FROM UNCERTAIN PROTEIN INTERACTION NETWORKS TO SIGNALING PATHWAYS THROUGH INTENSIVE COLOR CODING

Haitham Gabr, Alin Dobra and Tamer Kahveci*

CISE Department, University of Florida,
Gainesville, FL 32611, USA
E-mail: {hgabr, adobra, tamer}@cise.ufl.edu*

Discovering signaling pathways in protein interaction networks is a key ingredient in understanding how proteins carry out cellular functions. These interactions however can be uncertain events that may or may not take place depending on many factors including the internal factors, such as the size and abundance of the proteins, or the external factors, such as mutations, disorders and drug intake. In this paper, we consider the problem of finding causal orderings of nodes in such protein interaction networks to discover signaling pathways. We adopt color coding technique to address this problem. Color coding method may fail with some probability. By allowing it to run for sufficient time, however, its confidence in the optimality of the result can converge close to 100%. Our key contribution in this paper is elimination of the key conservative assumptions made by the traditional color coding methods while computing its success probability. We do this by carefully establishing the relationship between node colors, network topology and success probability. As a result our method converges to any confidence value much faster than the traditional methods. Thus, it is scalable to larger protein interaction networks and longer signaling pathways than existing methods. We demonstrate, both theoretically and experimentally that our method outperforms existing methods.

Keywords: protein interaction networks; signaling pathways; color coding; chromatic polynomial

1. Introduction

Studying interactions between proteins has been of utmost importance in understanding how proteins work collectively to govern cellular functions.[1,2] Such collection of interactions among proteins is called a protein interaction network. The interactions are uncertain events. They may or may not take place depending on the internal factors, such as the size and abundance of the proteins, or the external factors, such as mutations, disorders and drug intake. Mathematically, a protein interaction network is often modeled as an edge-weighted undirected graph where each node denotes a protein and each edge represents an interaction between a pair of proteins. The weight of an edge denotes the level of confidence that this interaction truly exists.

Computational analysis of protein interaction networks has been essential in identification of signaling pathways. A signaling pathway is a series of proteins in which each protein participates in transmitting biological information by modifying its successor through an interaction. Thus, signaling pathways can be viewed as simple paths in protein interaction networks.[3] One outcome of the uncertainty of the interactions is that the pathway that transmits signals between two specific sets of proteins (e.g., from membrane receptors to transcription factors) may differ as the set of interactions change. Finding possible pathways in the presence of such uncertainty has great potential in numerous applications including identification of drug targets, studying complex diseases, drug-drug interaction and metabolic engineering.

The confidence value of an interaction between two proteins is often considered as the probability that a signal is transmitted between those two proteins. Scott *et al.* conjectured that a signal tends to move through the most probable pathway[4] (i.e., the pathway with the highest product of interaction confidence values). The following defines the *Minimum Weight Pathway Identification* problem which is identical to the problem of identifying the most probable pathway in a protein interaction network.

Problem. (Minimum Weight Pathway Identification) Consider a protein interaction network $G = (V, E, w)$ where V denotes the set of proteins and E denotes the set of interactions. Let us denote the confidence for each interaction in E with function $\lambda() : E \Rightarrow [0, 1]$. We define the function $w()$ on the edges as $w() = -\log \lambda()$. Assume that we are given a set of starting proteins $S \subseteq V$ and a set of target proteins $T \subseteq V$. Given a path length denoted by m, the problem is to find a path $\Phi = v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_m$ with no repeating proteins, where $\sum_{i=1}^{m-1} w(v_i, v_{i+1})$ is the minimum among all paths with $v_1 \in S$, $v_m \in T$ and $v_i \in V$, $\forall i \in \{1, 2, \ldots, m\}$.

Scott *et al.* showed that the traveling-salesman problem is polynomial-time reducible to the problem above;[4] therefore it is NP-hard. They developed a method using the *color-coding* technique of Alon *et al.*[5] The idea of this method is to randomly assign each node in the graph one of m different colors. A pathway is *colorful* if and only if all of its nodes are in different color. They then search for an optimal colorful pathway. Finding a colorful path is computationally much cheaper than finding a path without assigning colors. The drawback is that the optimal path may not be colorful in a random color assignment, leading color coding to find a sub-optimal result. To deal with this, it repeats the coloring process for several iterations. The confidence in the optimality of the result monotonically increases with each iteration until it reaches a given level of confidence. As we elaborate later in Section 2, the confidence value depends solely on the pathway length m and does not capitalize on readily available information such as the network topology and color assignment. As a result, the method provides a theoretically correct but very conservative confidence value. Hence it requires many iterations in order to achieve a given confidence level, leading to an unnecessarily inefficient running time performance.

