A Memetic Algorithm Approach to Network Alignment

Mapping the Classification of Mental Disorders of DSM-IV with ICD-10

Mohammad Nazmul Haque
School of Elect. Eng. and Computing
The University of Newcastle
Callaghan, NSW, Australia
mohammad.haque@newcastle.edu.au

Luke Mathieson
School of Software
University of Technology Sydney
Ultimo, NSW, Australia
luke.mathieson@uts.edu.au

Pablo Moscato
School of Elect. Eng. and Computing
The University of Newcastle
Callaghan, NSW, Australia
pablo.moscato@newcastle.edu.au

ABSTRACT

Given two graphs modelling related, but possibly distinct, networks, the alignment of the networks can help identify significant structures and substructures which may relate to the functional purpose of the network components. The Network Alignment Problem is the NP-hard computational formalisation of this goal and is a useful technique in a variety of data mining and knowledge discovery domains. In this paper we develop a memetic algorithm to solve the Network Alignment Problem and demonstrate the effectiveness of the approach on a series of biological networks against the existing state of the art alignment tools. We also demonstrate the use of network alignment as a clustering and classification tool on two mental health disorder diagnostic databases.

CCS CONCEPTS

• **Theory of computation** → **Graph algorithms analysis**; • **Computing methodologies** → **Search methodologies**; • **Applied computing** → *Life and medical sciences.*

KEYWORDS

Network Alignment, Memetic Algorithm, Graph Theory, DSM-IV, ICD-10, protein-protein Interaction network, psychopathology

ACM Reference Format:
Mohammad Nazmul Haque, Luke Mathieson, and Pablo Moscato. 2019. A Memetic Algorithm Approach to Network Alignment: Mapping the Classification of Mental Disorders of DSM-IV with ICD-10. In *Genetic and Evolutionary Computation Conference (GECCO '19), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3321707.3321753

1 INTRODUCTION

Network alignment (NA) is the process of mapping one network to another based on their structural similarities. This topological matching approach between networks plays a pivotal role in projecting biological significance from well-known species to less-studied species. It also has profound applications in Protein-Protein interaction (PPI) networks, genome sequence alignment, metabolic networks and other bioinformatics applications [5, 6, 18].

Network alignment can be seen as a generalized and more challenging case of the subgraph isomorphism problem. In the subgraph isomorphism problem, we have to determine if one graph (G) is an *exact* sub-graph of another graph (H). However, in network alignment problem we have to find the best way to "fit" G into H, even if there *does not exist* any exact sub-graph [4].

Algorithms for aligning networks fall into two main classes: global and local. Global network alignment (GNA) [15] aims to maximise the overall similarity (at a global scale) of the networks. In contrast, local network alignment (LNA) attempts to identify a single highly similar sub-region between networks. Hence, GNA produces a one-to-one mapping of the nodes of one network into another. Many algorithms have been proposed so far for GNA [6, 10, 14]. Generally, GNA works in two phases: (1) compute pairwise network similarities as a *cost function* and (2) use an *alignment strategy* to achieve the highest total scores for the whole networks. The aforementioned algorithms and many others were designed specifically for biological networks [8], but can be applied elsewhere. However, the comparison of whole networks for Network Alignment is NP-Complete [11]. Recently, Mathieson *et al.* [12] proved that this problem is W[1]-complete for several parameterizations, further supporting the use of heuristic approaches.

For very large networks, current GNA algorithms run slowly due to using local network alignment algorithms in a greedy manner to solve GNA [23]. Additionally, the biologically inspired algorithms have low alignment accuracy due to utilising weights to give preferences to the specific biological modules. These shortcomings of existing algorithms motivated us to propose the new method for GNA utilising the structural similarity measures to improve alignment quality in a more general setting.

Our new algorithm for GNA utilizes a memetic algorithm framework [13]. The algorithm exploits an index shuffling mutation and a modified partially matched crossover recombination operator in the evolutionary phase, and a simple 2-opt style local search procedure. The fitness function measures the overlap in edges under the alignment, but also includes a term for encouraging the alignment of vertices in the two networks that are the "same".

Although this approach is relatively straight-forward, it is in fact competitive with state-of-the-art alignment tools. We use five different metrics and three PPI networks to compare the performance of our approach with existing tools, with our approach exceeding the performance of the other tools in most cases.

GECCO '19, July 13–17, 2019, Prague, Czech Republic
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6111-8/19/07.
https://doi.org/10.1145/3321707.3321753

To demonstrate the effectiveness of our approach in knowledge discovery, we also apply our algorithm to networks of diagnostic criteria for mental health disorders developed from the two leading diagnostic handbooks (ICD-10 and DSM-IV). Although these handbooks were developed independently and employ different diagnostic criteria, our algorithm not only aligns similar conditions, it also aligns semantically similar diagnostic criteria using only the topological information in the graph.

Section 2 introduces definitions and formalises the GNA problem. Section 3 details the algorithm and its components, along with technical aspects of the performance tests. Section 4 discusses the experiments performed and their results. In Section 5 the results of the alignment of the two mental health networks is discussed.

2 PRELIMINARIES AND DEFINITIONS

Firstly, we present a brief introduction to some basic notions here in order to provide more context about the problem.

An undirected graph (or simply *graph*) is denoted $G(V, E)$ where V is a non-empty set of *vertices* (also called *nodes*) and E is a set of unordered pairs of distinct elements of V, called edges. We refer to the set of edges as $E(G)$ and set of vertices as $V(G)$. The *cardinality* of the sets of vertices and edges are denoted as $|V|$ and $|E|$ respectively.

