Topology aware coloring of gene regulatory networks

Günhan Gülsoy , Bhavik Gandhi , Tamer Kahveci
Computer and Information Sciences and Engineering
University of Florida, Gainesville, FL 32611
{ggulsoy,bgandhi,tamer}@cise.ufl.edu

May 20, 2011

Abstract

We consider the problem of finding a subnetwork in a given biological network (i.e., target network) that is the most similar to a given small query network. We aim to find the optimal solution (i.e., the subnetwork with the largest alignment score) with a provable confidence bound. There is no known polynomial time solution to this problem in the literature. Alon *et al.* has developed a state of the art coloring method that reduces the cost of this problem. This method randomly colors the target network prior to alignment for many iterations until a user supplied confidence is reached. Here we develop a novel coloring method, named *k-hop coloring* (k is a positive integer), that achieves a provable confidence value in a small number of iterations without sacrificing the optimality. Our method considers the color assignments already made in the neighborhood of each target network node while assigning a color to a node. This way, it preemptively avoids many color assignments that are guaranteed to fail to produce the optimal alignment. We demonstrate both theoretically and experimentally that our coloring method outperforms that of Alon *et al.* which is also used by a number network alignment methods including QPath and QNet by a factor of three without reducing the confidence in the optimality of the result. Our experiments also suggest that the resulting alignment method is capable of identifying functionally enriched regions in the target network successfully.

1 Introduction

Biological processes are often governed through interactions of many molecules. Depending on the types of interactions, collections of such molecules are explained using metabolic [12], gene regulatory [17] or protein interaction [14] networks. We will use the term *biological network* to describe them in this paper. Understanding biological networks has been one of the main goals of biological sciences as they contain key information regarding how organisms work. To achieve this goal, biological networks are analyzed in a number of ways. One of them, comparison based analysis, identifies similar parts of two biological networks by aligning them. Such analysis has been successfully used in many applications such as finding functional annotations [9], identifying drug targets [24], reconstructing metabolic networks from newly sequenced genome [12], and building phylogenetic trees [8].

Existing literature often considers network alignment as a graph or subgraph isomorphism problem. In order to do this, they consider each molecule (i.e., gene or protein) as a node and each interaction as an edge that connects the nodes corresponding to that interaction. These methods measure the quality of an alignment in terms of a scoring function. We explain the concept of network alignment and scoring function later in this section. The graph and subgraph isomorphism problems, however, do not have known polynomial time solutions [13, 10]. We can classify existing solutions under two categories. The first one develops heuristic solutions with polynomial time complexities [4, 5, 7, 18]. Although these methods often work in practical time, it is impossible to tell how the result compares to the optimal one (i.e., the alignment with the highest score). The second one (which is the focus of this paper) searches for the optimal alignment [11, 16, 20, 22]. In order to simplify the problem, existing methods impose restrictions on network topologies.

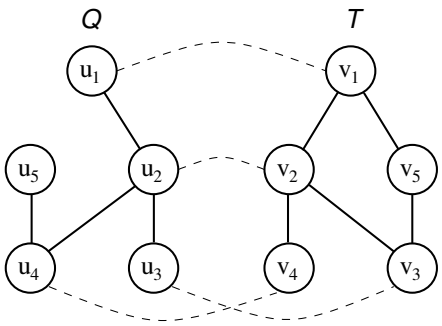

Figure 1: An alignment of the networks Q and T. $\{u_1, 22_2, ..., u_5\}$ and $\{v_1, v_2, ..., v_5\}$ are the set of nodes in networks Q and T respectively. The solid lines represent the interactions between pairs of molecules. The dashed lines connect the aligned nodes to each other. Note that unaligned nodes (e.g., v_5) can exist in any network. They correspond to insertions or deletions in the alignment.

Also, in order to reduce the running time, they ensure optimality with a provable confidence bound, which is less than 100%. For example, some use randomized techniques such as color coding that returns a result which is optimal with a predefined confidence [11, 22]. We will elaborate on existing methods, particularly on color coding methods, in Section 2. Formal definition of the problem we consider in this paper is as follows:

Problem Definition: Assume that we are given two biological networks denoted with $Q = (V_Q, E_Q)$ and $T = (V_T, E_T)$, where V_Q and V_T are the set of nodes (i.e., genes or proteins) and E_Q, E_T are the set of edges (interactions) respectively. An *alignment* of Q and T is a bijection between the nodes of Q and T, where a mapping between two nodes implies that the edges between them (if they exist) align with each other. Figure 1 shows the alignment between two hypothetical networks. Dashed lines in the figure show the node mappings between the nodes of the two networks. An alignment allows insertions and deletions of nodes or edges by matching them to $NULL$. The score of an alignment of Q and T is the sum of following three terms:

1. Pairwise similarity scores of the matching nodes
2. Weights of the matching edges,
3. Insertion and deletion penalties.

An alignment is optimal if it has the largest score among all possible alignments. We say that we have a confidence of ϵ in the result if the alignment is optimal with probability ϵ. *Our aim is to develop an algorithm to find the optimal alignment between Q and T with a given confidence ϵ in practical time.*

Contributions: In this paper, we develop a novel randomized biological network alignment method which can align two networks optimally with a provable confidence bound. More specifically, our main contribution here is a smart color coding method. Unlike the state of the art coloring method of Alon *et al.* [1], our method considers the color assignments already made in the neighborhood of each node while assigning a color to that node. This way it preemptively avoids many nonpromising color assignments. We formulate the confidence calculation for our coloring method. We evaluate the impact of our coloring method when the target network has an arbitrary topology and the query network is a tree or a bounded treewidth graph. We demonstrate both theoretically and experimentally that our coloring method outperforms that of Alon *et al.* [1] which is also used in QPath [22] and QNet [11] by a factor of three without reducing the confidence in the optimality of the result.

