In this section, we start by properly formulating the problem and defining common terms that we use in our methods. We then present new thoughts about pathway detection using color coding. We study the network topology and how we can involve it in our calculation to obtain a better success probability, and hence needing less number of iterations and improving performance. Last, we present an enhanced color-coding method for detecting pathways in protein interaction networks.

Problem formulation

Given a graph G = (V, E, w), where V is its set of nodes, E is its set of edges and w is the edge-weight function, and given a set S \subset V and a path length m. Assume Pi is the set of all simple paths of length m starting at any node s \in S and ending at node i. Our goal is to find, for each node i, the path p \in Pi whose sum of edge weights is minimum.

The following are the definition of some commonly used terms:

* k neighborhood. For a given node v \in V and an integer k, the k neighborhood of v is a set of nodes U \subset V where a node u \in U if and only if u can be reached from v in k hops or less.
* max\_k. For a given node v \in V, max\_k(v) is the maximal value of k such that \forall u \in k-neighborhood of v, the color assigned to u is not equal to the color assigned to v.
* max\_k configuration. For a given path P, the max\_k configuration of P is the sequence of max\_k values corresponding to the sequence of nodes in P.

Based on the work presented by Scott et al.\cite{scott}, a generic color-coding approach to solving the problem consists of three main steps. The first step is coloring the network; \foreach v \in V we independently select a color drawn uniformly at random from a set of m different colors. The second step is finding an optimal colorful path; \forall v \in V we want to find the minimum-weight colorful path of length m starting in S and ending at v, and then we extract the minimum of these paths. The third step is calculating the success probability Ps; we calculate a lower bound on the probability that the unknown overall optimal path is indeed colorful, hence the probability that it is indeed the optimal colorful one we found. These three steps are repeated r times until 1 – Pi(1:r) 1 – Psi >= E, where E is a required confidence level. It is obvious that a higher success probability would result in less number of iterations required, hence less execution time.

Calculating success probability in general is an obvious counting problem. Ps = m!/Nc where m! is the number of coloring possibilities in which the path is colorful, and Nc denotes the total number of coloring possibilities for the path. Scott et al.\cite{scott} calculated Nc as equal to mm. This calculation considered no restrictions on the color selection of each node, and is discarding available knowledge of the network topology and colors already assigned to its nodes.

As guided by Gulsoy et al.\cite{gulsoy}, knowing the network topology can be useful in calculating success probability, specifically the number of coloring possibilities of the optimal path. \forall v \in V, \forall u \in k neighborhood of v, if the color of u is not equal to that of v, then the number of coloring possibilities can be calculated as follows: Nc <= (m – k)m – k Pi(0:k-1) (m – i) < mm \forall k > 0 which results in a lower bound on success probability: Ps >= m!/ (m – k)m – k Pi(0:k-1) (m – i) > m!/mm \forall k > 0 However, according to this scheme, the node with a minimum value of max\_k dominates the whole network, which produces a correct but very conservative lower bound on success probability.

Our approach relies on individual max\_k values of all nodes in an optimal path. Assuming knowledge of the max\_k configuration of the optimal path, we use it to calculate Nc under the restrictions induced by these values. We also assume that each node in the path is not connected to any other nodes except the ones before and after it in the path. This assumption is valid because any more connections will only induce more coloring restrictions and hence cause Nc to decrease; therefore we get a solid upper bound on Nc, hence a solid lower bound on Ps. For a given node v in a given path, all max\_k(v) nodes in either direction from v are not allowed to have the same color as v. We represent this rule as an unweighted constraint graph W = (H, L) where H is the set of nodes and L is the set of edges. H contains a node corresponding to each node in the path, and L contains an edge for each pair of nodes that are not allowed to have the same color, according to the aforementioned rule. Fig 1 shows an example of a path, its max\_k configuration and the corresponding constraint graph W. The problem now translates to calculating the value of the chromatic Polynomial P(W, m): the number of ways of coloring W using m colors without any pair of adjacent nodes having the same color. We calculate this value using the following edge-contraction recursive rule based on the fundamental reduction theorem\cite{dong}:

According to this method, the value of Nc for the example path shown in Fig 1a is 5,760, while Scott et al.\cite{scott} and gulsoy et al.\cite{gulsoy} would yield Nc = 46,656 and 18,750 for the same example. Such a decrease in the value of Nc leads to an increase in the value of Ps.