We define the alignment of a pair of graphs (shown in Fig. 1) in Definition 2.1 and the quality of the alignment in Definition 2.2.

Definition 2.1 (Alignment)**.** An *alignment* between two graphs G_1 and G_2 where $|V(G_1)| \leq |V(G_2)|$ is an injective function $f_{G_1,G_2} : V(G_1) \to V(G_2)$.

Note that we do not actually lose any generality by requiring that $|V(G_1)| \leq |V(G_2)|$, as we can simply add isolated, dummy vertices to $V(G_2)$. Where context allows, we will omit the subscripts.

Definition 2.2 (Value of an Alignment)**.** Given two graphs G_1 and G_2, the value of an alignment f, denoted val f, is defined as

$$\operatorname{val} f = \frac{1}{2} \sum_{u,v \in V(G_1)} \tau_f(u, v) \tag{1}$$

where $\tau_f : V(G_1) \times V(G_1) \to \mathbb{N}$ is given by

$$\tau_f(u, v) = \begin{cases} 1 \text{ if } uv \in E(G_1) \text{ and } f(u)f(v) \in E(G_2) \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

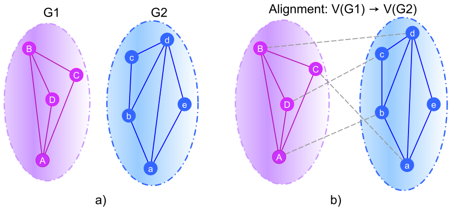


Figure 1: An example showing a) two graphs and b) an alignment of those graphs.

We note that this is possibly the simplest (sensible) valuation of an alignment and much more complex ones are possible. For example we can introduce a term that favours the alignment of particular pairs of vertices, or gives different weights for each aligned edge or vertex and of course non-integral and non-increasing (even non-monotone) valuations are possible.

We observe informally that it is a non-trivial valuation of the edges that seems to give rise to the complexity (or more precisely, valuations that are dependent on more than one vertex and its image under the valuation). For example, given a labelled graph, the valuation that simply adds 1 for each label that matches the label of its image makes the Network Alignment problem almost trivially polynomial. Similar valuations can be simply maximised.

We can now state the Network Alignment problem:

Network Alignment:
Instance: Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with $|V_1| \leq |V_2|$, positive integer b.
Question: Is there an alignment f with val $f \geq b$?

3 ALGORITHMIC IMPLEMENTATION

The Network Alignment with Memetic Algorithm (NAMA) prototype was built using Java 1.8. The Java FixedThreadPool method was used to provide parallel and distributed processing functionality. As the two input graphs are not modified, we represent them using a lightweight graph data structure with a list storing the vertex labels and a HashMap for the adjacencies.

The overall algorithm follows the typical memetic algorithm structure. Populations of individual solutions are generated then repeatedly mutated and crossbred with the "fittest" solutions being selected for mutation and crossover. Periodically the evolutionary cycle is interrupted by an iterative improvement phase where every individual is subjected to deterministic optimisation (in our case via a local search procedure). The details of the number of generations, the frequency of the deterministic improvement and the size and number of the populations are all parameters which can be tuned to improve performance, either from an optimality standpoint or from a resource-use perspective.

3.1 Preprocessing

Although the Network Alignment problem is formulated such that the two graphs G_1 and G_2 can have different numbers of vertices, it is more convenient algorithmically to assume that the two graphs have the same number of vertices. As noted earlier we may'pad' the smaller instance with degree zero vertices. Excepting rather unusual fitness functions, the vertices simply act as placeholders and do not affect the solution.

To enhance the speed of determining adjacency in the graphs when computing the fitness function, the algorithm memorizes the adjacency results for all pairs of vertices in each graph. If not done with care this process can introduce a large, if not crippling, space overhead. For graphs of the size dealt with in this paper, a naïve approach would require several gigabytes of memory – feasible, but not desirable. Increasing the graph size by an order of magnitude would increase the memory usage by two orders of magnitude quickly becoming infeasible. To strike a balance, we store only the instances where two vertices are adjacent, exploiting exception handling mechanisms to deal with the non-adjacent cases. This approach will encounter the same problems for highly connected graphs. In this case we must resort to either computing the adjacencies as needed, or using external memory techniques [1].

3.2 Individual Representation, Mutation, Recombination and Selection

The representation of the individual solutions, the mutation operators and the crossover operators are discussed below.

3.2.1 Individual Representation: As we ensure that the two graphs have the same number of vertices, the alignment is a bijective function from the vertices of one graph to the vertices of the other. This can easily be represented by a function from $\{0, \ldots, n-1\}$ to $\{0, \ldots, n-1\}$ – i.e. a permutation of the indices of the vertices. Thus an individual solution is represented in the algorithm as a permutation of the numbers $\{0, \ldots, n-1\}$, stored as a list with the indices of the list representing the domain of the alignment function. An example of the individual representation for two networks and corresponding mapping is shown in Fig. 2.