We use a number of symbols to describe our method. Table 1 lists the most frequently used symbols throughout this paper.

Table 1: Frequently used symbols in this paper

Q,T	Query and the target networks to be aligned
V_Q,V_T	The set of nodes in query and target networks
E_Q,E_T	The set of interactions in query and target networks
u_i,v_j	Nodes of networks Q and T
m,n	Number of nodes in query and target networks
$Sim(u_i, v_j)$	Similarity score between the nodes u_i and v_j
δ	Insertion or deletion penalty
ϵ	Confidence in the optimality of the alignment
C	Set of colors to color the target network
$c(v_j)$	Color of the node v_j
R_{u_i}	Complete set of children of node u_i in Q
$u_i(k)$	kth child of u_i in Q
$N(v_j)$	Set of neighbors of node v_j in network T

The rest of this paper is organized as follows: Section 2 summarizes existing biological network alignment tools and elaborates on color coding and dynamic programming algorithms. Section 3 explains our coloring method. Section 4 presents our experiment results. Finally, Section 5 briefly concludes the paper.

2 Background

Existing solutions for aligning biological networks can be grouped under two main classes; heuristic [4, 5, 7, 18] and optimal [11, 16, 20, 22] alignment strategies. Next, we explain existing alignment methods.

Heuristic strategies: Most of the existing literature fall under this category. MetaPathwayHunter aligns a query network of a specific topology to a collection of metabolic networks [20]. Tohsato *et al.* proposed an algorithm which aligns linear paths within metabolic networks based on the features of individual enzymes in the network [25]. MetNetAligner allows insertion and deletion of enzymes [21]. Torque aligns protein interaction networks by ignoring the query topology imposed by the interactions [6]. IsoRank [23, 18] and SubMAP [5, 4] are two recent algorithms which combine node similarity and network topology information for biological network alignment. These studies showed that for network alignment taking the network topologies into account increases the accuracy of the resulting alignment. However, all these solutions are heuristic algorithms. They do not calculate a provable confidence value for the resulting alignment, nor provide error percentage for the alignment score. Thus, it is impossible to tell how well they do as compared to the optimal alignment.

Optimal alignment strategies: In order to find the optimal alignment between two biological networks, the algorithms in this class reduce this problem to the subgraph isomorphism problem. However, subgraph isomorphism problem has no known polynomial time solutions. Thus, in order to find the optimal alignment in practical time, these algorithms enforce additional restrictions on query network size and topology. PathBLAST, for instance can only align linear pathways to large networks [16]. This algorithm also limits the number of node insertions and deletions in the alignment. QPath [22] aligns linear networks with protein interaction networks using a probabilistic method called color coding [1]. QNet extends QPath to align tree networks and bounded treewidth graphs with protein interaction networks [11]. Hüffner *et al.* propose improvements on color coding which improves the performance of color coding [15]. We elaborate on QNet later in this section. These algorithms aim to find the optimal alignment between two networks. Although they do not guarantee optimality, they provide provable confidence values for their results. Next, we discuss how the confidence value is computed using a technique called color coding.

Color coding and dynamic programming: Color coding is a randomized method for finding small subnetworks within large networks [1]. Let us denote the number of nodes in the query network with m. Color coding begins by coloring the nodes of the target network using m colors randomly. We denote the set of m colors with C. We say that a subnetwork of the target network is *colorful* if all of its nodes have different colors. Figure 2 demonstrates two instances of coloring of a three node network, among which only one is

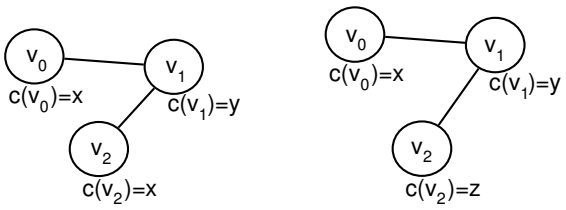

Figure 2: Two different coloring instances of a subnetwork of three nodes. The letters x,y and z represent different colors which can be assigned to the network nodes. $c(v_i)$ represents the color of a node v_i. (Left) An instance which is not colorful ($c(v_0) = c(v_2)$). (Right) A colorful assignment of colors.

colorful. Instead of finding an instance of the query, the problem now becomes finding a colorful instance of the query network. This reformulation of the problem reduces the time and the space complexity of the problem by a factor of $(\frac{n}{2})^m$. This is a significant improvement as network sizes in the order of hundreds to thousands are common.

One of the most recent algorithms that use color coding is QNet [11]. Next, we explain how coloring is used to measure the confidence in an alignment by focusing on QNet.

QNet develops a dynamic programming algorithm that uses three different functions to handle match, insertion and deletion of the nodes of the given networks. Let M, I and D respectively denote these functions. M has four arguments which are: query tree node u_i, target network node v_j, the color set used by the partial alignment S which is a subset of C, and set of first k children of u_i denoted by $R_{u_i}(k)$. D and I have all the arguments M has except $R_{u_i}(k)$. For every node pair (u_i,v_j) where $u_i \in Q$ and $v_j \in T$, the base case is set as follows:

$$M(u_i, v_j, \{c(v_j)\}, \emptyset) = Sim(u_i, v_j)$$

$$I(u_i, v_j, \emptyset) = D(u_i, v_j, \emptyset) = 0$$

After initializing M, I and D, dynamic programming recursions take place. Following defines these recursive functions:

$$M(u_i, v_j, S, R_{u_i}(k)) =$$

$$\max_{\substack{v_j{}' \in N_j, \\ S' \subset S}} \begin{cases} M(u_i, v_j, S', R_{u_i}(k-1)) + M(u_i(k), v_j{}', S - S', R_{u_i(k)}) \\ M(u_i, v_j, S', R_{u_i}(k-1)) + I(u_i(k), v_j{}', S - S') \\ M(u_i, v_j, S', R_{u_i}(k-1)) + D(u_i(k), v_j, S - S') \end{cases}$$

$$I(u_i, v_j, S) = \max_{v_j{}' \in N_j} \begin{cases} M(u_i, v_j{}', S, R_{u_i}) + \delta \\ I(u_i, v_j{}', S) + \delta \end{cases}$$

$$D(u_i, v_j, S) = \max_{v_j{}' \in N_j} \begin{cases} M(u_i(1), v_j{}', S, R_{(u_i(1))}) + \delta \\ D(u_i(1), v_j{}', S) + \delta \end{cases}$$

After the recursions are complete, one can find the score of the best alignment as:

$$\max \begin{cases} \max_{v_j} M(u_0, v_j, C, R_{u_0}) \\ \max_{v_j} I(u_0, v_j, C) \\ \max_{v_j} D(u_0, v_j, C) \end{cases}$$

We can find the alignment by backtracking the matrices corresponding to M, I and D. A more detailed explanation of dynamic programming formulation is in Dost *et al.* [11].

Dynamic programming can discover the optimal alignment if and only if the subnetwork of in T that corresponds to the optimal alignment is colorful. Since all coloring instances have the same likelihood, the

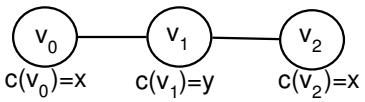

Figure 3: A three node network with colors assigned. x and y are the colors used to color this network. This instance of coloring is 1-hop colorful. However, it is not 2-hop colorful.

chance of finding the optimal alignment is equal to the ratio of the number of colorful instances to the total number of all coloring instances of that subnetwork. For random coloring, the total number of coloring instances is m^m for the m node subnetwork. The number of instances which the subnetwork is colorful is $m!$. Thus, the chance of finding the optimal alignment is $\frac{m!}{m^m}$. In order to increase the chance of finding the optimal alignment, color coding method repeats coloring and searching until a required confidence ϵ (where $\epsilon \in [0, 1]$) is met. For a given confidence value ϵ, we compute the number of iterations r needed to achieve at least ϵ confidence from the following equation:

$$\epsilon = 1 - \left(1 - \frac{m!}{m^m}\right)^r.$$

Thus, we have:

$$r = \frac{\log(1-\epsilon)}{\log(1-\frac{m!}{m^m})}. \tag{1}$$

3 Our Coloring Method

Color coding fails to find the optimal alignment if the true solution is not colorful. In order to address this, color coding performs multiple Bernoulli trials. At each trial, it colors the target network randomly and performs alignment with the expectation that at least one of those trials (i.e., iterations) will be successful. As the number of iterations increases, the confidence in the optimality of the result grows. Large number of iterations is however undesirable as the running time grows with the number of iterations.

The number of iterations in any coloring method depends on two parameters. These are the confidence value and the probability of coloring the correct subnetwork colorfully in the target network in a single iteration. Among these, the first one is supplied by the user. As a result it cannot be altered. Our coloring method increases the value of the second parameter to reduce the number of iterations.

Before we explain our coloring method, we define the concept of *distance* between two network nodes and the *neighborhood* of a node:

Definition 1. *(*DISTANCE*): The* distance *between two nodes in a network is the number of edges on the shortest path between them.* □

Definition 2. *(*k-NEIGHBORHOOD*): Given a positive integer k, the k-neighborhood of a node v_j in a network T is the set of all nodes of T with distance at most k to v_j.* □

Our coloring method arises from the following observation. A query is a connected biological network. An alignment maps the nodes and the edges of the query network to those of the target network. As a result, most of the nodes of this subnetwork are connected to each other directly or through a sequence of edges. Our coloring method exploits this by assigning different colors to the nodes whose distance to each other is small. We formalize this in the following definition.

Definition 3. *(*k-HOP COLORING*): We call that an assignment of colors to all the nodes of a given network as* k-hop colorful *if the graph contains no two nodes of the same color within a distance of k to each other.*

Figure 3 depicts an assignment of colors for a three node network. 1-hop coloring allows this assignment of colors since every node has different colors with its neighbors. However, this specific assignment of colors

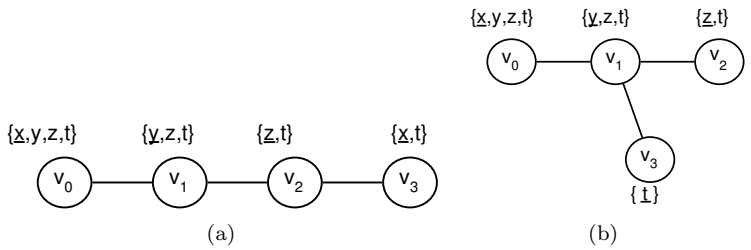

Figure 4: Coloring instances of two different networks of four nodes using 2-hop coloring. Node labels are written inside the nodes. The coloring is performed in v_0, v_1, v_2, v_3 order. The colors are represented with x, y,z and t. List of possible colors for a node is written in curly brackets near each node. The chosen color for each node is underlined. The figures show the process of (a) coloring a simple linear path and (b) coloring a branching network. Note that in (b) list of possible colors of v_3 is smaller. This is because in (b) all the remaining nodes are in 2-neighborhood of v_3.

is not possible using k-hop coloring, where $k \geq 2$. This is because v_0 and v_2 have the same color, and the distance between these two nodes is 2.

