Analysis of Biological Networks: Protein modules – Color Coding*

Lecturer: Eithan Hirsh Scribe: Shelly Mahleb and Benny Davidovich

Lecture 6, November 30, 2006

1 Introduction

In this lecture we present:

- Color coding as a combinatorial technique for finding modules. The algorithm looks for simple paths, cycles, and other small subgraphs of specified size, within a given graph [1].
- The applications of color coding to analyzing the protein-protein interaction network of yeast [3].
- A greedy based technique for identifying protein complexes by searching for significantly dense subgraphs in a given graph [2].

2 Color Coding

Color Coding is a method for finding in a given graph $G = (V, E)$ a simple path or cycle of a specified length k. Its running time is exponential in k and linear in the size of the graph. In the following discussion we will focus on path finding.
The problem of finding a simple path of a specified length k is NP- hard, by reduction from the Hamilton path problem. A trivial algorithm will solve it in $O(n^k)$, where n denotes the number of vertices in the graph. Our goal is to show a *fixed parameter* algorithm:

Definition 1 *a problem is fixed parameter tractable **(FPT)** if we can isolate one of its parameters k (so it won't be a part of the input), such that there exists an algorithm for it whose running time is $f(k)n^{O(1)}$, where n is the input size.*

2.1 Random Orientations

The difficulty of finding a simple path of length k in a given graph G is NP-hard as a result of requiring the path to be *simple*. Otherwise it is possible to find all paths of length k by raising the adjacency matrix A to the k-th power, but most of those paths will not be simple. One way of avoiding non simple paths is by choosing a random orientation in G; this can be done by directing the graph in the following way:

- choose a random permutation $\pi : V \to \{1, .., |V|\}$
- direct each edge $(u, v) \in E$ from u to v if and only if $\pi(u) < \pi(v)$.

*Based on last's year scribe, written by Anat Kreimer and Yariv Eisenberg

Every directed path of length k in $\overrightarrow{G}$ is simple and corresponds to a simple path of length k in G.
Every simple path of length k in G, on the other hand, has a $\frac{(k+1)!}{2}$ chance of becoming a directed path in $\overrightarrow{G}$ ($\frac{(k+1)!}{2}$ in each direction).
This yields the following theorem:

Theorem 1 *A simple undirected path of length k in a graph $G = (V, E)$ that contains such a path can be found in $O((k+2)!|V|)$ expected time.*

Proof: An algorithm with $O((k+1)! \cdot |E|)$ expected time is immediate. Simply choose a random acyclically oriented version $\overrightarrow{G}$ of G and find the longest directed path in it. This can easily be done in $O(|E|)$ time. The longest path in $\overrightarrow{G}$ would be of length at least k with a probability of at least $\frac{2}{(k+1)!}$. If the longest path is of length less than k, repeat the process. The expected number of times this process is repeated before the desired path is found is at most $\frac{(k+1)!}{2}$. To reduce the $O((k+1)! \cdot |E|)$ complexity to the desired $O((k+2)!|V|)$ we use the well known fact that every graph with $|V|$ vertices and at least $k \cdot |V|$ edges contains a path of length k. This fact easily supplies a method of finding such a path in $O(k \cdot |V|)$ time. A specific way of incorporating this into the algorithm is as follows: start a DFS (depth-first search) on the graph. If a vertex of depth k is ever found, stop and output the path from the root to this vertex. If no such vertex is found, the graph contains at most $k \cdot |V|$ edges (as all back-edges point to ancestors) and we may apply the algorithm described above. ■

2.2 Random Coloring

The method of color coding is done by choosing a random coloring of the vertices of G with k colors (see Figure 1). A path in G is said to be *colorful* if each vertex on it is colored by a distinct color. Thus, a colorful path in G is clearly simple; that means that in order to find a simple path of specified length k we need to find in the random colored by k colors graph G a *colorful* path of length k. In order to analyze the complexity of finding such path, we will use the following lemma:

Lemma 2 *Let $G = (V, E)$ be a directed or undirected graph and let $c : V \to \{1, ..., k\}$ be a coloring of its vertices with k colors. A colorful path of length $k-1$ in G, if one exists, can be found in $2^{O(k)} \cdot |E|$ worst-case time.*

Proof: We describe at first an $2^{O(k)}|E|$ time algorithm that receives as input the graph $G = (V, E)$, the coloring $c : V \to \{1, ..., k\}$ and a vertex $s \in V$, and finds a colorful path of length k that starts at s, if one exists. A colorful path of length k that starts at some specified vertex s is found using a Dynamic programming approach: suppose that for each vertex $v \in V$ the possible sets of colors on colorful paths of length i that connect s and v was found. Note that there is no record of all the colorful paths connecting s and v, only the record of color sets appearing on such paths. Therefore, for each vertex v there is a collection of at most $\begin{pmatrix} k \\ i \end{pmatrix}$ color sets. Then inspect every subset C that belongs to the collection of v, and every edge $(v, u) \in E$. If $c(u) \notin C$, the set $C \cup \{c(u)\}$ is added to the collection of u that corresponds to colorful paths of length $i+1$.
The graph G contains a colorful path of length k with respect to the coloring c if and only if the final collection, that corresponding to paths of length k, of at least one vertex is non-empty. The number of operations performed by the algorithm outlined is at most

$$O(\sum_{i=1}^{n} i \begin{pmatrix} k \\ i \end{pmatrix} \cdot |E|) \tag{1}$$

which is clearly $O(k2^k)|E|)$.
To find a colorful path of length k in G that starts somewhere, just add a new vertex s' to V, color it with a new color 0 and connect it with edges to all the vertices of V. Now look for a colorful path of length $k+1$ that starts at s'.

