Compression of Biological Networks using a Genetic Algorithm with Localized Merge

Sheridan Houghten†, Angelo Romualdo†, Tyler K. Collins†, Joseph Alexander Brown*
†Department of Computer Science, Brock University, Ontario, Canada
*Artifical Intelligence in Games Development Lab, Innopolis University, Innopolis, Russia
shoughten@brocku.ca, ar14rk@brocku.ca, tk11br@brocku.ca, j.brown@innopolis.ru

***Abstract*—Network graphs appear in a number of important biological data problems, recording information relating to protein-protein interactions, gene regulation, transcription regulation and much more. These graphs are of such a significant size that they are impossible for a human to understand. Furthermore, the ever-expanding quantity of such information means that there are storage issues. To help address these issues, it is common for applications to compress nodes to form supernodes of similarly connected components. In previous graph compression studies it was noted that such supernodes often contain points from disparate parts of the graph. This study aims to correct this flaw by only allowing merges to occur within a local neighbourhood rather than across the entire graph. This restriction was found to not only produce more meaningful compressions, but also to reduce the overall distortion created by the compression for two out of three biological networks studied.**

I. Introduction

Many problems in bioinformatics require the analysis of huge amounts of data that is stored in the form of a *network* and indeed such data is of increasing interest for many bioinformatics researchers. Mathematically, networks are generally referred to as *graphs* and in this paper we use the two terms interchangeably.

There are many types of biological networks [3]. Protein-protein interaction networks record information on interactions between proteins and have a strong impact on various genetic diseases [17], [7]. Others include gene regulatory networks and transcription regulatory networks. The analysis of information in biological networks is difficult as they generally have a complex structure and are very large in size. Various statistical and topological information as to the nature of a given network may help to provide some insight [2].

One approach to help analyze information in large networks is compression, which helps to solve the associated storage issues. In addition, an appropriate compression may help to simplify the structure and thereby also simplify the analysis.

Graph compression is known to be a special instance of the Set Partitioning Problem, making the problem of compression against a known compression ratio NP-Complete [10]. Deterministic algorithms that will provably compress graphs to an optimum are known, but have an exponential run-time.

A. Graphs and Compression

A *graph* consists of a set of *nodes* and a set of *edges*, where each edge connects two nodes. A node is *incident* with an edge if it is one of the two nodes that the edge connects. Two nodes are *adjacent* if they are incident with the same edge; such nodes are *neighbours* of each other. The *degree* of a node is the number of edges with which it is incident. Information on the edges of a graph, i.e. its adjacency information, is most commonly represented by an *adjacency matrix* or an *adjacency list*. A *path* between two nodes is a sequence of edges that connect them; the *distance* between two nodes is the length of the shortest path between them.

Graphs may be *weighted*, in which case every edge has an associated weight, or *unweighted*, in which case all edges are assumed to have the same weight. Graphs may also be *directed*, in which case edges may only be traversed in one direction similar to a one-way street, or *undirected*, in which case all edges may be traversed in both directions.

The *size* of a graph is its number of edges while its *order* is its number of nodes. Both of these measures have an impact on the difficulty of using and analyzing the graph. Whether a graph is *dense* (has many edges relative to the number of nodes) or *sparse* (has relatively few edges) has an impact on both representation and which algorithms work well, and going further a myriad of other topological features and measures also have an effect. A *complex network* is one that is considered to have non-trivial topological features, and many biological networks fall under this category.

There are many different forms of graph compression schemes [4]. Many algorithms have been proposed for the problem of summarizing such networks in a logical manner (see, for example, [16] [20]). The graph compression algorithm Slashburn [11] permutes graphs into a form that can be more easily compressed. It relies on the recursive removal of "hub" nodes (highly-connected internal nodes) to be stored in their own stand-alone structure, as these are expensive to store in the adjacency matrix. The authors note that Slashburn is most effective on graphs that contain multiple hubs which are recursively connected to larger hubs.

In a hierarchical approach to compression (e.g. [15], [18]) the graph is compressed by merging sets of nodes into *supernodes* and edges into *superedges*. It is this approach which is used in the current study, by means of a genetic algorithm (GA) that identifies which nodes should be merged according to a fitness function that measures the distortion created by the merges. Only undirected and unweighted graphs are considered, although the methodology may be adapted for directed and/or weighted graphs.

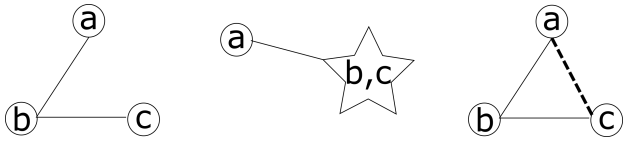

Fig. 1. Merge between nodes b and c produce the supernode b, c. After the resulting graph is decompressed there is a new edge between a and c, highlighted by dashed lines; this is a *fake edge*.