The first question we need to answer at this point is how fast the k-hop coloring reaches to a given confidence. Following theorem establishes a lower bound to the probability that a k-hop coloring instance is successful.

Theorem 1. *The probability of success (p) of k-hop coloring is bounded as:*

$$p \geq \frac{m!}{(m-k)^{m-k} \prod_{i=0}^{k-1}(m-i)}.$$

Proof. Let us denote the set of nodes to be colored with $\{v_0, v_1, ..., v_{m-1}\}$. We first prove this theorem on a specific network topology, where the nodes are connected as a path. That is there is an edge (v_i, v_{i+1}) for all $i \in \{0, 1, ..., m-1\}$. Figure 4(a) illustrates this on a small network. Later we will generalize this theorem to arbitrary topologies.

We first compute the number of k-hop colorings. Total number of legal k-hop colorings does not depend on the order at which we color the nodes. Therefore, to simplify our proof, we color the nodes in $v_0, v_1, ..., v_{m-1}$ order. We can assign m colors to v_0. However, once v_0 is determined there are $m-1$ choices for v_1. This is because the distance between v_0 and v_1 is less than k. So they are not allowed to have the same color. Similarly for all $i \geq k-1$ we have $m-i$ choices for v_i since i colors are used by other nodes in its k-neighborhood. For each of the remaining $m-k$ nodes v_i $(i \geq k)$ there are $m-k$ possible colors as there are k unique colors used among their k-distance neighbors. Thus, the number of k-hop colorings is $(m-k)^{m-k} \prod_{i=0}^{k-1}(m-i)$.

Now, we are ready to extend our computation to arbitrary topologies. Each node in the k-neighborhood of a node v_i asserts a restriction on the number of colors that v_i can be colored with. As a result, the number of all k-hop colorings reduces or remains the same if we increase the size of the k-neighborhood of a node. In any connected network, each node has at least k nodes in its k-neighborhood. Linear path minimizes the size of the k-neighborhood as each node is connected to at most two other nodes. Therefore, the total number of k-hop colorings in connected networks with m nodes is at most $(m-k)^{m-k} \prod_{i=0}^{k-1}(m-i)$.

The number of colorful instances of the same m nodes is simply the number of permutations of the m colors to these nodes, i.e., $m!$. The probability of success is the ratio of colorful instances to that of all k-hop colorful instances, that is:

$$p \geq \frac{m!}{(m-k)^{m-k} \prod_{i=0}^{k-1}(m-i)}.$$

□

Notice that k-hop coloring limits many useless coloring instances by including an additional constraint parameterized using k. Figure 4(a) presents this on a concrete example when the network contains four nodes. For $k = 2$, the number of possible 2-hop colorings is $2^2 \times 4 \times 3 = 48$. On the other hand the total number of colorings using the standard coloring method of Alon *et al.* [1] is much larger ($4^4 = 256$). The network in Figure 4(b) violates the linearity in Figure 4(a) by connecting v_3 with v_1 instead of v_2. This increases the size of the 2-neighborhood of v_0 and v_3. As a result, the number of possible 2-hop colorings reduces from 48 to 24. As the value of k grows, the k-hop coloring method becomes more and more stringent. For instance, when k is equal to $m-1$, using the k-hop coloring method, we can ensure that any connected subnetwork of size m is colorful. This implies that $(m-1)$-hop coloring guarantees to return the optimal connected subnetwork in only one iteration. However, this restriction comes at a price. Not all the networks can be colored using k-hop coloring. For example, if there is a clique of size m in the network, it cannot be colored with $(m-1)$-hop coloring using m colors. As the value of k drops the set of networks that can be colored grows. Our experiments demonstrate that more than 99% of all the networks in KEGG [19] are colorable using 1-hop coloring method. We elaborate on this issue in Section 4.1.

Following corollary sets an upper bound to the number of iterations required by our method.

Corollary 1. *The number of iterations needed by k-hop coloring to achieve a confidence ϵ is at most:*

$$\frac{\log(1-\epsilon)}{\log\left(1-\frac{m!}{(m-k)^{m-k}\prod_{i=0}^{k-1}(m-i)}\right)}. \quad \square$$

One can prove Corollary 1 by substituting the success probability in Equation 1 with that of Theorem 1. As k grows, the upper bound to the number of iterations drops as well as the percentage of colorable networks. We observe that small values of k (such as $k = 1$) results in a significant improvement in the number of iterations with negligible drop in colorable networks (See Section 4.2 for details).

4 Experiments

This section evaluates the performance of our method experimentally.

Dataset. We used networks from the KEGG database [19] in our experiments. We downloaded all the gene regulatory networks that have more than 15 genes (i.e., nodes). In total, there are 297 such networks as of September 2010. In this database, we had 21 different types of networks (such as MAPK signaling pathway) from 46 different organisms.

We created three sets of query networks from this dataset by performing random walks on our gene regulatory networks. These sets contain seven, eight and nine node queries respectively. Each query set consisted of 50 query networks. All the query and database networks we used in this paper can be downloaded from our server at `http://bioinformatics.cise.ufl.edu/khop.html`.

We downloaded gene annotation data for the genes in our dataset from the gene ontology database [3]. We used "2011-02" release of type "assocdb" for our experiments. There are a number of networks, which do not have any annotations in gene ontology database. We did not use such networks in our experiments requiring gene function analysis.

Implementation Details. We implemented all the methods we used in our experiments in C++. We use our own implementation of color coding and dynamic programming algorithm in our experiments. We used the same dynamic programming formulation used by Dost *et al.* [11]. In order to calculate pairwise similarity scores between two nodes, we use BLAST [2]. We normalize the minus logarithm of the E-values to get the scores. We use insertion/deletion penalties as high as 0.5 times the largest possible match score.