Furthermore, each simple path of length $k-1$ has a chance of $\frac{k!}{k^k} > e^{-k}$ to become colorful. $k!$ is the number of possible permutations of k colors, and k^k is the number of possibilities of coloring the vertices. Therefore, the probability of failure is $(1-\frac{1}{e^k})^t$, where t is the number of iterations. We would like that the value of the failure probability would not exceed a certain value denoted as ϵ:

$$P(fail) = (1-\frac{1}{e^k})^t < \epsilon \tag{2}$$

implying that $t = e^k ln\frac{1}{\epsilon}$. Hence, the overall running time is $2^{O(k)}|E|$. ■

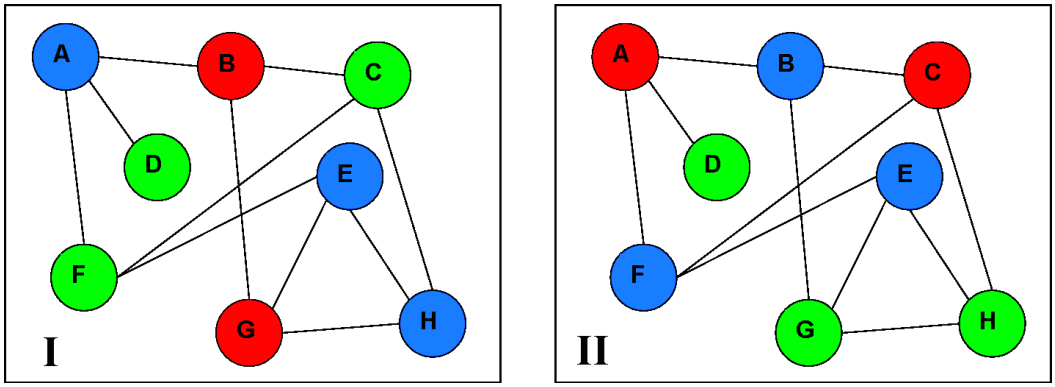

Figure 1: Coloring example:two different colorings on toy graph, k=3. In coloring I: P(A,RGB) is built $C- > BC- > ABC$. In coloring II: P(A,RGB) is built $G- > BG- > ABG$. ABC is not colorful in coloring II. Where R is red, G is green, B is blue.

3 Biologically Motivated Constrains

Until now we have described how the color coding method finds a simple path of length k in a given graph G. When implementing this method in a biological network in order to find protein - protein interactions in a biological path, (i.e. signal transduction), there are several constrains that have to be considered. Scott *et al.* presented biologically-motivated extensions of the basic technique [3]. The weight of each on each edge (u,v) in a path p should be $-\lg P(u,v)$, where $P(u,v)$ is the probability of those two proteins denoted as u and v to interact. The score of each path then will be : $-\sum_{(u,v)\in E} \lg P(u,v)$, and the length of a path is the number of vertices it contains. There can be sets of starting or ending proteins in the path(i.e.,finding the cascades that begin from membrane receptors). In the following discussion, we will denote the set of possible starting proteins as I. One can force a certain protein to be included in the path by giving it a unique color. In addition , using counters, we can limit the number of proteins of a given type. This extension multiplies the storage requirement and running time of each trail by a constant. Several counters can be added to enforce different constrains, each multiplies the time and storage requirement by a constant factor and does not affect the probability that the optimal path is colorful in any given trail.[3].

3.1 The Path Recursion

The biological problem can be described algorithmically as follows: Given an undirected weighted graph $G = (V, E, w)$ with n vertices, m edges and a set I of start vertices, we wish to find, for each vertex v, a minimum-weight simple path of length k that starts within I and ends at v. If no such simple path exists, the algorithm should report this fact. There is a dynamic programming algorithm for the problem ,the running time of this algorithm is $O(kn^k)$ and its space requirement is $O(n^k)$.

It can be improved by Color Coding. This problem can be solved using the following dynamic programming algorithm: for each nonempty set $S \in 1, 2, ..., k$ and each vertex v such that $c(v) \in S$, let W(v, S) be the minimum weight of a simple path of length $|S|$ that starts within I, visits a vertex of each color in S, and ends at v. This function can be tabulated (again, in increasing order of the cardinality of S) using the following recurrence:

$$W(v, S) = \min_{u:c(u)\in(S\setminus c(v))} W(u, S \setminus \{c(v)\}) + \omega(u, v), |S| > 1 \tag{3}$$

where the base of the recursion is: W(v, {c(v)}) = 0 if $v \in I$ and ∞ otherwise.
The weight of a minimum-weight colorful path ending at v is W(v, {1, 2, . . . , k}). For every vertex v, each trial yields a simple path of length k starting within I and ending at v, which is optimal among all the paths that are colorful under the random coloring in that trial. The running time of each trial is $O(2^k k \cdot |E|)$ and the storage requirement is O($2^k \cdot |V|$).

3.2 Segmented Pathways

Many signaling pathways start from membrane proteins and end at transcription factors in the nucleus, proceeding monotonically from outer to inner compartments of the cell. Therefore, The cell can be segmented from the membrane to the nucleus, labeling each segment in a non-decreasing order.

3.2.1 Unique Labeling

Assuming that each segment contains proteins of a particular class, and each protein assigned to exactly one class. The Mathematical framing and course of the algorithm: subject to this restriction find, for each vertex v, a minimum-weight simple path of length k from some vertex in I to v. The segments are numbered successively in the order of their occurrence along the desired path, Depending on biological information . Each protein u is assigned an integer label $L(u)$ which uniquely specifies the segment in which the protein may occur. The requirement is that the labels of the proteins along the path form a monotonically non-decreasing sequence. Such a path is called *monotonic*.