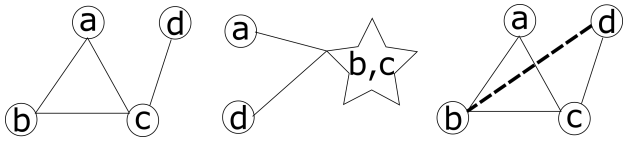

Fig. 2. Merge between nodes b and c produce the supernode b, c. After the resulting graph is decompressed there is a new edge between b and d, highlighted by dashed lines; this is a *fake edge*.

When graph G is compressed to produce G' and then subsequently decompressed to produce G'', a *lossless* compression method will have $G = G''$. In contrast, with a *lossy* compression method G'' is only an approximation of G.

Although lossy methods do not recover the original graph exactly, they may be able to obtain a better compression ratio than lossless methods. In addition, it has been noted that lossy methods might help to eliminate "noise" in graphs and thereby simplify the task of researchers analyzing them [14]. The implication for biological graphs is that lossy methods may filter out less significant or noisy data, thereby helping researchers to identify important components and relationships.

Consider the merge of nodes n_1 and n_2 in the original graph G to form node n_3 in compressed graph G'. Then in G': n_3 replaces n_1 and n_2; there is an edge (n_3, u) for every edge (n_1, u) in G; and there is an edge (n_3, v) for every edge (n_2, v) in G. If both n_1 and n_2 have edges incident with the same other node $u = v$ in G then when G' is decompressed to form G'' these same edges are in G'', just as they are in G. However when $u \neq v$ then there will be additional edges, (n_1, v) and (n_2, u), in G'' that were not in the original graph G. In addition, conceptually all nodes that are merged together are seen as a single unit and this forms a clique when decompressed into G'', so that if (n_1, n_2) was not in the original graph G then it is now an edge in G''. All additional edges that appear in G'' but not G are called "fake edges" [6]. Examples of fake edges are shown in Figures 1, 2 and 3. In each of these figures the original graph G is on the left, the compressed graph G' is in the centre, and the decompressed graph G'' is on the right.

In the case of a lossless compression, the number of fake edges is zero. In the current study the GA has a fitness function based on the number of fake edges, but it does not directly attempt to find a lossless compression.

B. Goals and Motivation

The current study is motivated by previous work in [6], in which compression of biological networks was examined with two different approaches using genetic algorithms (see Section II). In the first, a single-objective GA was used that merged nodes based on a similarity measure of the edges that those nodes had in common. In the second, a single-objective GA was used that tracked only the number of fake edges that would be created by merging nodes; these fake edges are as defined in Section I-A.

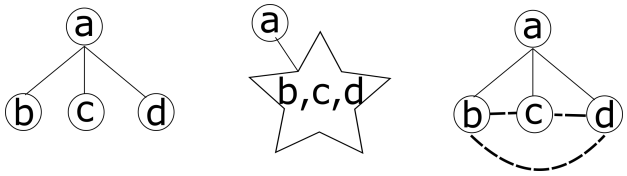

Fig. 3. Merges between nodes b, c, and d produce the supernode b, c, d. After the resulting graph is decompressed b, c and d form a clique; the edges in this clique are *fake edges* and are highlighted by dashed lines.

This second approach was flexible and was also extended to a multi-objective GA that allowed it to be used to establish a balance between the number of fake edges created while maximizing compression. However, it was noted that by concentrating solely on the number of fake edges created by a merge, sometimes nodes were selected for merge despite having very little in common with one another. For example, in an extreme case, two nodes could be a greater distance apart than any other pair of nodes in the graph, but if each were incident with only a single edge then merging those nodes would create a very small number of fake edges. This type of merge not only seems unnatural, but it also takes away from any meaning that one might read from the graph.

Essentially, nodes that are nearby are deemed to have a much stronger relationship than those that are far apart. Merging nodes that are far apart from one another in the original graph G creates a situation in which they are now seen as a unit in the compressed graph G', and indeed when G' is decompressed to form G'' these nodes share an edge. This implies a strong relationship that is not based on the original data.

This problematic situation is addressed in the current study by only allowing nodes to be merged if they are within a specified distance of one another. Although this significantly complicates calculations, taking this into account helps to ensure that the compression is more true to the original data.

C. Organization of Paper

The remainder of this paper is organized as follows. Section II describes the methodology used, including full details on the genetic algorithm. Section III describes the three biological networks examined in the current study. Section IV provides the results of the study, and Section V provides conclusions and describes possible future work.

II. Methodology

Genetic Algorithms (GAs) were developed primarily by Holland [9]. GAs are population based Evolutionary Algorithms in which a set of candidate solutions known as *chro-*

mosomes are examined for their ability to solve the problem via a *fitness evaluation*. Those with a higher fitness evaluation score are more likely to move on and be subjected to *variation operators* such as *crossover* and *mutation*. Crossover applies a binary operator taking two chromosomes and the notion of breeding them together to produce children. Mutation is a unary operation which makes a small change to a chromosome. After a number of *generations* in which the population of chromosomes is subjected to fitness evaluation and variation operators, the population will move towards higher fitness areas of the search space.