Evaluation Criteria. We measure *running time* and *confidence* as performance indicators for the coloring methods we compared. We measured running time as the number of iterations. We measured confidence after interaction r as the percentage of the alignments which find the optimal alignment after r iterations.

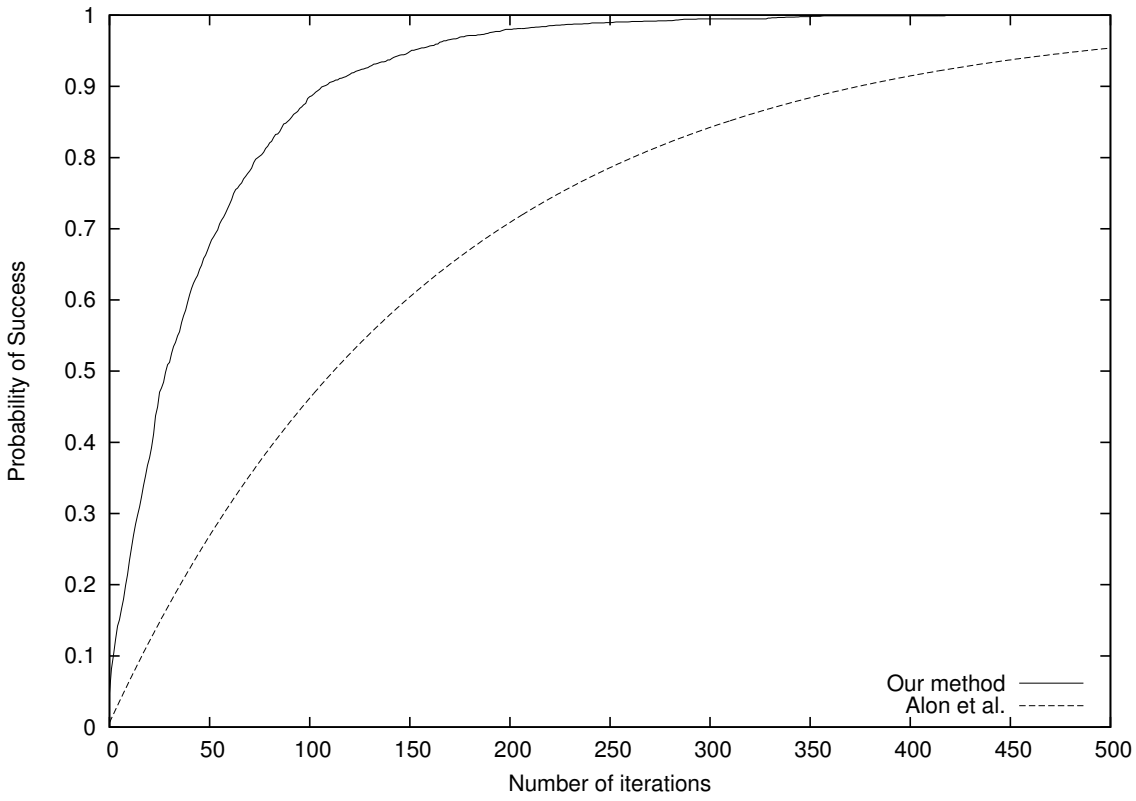

Figure 5: Probability of success versus the number of iterations for 1-hop coloring. Probability of success for t iterations shows the fraction of the alignments among all 2500 alignments for which the optimal alignment is found within the first t iterations. For instance, probability of success = 0.9 means that the corresponding method optimally aligned 90% of all the query and target network pairs. We used seven node query networks in this figure.

In order to use as the optimal alignments for our experiments, we aligned all the pairs of networks using random coloring with 99.99% accuracy.

Test Environment. We performed our tests on Linux servers equipped with dual AMD Opteron dual core processors running at 2.2 GHz, and 3 GBs of main memory.

In Section 4.1, we compare 1-hop coloring to the random coloring method. In Section 4.2, we analyze the effects of k on our coloring method.

4.1 Performance comparison against the state of the art coloring

In Section 3, we showed theoretically that our coloring method outperforms random coloring of Alon *et al.* [1] in terms of the speed at which they achieve a given confidence value. In order to demonstrate the difference in performance experimentally, we compare 1-hop coloring with the coloring method of Alon *et al.* on our dataset. In this experiment, we randomly picked 50 networks from our database. We then aligned each of the 50 query networks in each query set with all of these 50 database networks. So, in total we performed $50 \times 50 = 2500$ pairwise alignments for each query network size. We carried out each of these 2500 alignments using the coloring method by Alon *et al.* [1] and using our 1-hop coloring method. For each alignment, we fix the total number of iterations to 500. We record the resulting alignment score after each iteration for every alignment. We report the fraction of the alignments among all the 2500 alignments which

