusters, etc. As it can be seen, the total time is likely to be under a minute, in several minutes or even a few hours, perhaps a bit more in the worst cases. Unlike balanced GNA algorithms, it becomes challenging to predict the total duration properly if such an algorithm is trained for solving the constrained GNA problem.

Despite this ambiguity, a general assessment can be done with regard to what we have experienced for many different sizes of input: Constructing the graphs of PPI networks does not take much time almost and so placing the covered nodes into the appropriate clusters. However, for the detection of edge additions to E_{CG} between clusters, the overall time goes up if the number of clusters are high as well as available interactions of the networks of species. Then, these are used in other various stages such as getting fixed scores, plausible edge counts and, indeed, in the message-passing, proportionally affecting the elapsed time. Having more cluster permutations can also add more time for examination of fixed conservations, edge counts, external conservations and each selected pairs of matches. Note that one may hope in the message-passing stage that a cluster with permutations in a huge size does not have any neighbor, so every iteration is passed in a fast manner.

For a cluster containing lots of pairs of nodes (say, 10 or 15), the amount of permutations drastically increases to millions even billions, extending the total time much more, mainly because of $n!$. This points out to one of the weaknesses of MP-CNet so that storing an incredible size of permutations requires a very high amount of memory that current modern computers may not hold them together, causing the algorithm to fail before the completion. To address this critical issue, we have applied a limit to clusters to make them have maximum number of node subset (mostly chosen 8, for this purpose), thus reducing the size of permutations and allowing comfortable progression. However, this also influences the total number of proteins covered as removed ones may have been in part of conserved interactions or have provided connections between clusters, which is not desirable. This restriction was reflected to the experiments we have materialized above for our algorithm and others in the next section in order to be able to complete for many instances.

3.2. Biological Significance

One of the other necessary measurement of alignments is the overall biological quality they own across all one-to-one matches, mainly based on gene ontology terms (GO). These are widely used to assimilate the representation of genes or proteins characteristics across the species and they consist of detectable or directly observable stuff that are represented and the relationships are shown. The ontology involves three main domains: *Biological Process*, for operations or sets of molecular events; *Cellular Component*, indicating the parts of a cell or its environment out of the cell and *Molecular Function*, showing the elemental activities of genes at the molecular level.

GO annotations of each species are retrieved from the Gene Ontology Consortium website[7] (Ashburner et al., 2000). In our experiments, we took advantage of the sources of SPINAL algorithm as they already uncovered which GO annotations have been possessed by the proteins of species. The admissable calculation method of GO consistency score is composed of counting the overlapped GO terms that annotate pairs of matched proteins, collectively. We benefit from a formulization introduced in SPINAL to measure the score for all algorithms in comparison (Aladag and Erten, 2013):

$$GOC(\mathcal{M}) = \sum_{\forall \kappa \in Z^+} (\kappa \times |GO_\kappa|)$$
$$GO_\kappa = \{(g_a, h_b) \in \mathcal{M} : |GO(g_a) \cap GO(h_b)| = \kappa\} \quad (3.1)$$

Here, κ denotes how many common GO annotations between g_a and h_b, easily accomplished by the intersection of these proteins, where $GO(g_a)$ and $GO(h_b)$ indicates all available annotations of GO terms possessed by the proteins g_a and h_b, respectively. The total score is assessed by accumulating all found common ones, multiplying κ with its respective total set of overlap occurences among all proteins, exactly that number.

A comprehensive experiment has been carried out by taking all matches from unfiltered and filtered alignments of balanced GNA algorithms against MP-CNet. Table 3.5 and 3.6 represents the results of these examinations. First of all, the GO consistency scores of MP-CNet look impressive. Receiving such quite good results are more than what we have expected. The reason why we keep it in the lower level at the beginning is because total conserved interactions may also have an affect to overall GO consistency. However, these results showed that this situation should be

[7] Freely available at http://www.geneontology.org

concatenated to clusters and more than that the generation made by Inparanoid. As this algorithm does its best to cover the most orthologous proteins in many clusters; on one hand, these may carry the GO annotations that are mostly common to each other, and our algorithm has the potential to reveal these similar biological functions by a high proportion. This convinces that finding the annotations has no dependence to the availability of conserved interactions.