Gülsoy *et al.*[6] presented an enhanced color-coding technique called *k-hop coloring.* A colored network is k-hop colorable if the shortest path between all pairs of same-color nodes is more than k hops in length. This method exploits the network topology and the node colors to assign the network a maximal value k such that the network is k-hop colorable. This additional piece of information allows for higher success probability at each iteration, yielding fewer iterations than that by Scott *et al.*[4] However, subnetworks with high connectivity quickly diminish the ability to k-hop color the whole network for large values of k. For example, a network containing a clique of size m cannot be colored with $(m-1)$-hop coloring using m colors.[6]

Our contribution. In this paper, we consider the problem of finding signaling pathways in protein interaction networks. We develop a new coloring method that overcomes the bottlenecks of existing coloring methods by Scott *et al.*[4] and Gülsoy *et al.*[6] Our contribution comes from a deeper understanding of the relation between network topology, random color assignment and confidence value. We assign a value that we call k_{max} to each node individually by studying the colors of all the nodes in the network. k_{max} value of a node v at an iteration is the maximal value of k such that there is no other node u that is reachable from v in k hops such that both u and v have the same color. We also study how this reflects on the resulting success probability for each iteration. Given different k_{max} values for each node on a pathway, we show how to obtain a bound on success probability. Based on these findings, we present a new method for detecting signaling pathways in protein interaction networks using an enhanced k-hop coloring technique. Given the parameter pathway length m, we start by randomly assigning one of m colors to each node in the graph, we then extract the optimal colorful pathway. We then calculate our new bound on success probability. We repeat this process until the cumulative success probability is at least equal to a given confidence level. Our experiments demonstrate that our method converges to high confidence values much faster than the existing methods including Scott *et al.*[4] This enables computational analysis of larger networks or longer pathways.

The rest of the paper is organized as follows. Section 2 discusses the background and related work. Section 3 describes our method in detail. Section 4 presents experiments evaluation. Finally, Section 5 concludes the paper.

2. Background

A number of methods have been developed so far to identify signaling networks from protein interaction networks. Kelley *et al.*[3] detected conserved signaling pathways between related organisms by performing global alignment between their protein interaction networks. Shlomi *et al.*[7] introduced QPath, a method for querying protein interaction networks for pathways using known homologous pathways as queries. Both Kelley *et al.*[3] and Shlomi *et al.*[7] are comparative methods. They require knowledge of multiple interaction networks.

Lu *et al.*[8] presented a divide-and-conquer algorithm to find signaling subnetworks in protein interaction networks. They scored the resulting subnetworks based on the similarity of expression profiles of their nodes to the given source and destination nodes. This method aims to detect paths whose proteins are highest in expression similarity, and thus it does not utilize the confidence in the interactions. Steffen *et al.*[9] used exhaustive search to list pathway candidates in protein interaction networks, and scored each one based on how similar the expression profiles of its genes are. Bebek *et al.*[10] presented a method for finding new signaling pathways using association rules of known ones. The time complexity of exhaustive graph search is exponential in terms of the network size, and hence is very inefficient. Gitter *et al.*[11] presented a method for discovering signaling pathways by adding edge orientation to protein interaction networks. They selected an optimal orientation of all edges in the network that maximizes the weights of all satisfied length-bound paths. They proved that this problem is NP-hard, and provided three approximation algorithms for it. As shown in their results, these methods do not scale well with increasing number of source and destination nodes and path length.

The closest studies to that presented in this paper are those by Scott *et al.*[4] and Gülsoy *et al.*[6] The former detected signaling pathways in protein interaction networks using color coding. The latter developed topology-aware color coding for network alignment. We describe both methods in Section 1. Both methods run multiple coloring iterations. Let us denote the probability that the coloring at an iteration is successful (i.e., true optimal path is colorful) with P_s. The probability that at least one out of r iterations is successful is $1 - (1 - P_s)^r$. Following from this, in order to ensure confidence of at least ϵ $(0 \leq \epsilon \leq 1)$, they run r iterations, such that $1 - (1 - P_s)^r \geq \epsilon$. Both methods calculate success probability as

$$P_s = \frac{m!}{N_c} \tag{1}$$

where N_c is the number of coloring assignments possible for the optimal pathway. They differ in the way they compute N_c. Scott *et al.*[4] calculated $N_c = m^m$. Gülsoy *et al.*[6] calculated $N_c \leq (m-k)^{m-k} \prod_{i=0}^{k-1}(m-i)$ where k is the value assigned to the network such that it is k-hop colorable. Notice that in Equation 1, smaller values for N_c are desirable. This is because small values for N_c increase success probability, and thus reduce the number of iterations needed to attain a given confidence level ϵ. *This paper develops a novel method that computes a much smaller upper bound on N_c than both of these approaches, leading to higher bound on P_s.*

3. Method description

This section describes our method in detail. Section 3.1 presents a high level description of our method. Section 3.2 makes key definitions needed by our method. Section 3.3 defines how we compute probability of success for our method. Section 3.4 theoretically shows why the performance of our method is better than or the same as that of existing methods.