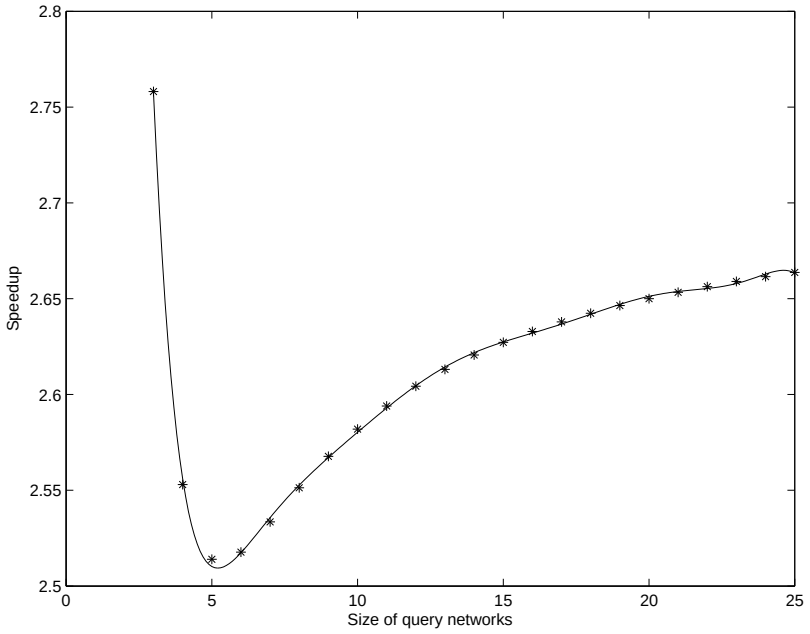

Figure 6: The performance improvement of 1-hop coloring over Alon *et al.* for query sizes of 4 to 25. The speedup is calculated by dividing the number of iterations for Alon *et al.* by that of 1-hop coloring until they both reach to the same confidence. We also draw a polynomial fitting the calculated values.

reach to the optimal one in first t iterations for $1 \leq t \leq 500$ for each coloring method.

Figure 5 shows our experiment results when the query network contains seven nodes. Our experiments demonstrate that 1-hop coloring has a significant performance improvement over Alon *et al.* [1]. In order to achieve same level of confidence on the alignment result, random coloring requires about twice the number of iterations, thus the amount of time. For example, random coloring reaches probability of success of 0.9 after more than 350 iterations. On the other hand, 1-hop coloring needs less than 120 iterations for same amount of results. Furthermore, in only 150 iterations, 1-hop coloring achieves a probability of success which random coloring failed to approach even after 500 iterations. Results of this experiment reveal that alignment using 1-hop coloring is more than two times faster than alignment using random coloring for the same success probabilities.

Figure 5 shows that k-hop coloring outperforms Alon *et al.* for seven node query networks. An interesting question at this point would be how the size of query networks changes the difference in performance. In order to figure out the relation between the size of query networks and the performance gap between two methods, we theoretically calculate the speedup 1-hop coloring creates for different query sizes. We do this by dividing the number of iterations required by Alon *et al.* to achieve a specific confidence value by that of 1-hop coloring. We calculate the number of iterations using the formulas we provided in Equation 1 and Corollary 1. Notice that in the ratio, as the term with confidence is canceled out, ϵ does not have an effect on the ratio of running times.

Figure 6 shows the results of this experiment. Our results show that the gain in performance by using 1-hop coloring is greater than 2.5 for every query size. For query networks which have less than five nodes, the performance difference is higher than expected, because the confidence of 1-hop coloring for such query networks is very high. Considering the practical query network size of bounded treewidth queries, which is 6 to 10, our method performs between 2.5 to 2.6 percent faster than Alon *et al.*. For example, in order to align a nine node query network to any target network with 80% confidence, Alon *et al.* requires 1718 iterations. On the other hand, our 1-hop coloring method requires only 669 iterations for the same alignment with the same confidence on the result.

Table 2: Statistics of k-hop coloring for $k = 1, 2$ and 3. The first column shows the percentage of target networks which can be colored using k-hop coloring. Percentage of nodes in k-neighborhood is size of k-neighborhood divided by the total number of nodes in the network. The last three columns report the number of iterations required to achieve 95%, 98% and 99% confidence respectively.

	Percentage of colorable networks	Percentage of nodes in k-neighborhood	Number of iterations to achieve confidence		
			0.95	0.98	0.99
1-hop	100%	8%	193	252	297
2-hop	20%	27%	77	100	118
3-hop	4%	42%	31	40	47

4.2 Effects of k on coloring

In Section 3, we discussed that with k-hop coloring, higher confidence can be achieved using larger k. However, as the value of k grows, the number of networks that can be colored with k-hop coloring drops. In this experiment, we aim to show the effects of k on the accuracy and performance of our coloring method. In order to do this, similar to the experiment in Section 4.1, we randomly choose 50 database networks and align each of them with all the query networks, resulting in 2500 pairwise comparisons. We repeat this experiment using the k-hop coloring method for $k = 1, 2$ and 3. In order to observe different aspects of the k-hop coloring method, we measure three different types of statistics for each value of k.

1. Percentage of networks that can be colored using k-hop coloring. The larger this value is the better. This is because our method will find the optimal alignment only if the target network is k-hop colorful.

2. Percentage of nodes that are in the k-neighborhood of a target node on the average. The smaller this value is the better as each node in the k-neighborhood limits the set of colors that can be assigned to that node.

3. The number of iterations needed to reach to a given confidence value ϵ when the target network can be colored using the k-hop coloring method. We report this for $\epsilon = 0.95, 0.98$ and 0.99.

Table 2 shows our results for this experiment. Our experiment results reveal that 1-hop coloring can successfully color every network in our dataset. Moreover, it can achieve 99% confidence in alignment results after less than 300 iterations. We also observed that, as k increases the number of iterations to attain a certain confidence value decreases significantly. For example at 99% confidence, 2-hop coloring needs only 40% of the number of iterations of 1-hop. 3-hop coloring needs about 15% of the number of iterations that 1-hop coloring requires for the same confidence. For the networks that can be colored, we observe that increasing k only by one speeds up by more than two times. However, we see that the improvement in the running time comes at the price of reduced percentage of colorable networks. This is because as k grows, so does the k-neighborhood of each node in the target network. Therefore, based on our experiments, we suggest using the largest k, where the size of k-neighborhood does not exceed 10% of all network nodes.