Table 3.5 GO consistency scores of all alignments of the algorithms (unfiltered)

Pairs	**MP-CNet**	**SPINAL-1 / SPINAL-2**				
		$\alpha = 0.3$	**$\alpha = 0.4$**	**$\alpha = 0.5$**	**$\alpha = 0.6$**	**$\alpha = 0.7$**
C.eleg-D.mela	**1504**	1874 / 736	1836 / 693	1814 / 640	1771 / 668	1761 / 682
C.eleg-H.sapi	**205**	375 / 236	371 / 205	369 / 203	382 / 219	363 / 197
C.eleg-S.cere	**1385**	1948 / 827	1945 / 880	1925 / 894	1923 / 875	1916 / 845
D.mela-H.sapi	**566**	734 / 257	718 / 250	672 / 227	646 / 243	621 / 191
D.mela-S.cere	**3539**	X / 1586	X / 1562	X / 1451	X / 1426	X / 1349
H.sapi-S.cere	**512**	919 / 487	880 / 450	877 / 444	862 / 487	845 / 514
Pairs	**MI-GRAAL**	**IsoRank**				
		$\alpha = 0.3$	**$\alpha = 0.4$**	**$\alpha = 0.5$**	**$\alpha = 0.6$**	**$\alpha = 0.7$**
C.eleg-D.mela	374	1169	1160	1135	1126	1116
C.eleg-H.sapi	142	263	266	265	258	244
C.eleg-S.cere	489	1553	1534	1542	1522	1530
D.mela-H.sapi	160	322	311	306	309	301
D.mela-S.cere	859	3407	3444	3448	3450	3405
H.sapi-S.cere	315	596	617	613	613	592

Table 3.6 GO consistecny scores of all alignments of the algorithms (filtered); while MP-CNet alignment remains unfiltered

Pairs	**MP-CNet**	**SPINAL-1 / SPINAL-2**				
		$\alpha = 0.3$	**$\alpha = 0.4$**	**$\alpha = 0.5$**	**$\alpha = 0.6$**	**$\alpha = 0.7$**
C.eleg-D.mela	**1504**	1286 / 345	1256 / 337	1226 / 243	1200 / 280	1177 / 302
C.eleg-H.sapi	**205**	168 / 67	152 / 43	146 / 46	131 / 44	134 / 29
C.eleg-S.cere	**1385**	1211 / 255	1208 / 345	1188 / 280	1190 / 299	1186 / 278
D.mela-H.sapi	**566**	448 / 77	436 / 76	408 / 61	407 / 70	381 / 38
D.mela-S.cere	**3539**	X / 538	X / 522	X / 461	X / 473	X / 393
H.sapi-S.cere	**512**	458 / 131	439 / 100	437 / 89	426 / 125	413 / 115
Pairs	**MI-GRAAL**	**IsoRank**				
		$\alpha = 0.3$	**$\alpha = 0.4$**	**$\alpha = 0.5$**	**$\alpha = 0.6$**	**$\alpha = 0.7$**
C.eleg-D.mela	61	634	629	610	618	605
C.eleg-H.sapi	9	75	75	68	68	60
C.eleg-S.cere	65	823	818	814	797	799
D.mela-H.sapi	33	105	99	97	97	96
D.mela-S.cere	121	2014	2023	2007	1992	1970
H.sapi-S.cere	4	193	193	207	201	196

By comparing the results with other balanced GNA algorithms, ours can outperform in many instances against unfiltered matches of the alignments. Here, SPINAL-1, the first version of the algorithm, provide the best GO consistecy score in all pairwise species. We could have said the same for *D. melanogaster- S. cerevisiae* pairs if the algorithm had accomplished the alignment without a problem, therefore marked with (X) as we stated in Table 3.2. IsoRank also performs nicely, and accumulates more score than MP-CNet in all α parameter values for *C. elegans-H. sapiens*, *C. elegnans-S. cerevisiae* and *H .sapiens-S. cerevisiae*, but still not higher than SPINAL-1. Additionally, there is another case, attracting attention that MI-GRAAL does not seem to be designed for extracting the similar GO annotations of matched proteins as confirmed by our results that it always stays behind than other algorithms, including MP-CNet. Observing the other table that involves the results of filtered alignments, every balanced GNA algorithm, including those with the variants of the control parameter values, GO scores are not higher than our algorithm. This is obviously because all matches of alignments are checked for the existence in clusters in terms of Inparanoid outcomes and that decreases the total number of matches pretty much. Despite this filtering, the superiorities between the algorithms (if we do not count ours), still continues exactly the same. Last, but not the least, this series of experiments constitute a substantial case study for balanced GNA algorithms to measure the performance overall including those were materialized in Section 3.1, and allows MP-CNet to be placed anywhere through them even though the differences of problem solution approach.