3.1. *An overview of our method*

Consider a weighted undirected graph $G = (V, E, w)$, a path length m, a set of starting and target nodes S and T respectively, with $S, T \subseteq V$. Scott *et al.* has shown that it is possible to find the minimum weight path of a m nodes from S to T in G using dynamic programming.[4] In principle, our method follows the same steps. Algorithm 3.1 presents our method at a high level. The algorithm works iteratively. At each iteration we randomly color the network (Step 3). We then use dynamic programming to find the minimum weight colorful path (Step 4). The dynamic programming works as follows. Let us denote a coloring function with $c() : V \Longrightarrow C$. We dynamically tabulate the minimum weight of a colorful path colored only using C', starting within S and ending at v, using the following recurrence:[4]

$$W(v, C') = \min_{u: c(u) \in (C' \backslash \{c(v)\})} W(u, C' \backslash \{c(v)\}) + w(u, v), |C'| > 1 \tag{2}$$

where $W(v, \{c(v)\}) = 0$ if $v \in S$ and ∞ otherwise. Once we find the best colorful path in that iteration, we store it in a min-heap according to the weight of the path (Step 5). We then compute the probability that the current iteration was successful in finding the optimal path (i.e., minimum weighted path regardless of being colorful or not) (Step 6) and update our confidence in the best result seen so far (Step 7).

Algorithm 3.1 Compute the minimum weight path

```
Require: Input network $G = (V, E, w)$, starting and target node sets $S \subseteq V$ and $T \subseteq V$
Require: Color set $C = \{c_1, c_2, \ldots, c_m\}$
Require: Confidence cutoff $\epsilon$
1: $P \leftarrow 0$ {Initialize overall success probability}
2: while $P < \epsilon$ do
3:     Assign colors to the nodes in $V$ randomly from the set $C$
4:     $\Phi \leftarrow$ Find the minimum weight colorful path of length $m$ in $G$
5:     Store $\Phi$ in the min-heap of solutions observed so far if it is a new solution.
6:     Compute the probability of success $P_s$ for the current coloring iteration.
7:     $P \leftarrow 1 - (1 - P)(1 - P_s)$ {Update the overall success probability}
8: end while
```

As we noted earlier, Algorithm 3.1 is very similar to the method by Scott *et al.*[4] So, a legitimate question is what is the big challenge addressed in this paper? The answer lies in Step 6 of the algorithm where we compute the probability of success at each iteration. This step is missing in all the color coding methods to the best of our knowledge, including Scott *et al.*[4] among others.[5–7,12] All these existing methods precompute a probability of success prior to the iterations and use the same probability value throughout the iterations (see Equation 1 and Section 2). As a result, they make extremely conservative assumptions which have to hold regardless of which node gets which color. *Our contribution is to eliminate those worst case assumptions and recompute the probability of success at each iteration by carefully inspecting the colors of all the nodes.* We explain how we do this in the following sections.

3.2. *Basic definitions and model*

In this section, we build the mathematical model that will help us compute the probability of success in each iteration. Assume that we are given a protein interaction network similar to the one described in Section 1, denoted by $G = (V, E, w)$, where $w(u, v) = -\log \lambda(u, v)$. Also assume that the colors of the nodes are already assigned in the current iteration. We denote the set of possible colors with $C = \{c_1, c_2, \ldots, c_m\}$ and the color of node $v \in V$ with $c(v)$. We start by discuss several key concepts.

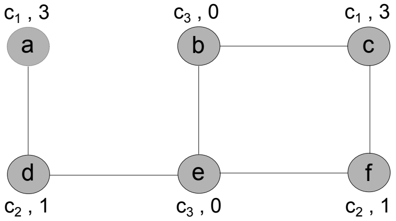


Fig. 1. A hypothetical protein interaction network with six nodes {a, b, c, d, e, f}. The network is colored using three colors $\{c_1, c_2, c_3\}$. Each node carries two labels. The label on the left denotes the color assigned to this node. The one on the right is the node's k_{max} value. For instance node d is assigned to color c_2 and its k_{max} value is 1 (i.e., there is no other node assigned to color c_2 within 1-hop of node d).

Definition 1. (Simple path) Given a network $G = (V, E)$, a *simple path* from u to v (u, $v \in V$) is an ordering $< v_1, v_2, \ldots, v_k >$, of a subset of the vertices of G such that $v_1 = u$, $v_k = v$, $(v_i, v_{i+1}) \in E$ and $v_i \neq v_j$ for all $i \neq j$.

Consider two nodes u and v in G. Let k be a positive integer. We say that v is *reachable* from u in k hops if there is a simple path from u to v that contains k edges.

Definition 2. (k neighborhood of a node). Let $v \in V$ be a node in G, and k be a nonnegative integer. We define the k neighborhood of node v as the set of nodes in $V \backslash \{v\}$ which are reachable from v in k hops or less. We denote this set using notation $\Psi_k(v)$.

Figure 1 shows an example of a colored network. In this example, $\Psi_1(a) = \{d\}$ because the node d is the only node that is reachable from the node a in 1 hop (or less). Similarly, $\Psi_1(f) = \{c, e\}$, $\Psi_2(a) = \{d, e\}$ and $\Psi_2(f) = \{c, e, b, d\}$. Following definition establishes the relationship between each node of the network and the rest of the network based on the colors assigned to all the nodes.