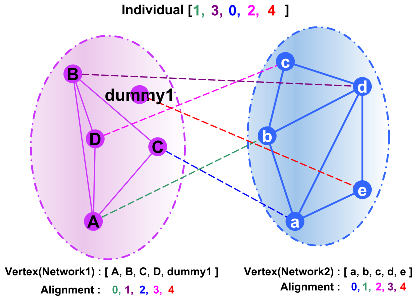

Individual [1, 3, 0, 2, 4]
B
dummy1
D
C
A
c
d
b
e
a
Vertex(Network1) : [A, B, C, D, dummy1]
Alignment : 0, 1, 2, 3, 4
Vertex(Network2) : [a, b, c, d, e]
Alignment : 0, 1, 2, 3, 4

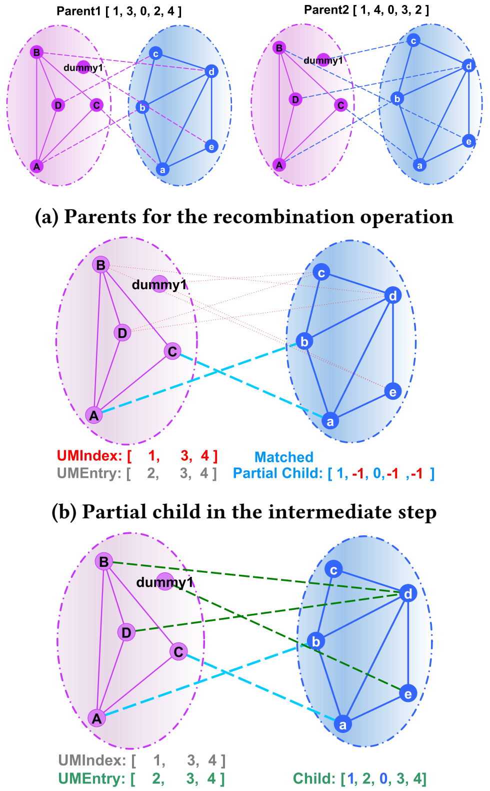
Figure 2: An example of the individual representation used in the algorithm. The value, m, in the individual at index i represents the mapping of i^{th} vertex of *Network 1* with the m^{th} vertex of *Network 2*.

3.2.2 Mutation Operator: To mutate the individuals we use a index shuffling mutation[1], which considers each element of the individual and, with a given probability, swaps it with a randomly chosen element of the individual. The probability of swapping an element is user defined, for this work, we used 0.05. Although simple, this operator helps prevent the ossification of substructures, where groups of vertices are stuck in structural local minima.

3.2.3 Recombination Operator: Here we use a partially matched crossover (PMX) which randomly selects a subsequence of the indices of the two parents, using the element at that index in the other parent to pick which element to swap the current element with. Goldberg and Lingle Jr. [7] give more detail. Fig. 3 shows an example of the recombination operation used in the algorithm. In the context of network alignment, this operator selects different substructures from the two parents to map into the child.

3.2.4 Selection: The individuals to be bred and mutated are selected by a tournament selection process (whereby the same individual can be selected multiple times). For a given k and t, k tournaments are conducted, each between t randomly selected individuals. The fittest of the t is selected as the "winner" of that tournament. Within this paper, we choose k to be the number of individuals and t to be 3.

[1] deap.tools.mutShuffleIndexes.



(a) Parents for the recombination operation

(b) Partial child in the intermediate step

(c) Child generated from the recombination operation

Figure 3: An example of the *recombination* operation used in the algorithm. (a) The parents taking part in the recombination process. (b) The step in the recombination process which keeps only the matched mapping between parents and tracks the indices where the mappings were mismatched. (c) The final mapping reconciliation in generating the child achieved by aligning the unmatched indices from the sorted list.

3.3 Local Search Optimisation

For individual optimization, we employ a simple local search procedure, similar to the 2-opt local search procedure [3]. Here, we repeatedly take one vertex and try to swap it with another vertex of the individual. To evaluate the swap-ability of the pair, we compute the fitness score for the vertex-swapped individual. If the new fitness score is improved compared to the previous fitness, we consider this swap. We repeat this process of improvement for all available vertices in the individual. We repeat the process until no improvement can be made. Eventually, we discover the best pair of vertices to swap through the local search process for which it provides the best fitness score.

3.4 Fitness Function

The fitness function employed in the algorithm is slightly more complex than that of Section 2 in that we include a term accounting for aligning designated pairs of vertices (for example we may wish align vertices representing the same protein in two PPI networks), along with scalar weights allowing adjustment of the relative influence of the vertex alignment and edge alignment. The fitness function $fitness(x)$ where x is an individual consisting of n elements is then

$$fitness(x) = \sum_{i=0}^{n-1} \sigma(i, x[i]) + \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} \tau(i, j, x[i], x[j]) \qquad (3)$$

with

$$\sigma(i, x[i]) = \begin{cases} c_1 \text{ if } i \text{ is ``the same'' as } x[i] \\ 0 \text{ otherwise} \end{cases} \qquad (4)$$

and

$$\tau(i, j, x[i], x[j]) = \begin{cases} c_2 \text{ if } ij \in E(G_1) \text{ and } x[i]x[j] \in E(G_2) \\ 0 \text{ otherwise.} \end{cases} \qquad (5)$$

We assume that vertices can be referred to by their indices. The constants c_1 and c_2 can be chosen to favour aligning edges, vertices or both. More generally these need not be constants, but could be replaced by more complex expressions. In the definition of σ, we deliberately leave the meaning of "the same" open to interpretation, as the meaning may change with the changing context of the input. For example we may consider two vertices "the same" if they have the same label (which does not necessarily denote uniqueness of the labelling), or if they represent proteins from the same family, etc. We note also that the fitness function implicitly assumes a *directed* graph where self loops are allowed. Given an undirected graph, each alignment of two edges will be counted twice. While this is not a significant problem – effectively it simply doubles the value pf c_2, it may be preferable for interpretative reasons to adjust the second sum in the fitness function to:

$$\sum_{i=0}^{n-1}\sum_{j=i+1}^{n-1} \tau(i, j, x[i], x[j]). \qquad (6)$$

3.5 Performance Testing

From a theoretical perspective, the $O(n^2)$ asymptotic running times of the implementation of the fitness function and the local search optimisation dominate the running time of a given generation (where n is the number of vertices in the each graph).