In each trial, assign each vertex a color drawn uniformly at random from {1, 2, . . . , k}. Since each vertex is restricted to a unique segment, the path will be simple provided that vertices in the same segment are assigned to different colors. For a vertex v and a subset of the colors $S \in \{c(v)\}$, W(v, S, k) is defined as the minimum weight of a simple monotonic path of length k from I to v, in which no two vertices with the same label have the same color, and the set of colors assigned to vertices with label $L(v)$ is S. The algorithm should be modified as follows:

$$W(v, \{c(v)\}, l) = \min_{u:L(u)<L(v)} \min_{S} W(u, S, l-1) + w(u, v), l > 1 \tag{4}$$

$$W(v, S, l) = \min_{u:L(u)=L(v),c(u)\in S\setminus\{c(v)\}} W(u, S \setminus \{c(v)\}, l-1) + w(u, v), 1 < |S| \leq l \tag{5}$$

Where the base of the induction:
$W(v, \{c(v)\}, 1) = 0$ if $v \in I$ and ∞ otherwise.

Time Complexity: suppose there are at most h vertices in each segment. Then each trial has a running time of $O(2^h h \cdot k \cdot |E|)$ and a storage requirement of $O(2^h \cdot k \cdot |V|)$. The probability that all vertices within the same segment will be colored by distinct colors is e^{-h} (as shown in 2.1). Thus the expected number of trails needed to discover an optimal segmented pathway with at most h proteins per segment is of order e^h which is much smaller than e^k.

3.2.2 Interval Constraints

It may be unrealistic to assume that every protein can be assigned a priori to a unique segment. Instead, as described in Scott *et.al*, we can assume that for each protein there is a lower bound $L_1(u)$ and an upper bound $L_2(u)$ on the number of its segment. (an example of a segmented path is given in Figure 2 ; if on the second path in the fifth cycle the caption was [2,2] it wouldn't be monotonic). A path $(u_1, ..., u_k)$ is *consistent with segmentation* if it is possible to assign to each protein u_i a segment number s_i such that the sequence of segment numbers along the path is monotonically non-decreasing and, for each i, $L_1(u_i) \leq s_i \leq L_2(u_i)$. The reformulation this condition is as follows: For any path P, let $s(P)$ be the maximum of $L_1(u)$, over all proteins u in P. Then the path $(u_1, ..., u_k)$ is consistent with segmentation if and only if, for all i, $L_2(u_i) \geq s(u_1, ..., u_{i-1})$.

Let each vertex u be assigned a color $c(u)$ drawn uniformly at random from $\{1, 2, ..., k\}$. For each vertex v, the color coding method seeks a minimum-weight path of length k from I to v which is both colorful and consistent with segmentation.

Define $W(v, s, S)$, where $L_1(v) \leq s \leq L_2(v)$, as the minimum weight of a simple path P of length $|S|$ from I to v that is consistent with the segmentation, such that $s(P) = s$ and S is the set of colors assigned to the vertices in P. The following dynamic programming recurrence is obtained:

$$W(v, L_1(v), S) = \min_{u:c(u)\in(S\setminus\{c(v)\})s'\leq L_1(v)} \min W(u, s', S \setminus \{c(v)\}) + w(u, v), |S| > 1 \tag{6}$$

$$W(v, s, S) = \min_{u:c(u)\in(S\setminus\{c(v)\})} W(u, s, S \setminus \{c(v)\}) + w(u, v), L_1(v) < s \leq L_2(v), |S| > 1 \tag{7}$$

Where the base of the induction is: $W(v, L_1(v), \{c(v)\}) = 0$ if $v \in I$ and ∞ otherwise.
Time Complexity: The weight of a minimum-weight colorful path ending at v and consistent with the segmentation is $min_s W(v, s, \{1, 2, ..., k\})$. The running time per trial is $O(2^k k \cdot t \cdot |E|)$ and the storage requirement is $O(2^k t \cdot |V|)$, where t is the number of ordered segments.

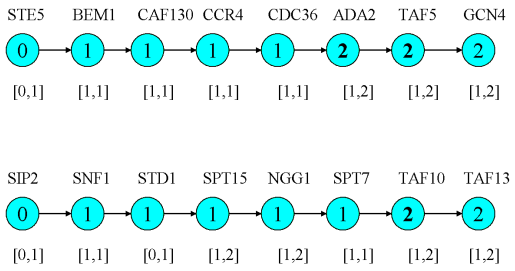

Figure 2: Segmented path examples.

3.3 More General Structures

In general, signaling pathways need not consist of a single linear path. In Scott *et al.* it is demonstrated that the color coding method can be used to find high-scoring signaling pathways with a more general structure [3].

3.3.1 Rooted Trees

Let G = (V,E) be a weighted graph with $I \subseteq V$, and let k be a positive integer. We wish to find, for each vertex v a tree of minimum weight among all k-vertex subtrees in G that are rooted at v and in which every leaf is an element of I. In each color coding trial, each vertex u assigned to a color drawn uniformly at random from {1, 2, . . . , k}. For $v \in V$ and $\{c(v)\} \subseteq S \subseteq \{1, 2, ..., k\}$, let W(v, S) be the minimum weight of a subtree with $|S|$ vertices that is rooted at v, contains a vertex of each color in S, and whose leaves are in I. The following recurrence can be used to compute W(v, S):

$$W(v, S) = \min\{\min_{u:c(u)\in(S\setminus\{c(v)\})} W(u, S \setminus \{c(v)\}) + w(u, v), \tag{8}$$

$$\min_{(S_1,S_2):S_1\cap S_2=\{c(v)\},S_1\cup S_2=S} W(v, S_1) + W(v, S_2)\} \tag{9}$$

Where the base of the induction is: $W(v, \{c(v)\}) = 0$ if $v \in I$ and ∞ otherwise.