Definition 3. (k_{max} value of a node). Let $v \in V$ be a node in a colored network G. The k_{max} value of v, denoted with $k_{max}(v)$, is the maximal value of k such that the k neighborhood of v does not contain a node with the same color as v. i.e., $k_{max}(v) = \text{argmax}_k\{\forall u \in \Psi_k(v), c(u) \neq c(v)\}$.

Figure 1 shows the k_{max} values for the nodes in the network. For example, the colors of all the nodes in $\Psi_1(f) = \{c, e\}$ are different than the color of f. Expanding the neighborhood of f to two, we get $\Psi_2(f) = \{c, e, b, d\}$. In this set, $c(d) = c(f) = c_2$. Therefore $k_{max}(f) = 1$. Similarly, $k_{max}(a) = 3$ and $k_{max}(b) = 0$. Next definition characterizes a simple path of the network.

Definition 4. (k_{max} CONFIGURATION OF A PATH). Consider a simple path $\Phi = v_1 \rightarrow \ldots \rightarrow v_m$ of m nodes in G. The k_{max} configuration of Φ is the vector $[k_{max}(v_1), \ldots, k_{max}(v_m)]$.

As an example, in Figure 1, the k_{max} configuration of the path $\Phi = a \rightarrow d \rightarrow e \rightarrow f$ is [3, 1, 0, 1]. That for $a \rightarrow d \rightarrow e \rightarrow b$ is [3, 1, 0, 0].

3.3. *Bounding the probability of success tightly*

In this section, we focus on one coloring iteration and describe how we compute the probability of success in that iteration. Consider any colorful path with m nodes. The number of ways to assign colors to the nodes of that path while keeping it colorful is $m!$. Notice that this is equal to the numerator in Equation 1 for probability of success. The denominator in that equation, denoted by N_c, is the total number of ways to color that path regardless of whether it yields a colorful path or not.

Notice that there can be many different color assignments that yield the same k_{max} configuration for the same path. Also, as we will show later, the number of possible color assignments to the nodes of a path can be different for different k_{max} configurations. Indeed, the k_{max} configuration of a path describes the constraints imposed on all the nodes of that path about how many alternative colors can be assigned to them. Following from this observation, we first build a new undirected and unweighted graph, called the *constraint graph* from the k_{max} configuration. By utilizing the constraint graph we transform the problem of finding the number of possible colorings to the chromatic polynomial computation problem. Next, we describe how we build the constraint graph and how we utilize it to find the number of colorings.

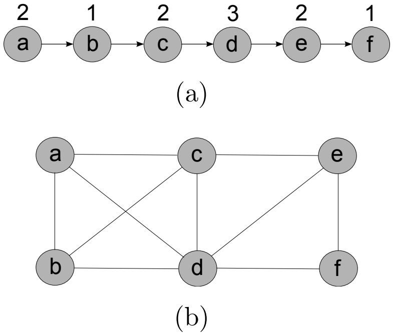


Fig. 2. (a) An example 6-node path with its k_{max} configuration shown above it. (b) The corresponding constraint graph G^Φ.

Building the constraint graph. Assume that we are given a simple path $\Phi = v_1 \rightarrow v_2 \rightarrow \ldots \rightarrow v_m$ of m nodes along with its k_{max} configuration $[k_{max}(v_1), k_{max}(v_2), \ldots, k_{max}(v_m)]$. We build a constraint graph with m nodes $\{u_1, u_2, \ldots, u_m\}$. We denote the constraint graph as $G^\Phi = (V^\Phi, E^\Phi)$ where V^Φ is its set of nodes and E^Φ is its set of edges. For each pair of nodes u_i and u_j in V^Φ, we draw an undirected edge between them if the following condition holds:

$$j - i \leq \max\{k_{max}(v_i), k_{max}(v_j)\}.$$

Notice that the indices i and j above show the positions of the nodes on the given path Φ. As a result, an edge between u_i and u_j in the constraint graph indicates that v_i and v_j can not be of the same color according to the underlying k_{max} configuration. Consider any coloring instance I that obeys the k_{max} configuration. Let v_i and v_j be any two nodes having the same color in I. Therefore $j - i > \max\{k_{max}(v_i), k_{max}(v_j)\}$.

Therefore, the corresponding u_i and u_j in G^Φ are not adjacent. Hence, I also obeys the constraints of G^Φ. A similar argument can be made in reverse. Thus, each possible coloring of the given path Φ that obeys the k_{max} configuration corresponds to a chromatic coloring of the constraint graph G^Φ and vice versa. Figure 2 shows an example of a path, its k_{max} configuration and the corresponding constraint graph.