The empirical performance of the algorithm was examined using randomly generated graphs varying in the number of vertices and average degree. The graphs were generated using the Watts-Strogatz model for small world graphs [22]. This model was chosen as the application areas forming partial inspiration for this work involve social networks and protein-protein interaction networks, which exhibit small world properties.

The performance tests were performed on a Dell Precision T1700 workstation with an Intel Xeon E3-1271 v3 3.60GHz 64-bit 4-core hyperthreaded processor and 32Gb of DDR3 1866MHz RAM in four 8Gb sticks, running Ubuntu Linux version 16.04. To compile and run the program we use Java version 1.8.2.

From Fig. 4 we can see that the running time of the algorithm increases cubicly in terms of the number of vertices in the graph, with a pleasingly small leading coefficient. We note that the choice to parallelise the algorithm introduces a further tradeoff. For small graphs, the overhead incurred in parallelisation far exceeds the time spent on computing the solution. The break-even point (with this data) is at 623 vertices, though of course this will be easily shifted with relatively mild alteration in computing conditions. For comparison purposes, Fig. 5 shows the running times for non-parallel execution with small graphs.

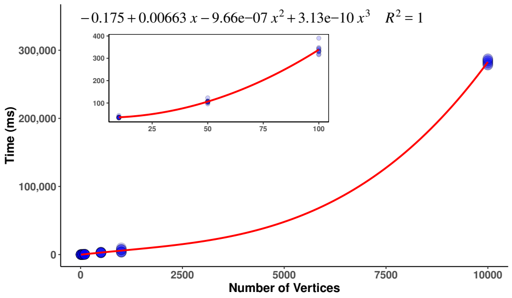


Figure 4: Running time of the memetic algorithm compared to the number of vertices in the graph.The fitted curve gives an approximation of runtime constants in the complexity.

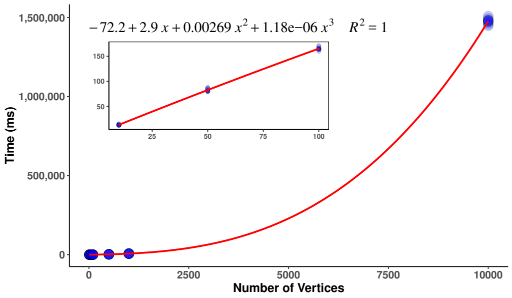


Figure 5: Running time of the memetic algorithm compared to the number of vertices in the graph using only a single process. The fitted curve gives an approximation of runtime constants in the complexity

To illustrate the effect of increasing degree on the performance of the algorithm, we plot the time taken compute alignments of 100-vertex graphs with increasing average degree, see Fig. 6. The effect of increasing edge density has a clear impact on performance, displaying a roughly linear relationship (as would be expected). At the lower end of the curve, the overhead of running the software influences the running time, flattening the curve. At the higher end, we see a more interesting effect, with a drop off in running time as the average degree approaches the number of vertices. We speculate that at high average degree, many of the vertices are essentially interchangeable, leading to fast convergence on a solution. Possibly, the near degeneracy of the optimal solution for high node-density instances may reduce the time required for the the periods of local search and the overall time required by the MA would then be shortened. To further illustrate the running time dependencies on size and density, we give in Fig. 7 a surface plot of the average running time of the algorithm on graphs of varying sizes and degrees. As this produces a large amount of data, we limit the size (and hence the maximum degree) of the graphs to 50 vertices. Nonetheless this gives a broad impression of the complexity landscape.

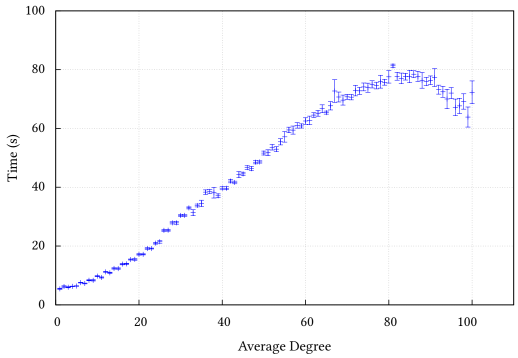


Figure 6: Memetic Algorithm Running Time against Degree. Graphs with 100 vertices, but varying average degree are used to compute the time. Each point represents the average (with standard deviation) of 10 runs at each average degree.

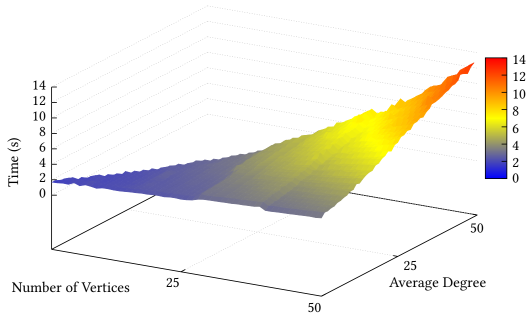


Figure 7: Memetic Algorithm Running Time against Number of Vertices and Average Degree. Each data point is the average of 10 runs of the algorithm, with settings as used for the plots in Figures 5 and 6.

4 EXPERIMENTAL OUTCOME

To explore the effectiveness of our approach, the algorithm was applied to pairs of PPI networks and pairs of non-PPI networks. This has the added benefit of helping determine the effectiveness outside a particular domain (which may or may not engender particular network structures). We also compare the performance of our proposed algorithm with other public domain, state-of-the-art network alignment tools. We will use several scores to compare those algorithms for alignment of different real-world networks.