The running time for a trial is $O(3^k km)$ and the storage required is $O(2^k n)$. The factor of 3^k in the running time comes from the second line of the recurrence, which examines for each subset of S all possible bipartitions. This is equivalent to 3^k, the number of ways to divide k colors into 3 distinct groups.

3.3.2 Two-terminal series-parallel graphs

The definition of a two-terminal series-parallel graph (2SPG) (see Figure 3) is recursive:
Base case: a graph with two vertices u and v connected by an edge is a 2SPG between terminals u and v.
Series connection: if G_1 is 2SPG between u and v, G_2 is a 2SPG between v and w and G_1 and G_2 have no vertices in common except v, then $G1 \cup G2$ is a 2SPG between u and w. Parallel connection is defined as follows: if $G1$ and $G2$ are 2SPGs between u and v, and they have no vertices in common except u and v, then $G_1 \cup G_2$ is a 2SPG between u and v.

T he goal is to find, for each vertex v, a minimum-weight k-vertex 2SPG between some vertex in I and v. The recursion is: let W(u, v, S) be the minimum weight of a 2SPG between u and v with $|S|$ vertices in which the set of colors occurring is S. Then, following the recursive definition of a 2SPG they obtain:

$$W(u, v, S) = \min\{\min_{w,S_1,S_2:S_1\cup S_2=S,S_1\cap S_2=\{c(w)\}} W(u, w, S_1)+W(w, v, S_2), \min_{T_1,T_2:T_1\cap T_2=\{c(u),c(v)\},T_1\cup T_2=S} \tag{10}$$

$W(u, v, T_1) + W(u, v, T_2)\}$

Where $W(u, v, \{c(u), c(v)\}) = w(u, v)\ for\ every\ edge\ (u, v)$.

The execution time of a trial is $O(3^k kn^2)$ and the storage requirement is $O(2^k n^2)$.
In Figure 4 we can see a comparison of the running time of all the types of paths that were mentioned so far.

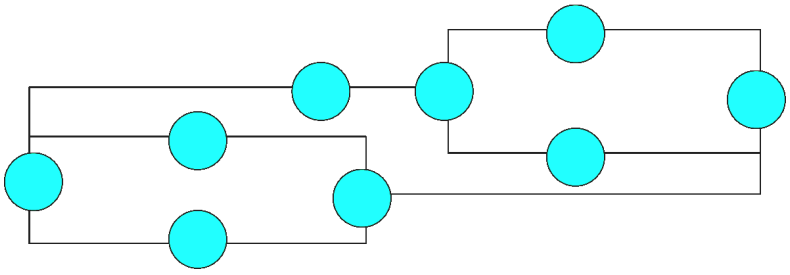

Figure 3: Two-terminal series-parallel graphs example.

Type	DP Time	Space
Path	$O(2^k km)$	$O(2^k n)$
Segmented Path	$O(2^k kml)$	$O(2^k nl)$
2Terminal SPG	$O(3^k kn^2)$	$O(2^k n^2)$
Rooted Tree	$O(3^k km)$	$O(2^k n)$

Figure 4: Summary of the complexity of identifying certain graph classes using color coding.

4 Applications to Yeast Protein Interaction Network

In Scott *et al.* the color coding method was implemented for finding both simple and rooted trees in the yeast protein interaction network [3]. Protein-protein interaction data were obtained from the Database of Interaction of proteins and consisted of 14,319 interactions among 4,389 proteins in yeast, when varying the desired path length, the probability of success, and the number of required paths. In all those runs, the paths were constrained to start at a membrane protein and end at a transcription factor, and rooted trees were constrained to be rooted at a membrane protein and have all leaf nodes be transcription factors . As a first test, the algorithm was applied to compute optimal paths and trees of size 8.Figure 5 presents a benchmark of the running time of various path lengths in this network.

Path length	Success probability	#Paths	Time (sec)
10	99.9%	100	11650
9	99.9%	100	2149
8	99.9%	500	498
8	99.9%	300	460
8	99.9%	100	435
8	90%	100	303
8	70%	100	269
8	50%	100	257
7	99.9%	100	97
6	99.9%	100	32

Figure 5: Source: [3]. Running times of the path finding algorithm for different parameter settings.

4.1 Validation Techniques

The following statistical methods were:

- **Weight p-value:** given a subnetwork with weight w, its weight p-value is defined as the percent of top-scoring subnetworks in random networks (computed, in all cases, using the same algorithm and parameters that are applied to the real networksee below) that have weight w or lower, where random networks are constructed by shuffling the edges and weights of the original network, preserving vertex degrees. Edges whose weights differ by more than a factor of two are not shuffled in order to approximately preserve, for each vertex, the sum of the weights of the edges touching it.

- **Functional enrichment:** to evaluate the functional enrichment of a subnetwork N its proteins associated with known Biological Processes using the Gene Ontology (GO) annotations. The tendency of the proteins then was computed in order to have a common annotation. The scoring is done as follows: define a protein to be below a GO term t, if it is associated with t or any other term that is a descendant of t in the GO hierarchy. For each GO term t with at least one protein assigned to it, they compute a hypergeometric p-value based on the following quantities: (1) The number of proteins in N that are below t; (2) the total number of proteins below t; (3) the number of proteins in N that are below all parents of t; and (4) the total number of proteins below all parents of t. The p-value is further corrected for multiple testing.