Computing the number of colorings. Formally, the value of the chromatic polynomial $A(G^\Phi, m)$ is equal to the number of ways of coloring G^Φ using m colors without any pair of adjacent nodes having the same color. Applying chromatic polynomials on the constraint graph of a path yields the number of possible colorings of that path. We use an edge-contraction recursive rule based on the fundamental reduction theorem.[13] To describe this, we first define two contraction operators on graph G^Φ. The first one removes one edge, (u, v) from the edge set of G^Φ. We denote this with $G^\Phi - (u, v)$. The second one merges two nodes, u and v, into a single node uv. To do this, we insert a new node uv to G^Φ. We also insert an edge between uv and all the nodes which are adjacent to either u or v. We then remove the nodes u and v along with all the edges incident to them. We denote this merge operation with $G^\Phi/\{u, v\}$. Using this notation, the chromatic polynomial is computed using the following recurrence relation

$$A(G^\Phi, m) = A(G^\Phi - (u, v), m) - A(G^\Phi/\{u, v\}, m) \tag{3}$$

Finally, an important question is: which path should we choose to use its corresponding k_{max} configuration as input to our method? Ideally, this path should be the optimal path that we don't know and are looking for. Instead, we use the optimal colorful path we find at each iteration. The main rationale behind this choice is that we expect that the local optimal path of a random coloring instance will have common nodes and edges with the overall optimal path. This is because the optimal path will contain edges with small weights. In Section 4.1 we empirically show that this indeed yields a good approximation to the value of P_s in practice.

Now we are ready to compute the probability of success, P_s, for a coloring instance of our method (i.e, Step 6 of Algorithm 3.1). At each iteration, we first build the constraint graph G^Φ of the best colorful path Φ found at that iteration. We compute the number of chromatic colorings of G^Φ as $A(G^\Phi, m)$ as described above. We then set $N_c = A(G^\Phi, m)$ and compute the probability of success using Equation 1 as $P_s = m!/N_c = m!/A(G^\Phi, m)$.

3.4. *Analysis of the probability of success*

One key question would regarding how we compute the probability of success is: Is it guaranteed to be better than existing methods including Scott *et al.*[4] and Gülsoy *et al.*[6]? In this section, we answer this theoretically. We start by defining a partial order between k_{max} configuration of a paths as follows: Consider two such configurations $\mathbf{x} = [x_1, x_2, \ldots, x_m]$ and $\mathbf{y} = [y_1, y_2, \ldots, y_m]$. We say that $\mathbf{x} \leq \mathbf{y}$ if and only if $\forall_i$, $x_i \leq y_i$.

Proposition 3.1. *Consider two k_{max} configurations $\mathbf{x}$ and $\mathbf{y}$ of two simple paths each having m nodes. Let us denote their corresponding constraint graphs G_x and G_y respectively. If $\mathbf{x} \leq \mathbf{y}$ then $A(G_x, m) \geq A(G_y, m)$.*

We omit detailed proof of Proposition 3.1 due to space limitation. However, briefly it follows from the observation that $\mathbf{x} \leq \mathbf{y}$ implies that every edge in G_x also appears in G_y. However, the opposite may not be true. In other words, G_x has only a subset of the constraints imposed by G_y. Thus, the chromatic polynomial $A(G_x, m)$ cannot be less than $A(G_y, m)$.

Proposition 3.1 has two important implications. First, traditional color coding method (such as Scott *et al.*[4]) computes $N_c = m^m$. This is the most conservative case in our model when the k_{max} configuration is [0, …, 0]. Clearly, this will yield the worst (i.e., largest) possible value for the chromatic polynomial since $[0,\ldots,0] \leq \mathbf{y}$ for any k_{max} configuration $\mathbf{y}$. Second, let t be the smallest k_{max} value among all the nodes in the network. The formulation by Gülsoy *et al.*[6] corresponds to k_{max} configuration is $[t, \ldots, t]$. Let $\mathbf{y}$ be the k_{max} configuration of any m-node path in the same network. We have $[t, \ldots, t] \leq \mathbf{y}$ since all the entries of $\mathbf{y}$ have value t or more. *We conclude from these two implications that our method is guaranteed to produce less or same N_c value as the mentioned existing methods depending on the network topology and the color distribution. Smaller values for N_c implies larger success probability, and thus, faster convergence to the desired confidence value.*

As an example, our method computes the value of N_c for the path shown in Figure 2(a) is 5,760, while Scott *et al.*[4] and Gülsoy *et al.*[6] yield N_c = 46,656 and 18,750 respectively for the same example. According to Equation 1, such a decrease in the value of N_c leads 8.1 and 3.2 times larger success probability than the two above-mentioned methods respectively.

4. Experiments

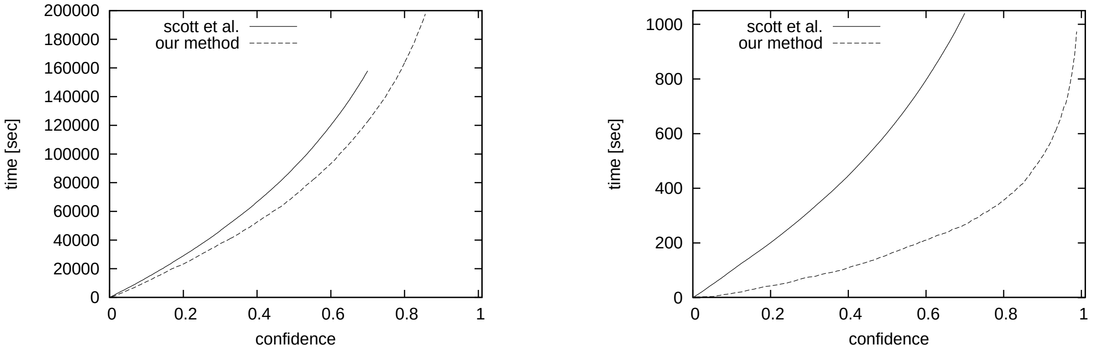
In this section, we evaluate our method on real protein interaction networks. We implemented our method in Java. We ran our experiments on Linux machines with 2.2-GHz dual AMD Opteron dual core processors and 3 GBs of main memory.