4.1 Alignment Quality Measure

To measure the goodness of alignment between networks, structural and functional scoring methods are available. Functional measures consider the significance of biological function in the calculation. Those scores can be only computed for protein-protein interaction (PPI) networks where existing GO (gene ontology) terms are available. As we are dealing with the structural similarity mapping for all types of networks, computing the significance of biological functional score is out of the scope of current work.

Table 1: Characteristics of PPI networks considered in this study.

Species	Identifier	Vertices	Edges (Interactions)
thread worm	`cel`	5948	23,496
brown rat	`rno`	8002	32,527
mouse	`mmu`	9109	38,414

For the structural scores, the value is calculated for the topological similarities of two aligned networks. Following are some of the structural scores measuring global network alignment:

- `Conserved Interaction Under Alignment (CIUA)`: This score calculates the number of interactions in the graphs (*i.e.* edges) conserved under the alignment $f(E_1, E_2)$:
$$CIUA = |f(E_1) \cap E_2|. \tag{7}$$
- `Edge Correctness (EC)`: The edge correctness (EC) score measures the percentage of edges E_1 from the smaller network G_1 that are mapped to edges E_2 from the larger network G_2 under the given alignment f:
$$EC = \frac{|f(E_1) \cap E_2|}{|E_1|} \times 100\%. \tag{8}$$
- `Induced Conserved Structure (ICS)`: The ICS measures the percentage of edges matched in the larger network G_2 (*c.f.* Edge Correctness):
$$ICS = \frac{|f(E_1) \cap E_2|}{|E_2|} \times 100\%. \tag{9}$$
- `Symmetric Substructure Score` (S^3): Vikram *et al.* [17] proposed the S^3 score. This score measures the number of edge conservation between the aligned networks:
$$S^3 = \frac{|f(E_1) \cap E'_2|}{|E_1| + |E'_2| - |f(E_1) \cap E'_2|} \times 100\%. \tag{10}$$
- `Size of the Largest Connected Component (LCC)`: The number of vertices present in the largest connected component of the aligned network.

4.2 Datasets

To compare the performance of our proposed network alignments we have used protein-protein-interaction (PPI) networks for three species from the STRING database v8.3 and a pair of datasets related to mental health disorders.

4.2.1 PPI Networks from the STRING database. We used PPI networks from the STRING database v8.3 [20] for three species: **thread worm**, **brown rat** and **mouse**; respectively referred as *C.elegans* (`cel`), *R.norvegicus* (`rno`), *M.musculus* (`mmu`). El-Kebir *et al.* [6] considered only the interactions which were experimentally verified and computed the network alignment using their proposed algorithm. The number of vertices and edges present for those PPI networks are shown in Table 1.

4.2.2 Classification of Mental Health Disorders. In the field of psychopathology, the current dominant frameworks for diagnostic classification of mental health disorders are the International Classification of Diseases and Related Health Problems (ICD10) [16] and

Table 2: Network alignment score comparison for cel and mmu networks. Column with value '–' correspond to exceeded time/memory limits.

Algorithm	CIUA	EC	ICS	S^3	LCC
Natalie	–	–	–	–	–
HubAlign	856	14.907	21.193	9.591	6.765
NETAL	2369	41.257	56.715	31.378	43.097
NAMA	**2437**	**56.846**	**61.432**	**41.894**	**39.097**

Table 3: Network alignment score comparison for cel and rno networks.

Algorithm	CIUA	EC	ICS	S^3	LCC
Natalie	542	47.008	21.828	17.518	20.649
HubAlign	805	69.818	38.665	33.128	**42.117**
NETAL	805	69.818	76.521	57.500	21.391
NAMA	**883**	**76.583**	**76.583**	**62.052**	20.050

the Diagnostic and Statistical Manual of Mental Disorders (DSM-IV). Even though both manuals classify similar mental and behavioural disorders, research shows that there are striking difference in concordance and prevalence of disorders depending on which manual (or manual-based instrument) is used. Tio *et al.* [21] exported those manuals into network format where each of the symptoms were represented as a vertex and an edge existed if two vertices in the network co-occurred for at least one disease (from a set of 148 disorders). The resultant graph for DSM-IV consists of 439 vertices (symptoms) and 2626 edges. On the other hand, ICD-10 network consists of 588 vertices and 6169 edges.

4.3 Results

We executed the proposed Network Alignment with Memetic algorithm (NAMA) on these networks. For each set of experiments, we executed the memetic algorithm for 2000 iterations with a population size of 10 and the local search improvement was applied at every 100th iteration (parameter values were tuned by the performance testing on small world graphs [22] in Sect. 3.5). We used three state-of-the-art network alignment algorithms (Natalie [6][2], HubAlign [10][3], and NETAL[4] [14]) to compare the performance of our proposed method. All of the algorithms were executed on same machine with the same computing resources (as described in Sect. 3.5).

4.3.1 Network Alignment results on PPI Networks. Tables. 2–4 compare the different scores achieved for the alignment of the PPI networks. For the alignment of cel with mmu, the Natalie algorithm did not finish the execution with allocated resources, hence we are unable to produce any alignment scores for it (Table 2). However, our proposed NAMA outperformed all other algorithms for each of the five scores. Next in Table 3, HubAlign produced the best score for LCC, however, NAMA has beaten all algorithms in the remaining four scores for the alignment of cel and rno (Table 2). Finally, for the alignment task of mmu and rno, the outcomes are more mixed. Here NETAL has shown best performance for ICS and S^3, HubAlign for LCC, and NAMA has exhibited the best performance for the CIUA and EC scores (Table 4).