The current study uses a single-objective genetic algorithm for which, similarly to [6], fitness is based upon the notion of fake edges. However, two nodes are considered for merge only if they are within a given distance of each other. Furthermore, a more appropriate count of fake edges is used. See Section II-F for full details on fitness. The approach is applied to three biological networks, as specified in Section III.

A. Representation

Following the representation in [6], the graph is represented using an adjacency list and each individual node is represented by a unique integer value between 0 and $N_o - 1$, where the original graph has N_o nodes.

The *compression ratio* C of a graph measures the proportion by which the number of nodes in the original graph is reduced by compression. It is calculated as $C = 1 - \frac{N_c}{N_o}$, where N_o is as defined above and N_c is the number of nodes in the compressed graph. In fact the desired compression ratio is given as an input parameter. Therefore given N_o and C, the number of nodes in the compressed graph is calculated as $N_c = N_o * (1 - C)$.

The compressed graph is created from the original graph using a sequence of $N_o - N_c$ merges. Each individual merge is represented by two integer values: the *root*, which is the index of the first node to be merged, and the *offset*, which is a positive integer used to calculate the index i of the second node to be merged as $i =$ *(root + offset) mod* N_o. The use of *offset* is to ensure a node is not merged with itself, and hence this must be a positive value.

The chromosome consists of two one-dimensional arrays of length $N_o - N_c$, where corresponding indices in the arrays each store the root and offset for a single merge as specified above. For example, to compress a graph with 100 nodes by a compression ratio of 5% requires 5 merges, with the resulting graph having 95 nodes. In this case, the chromosome consists of two arrays each of length 5. See Figure 4 for an example chromosome. In the given example the following pairs of nodes will be merged: 22 with $(22 + 21)$ *mod* 100 = 43, 12 with $(12 + 94)$ *mod* 100 = 6, 24 with $(24 + 12)$ *mod* 100 = 36, 71 with $(71 + 19)$ *mod* 100 = 90, and 20 with $(20 + 7)$ *mod* 100 = 27.

B. Initial Population

Each chromosome in the population is a sequence of merges which must be *local* in that both nodes to be merged are within a specified distance of each other. For each of the $N_o - N_c$ merges, the first node is chosen randomly and its index is stored in the root. Next, a breadth-first search is performed from that node to find all other nodes within the specified distance; once these are found, one is chosen at random and the offset is set accordingly.

root	22	12	24	71	20
offset	21	94	12	19	7

Fig. 4. Example Chromosome

root	22	12	**52**	71	20
offset	21	94	**78**	19	7

Fig. 5. Result of Mutation

C. Selection

Tournament selection is used for selection of the two parents. For each parent, a separate tournament selection process occurs: k chromosomes are chosen at random from the population and evaluated, and then the best selected for reproduction as a parent. These parents are then subjected to crossover and mutation based on the settings to create two child chromosomes. This process repeats to create all chromosomes for the next generation.

D. Mutation

Mutation is applied simultaneously to both arrays of the chromosome. Single-point mutation is used, which has the effect of changing a single merge. The mutation point is a random value j between 1 and $N_o - N_c$, the size of the chromosome. The entry at index j of the root array is changed to a random value between 0 and $N_o - 1$.

In the previous study [6] the entry at the chosen mutation point in the offset array was simply changed to a random value between 1 and N_o. This could result in some merges of nodes that were far apart in the original graph, and is a situation avoided during mutation in the current study. As when generating the initial population (see Section II-B), once the first node has been chosen, the other node is chosen randomly from among all other nodes within the specified distance.

Figure 5 shows the result of single-point mutation in the 3rd entry of the chromosome from Figure 4. The result of this mutation is that the third merge is now between nodes 52 and $(52 + 78)$ *mod* 100 = 30, while all others stay the same.

E. Crossover

Crossover is applied to both arrays of the chromosome simultaneously. Two-point crossover is used, with the first crossover point being a random value between 1 and $N_o - N_c$, the size of the chromosome, and the second point being a random value between the first crossover point and $N_o - N_c$. Figure 7 shows the result of crossover on the pair of chromosomes from Figure 6, with the third and fourth entries both being exchanged between the two chromosomes.

root A	22	12	52	71	20
offset A	21	94	78	19	7

root B	76	22	82	15	12
offset B	12	57	84	25	79

Fig. 6. Before Crossover

root A	**22**	**12**	82	15	**20**
offset A	**21**	**94**	84	25	**7**

root B	76	22	**52**	**71**	12
offset B	12	57	**78**	**19**	79

Fig. 7. After Crossover, with original chromosome A shown in **bold**

It is to be noted that a sequence of merges may create a situation in which two nodes will be merged despite being at a distance higher than allowed, because they have distance less than or equal to the allowed threshold for some intermediate node. For example, suppose that node a is distance 10 from node b and distance 20 from node c, while node b is distance 10 from node c. If the maximum distance allowed is 10, then a and b are allowed to merge into supernode ab, which has distance 10 from c. Node ab may be subsequently merged with c to form supernode abc. Crossover may cause this sequence of merges to change so that ab is in one chromosome and ac is in another, despite the fact that a and c are at a greater distance than allowed. In the current study this situation is not addressed because it is seen as relatively rare, however future versions should possibly take this into consideration.