Datasets We used the protein interactions of *H. sapiens* and *R. norvegicus* taken from the MINT database.[14] The first one is a large dataset of 15,472 interactions among 6,122 proteins. The second one is a smaller dataset containing 806 interactions among 631 proteins. Each interaction is described by two interacting proteins and a reliability score between 0 and 1 that represents the level of confidence that this interaction exists. MINT calculates reliability scores of interactions from available evidence, such as the size and type of the experiment reporting the interaction, sequence similarity of ortholog proteins.[15]

We use the negative logarithm of MINT reliability scores as edge weights. In all experiments, we find pathways starting within the set of membrane proteins and ending within the set of transcription factors. We use the Gene Ontology database[16] to identify these sets. We identify membrane proteins as the ones annotated with the terms GO:0005886 and GO:0004872, and transcription factors as those with GO:0000988, GO:0001071 and GO:0006351.

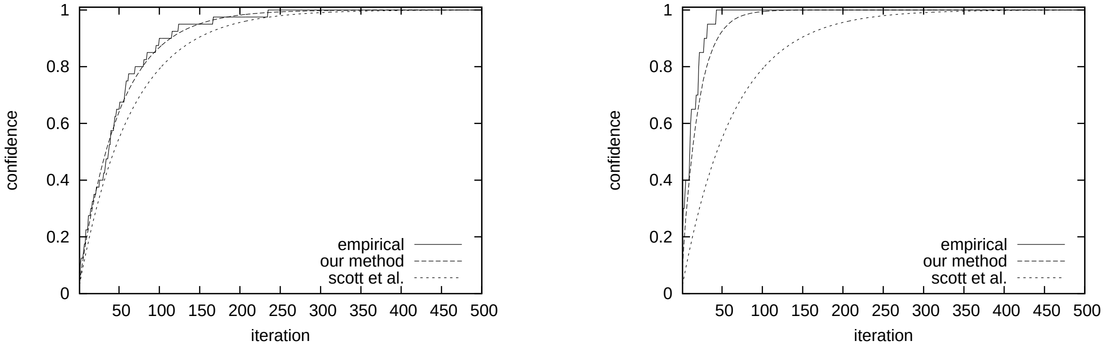
4.1. *Performance assessment*

In Section 3.4, we have already shown theoretically that our method is guaranteed to be at least as fast as the traditional color coding methods. The gap however depends on the topology of the underlying protein interaction network. In this section, we experimentally evaluate how the performance of our method compares to Scott *et al.*[4] as a leading method. We run both methods on our datasets for 500 iterations. We repeat this experiment for pathway lengths = {4, 5, 6, 7, 8, 9}. We measure the total time taken and the confidence value computed by each method at each iteration. We run this process multiple times (at least 20 times) and report the



(a) *H.sapiens* (b) *R.norvegicus*

Fig. 3. Total time needed to achieve a given level of confidence by our method and Scott *et al.* for *H.sapiens* and *R.norvegicus* for path length = 8.



(a) *H.sapiens* (b) *R.norvegicus*

Fig. 4. Confidence level achieved after a given number of iterations using our method and to Scott *et al.* for *H.sapiens* and *R.norvegicus* when path length is fixed at 6. Empirical results denote the fraction of experiments in which the optimal path is found at or before a given iteration.

average of these runs. Below, we report a small subset of these experiments due to page limits.

Figure 3 shows the time it takes to reach to various confidence levels for path length = 8. Our method takes much less time than Scott *et al.* to achieve the same level of confidence. The gap between the two increases as the confidence level increases. We observe that the gap is significantly larger for the *R. norvegicus* dataset. This is mainly because this dataset is more sparse than the other one. As a result, it often produces very dense constraint graphs leading to high success probability values. Scott *et al.* is, on the other hand, oblivious to the density of the network. It produces the same conservative success probability for both datasets. As a result, as we can see in Figure 3, Scott *et al.* can only reach to around 70% confidence for both datasets after 500 iterations. Our method, on the other hand, reports 85% and more than 99% confidence for the *H. sapiens* and *R. norvegicus* datasets respectively after the same number of iterations. The difference between the largest confidence we report for the two datasets can be explained from the density of the two networks. As the network gets sparser, our method tends to gets larger confidence value. In Figure 3(a), we see that our method takes more time to complete

Table 1. Z-scores calculated for the optimal paths found by our method for *H.sapiens* and *R.norvegicus* for different path lengths. Here, μ is the mean of the weight of a random path in the same network with the same length. θ is the weight of the optimal path found by our method. Z is the Z-score of our method.