Table 4: Network alignment score comparison for mmu and rno networks.

Algorithm	CIUA	EC	ICS	S^3	LCC
Natalie	417	36.167	36.167	22.075	14.77
HubAlign	727	63.053	56.182	42.267	**22.487**
NETAL	765	66.349	**85.189**	**59.487**	13.205
NAMA	**822**	**71.292**	71.292	55.391	13.582

In summary, NAMA outperformed all of the algorithms for the EC and CIUA alignment scores and performed comparatively when using other scores against three state-of-the-art algorithms when aligning PPI networks.

Table 5: The network alignment score comparison for DSM-IV and ICD-10 networks.

Algorithm	CIUA	EC	ICS	S^3	LCC
Natalie	1768	67.326	40.338	33.734	53.522
HubAlign	1632	62.148	32.588	27.191	56.068
NETAL	1943	73.991	**54.702**	45.880	49.920
NAMA	**2464**	**93.831**	49.054	**47.522**	**59.240**

4.3.2 Network Alignment results on the Classification of Mental Disorders. The DSM-IV graph has 439 vertices and 2626 edges. While the ICD-10 has 588 vertices and 6169 edges. The network alignment performance of the algorithms on the classification of mental disorders network for five scores are shown in Table 5. Among those edges, the alignment by our algorithm conserves 2464 edges (Conserved Interaction Under Alignment Score). The Edge Correctness (EC) Score for the alignment is 93.83%. These measures clearly advocate for the strong topological similarity preservation power of our proposed network alignment method (NAMA). On the other hand, NETAL achieved best performance only for the ICS score for the alignment of DSM-IV with ICD-10. Hence, the proposed method, NAMA, clearly outperformed all other approaches for four scores (CIUA, EC, S^3 and LCC) when comparing with Natalie, HubAlign and NETAL.

5 DISCUSSION ON THE CLASSIFICATION OF MENTAL DISORDERS

Now, we shift our analysis to another dimension of the DSM-IV network. The giant connected component of the DSM-IV graph (the DSM-IV Core network has 156 vertices and 1441 edges) has been previously used in the analysis of community detection [9]. We used the same network of the DSM-IV Core to align with the ICD-10 network using our proposed network alignment method. The intuitive idea of mapping the core with the whole network is to see how the DSM-IV core aligns with the ICD-10 network