In summary our experiments suggest that for a small percentage (about 4 to 20%) of the networks in KEGG, 2-hop or 3-hop coloring are preferable. For the remaining ones, 1-hop coloring is the best choice.

4.3 Biological significance of the results

In this experiment, we demonstrate how much information can be harvested from gene regulatory networks using our method. In order to do this, we measure the information content of our networks. We use the *functional enrichment score* to evaluate the information content of a network. For a gene annotation term g, we calculate its functional enrichment score in a network Q as follows. Let us denote the ratio of the nodes annotated with g as $p(g|Q)$. Let us denote the mean and standard deviation of the probability distribution of a protein in the whole dataset being annotated with g with $\mu(g)$ and $\sigma(g)$. The functional enrichment

Table 3: Statistical information regarding the functional enrichment of the query networks we used in our experiments for 7,8 and 9 node query networks. First three columns are the mean, standard deviation and the maximum of the enrichment scores of all enriched terms. Last column shows the percentage of queries which contain at least one enriched term.

	The functional enrichment score distribution			% of enriched networks
	Mean	Std. Deviation	Maximum	
7-node	3.5	1.04	5.26	48%
8-node	3.85	0.7	5.17	30%
9-node	3.89	0.89	5.49	56%

score (or simply the *enrichment score*) of a term g in Q is then calculated as $(p(g|Q) - \mu(g))/\sigma(g)$. In the literature, this measure is also called the *Z-score*. We say a term g is enriched in Q if the enrichment score of g in Q is greater than 2.0. We say a network Q is enriched, there exists at least one term that is enriched in Q. We ignore terms with smaller enrichment scores to account for false discoveries.

Information content of the query networks. Using the enrichment score, we first evaluate our query networks. We measure how increasing network size affects the information content of a network. In order to do this we calculate the enrichment scores for each annotation in all of our query networks. We record the number of enriched query networks and the enrichment scores of each query network. For query networks of size 7, 8 and 9, we report the mean and the standard deviation of enrichment score distribution, as well as the average of the maximum enrichment score observed and the percentage of enriched query networks in respective network sets.

Table 3 displays our results for this experiment. Our experimental results demonstrate that there is a high ratio of enriched networks among our queries. This result implies that our query set consists of many networks, whose genes have large amount of common functions. Another observation to be made is that, query size does not have significant effect on the enrichment scores of the networks. For example, in our experiments enriched networks among the set of 7, 8 and 9 node query networks have similar enrichment score distributions. However, there is a large variation in the percentage of the enriched networks. We attribute this to the random walk algorithm, which uses a total random network generation strategy, without the knowledge of any functional data.

We conclude that there can be many large query networks with high enrichment scores. Thus, our coloring algorithm is essential in making it computationally feasible to align them to target networks in practical time.

Information content of the alignments. In the previous experiment, we evaluated the information content of our queries using the enrichment score. In this experiment, we use the same tool to interpret our alignment results. In order to do this, we use the alignments with the highest information content. Query networks and the subnetworks that align to them in these alignments are annotated with over 35 terms each. For each of these networks, we calculate the enrichment scores of each gene annotation term in the query network and the subnetwork of the target network it is aligned to. We report the enrichment score of every term in our results.

Figure 7 shows our experimental results for the enriched terms for top five queries with the largest number of enriched terms. Notice that there are a large number of terms which have enrichment scores greater than 2.0 in both query and target networks. An enrichment score of 2.0 or larger means that the probability to observe a specific annotation in our network is more than two standard deviations larger than the mean probability of all of the dataset. In other words, these annotations in query and target networks are statistically significant. Our results demonstrate that the enrichment scores of the terms in the query network and the matching target network converge almost converge on the same line for each query. This shows that there is a high correlation between the enrichment scores in query and the target networks. High correlation between the enrichment scores of highly enriched terms show that our alignment method

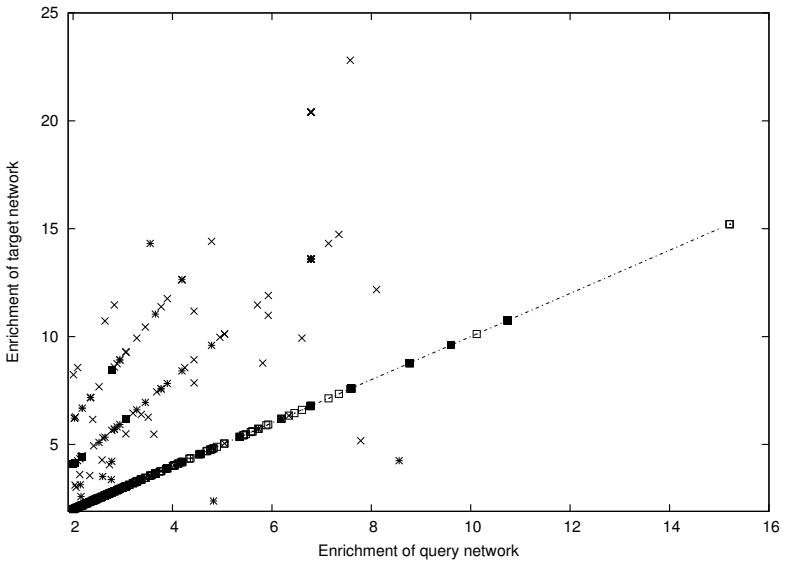

Figure 7: The enrichment scores of each gene annotation term in query and target networks in top five queries with the largest number of enriched terms. The enrichment score of the terms in each alignment is shown using a different symbol in the figure. The dashed line is the set of hypothetical points where the enrichment of a term in the target network is equal to the enrichment of the term in the query network.

is successful at discovering functional similarities between networks. Another interesting observation that follows from this figure is that each term is more enriched in the target network than the query network most of the time. This result suggests that, our method can discover highly enriched subnetworks.