F. Fitness Function

The fitness function was initially developed in [19] and [6]. This fitness function counts fake edges, which are those which do not appear in the original graph G but which do appear in G'', where G'' is produced by first compressing G to produce G' and then decompressing G'. See Figures 1, 2 and 3 for examples.

The fitness, which should be minimized, is simply a count of the number of fake edges created as a result of compression followed by decompression. The number of fake edges overall is calculated by tallying the number of fake edges contributed by each individual merge.

As described in Section I-A, when nodes n_1 and n_2 in original graph G are merged to form node n_3 in compressed graph G' then for every edge (n_1, u) in G there is an edge (n_3, u) in G' and for every edge (n_2, v) in G there is an edge (n_3, v) in G'. This process creates fake edges (n_1, v) and (n_2, u) when G' is decompressed to form G'' unless they already existed in G. Also, G'' will contain the edge (n_1, n_2), which is also a fake edge if it did not already exist in G.

In the previous studies [19], [6], the number of fake edges contributed by merging nodes n_1 and n_2 simply considered the exclusive-or of the neighbours of n_1 and n_2. This did not always take into account fake edges from previous merges that may have created n_1 or n_2 as an intermediate step.

Therefore in the current study we use a more accurate count that essentially considers each set of nodes that have been merged together to have created a unit that, when decompressed, forms a clique. Within that clique, any edges that were not in the original graph are fake edges. Also, any node which has an edge to *any* of the nodes in this clique will now have an edge to *every* node in the clique, and all such extra edges are also fake edges.

III. Datasets

Compression is of significant importance for biological networks not only because they are usually very large in size, but also because compression may help in their analysis: when the GA chooses to merge nodes, it perceives those nodes as "similar" because they are relatively close to one another and create few fake edges when merged. This perception of similarity may indicate some possible biological relationship that should be further examined.

In this study we concentrate on the compression of biological networks, however the methods may be applied to any large graph. We use the same biological networks as in [6], all of which are briefly described below.

A. Yeast transcriptional regulatory network

The first dataset is the yeast transcriptional regulatory network [12]. No modifications were made to this dataset. This network contains 690 nodes and 1083 edges, and is the smallest of the networks examined.

B. E. coli

The second dataset is the gene regulatory network of Escherichia coli (E. coli) [12]. It was cleaned of all duplicate links (nodes indicating both activation and inhibition) and all unknown links, with 5 in total removed. After cleaning, the final graph consists of 1123 nodes and 2108 edges.

C. Protein-protein interactions

The final dataset, from Figeys, contains human protein-protein interactions [5]. No modifications were made to this dataset. The largest of the three networks examined, it contains 2239 nodes and 6452 edges.

IV. Results

This section examines the performance on the datasets specified in Section III. All test cases listed in Table I were applied to the problem of compressing each of the three given biological networks. The high distance test case is intended primarily as a "sanity check", as for the type of data being considered it should result in almost all nodes being allowed to merge with almost all others; for example, the Figeys dataset described in Section III-C is known to have a diameter of 10 in its main cluster [8], meaning that any two nodes in this cluster are at a distance of at most 10 from each other. The medium and low distance test cases each examine compression ratios of 10%, 20% and 25%; these are designed to provide some preliminary insight as to useful values for maximum distance across different compression ratios.

TABLE I
TEST CASES APPLIED TO ALL DATASETS

Type	Compression Ratio	Maximum Distance
High Distance	0.25	10
Medium Distance	0.10	5
	0.20	5
	0.25	5
Low Distance	0.10	3
	0.20	3
	0.25	3

TABLE II
EXPERIMENTAL PARAMETERS FOR THE GENETIC ALGORITHM

Parameter	Value
Mutation Rate	10%
Crossover Rate	90%
Generations	500
Population Size	100
Tournament Size	5
Number of Elites	1
Number of Runs	5

The experimental parameters for the GA are summarized in Table II. All of these were determined empirically, and were applied to all test cases for all datasets. The authors recognize the small number of runs performed for each test case and each dataset. This was necessary due to time constraints: the process of finding local nodes to merge is a bottleneck as it requires a breadth-first search, which is a slow process that is repeated a very large number of times. This process is a target for future improvements, as described in Section V. For the purpose of this study, it was deemed important to consider a range of compression ratios and distances.

A. Yeast transcriptional regulatory network

Table III lists the global best fitness obtained for each of the test cases for the yeast transcriptional regulatory network described in Section III-A, along with the average for the best fitness found in each of the runs. Recall that the fitness is a count of the total number of fake edges created.