[2] Natalie : https://github.com/ls-cwi/natalie

[3] HubAlign : http://ttic.uchicago.edu/~hashemifar/software/HubAlign.zip

[4] NETAL : http://ttic.uchicago.edu/~hashemifar/software/NETAL.zip

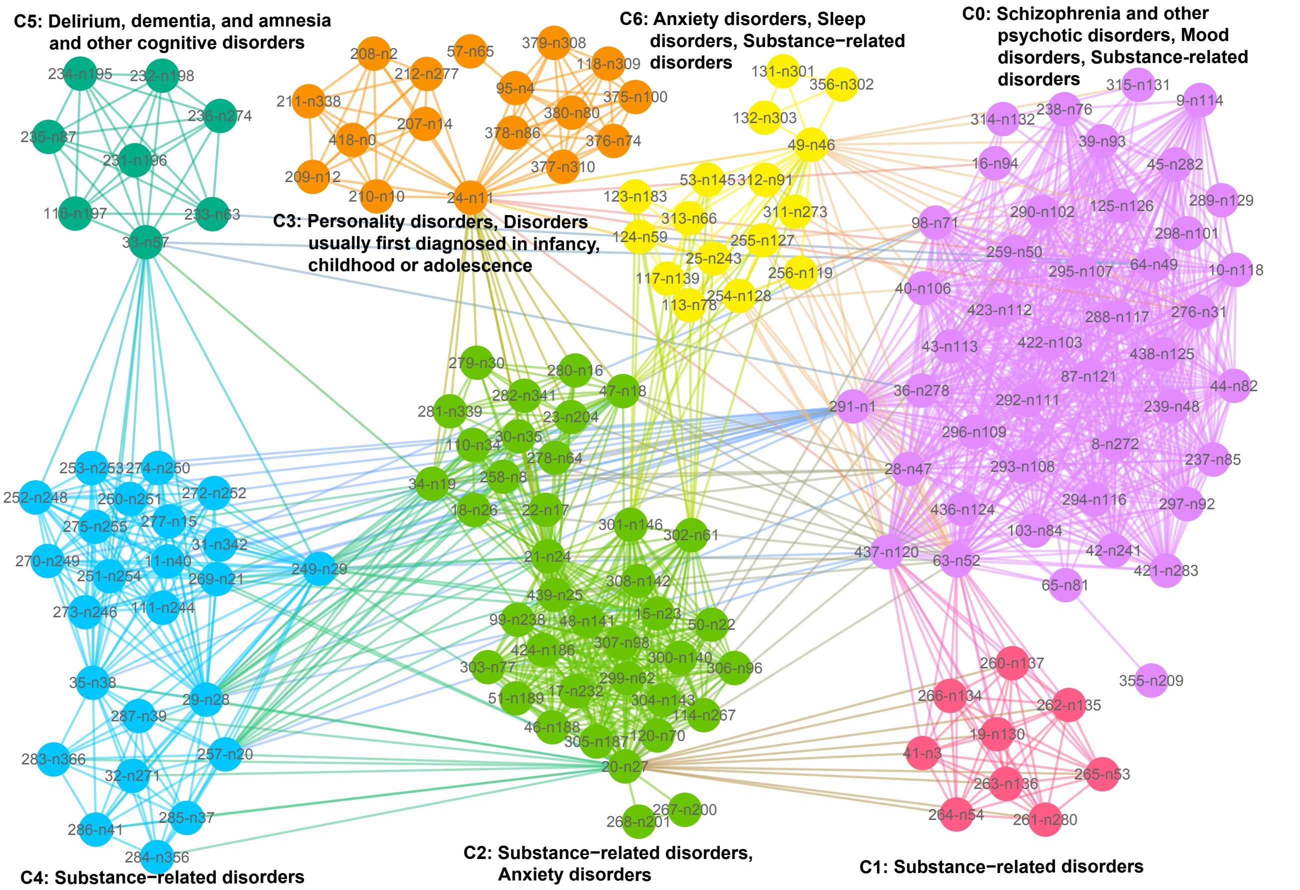


Figure 8: The *largest connected common subgraph* from the alignment of nodes between the Core of DSM-IV and ICD10. Node label represents the mapping of DSM-IV’s label connected by dash symbol “-” to the node label of ICD-10. The node colors are assigned based on the modularity class score computed from Gephi software.

and whether any prevalent topological characteristics exist. The numerical results of the alignment of the networks for the core of DSM-IV and the ICD-10 is shown in Table 6 and it is clear that NAMA able to find good alignment of DSM-IV Core with ICD-10 with an EC of 90.71%.

Table 6: The network alignment score of the proposed NAMA for the network alignment of the `Core of DSM-IV` and `ICD-10` networks.

Networks	CIUA	EC	ICS	S^3	LCC
Core-DSM-IV, ICD10	1768	90.713	53.722	50.922	63.254

Figure 8 shows the *largest connected common subgraph* from the alignment of networks achieved for the core of DSM-IV and ICD10 using the NAMA. The node labels from DSM-IV are numeric and ICD-10 starts with letter ‘n’. The labels in figures are constructed form the label in Core of DSM-IV to label of ICD-10 connected by dash (‘-’) symbol. The node coloring of the aligned network is achieved by applying modular communities (modularity score of 0.568) implemented by [2] in the Gephi graph visualization program. The community labels are constructed from the prevailing disorder classes for the group in core of the DSM-IV network.

5.1 Substance-related disorders

In Figure 8, there are some symptoms worth discussing for the different community memberships. For instance, the *substance-related disorders* are well separated into five communities (**C0**, **C1**, **C2**, **C4** and **C6**). Among them, the largest community (**C0**) in the network has nodes form other two disorder classes (*Schizophrenia and other psychotic disorders* and *Mood disorders*) and acting as a hub to other classes of disorders.

The community **C1** (*substance-related disorders*) has two articulation points (nodes `63-n52` and `437-n120`) with the largest community **C0**. The first node `63-n52` combines `63`:“insomnia / difficulty falling or staying asleep” with `n52`:“Difficulty in concentration / Difficulty in concentrating / Diminished ability to concentrate / Difficulty in concentrating, because of worrying or anxiety”. The node `437-n120` is interpreted as the combination of `437`:“psychomotor agitation / restlessness” and `n120`:“Restless sleep”, both related to sleeping difficulty. So we can infer that substance-related disorders may cause insomnia or restlessness due to anxiety which could also co-appear with the other disorders in the group **C0**.

Another substance-related disorder group **C4** has two nodes (291-n1 and 437-n120) connecting with the largest community **C0**. The first node is a combination of 291:"psychomotor retardation" and n1:"Decreased level of consciousness (sopor, coma) / Decreased levels of consciousness (sopor, coma)", which correlate well with the slowing-down of thought and a decreasing of physical movements in an individual. The other node consists of 437:"psychomotor agitation" and n120:"Restless sleep", where both symptoms could be the reactions to severe stress [16].

Interestingly, node 20-n27 is also one of the node which connects the community **C1** with **C2** ("Substance-related disorders, Anxiety disorders") and also has many connections with another substance-related community **C4**. Here, the symptom n27:"Insomnia / Difficulty getting to sleep because of worrying / Difficulty in falling or staying asleep" is likely to be associated with 20:"tachycardia / accelerated heart rate", which is also supported by literature [19].

Form these analyses, we found that the mapping of nodes from the Core of DSM-IV to the ICD-10 using the proposed network alignment algorithm, NAMA, not only found some interesting structural similarities but was also supported by the semantics of the networks.

6 CONCLUSION

We proposed a memetic algorithm-based network alignment algorithm using the structural similarity of the networks. The proposed algorithm has been tested on PPI networks to compare against three well-known network alignment tools with five scores. The empirical outcome found the NAMA to be at least comparable to those state-of-the-art algorithms with three of the scores and better for two of the scores (EC and CIUA). To compare the performance of our approach with other networks, we aligned DSM-IV and ICD-10 networks and the proposed approach outperformed all state-of-the-art algorithms for four scores out of five. These comparisons supported NAMA as the superior algorithm in terms of utilising the structural similarities in networks for aligning nodes.

Further analysis on alignment of the core of DSM-IV and ICD-10 reveals many interesting similarities between the aligned networks. One thing to note, we haven't used the symptoms and disorder information to map among DSM-IV and ICD-10 networks, but only used them while interpreting the results. However, we found those relationships are well supported by literature and provided meaningful relations between types of disorders and associated symptoms. This work of network alignment by memetic algorithm opens the avenue to cross-check the classification of disorders and their associated symptoms form DSM-IV to ICD-10.