Conclusion

We presented a new algorithm, MP-CNet, which executes on constrained pairwise protein-protein interaction (PPI) networks by taking advantage of the message-passing strategy with belief propagation to perform global alignment with the set of cluster groups, generated by Inaparanoid. It was explained in details that many data processing stages should be applied in order to have a relevant input for the enforcement of the message-passing method on the cluster network and not experience any serious errors. The algorithm can make the convergence, indicating the maximum number of conservation of interactions achieved after finite number of iterations, but mostly depends on the structure of cluster network and PPI networks to materialize as emphasized many times. Moreover, it was explained that the edge connectivity between proteins of individual PPI networks must be preserved across all connected clusters for convergence and higher contingency of optimal alignment.

Experiments showed that MP-CNet, in many instances, could yield more conserved interactions than balanced GNA algorithms, when their whole alignments were taken with filterings applied such that the matches from proteins were checked with the ones covered in Inparanoid clusters to make reasonable comparisons. In addition, the general quality of biological features on the alignment output was tested with these algorithms in terms of Gene Ontology (GO). With no filter to the alignments and then with filters, alignments of MP-CNet has more enhanced biological quality in many cases, sometimes with fascinating amounts, though more

researches may be required if the most similar proteins in respective clusters in terms of their sequences precisely give higher rate of common GO annotations.

Despite the admissable results by the aspect of conserved interaction numbers, our algorithm is highly dependent on the amount of connections carried out among and within all cluster nodes and the clustering results of Inparanoid. There is a significant correlation between the edge connections of clusters and conservation amount as the outcomes assess these values are very close to each other, and including almost no conservation within clusters (fixed scores) at all. This often causes a direct impact to stay behind to balanced GNA algorithms, assuming the alignments are not filtered. However, in reality, we are not limited to solely use Inparanoid for clustering information. MP-CNet can be adapted to those that are created from the other clustering algorithms as well, like HomoloGene, OrthoMCL (Li et al., 2003), etc. More than that, we can also implement our own clustering mechanism that covers almost all proteins. With this way, the comparisons can be repeated again to observe if it brings improvements to the global alignment of our algorithm overall and so the biological quality. In the meantime, instead of only using IsoBase (Park et al., 2011) and DIP databases (Xenarios et al., 2002), we can benefit from different databases, for instance BioGRID (Stark et al., 2006), HPRD (Mishra et al., 2006), Ensembl (Flicek et al., 2013), NCBI (Wheeler et al., 2007) and merge protein informations and networks together to have a more comprehensive input to MP-CNet. Thinking about the running time, sometimes a large portion of time elapsed is lost to process of the input data with regard to total proteins and interactions of PPI networks, connections of clusters, covered total number of proteins before the message-passing method begins. Therefore, this can be separated from the algorithm as a standalone program and appropriate inputs can be made

available readily in general, enabling to become competitive with other algorithms in terms of running time. Along with this, more heuristic approaches and scoring schemes could be added to the algorithm for making more powerful for aligning proteins, but it is still open for research and development. Another affair to discuss is MP-CNet fails to complete if such a cluster contains plentiful amount of protein pairs with the inclusion of dummy nodes, causing to have incredible size of permutations that total memory of a computer cannot handle them at all. We have applied a workaround to this issue by limiting clusters to have a predefined maximum amount of pairs, thus reducing the available permutations quantity. Despite this facility, these removed proteins may be valuable to the whole cluster network as it can affect connectivity of clusters and conservations of proteins. For this reason, more intelligent methods should be implemented in the future, handling proteins, permutations and matches more effectively.

By presenting the capability of our messge-passing algorithm for constrained GNA problem, we assure it creates remarkable insights to researchers; so methods with better results could be produced, as they altogether could make considerable contributions to the understanding of the problem, bioinformatics and the identification of organism interactions.

References

Aebersold, R. and Mann, M. (2003). Mass spectrometry-based proteomics. *Nature*, 422(6928), 198-207.

Aji, S. M., and McEliece, R. J. (2000). The Generalized Distributive Law. *IEEE Trans. Inform. Theory*, 46, 325-343.

Aladag, A. E., and Erten, C. (2013). SPINAL: Scalable Protein Interaction Network Alignment. *Bioinformatics*, 29(7), 917-924.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, 215(3), 403-410.