Dataset	Path Length								
	6			7			8		
	μ	θ	Z	μ	θ	Z	μ	θ	Z
H.sapiens	5.906	0.129	5.409	7.074	0.130	5.477	8.341	0.221	5.764
R.norvegicus	4.975	4.540	0.889	7.307	5.025	1.453	8.457	4.858	1.467

500 iterations than Scott *et al.* This is because it spends additional time to build constraint graph and solve a chromatic polynomial problem. Finally, we observed similar characteristics for other path lengths (results not shown). The main difference was that the performance gap between our method and Scott *et al.* further increases with larger path lengths.

In our next experiment, we evaluate whether our confidence computation is correct in practice. To do this, we computed an empirical confidence as follows. Recall that we repeated each experiment many times. At each iteration we computed the fraction of the experiments in which we were able to find the optimal result as the empirical confidence. Ideally, the theoretical value should not be larger than the empirical one; the closer the two values are the better. Figure 4 shows the empirical confidence value as well as the theoretical confidence value of our method and Scott *et al.*. The results demonstrate that the gap between the empirical results and our method is much smaller than that for Scott *et al.* This is because of the conservative way they use to calculate success probability of an iteration as discussed in section 2. This gap increases as the path length parameter increases (results not shown). Thus, we conclude that both Scott *et al.* and our method produces correct confidence values. Scott *et al.* is, however to conservative, and thus spends too many iterations to reach to the same confidence value.

4.2. *Validation Experiments*

So far we have shown that our method outperforms existing coloring strategies in terms of the running time performance. In this section, we evaluate the biological significance of the paths found using our method. It is worth mentioning that our method returns the same results as Scott *et al.*[4] when both of them are allowed to reach a high confidence value (such as 99% confidence). The main difference is that our method scales to larger networks and longer paths. Therefore, here we will only focus on the results obtained by our method.

4.2.1. *Statistical significance of the results*

In this section we assess the statistical significance of the paths found by our method. We use Z-score to measure statistical significance. Z-score indicates by how many standard deviations our optimal weight is better than the weight of an average random path, so higher values are better. For each dataset and path length m, we run our method to get the path with the minimum weight θ. We then generate 1000 random simple paths of length m, starting at a membrane protein and ending at a transcription factor. We compute the average weight μ of these random paths and their standard deviation σ. We then compute the Z-score as $Z = \frac{\mu - \theta}{\sigma}$.

Table 1 shows the results for *H.sapiens* and *R.norvegicus* for path lengths 6, 7 and 8. Our results are always better than the random paths. Particularly, for the *H.sapiens* network we obtain very significant results. The Z-score for *R.norvegicus* is less. This is mainly because the edge confidence values in this network have much less variation than those in *H.sapiens*. Our Z-score increases with increasing path length. This is not surprising because increasing the size of random selection leads to less chances of the selected path being better or closer to the optimal path. This implies that there is a great potential that methods that scale to large path length will yield important biological insights into signaling pathway identification.

4.2.2. *Biological significance of the results*

Another important question is: how biologically significant are our results? To answer this question, we validate our results using functional enrichment. We use the Gene Ontology[16] to compute functional enrichment of paths found at different iterations of our method. Let Φ be the path being tested, T be the universal set of GO terms, m be the path length, M be the total number of proteins in the dataset, G_i be the total number of proteins annotated with the Go term t_i in the dataset, and g_i be the number of proteins annotated with t_i in Φ. We compute functional enrichment of Φ as $\min_{t_i \in T} P(X \geq g_i | M, m, G_i)$ where X is a random variable under a hyper-geometric distribution with these parameters. Lower enrichment values indicate paths with common functions, and thus they are better.

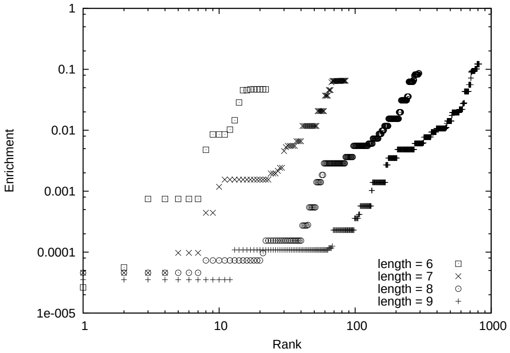


Fig. 5. Functional enrichment of best colorful paths found at different iterations of our method for *R.norvegicus* in sorted order. Smaller values are better.

Figure 5 plots the functional enrichment value of the best colorful paths found at different iterations of our algorithm in sorted order for the *R.norvegicus* network. We omit results for *H. sapiens* as it is very similar to those in Figure 5. We observe that as the distribution of the enrichment values follows power-law distribution. That is only a minority of the observed paths have very good enrichment while the majority tend to have bad ones. We observe that this behavior is consistent for all path lengths we tested. This suggests the following: (i) There can be multiple biologically interesting paths for the same start and end node sets. (ii) We need to have sufficiently high confidence in the result to avoid biologically meaningless paths since the enrichment drops quickly. (iii) Even long paths can be highly enriched. All of these observations show the importance of improving the running time performance of pathway discovery methods, and hence the importance of our contribution.