- **Searching for known pathways in the yeast:** The presented work concentrated on three MAPK signal transduction pathways: pheromone response, filamentous growth and cell wall integrity. For each of the pathways the network was searched for paths of lengths 6-10 using the pathways endpoints to define the start and end vertices. In all cases the results matched the known pathways well.

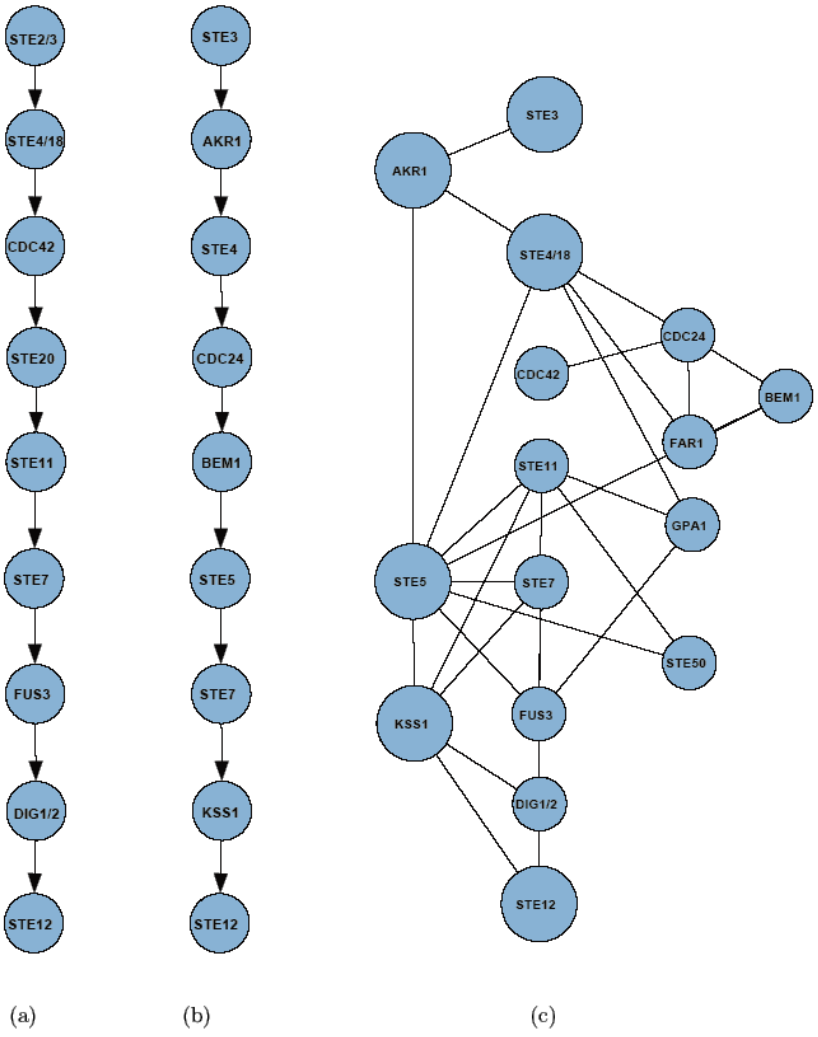

Figure 6: Source: [3]. The pheromone response signaling pathway in yeast. (a) The main chain of the known pathway.(b) The best path of the same length (9) in the network. (c) The assembly of all light-weight paths starting at STE3 and ending at STE12 that were identified in the network. Nodes that occur in at least half of the paths are drawn larger than the rest. Nodes that occur in less than 10% of the paths are omitted.