Ay, F., Kellis, M., and Kahveci, T. (2011). SubMAP: Aligning Metabolic Pathways with Subnetwork Mappings. *Journal of Computational Biology*, 18(3), 219-235.

Bandyopadhyay, S., Sharan, R., and Ideker, T. (2006). Systematic identification of functional orthologs based on protein network comparison. *Genome Research*, 16(3), 428-435.

Chen, J., Mackey, A. J., Vermunt, J. K., and Roos, D. S. (2007). Assessing Performance of Orthology Detection Strategies Appied to Eukaryotic Genomes. PLoS ONE, 2(4), e383.

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

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

Finley, R. L. and Brent, R. (1994). Interaction mating reveals binary and ternary connections between drosophila cell cycle regulators. *Proc. Natl. Acad. Sci. USA*, 91(26), 12980-4.

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. B., Srinivasan, B. S., and Batzoglou, S. (2008). Automatic parameter learning for multiple network alignment. *RECOMB*, pages 214-231.

Horn, G. B. (1999). *Iterative Decoding and Pseudocodewords*. Ph.D. dissertation, Dept. elect. Eng., Calif. Inst. Technol., Pasadena, CA.

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

Kelley, B. P., Sharan, R., Karp, R. M., Sittler, T., et al. (2003). Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proceedings of the National Academy of Sciences*, 100(20), 11394-11399.

Kelley, B. P., Yuan, B., Lewitter, F., Sharan, R., et al. (2004). PathBLAST: a tool for alignment of protein interaction networks. *Nucleic Acids Research*, 32(Web-Server-Issue), 83-88.

Klau, G. (2009). A new graph-based method for pairwise global network alignment. *BMC Bioinformatics*, 10, S59.

Koyutürk, M., Kim, Y., Topkara, U., Subramaniam, S., et al. (2006). Pairwise alignment of protein interaction networks. *Journal of Computational Biology*, 13(2), 182-199.

Kuchaiev, O., Milenkovic, T., Memisevic, V., Hayes, W., and Przulj, N. (2010). Topological network alignment uncovers biological function and phylogeny. *Journal of The Royal Society Interface*, 7(50), 1341-1354.

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

Lauritzen, S. (1996). *Graphical Models*. Oxford University Press.

Li, L., Stoeckert, C., J., and Roos, D. S. (2003), OrthoMCL: identification of ortholog groups for eukaryotic genomes. *Genome Research*, 13, 2178-2189.

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, 253-258.

Mehlhorn, K., and Naher, S. (1999). *Leda: A Platform for Combinatorial and Geometric Computing.* Cambridge University Press.

Milenkovic, T., Leong Ng, W., Hayes, W., and Przulj, N. (2010). Optimal network alignment with graphlet degree vectors. *Cancer Inform.*, 9, 121-137.

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, D295-D300.

Pearl, J. (1982). Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. In *Proceedings of the National Conference on Artificial Intelligence*, 133-136.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Francisco, CA: Morgan Kaufmann.

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.

Sato, T., Hanada, M., Bodrug, S., Shinji, I., et al. (1994). Interactions among members of the Bcl-2 protein family analyzed with a yeast two-hybrid system. *Proc. Natl. Acad. Sci. USA*, 91, 9238-9242.

Sharan, R., Suthram, S., Kelley, R. M., Kuhn, T., et al. (2005). Conserved patterns of protein interaction in multiple species. *Proceedings of the National Academy of Sciences of the United States of America*, 102(6), 1974-1979.

Singh, R., Xu, J., and Berger, B. (2007). Pairwise global alignment of protein interaction networks by matching neighborhood topology. *Research in Computational Molecular Biology*, pages 16-31. Springer.

Singh, R., Xu, J., and Berger, B. (2008). Global alignment of multiple protein interaction networks. *Proceedings of Pacific Symposium on Biocomputing*, pages 303-314.

Yedidia, J., Freeman, W., and Weiss, Y. (2000). Generalized Belief Propagation. *Mitsubishi Elect. Res. Lab.*, TR-2000-26.

Zaslavskiy, M., Bach, F. R., and Vert, J.-P. (2009a). A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12), 2227-2242.

Zaslavskiy, M., Bach, F. R., and Vert, J.-P. (2009b). Global alignment of protein-protein interaction networks by graph matching methods. *Bioinformatics*, 25, i259-i267.