This network prior to any compression is shown in Figure 8 and the same network after a compression of 10% using maximum distance 3 is shown in Figure 9. These visualizations were both created using GraphStream [1]. In comparing these visualizations it can be seen that the GA has performed many merges of nodes inside the small "satellite" clusters that are not part of the main cluster in the original graph: these consist of 2–5 nodes in Figure 8 and create either zero or very few fake edges when merged into a single node in Figure 9. Although less easy to see, other merges tend to occur in sections of the graph that consist of a "hub" node surrounded by "spokes". Again, this is sensible from the point of view of minimizing fake edges, however equally importantly they are also sensible from the point of view of identifying units that work closely together in a biological sense. Without requiring only local merges, the GA struggles to identify such locations and tends to have worse fitness. This can be seen when comparing 25% compression across high distance, medium distance and low distance: for this particular network, the fitness is actually better when the maximum distance between nodes is lower.

TABLE III
RESULTS FOR ALL TEST CASES – YEAST

Type	Cmp. Ratio	Merges	Max. Dist.	Glb. Best	Avg. Best
High	0.25	172	10	492	507
Med.	0.10	69	5	75	83
	0.20	138	5	275	306
	0.25	172	5	455	466
Low	0.10	69	3	60	61
	0.20	138	3	161	172
	0.25	172	3	262	267

TABLE IV
RESULTS FOR ALL TEST CASES – E.COLI

Type	Cmp. Ratio	Merges	Max. Dist.	Glb. Best	Avg. Best
High	0.25	280	10	867	887
Med.	0.10	112	5	173	182
	0.20	224	5	591	600
	0.25	280	5	882	891
Low	0.10	112	3	114	120
	0.20	224	3	412	425
	0.25	280	3	654	666

TABLE V
RESULTS FOR ALL TEST CASES – PROTEIN-PROTEIN INTERACTIONS

Type	Cmp. Ratio	Merges	Max. Dist.	Glb. Best	Avg. Best
High	0.25	559	10	3074	3137
Med.	0.10	223	5	683	689
	0.20	447	5	2092	2108
	0.25	559	5	3159	3246
Low	0.10	223	3	636	690
	0.20	447	3	2637	2720
	0.25	559	3	4295	4505

B. E. coli

Table IV lists the global best fitness obtained for each of the test cases for the E.coli network described in Section III-B, along with the average for the best fitness found in each of the runs. As can be seen in this table, there is a similar trend in that when the maximum distance is lower the fitness is better, however it is not as pronounced: while the fitness for maximum distance 3 is always better than the fitness for maximum distance 5 at the same compression ratio, the fitness for maximum distance 10 is almost identical to that for maximum distance 5 at 25% compression. The E.coli graph is similar in structure to the yeast graph, in that there is one main cluster with a number of satellite clusters; however, the satellites are larger on average, so that when merging their nodes more fake edges are created.