5 Conclusion

In this paper, we considered the problem of finding a subnetwork in a given biological network (i.e., target network) that is the most similar to a given small query network. Our aim was to find the optimal solution (i.e., the subnetwork with the largest alignment score) with a provable confidence bound. There is no known polynomial time solution to this problem in the literature. Alon *et al.* [1] has developed a state of the art coloring method that reduces the cost of this problem that needs to be executed for many iterations until a user supplied confidence is reached. Here we developed a novel coloring method, named *k-hop coloring*, that achieves a provable confidence value in a small number of iterations without sacrificing the optimality. Our method considers the color assignments already made in the neighborhood of each target network node while assigning a color to a node. This way it preemptively avoids many color assignments that are guaranteed to fail to produce the optimal alignment. We demonstrate both theoretically and experimentally that our coloring method outperforms that of Alon *et al.* which is also used by a number network alignment methods including QPath and QNet by a factor of three without reducing the confidence in the optimality of the result. Many application so far has benefited from the idea of coloring biological networks. Aligning a query network to a subnetwork of a target network is only one of them. We believe that our novel coloring methodology will improve the utilization of coloring and randomization in large scale biological network analysis in practical applications, and thus it will be of great value to biologists.

6 Acknowledgements

This work is funded by NSF under grants CCF-0829867 and IIS-084543.

References

[1] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42:844–856, July 1995.

[2] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403 – 410, 1990.

[3] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, and Gerald M. Rubin. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25, 2000.

[4] Ferhat Ay and Tamer Kahveci. SubMAP: Aligning Metabolic Pathways with Subnetwork Mappings. In *RECOMB*, pages 15–30, 2010.

[5] Ferhat Ay, Tamer Kahveci, and Valérie de Crécy-Lagard. A fast and accurate algorithm for comparative analysis of metabolic pathways. *J. Bioinformatics and Computational Biology*, 7(3):389–428, 2009.

[6] Sharon Bruckner, Falk Huffner, Richard M. Karp, Ron Shamir, and Roded Sharan. Torque: topology-free querying of protein interaction networks. *Nucleic Acids Research*, 37(suppl 2):W106–W108, 2009.

[7] Leonid Chindelevitch, Chung-Shou Liao, and Bonnie Berger. Local optimization for global alignment of protein interaction networks. *Pac Symp Biocomput*, pages 123–132, 2010.

[8] Jose C Clemente, Kenji Satou, and Gabriel Valiente. Reconstruction of phylogenetic relationships from metabolic pathways based on the enzyme hierarchy and the gene ontology. *Genome Inform*, 16(2):45–55, 2005.

[9] Jose C Clemente, Kenji Satou, and Gabriel Valiente. Finding conserved and non-conserved reactions using a metabolic pathway alignment algorithm. *Genome Inform*, 17(2):46–56, 2006.

[10] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

[11] Banu Dost, Tomer Shlomi, Nitin Gupta, Eytan Ruppin, Vineet Bafna, and Roded Sharan. QNet: a tool for querying protein interaction networks. *J Comput Biol*, 15(7):913–925, Sep 2008.

[12] Christof Francke, Roland J Siezen, and Bas Teusink. Reconstructing the metabolic network of a bacterium from its genome. *Trends Microbiol*, 13(11):550–558, Nov 2005.

[13] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[14] L. Giot and J. S. Bader et al. A protein interaction map of drosophila melanogaster. *Science*, 302(5651):1727–1736, 2003.

[15] Falk Hüffner, Sebastian Wernicke, and Thomas Zichner. Algorithm engineering for color-coding with applications to signaling pathway detection. *Algorithmica*, 52(2):114–132, 2008.

[16] Brian P Kelley, Bingbing Yuan, Fran Lewitter, Roded Sharan, Brent R Stockwell, and Trey Ideker. PathBLAST: a tool for alignment of protein interaction networks. *Nucleic Acids Res*, 32(Web Server issue):W83–W88, Jul 2004.

[17] Michael Levine and Eric H. Davidson. Gene regulatory networks for development. *Proceedings of the National Academy of Sciences of the United States of America*, 102(14):4936–4942, 2005.

[18] C.S. Liao, K. Lu, M. Baym, R. Singh, and B. Berger. IsoRankN: spectral methods for global alignment of multiple protein networks. *Bioinformatics*, 25(12):253–238, 2009.

[19] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res*, 27(1):29–34, Jan 1999.

[20] Ron Y Pinter, Oleg Rokhlenko, Esti Yeger-Lotem, and Michal Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–3408, Aug 2005.

[21] Cheng Q, Harrison R, and Zelikovsky A. MetNetAligner: a web service tool for metabolic network alignments. *Bioinformatics*, 25(15):1989–90, 2009.

[22] Tomer Shlomi, Daniel Segal, Eytan Ruppin, and Roded Sharan. QPath: a method for querying pathways in a protein-protein interaction network. *BMC Bioinformatics*, 7:199, 2006.

[23] R. Singh, J. Xu, and B. Berger. Pairwise Global Alignment of Protein Interaction Networks by Matching Neighborhood Topology. In *RECOMB*, pages 16–31, 2007.

[24] Padmavati Sridhar, Tamer Kahveci, and Sanjay Ranka. An iterative algorithm for metabolic network-based drug target identification. *Pacific Symposium on Biocomputing*, pages 88–99, 2007.

[25] Tohsato Y, Matsuda H, and Hashimoto A. A Multiple Alignment Algorithm for Metabolic Pathway Analysis Using Enzyme Hierarchy. In *ISMB*, pages 376–383, 2000.