ACKNOWLEDGMENTS

M.N.H. thanks Pia Tio from Department of Psychological Methods, University of Amsterdam, in relation to the graphs for DSM-IV and ICD-10 networks. P.M. acknowledges previous support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

REFERENCES

[1] Ahmed Shamsul Arefin, Mario Inostroza-Ponta, Luke Mathieson, Regina Berretta, and Pablo Moscato. 2011. Clustering Nodes in Large-Scale Biological Networks Using External Memory Algorithms. In *Algorithms and Architectures for Parallel Processing*, Yang Xiang, Alfredo Cuzzocrea, Michael Hobbs, and Wanlei Zhou (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 375–386.

[2] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[3] G. A. Croes. 1958. A Method for Solving Traveling Salesman Problems. *Operations Research* 6 (1958), 791–812.

[4] Natalie Jane de Vries, Jamie Carlson, and Pablo Moscato. 2014. A Data-Driven Approach to Reverse Engineering Customer Engagement Models: Towards Functional Constructs. *PLOS ONE* 9, 7 (07 2014), 1–19.

[5] W. E. Djeddi, S. B. Yahia, and E. M. Nguifo. 2018. A Novel Computational Approach for Global Alignment for Multiple Biological Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15, 6 (Nov 2018), 2060–2066.

[6] Mohammed El-Kebir, Jaap Heringa, and W. Gunnar Klau. 2015. Natalie 2.0: Sparse global network alignment as a special case of quadratic assignment. *Algorithms* 8, 4 (2015), 1035–1051.

[7] David E. Goldberg and Robert Lingle, Jr. 1985. Alleles Loci and the Traveling Salesman Problem. In *Proceedings of the 1st International Conference on Genetic Algorithms*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 154–159.

[8] Pietro Hiram Guzzi and Tijana Milenković. 2017. Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. *Briefings in Bioinformatics* 19, 3 (01 2017), 472–481.

[9] Mohammad Nazmul Haque and Pablo Moscato. 2018. The cohesion-based communities of symptoms of the largest component of the DSM-IV network. Retrieved the PrePrint from https://psyarxiv.com/spd8k.

[10] Somaye Hashemifar and Jinbo Xu. 2014. HubAlign: an accurate and efficient method for global alignment of protein-protein interaction networks. *Bioinformatics* 30, 17 (2014), i438–i444.

[11] Gunnar W. Klau. 2009. A new graph-based method for pairwise global network alignment. *BMC Bioinformatics* 10, Suppl 1 (2009), S59.

[12] Luke Mathieson, Natalie Jane de Vries, and Pablo Moscato. 2019. Using Network Alignment to Identify Conserved Consumer Behaviour Modeling Constructs. In *Business and Consumer Analytics: New Ideas*, Pablo Moscato and Natalie Jane de Vries (Eds.). Springer International Publishing, Cham, Switzerland, Chapter 12. https://www.springer.com/us/book/9783030062217

[13] Pablo Moscato. 1989. *On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms.* Technical Report 826. California Institute of Technology, Pasadena, California, USA.

[14] Behnam Neyshabur, Ahmadreza Khadem, Somaye Hashemifar, and Seyed Shahriar Arab. 2013. NETAL: a new graph-based method for global alignment of protein-protein interaction networks. *Bioinformatics* 29, 13 (2013), 1654–1662.

[15] Hiroyuki Ogata, Wataru Fujibuchi, Susumu Goto, and Minoru Kanehisa. 2000. A heuristic graph comparison algorithm and its application to detect functionally related enzyme clusters. *Nucleic acids research* 28, 20 (2000), 4021–4028.

[16] World Health Organization. 1993. *The ICD-10 classification of mental and behavioural disorders: diagnostic criteria for research.* Geneva: World Health Organization.

[17] Vikram Saraph and Tijana Milenković. 2014. MAGNA: Maximizing Accuracy in Global Network Alignment. *Bioinformatics* 30, 20 (2014), 2931–2940.

[18] Tie Shen, Zhengdong Zhang, Zhen Chen, Dagang Gu, Shen Liang, Yang Xu, Ruiyuan Li, Yimin Wei, Zhijie Liu, Yin Yi, et al. 2018. A genome-scale metabolic network alignment method within a hypergraph-based framework using a rotational tensor-vector product. *Scientific reports* 8, 1 (2018), 16376.

[19] Kai Spiegelhalder, Cathy Scholtes, and Dieter Riemann. 2010. The association between insomnia and cardiovascular diseases. *Nature and science of sleep* 2 (2010), 71.

[20] Damian Szklarczyk, Andrea Franceschini, Stefan Wyder, Kristoffer Forslund, Davide Heller, Jaime Huerta-Cepas, Milan Simonovic, Alexander Roth, Alberto Santos, Kalliopi P Tsafou, et al. 2014. STRING v10: protein–protein interaction networks, integrated over the tree of life. *Nucleic acids research* 43, D1 (2014), D447–D452.

[21] Pia Tio, Sacha Epskamp, Arjen Noordhof, and Denny Borsboom. 2016. Mapping the manuals of madness: Comparing the ICD-10 and DSM-IV-TR using a network approach. *International Journal of Methods in Psychiatric Research* 25, 4 (2016), 267–276.

[22] D. J. Watts and S. H. Strogatz. 1997. Collective dynamics of 'small-world' networks. *Nature* 393 (1997), 440–442.

[23] Jialiang Yang, Jun Li, Stefan Grünewald, and Xiu-Feng Wan. 2013. BinAligner: a heuristic method to align biological networks. In *BMC bioinformatics*, Vol. 14. BioMed Central, S8.