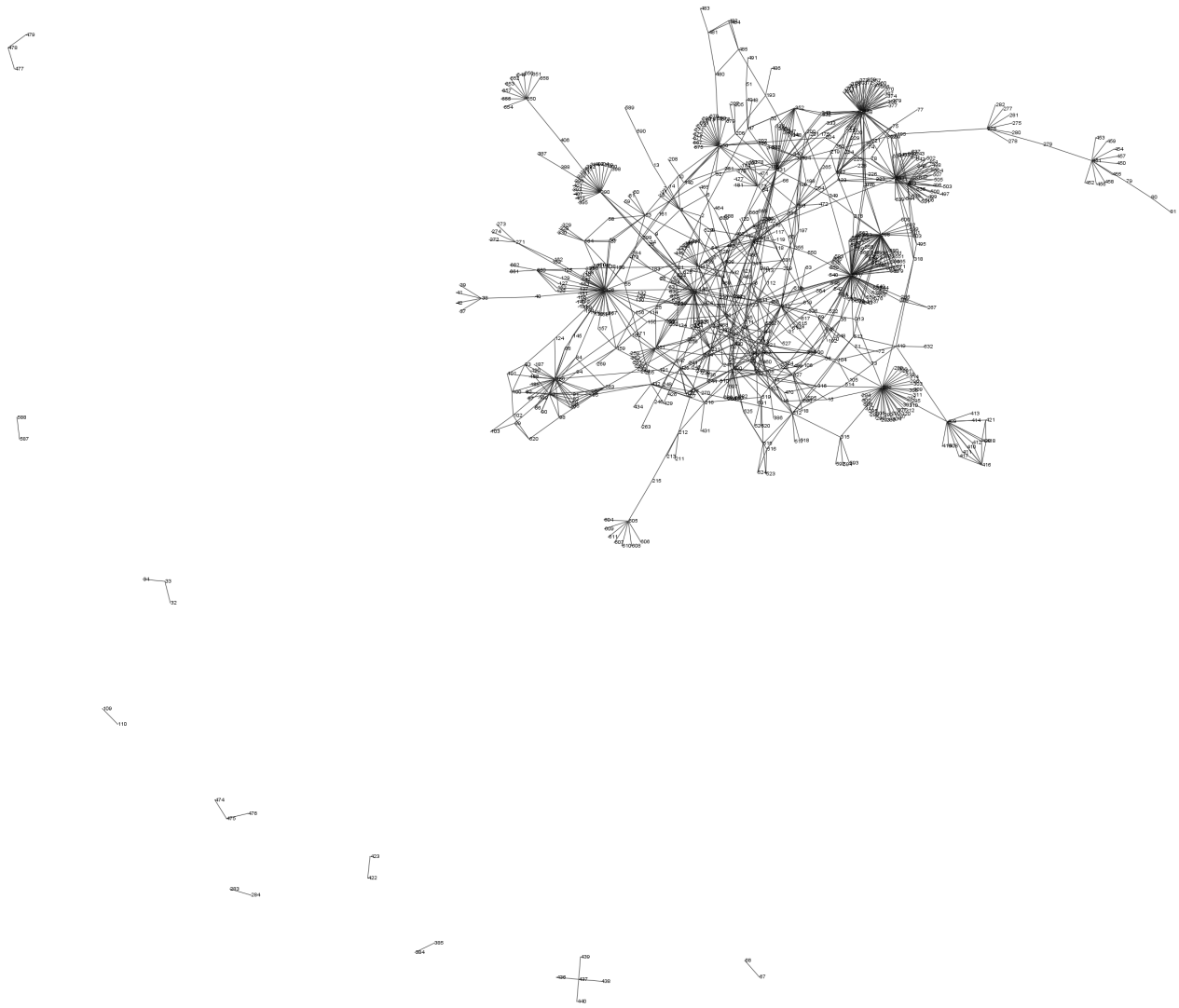

Fig. 8. Yeast transcriptional regulatory network prior to compression. Note the presence of one large main cluster along with several small "satellites".

C. Protein-protein interactions

Table V lists the global best fitness obtained for each of the test cases for the protein-protein interactions network described in Section III-C, along with the average for the best fitness found in each of the runs. In comparison to the other graphs studied, this is a very dense network with a large highly-connected main cluster and relatively few satellites. This table shows the effect this graph structure has on fitness: although at 10% compression the best fitness improves slightly from maximum distance 5 to maximum distance 3, for all other compression ratios the fitness is worse when the maximum distance is lower. At 10% compression the GA is able to obtain a number of low-cost merges from satellites and other sparse areas of the graph, however after this point it would create fewer fake edges by merging nodes that are further away, despite the fact these may be less sensible from a biological point of view (trying to extract "meaning" from the graph).

D. Comparison to Earlier Study

These results have improved on earlier work [6] for all three networks. Using a compression ratio of 25%, the previous study obtained a global best of 644 fake edges for yeast, 1149 for E.coli and 4217 for protein-protein interactions. Furthermore, as mentioned in Section II-F, the earlier study did not always account for fake edges from intermediate steps. This means that numbers from the earlier study would generally be slightly higher if using the current calculation.

V. Conclusion and Future Work

The results of this study demonstrate that by only allowing merges to occur between nodes within a specified distance of each other, more sensible merges are created. In particular, nodes that are nearby are much more likely to work closely together in the biological network, so that when the compression combines them into a unit, this unit is much more likely to be meaningful from a biological perspective.