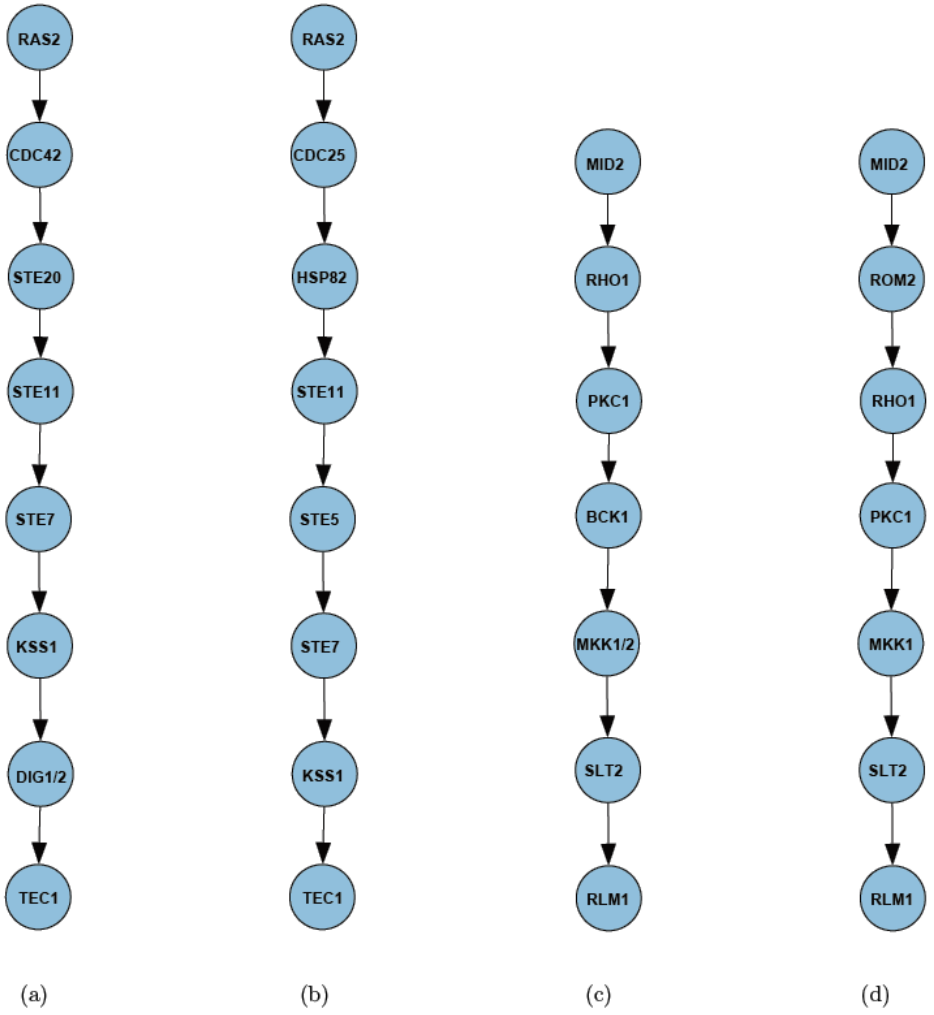

Figure 7: Source: [3]. Search results for the filamentous growth and cell wall integrity pathways. (a) The known filamentous growth pathway. (b) The best path of length 8 between RAS2 and TEC1. (c) The known cell wall integrity pathway. (d) The best path of length 7 between MID2 and RLM1.

4.2 Results

In this work the color-coding algorithm was implemented in order to simple paths and rooted trees in graph based on several biologically motivated extensions in protein interaction pathway in yeast protein network. In the pheromone response, pathway in which the yeast cells preparing for mating by inducing polarized cell growth toward a mating partner, cell cycle arrest in G1 , and increased expression of proteins needed for cell adhesion, cell fusion and nuclear fusion. This is shown in Figure 6a. Looking for the optimal path of length 9 in the yeast network yields the path shown in Figure 6b. The path mainly consist of of proteins from the pheromone response pathway. The aggregate of all the paths that the algorithm computed between Ste3p and Ste12p, across a range of lengths (6-10), is shown in Figure 6c.
All the proteins that were identified are part of the pathway, except Kss1p and Akr1p. In a similar work that was done by Steffen *et al.*(2002)[4] there were similar result, except three additional proteins that were found that are related to the pathway but not part of the main chain.

One other pathway that was tested is the filamentous growth pathway. This pathway is induced under stress conditions and causes yeast diploid cells to grow as filaments of connected cells. This pathway is shown in Figure 7a. Searching for a minimum-weight path of the same length (8), yields the path shown in Figure 7b, which is almost matching the known pathway.The interaction of Cdc25p to Hsp82p is an artifact due to a missing link between Ras2p and Cdc42p in the network data. The cell wall integrity pathway that mediates cell cycle regulated cell wall synthesis is presented in Figure 7c. The path that was found using the algorithm again matches the known path, and shown in Figure 7d. There is one falsely detected protein, Rom2p, and that can be a result of the fact that the network does not contain a direct interaction between Mid2p and Rho1p.

In addition, The algorithm was used in order to find the high osmolarity MAPK pathway. For this run, the exact known pathway was recovered, though it was the 11-th-scoring among 64 identified paths. To prevent a large number of minor variations on a small set of high-scoring subnetworks that could dominate the experiment's results, a heap structure was used to filter subnetworks with more than 70% of their proteins in common (the one of lowest weight was retained, and all others removed). With success probability set to 99.9%, the 100 best subnetworks of each type (rooted trees and simple paths) were recorded, and its weight p-value and functional enrichment were evaluated. The results are depicted in Figures 8 and 9.

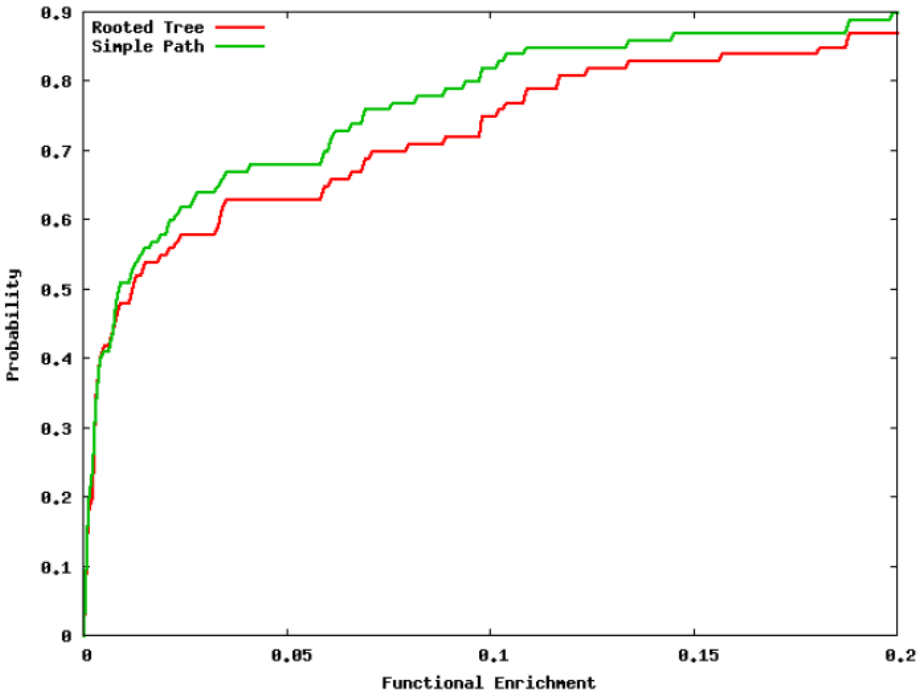

Figure 8: Source: [3]. Cumulative distributions of functional enrichment p-values. x-axis: p-value. y-axis: Percent of paths or trees with p-value x or better.