Next, we focus on a few of the most functionally enriched pathways our method finds on the *H. sapiens* network. Figure 6 shows three examples each having length of six. All the six genes in the path in Figure 6(a) regulate epidermal growth factor receptor signaling pathway. Among these the leftmost four genes appear in the ErbB signaling pathway. They also affect the development of various cancer types such as chronic myeloid leukemia, glioma and prostate cancer. In Figure 6(b), all the six genes are ephrin receptor binding. They affect cell growth and development and thus participate in cancer development. The five leftmost genes in Figure 6(c) negatively regulate the epidermal growth factor receptor signaling pathway. Notice that all the three pathways in this example overlap with each other, yet they also contain several genes that do not exist in others. For instance, the pathway in Figure 6(b) contains SRC unlike the other. SRC takes part in same pathways as most of the other genes in this figure, such as the ErbB signaling pathway. Thus, all of these significant paths reported by our method reveal different parts of the signaling networks through alternative paths.

5. Conclusion

In this paper, we presented an enhanced color-coding technique. We presented a novel way to calculate success probability for a single coloring iteration. We explained how to calculate the number of coloring possibilities for a path with a given k_{max} configuration. We also discussed the relation between configurations with different k_{max} values. We used the enhanced color-coding technique to find signaling pathways in protein interaction networks. We empirically showed that our method produces correct results, and that it needs less time than the leading method to produce these results. We also showed that the results of our method are of statistical and biological significance. Possible future extensions to the present work include extracting deregulated signaling pathways using a cancer gene expression dataset. The subject PPI network could be built from mRNA co-expression and high-throughput experiments.

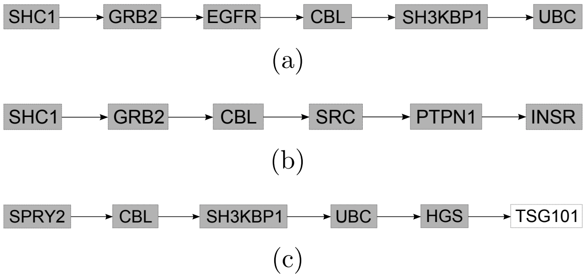


Fig. 6. Three sample pathways with functional enrichment value less than 10^{-11} found by our method in the *H.sapiens* dataset. The shaded nodes correspond to the genes which have common gene ontology term leading to the best functional enrichment. (a) The common term is GO:0042058. (b) The common term is GO:0046875.(c) The common term is GO:0042059.

References

1. B. Schwikowski, P. Uetz and S. Fields, *Nature Biotechnology* **18**, 1257 (December 2000).
2. P. Uetz, L. Giot and G. Cagney *et al.*, *Nature* **403**, 623 (February 2000).
3. B. P. Kelley, R. Sharan and R. M. Karp *et al.*, *PNAS* **100**, 11394 (September 2003).
4. J. Scott, T. Ideker, R. M. Karp and R. Sharan, Efficient algorithms for detecting signaling pathways in protein interaction networks, in *RECOMB*, (Springer-Verlag, Berlin, Heidelberg, 2005).
5. N. Alon, R. Yuster and U. Zwick, *J. ACM* , 844 (1995).
6. G. Gülsoy, B. Gandhi and T. Kahveci, Topology aware coloring of gene regulatory networks, in *ACM BCB*, (ACM, New York, NY, USA, 2011).
7. T. Shlomi, D. Segal, E. Ruppin and R. Sharan, *BMC Bioinformatics* **7**, p. 199 (2006).
8. S. Lu, F. Zhang, J. Chen and S.-H. Sze, *Algorithmica* **48**, 363 (August 2007).
9. M. Steffen, A. Petti and J. Aach *et al.*, *BMC Bioinformatics* **3**, p. 34 (2002).
10. G. Bebek and J. Yang, *BMC Bioinformatics* **8**, p. 335 (2007).
11. A. Gitter, J. Klein-Seetharaman, A. Gupta and Z. Bar-Joseph, *NAR* **39**, p. e22 (2011).
12. B. Dost, T. Shlomi and N. G. *et al.*, *Journal of Computational Biology* **15**, 913 (2008).
13. F. Dong, K. Koh and K. Teo, *Chromatic Polynomials And Chromaticity of Graphs* (World Scientific Pub., 2005).
14. A. Chatr-aryamontri, A. Ceol and L. M.-P. *et al.*, *Nucleic Acids Research* **35**, 572 (2007).
15. A. Ceol, A. Chatr Aryamontri and L. Licata *et al.*, *Nucleic Acids Research* **38**, D532 (2010).
16. M. Ashburner, C. A. Ball and J. A. Blake *et al.*, *Nature genetics* **25**, 25 (May 2000).