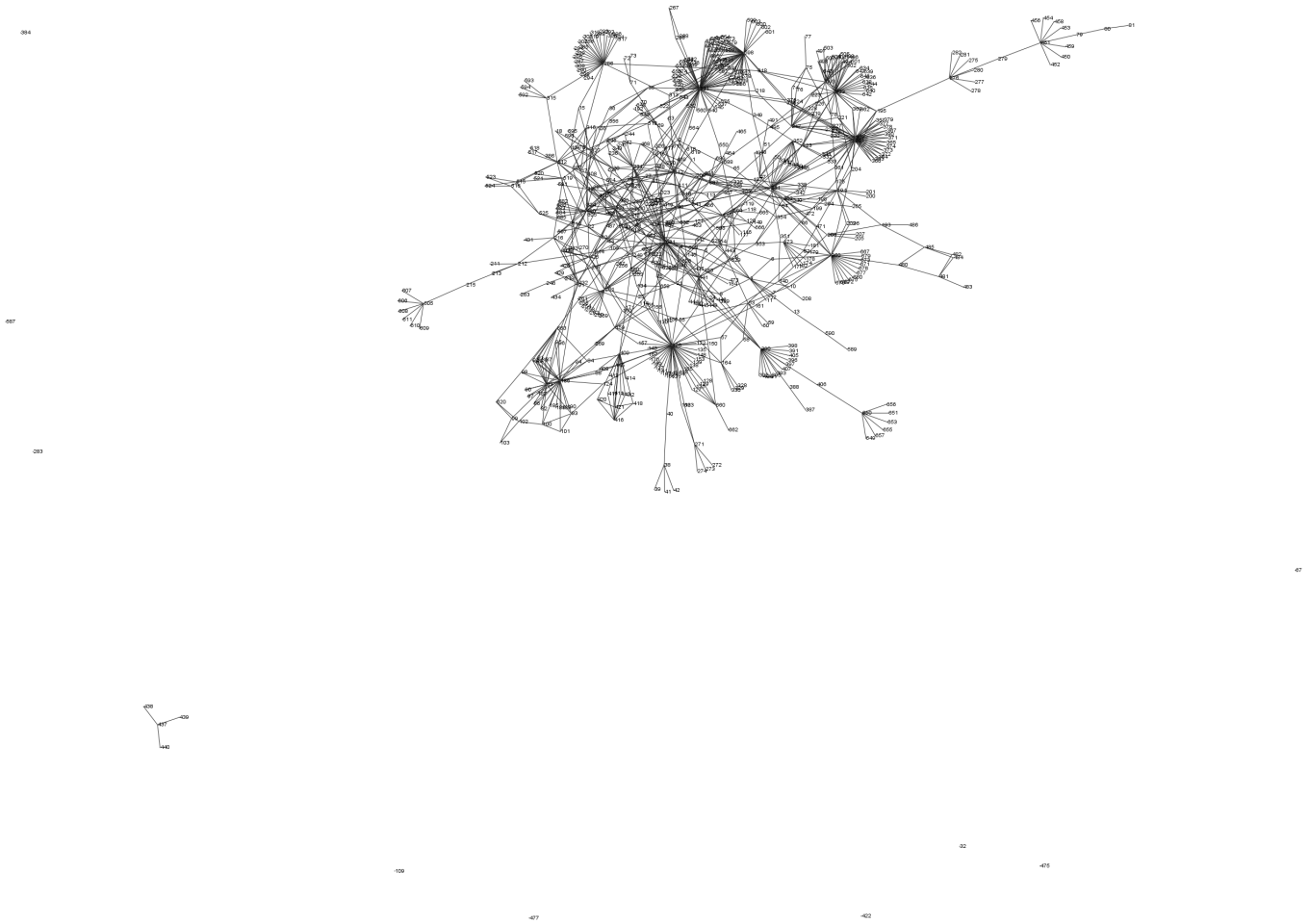

Fig. 9. Yeast transcriptional regulatory network after 10% compression with maximum distance 3. Only the "root" node is shown in a merged node. Note the concentration on merging the nodes in small connected components that are "satellites" and not in the main cluster.

In addition, such merges in general reduce the overall number of fake edges created during compression, with the exception of the highly-connected protein-protein interactions network. A smaller number of fake edges created by a compression can be seen as that compression creating less distortion of the original graph.

Appropriate values for the maximum distance parameter are key to the success of such a scheme. The current study considered values of 3 and 5 for maximum distance for a range of compression ratios, as well as a value of 10 for a single compression ratio. The concept of allowing only local merges should be further explored to determine the best values for maximum distance, noting that the best value will tend to differ from one graph to another. For example, it would be worthwhile to consider various metrics such as graph diameter (the greatest distance between any pair of vertices) to help select a value for maximum distance. The value may also depend on the level of compression that is desired.

The three biological networks analyzed in the current study were chosen primarily due to their use in earlier work [6]. However, in comparison to some other biological networks, they are still relatively small. For example, the datasets available from the Stanford Biomedical Network Dataset Collection [13] range in size from a few hundred nodes to an extreme of over a billion nodes, and also vary in terms of other topological features. As was seen in the current study, the structure of the graph significantly affects the success of the compression. Therefore, future work should include an examination of a wider range of biological networks. Additionally, it would be worthwhile to include support for directed and/or weighted graphs. Also, although the methodology has been usefully applied to three types of biological networks, it can also be applied to others with different features, including those not in the biological domain.

Recall that although one of the goals of compression is simply a reduction in size, another goal is to potentially use the compression to provide information about the data stored in the graph: if nodes are combined during a merge then what might one gain from thinking of them as a unit? With respect to biological networks, the compression could be inspected to determine if it is possible to interpret any information from the merges. As noted in Section I-A, the fact that the compression is lossy may actually help researchers to eliminate "noisy" data in the graph and thereby more easily find meaningful components and relationships. Comparison of this methodology to various graph clustering methods, specifically for the purpose of extracting meaningful information from the networks, would be a worthwhile endeavour.

It is useful to reflect upon the implications of requiring only local merges while trying to find a compression that minimizes the total number of fake edges. The first fitness measure described in [6] chooses to merge nodes based on a definition of similarity that includes consideration of whether the nodes have a neighbour in common. Requiring nodes to be within a given maximum distance of one another will tend to often select nodes with a neighbour in common as they are at most distance two from each other; however, it is a more general and flexible requirement as it also allows other nearby nodes to be selected for merge. It is worthwhile investigating modifying the requirement slightly, so that closer nodes are chosen with a higher probability than those that are further away but still within the specified maximum distance.