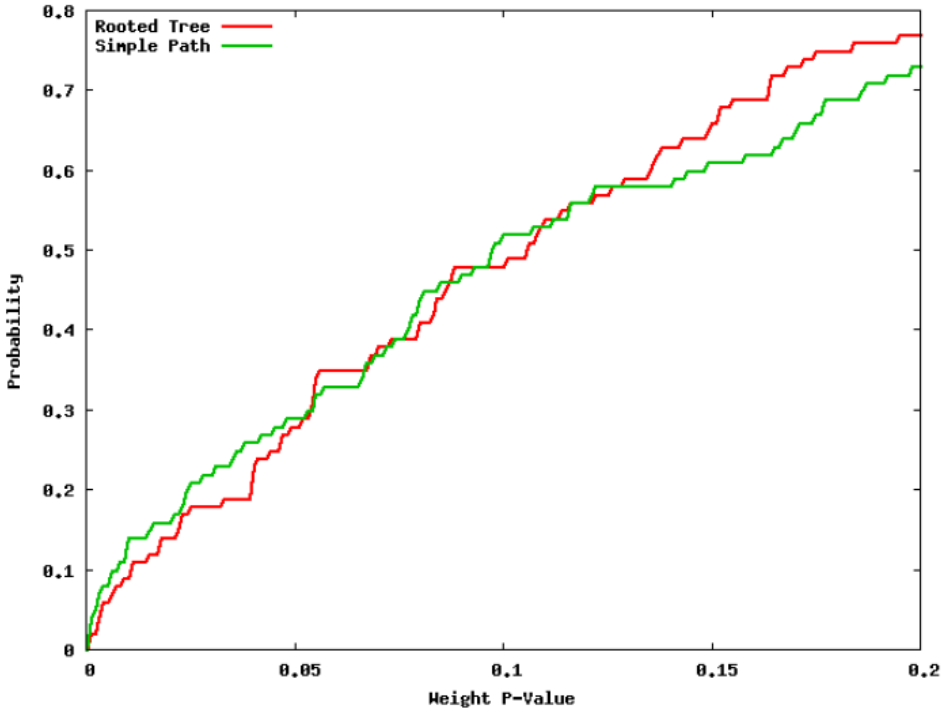

Figure 9: Source: [3]. Cumulative distributions of weight p-values. x-axis: p-value. y-axis: Percent of paths or trees with p-value x or better.

4.2.1 Comparison of Running Times

Table 15 presents a benchmark of the running time of the path search across a network with 4,389 nodes and 14,319 edges, when varying the desired path length, the probability of success and the number of required paths. The color coding algorithm runs in minutes when searching for a path of length 8 with success probability of 99.9%, and in just over three hours when searching for a path of length 10. The running time for exhaustive search was approximately 1.5-fold lower for length-7 paths (69 seconds), 2-fold higher for length-8 paths (866 seconds), and 7-fold higher for length-9 paths (15,120 seconds). See Figure 10 for comparison of the running times. Note that because the runtime for exhaustive search is proportional to the number of subnetworks to be examined, the difference in runtime between the color coding and the exhaustive search algorithms grows much wider when the endpoints of the paths are not constrained, or when looking for more general structures such as trees.

Path length	Color coding	Exhaustive
8	435	866
9	2,149	15,120
10	11,650	--

Figure 10: Source: [3]. Comparison of running times between exhaustive and color coding search : running time of the path search across a network with 4,389 nodes and 14,319 edges, when varying the desired path length, the times were measured in seconds.

5 Greedy Search

A common technique for module identification employs a greedy search. We describe below a greedy based framework for detecting protein complexes [2].

5.1 A Probabilistic Model For Protein Complexes

Sharan *et al.*[2] presented a probabilistic model for protein interaction data within a single species. Given a dataset of protein interactions for some organism, they translate it into an interaction graph G, a protein complex is typically dense therefore, dense subgraphs in G may be suggested as putative protein complexes . The model assumptions are: a clique is the ideal structure of a protein complex, which means protein complexes are manifested as dense subgraphs as in real life (an example is shown in Figure 6). The algorithmic problem is translated into finding heavy subgraphs in a graph.

The authors defined two models as follows: The protein-complex model, M_c , assumes that every two proteins in a complex interact with some high probability p. In terms of the graph, the assumption is that two vertices that belong to the same complex are connected by an edge with probability p, independently of all other protein pairs. In contrast, the null model, M_n, assumes that each edge (u, v) is present with the probability $p(u, v)$ that one would expect if the edges of G were randomly distributed but respected the degrees of the vertices.

5.2 The Scoring Scheme

A SAMBA like scoring scheme assigns the following score to a complex $C = (V', E')$

$$L(C) = \prod_{(u,v)\in E'} \frac{p}{p(u,v)} \prod_{(u,v)\notin E'} \frac{(1-p)}{1-p(u,v)} \tag{11}$$

A complicating factor in constructing the interaction graph is that we do not know the real protein interactions, but rather have partial, noisy observations of them. To deal with this difficulty the authors treat the protein protein interaction data as noisy observations.

L et T_{uv} denote the event that two proteins u, v interact. Let F_{uv} denote the event that they do not interact. Let O_{uv} denote the (possibly empty) set of available observations on the proteins u and v, that is, the set of experiments in which u and v were tested for interaction and the outcome of these tests. Using prior biological information one can estimate for each protein pair the probability Pr($O_{uv}|T_{uv}$) of the observations on this pair given that it interacts, and the probability Pr($O_{uv}|F_{uv}$) of those observations given that this pair does not interact. Also, one can estimate the prior probability Pr(T_{uv}) that two random proteins interact.