Another implication of attempting to minimize the number of fake edges is that there is a tendency for nodes to be selected if they have a relatively low degree in the original graph, as such nodes do not have many edges which could contribute to the creation of fake edges. In other words, the sparse areas of the graph are more likely to be selected for compression than the more dense areas, which is an idea employed directly by Slashburn [11].

Despite the benefits of the methodology as described above, there is a significant issue with respect to time requirements. In its current form, the methodology requires that a breadth-first search be performed each time that all local nodes need to be identified as possible merge candidates. This operation occurs many times during evolution and is very expensive. It should be noted that this operation is highly dependent on the maximum distance, as when maximum distance is smaller the breadth-first search to find all nodes within that range will consider a far smaller number of nodes before stopping. There are many options to explore that can identify such nodes in a more efficient manner, including, for example, randomized search or some level of precomputation.

Finally, it will be important to complete a greater number of runs, not only for the test cases used in the current study but also for other settings.

ACKNOWLEDGEMENTS

This research was funded in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

The authors would like to acknowledge the assistance of Avi Ma'ayan in providing information on the biological datasets and their sources.

REFERENCES

[1] GraphStream: A Dynamic Graph Library. http://graphstream-project.org/.

[2] Yassen Assenov, Fidel Ramírez, Sven-Eric Schelhorn, Thomas Lengauer, and Mario Albrecht. Computing topological parameters of biological networks. *Bioinformatics*, 24(2):282–284, 2007.

[3] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature reviews genetics*, 12(1):56, 2011.

[4] M. Besta and T. Hoefler. Survey and taxonomy of lossless graph compression and space-efficient graph representations. *arXiv preprint arXiv:1806.01799*, 2018.

[5] Daniel J B Clarke, Maxim V Kuleshov, Brian M Schilder, Denis Torre, Mary E Duffy, Alexandra B Keenan, Alexander Lachmann, Axel S Feldmann, Gregory W Gundersen, Moshe C Silverstein, et al. expression2kinases (x2k) web: linking expression signatures to upstream cell signaling networks. *Nucleic acids research*, 46(W1):W171–W179, 2018.

[6] T.K. Collins, A. Zakirov, J.A. Brown, and S. Houghten. Single-objective and multi-objective genetic algorithms for compression of biological networks. In *2017 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8, 2017.

[7] Ashkan Entezari Heravi, Koosha Tahmasebipour, and Sheridan Houghten. Evolutionary computation for disease gene association. In *2015 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8. IEEE, 2015.

[8] Human Protein: Figeys. http://konect.uni-koblenz.de/networks/maayan-figeys.

[9] J.H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, USA, 1992.

[10] Robin Lamarche-Perrin, Lionel Tabourier, and Fabien Tarissan. Information-theoretic Compression of Weighted Graphs. In *Poster session of the MSR-INRIA Join Center Workshop on Networks: Learning, Information and Complexity*, 2016.

[11] Yongsub Lim, U. Kang, and Christos Faloutsos. Slashburn: Graph compression and mining beyond caveman communities. *IEEE Trans. Knowl. Data Eng.*, 26(12):3077–3089, 2014.

[12] Avi Ma'ayan, Guillermo A. Cecchi, John Wagner, A. Ravi Rao, Ravi Iyengar, and Gustavo Stolovitzky. Ordered cyclic motifs contribute to dynamic stability in biological and engineered networks. *Proceedings of the National Academy of Sciences*, 105(49):19235–19240, 2008.

[13] Sagar Maheshwari Marinka Zitnik, Rok Sosič and Jure Leskovec. BioSNAP Datasets: Stanford biomedical network dataset collection. http://snap.stanford.edu/biodata, August 2018.

[14] H. Maserrat and J. Pei. Community preserving lossy compression of social networks. In *2012 IEEE 12th International Conference on Data Mining*, pages 509–518, Dec 2012.

[15] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. Graph Summarization with Bounded Error. In *SIGMOD*, pages 419–432, 2008.

[16] Qiang Qu, Feida Zhu, Xifeng Yan, Jiawei Han, Philip S. Yu, and Hongyan Li. Efficient Topological OLAP on Information Networks. In *DASFAA*, pages 389–403, 2011.

[17] Nahid Safari-Alighiarloo, Mohammad Taghizadeh, Mostafa Rezaei-Tavirani, Bahram Goliaei, and Ali Asghar Peyvandi. Protein-protein interaction networks (ppi) and complex diseases. *Gastroenterology and Hepatology from bed to bench*, 7(1):17, 2014.

[18] Hannu Toivonen, Fang Zhou, Aleksi Hartikainen, and Atte Hinkka. Compression of weighted graphs. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 965–973, 2011.

[19] A.N. Zakirov and J.A. Brown. NSGA-II for biological graph compression. *Advanced Studies in Biology*, 9(1):1–7, 2017.

[20] Feida Zhu, Zequn Zhang, and Qiang Qu. A direct mining approach to efficient constrained graph pattern discovery. In *SIGMOD*, pages 821–832, 2013.