G iven a subset U of the vertices, the score is the ratio of the likelihood of U under a protein-complex model and under a null model. Specifically the likelihood for the collection of all observations on vertex pairs in U, denoted O_u is calculated as follows:

$$Pr(O_u|M_c) = \prod_{(u,v)\in U\times U} Pr(O_{uv}|M_c) \tag{12}$$

$$= \prod_{(u,v)\in U\times U} (Pr(O_{uv}|T_{uv}, M_c)(Pr(T_{uv}|M_c) + (Pr(O_{uv}|F_{uv}, M_c)(Pr(F_{uv}|M_c) \tag{13}$$

$$= \prod_{(u,v)\in U\times U} (p(Pr(O_{uv}|T_{uv}) + (1-p)(O_{uv}|F_{uv})) \tag{14}$$

The first equation follows from the assumption that all pairwise interactions are independent. The second equation is obtained using the law of complete probability. The third equation follows by noting that given the hidden event of whether u and v interact, O_{uv} is independent of any model. It remains to compute Pr($O_{uv}|M_n$). Since their previous null model depended on having the degree sequence of the interaction graph, they cannot use it as is. To overcome this difficulty, they approximate the degree sequence of the hidden interaction graph: Let $d_1, \ldots, d_n$ denote the expected degrees of the vertices in G, rounded to the closest integer. In order to compute $d_1, \ldots, d_n$, they apply Bayess rule to derive the expectation of T_{uv} for any pair (u, v) given their observations on this vertex pair:

$Pr(T_{uv}|O_{uv}) = \frac{Pr(O_{uv}|T_{uv})Pr(T_{uv})}{(Pr(O_{uv}|T_{uv})Pr(T_{uv})+Pr(O_{uv}|F_{uv})(1-Pr(T_{uv}))}$

Hence, $d_i = [\sum_j [Pr(T_{ij}|O_{ij})]$, where $[\cdot]$ indicates rounding. Their refined null model assumes that G is drawn uniformly at random from the collection of all graphs whose degree sequence is $d_1, ..., d_n$. This induces a probability p_{uv} for every vertex pair (u, v). It's now possible to calculate the probability of O_U according to the null model:

$$Pr(O_U|M_n) = \prod_{(u,v)\in U\times U} (p_{uv}Pr(O_{uv}|T_{uv}) + (1-p_{uv})Pr(O_{uv}|F_{uv})) \tag{15}$$

Finally, the log likelihood ratio that they assign to a subset of vertices U is

$$L(U) = \log \frac{Pr(O_u|M_c)}{Pr(O_u|M_n)} \tag{16}$$

$$= \sum_{(u,v)\in U\times U} \log \frac{pPr(O_{uv}|T_{uv}) + (1-p)Pr(O_{uv}|F_{uv}))}{(p_{uv}Pr(O_{uv}|T_{uv}) + (1-p_{uv})Pr(O_{uv}|F_{uv})} \tag{17}$$

$$= \sum_{(u,v)\in U\times U} \log \frac{pPr(T_{uv}|O_{uv})(1-Pr(T_{uv})+(1-p)(1-Pr(T_{uv}|O_{uv}))Pr(T_{uv})}{(p_{uv}Pr(T_{uv}|O_{uv})(1-Pr(T_{uv}))+(1-p_{uv})(1-Pr(T_{uv}|O_{uv}))Pr(T_{uv})} \quad (18)$$

where the last equation presented follows by applying Bayes rule and canceling common terms in the numerator and denominator.

5.3 The Search Algorithm

The problem of searching for heavy subgraphs in a graph is NP-hard by reduction from CLIQUE [5]. The authors propose heuristic strategies for searching the interactions graph, the algorithm iteratively finds a heavy weight subgraph, adds it to the final output list, and removes all other highly intersecting subgraphs.

1. The algorithm performs a bottom-up search for heavy subgraphs in the interactions graph. It starts from high weight seeds, refines them by exhaustive enumeration: it computes a seed around each node v, which consists of v and all its neighbors u such that (u, v) has positive weight. If the size of this set is above a threshold it iteratively removes from it the node whose contribution to the subgraph score is minimum, until it reaches the desired size.

2. Then it expands them using local search: it enumerates all subsets of the seed that have size at least 3 and contain v. Each such subset is a refined seed on which the algorithm applies a local search heuristic. During the local search it iteratively adds a node, whose contribution to the current seed is maximum, or removes a node, whose contribution to the current seed is minimum, as long as this operation increases the overall score of the seed. Throughout the process, it preserves the original refined seed and do not delete nodes from it. For each node in the interactions graph, it records up to k heaviest subgraphs that were discovered around that node.

3. The resulting subgraphs may overlap considerably, so a greedy algorithm is used to filter subgraphs whose percentage of intersection is above a threshold: significant complexes were filtered by disallowing large overlap between two complexes. Precisely, if 60% of the nodes or 60% of the distinct proteins in each species were common to two complexes, the one with the poorer p-value was removed.

References

[1] N. Alon, R. Yuster, and U. Zwick. Color coding. *ACM*, 42:844856, 1995.

[2] R.Sharan, T. Ideker, B. Kelley, R. Shamir, and R.M. Karp. Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. *Journal of Computational Biology*, 12 , Number 6,Mary Ann Liebert, Inc.:835–846, 2005.

[3] J. Scott, T. Ideker, R.M. Karp, and R. Sharan. Efficient algorithms for detecting signaling pathways in protein interaction networks. *Proceedings of RECOMB*, pages 1–13, 2005.

[4] M. Steffen, A. Petti, J. Aach, P. D'haeseleer, and G. Church. Automated modeling of signal transduction networks. *BMC Bioinformatics*, 3:34–44, 2002.

[5] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(1):136–148, 2002.