Analysis of Biological Networks

Finding the best simple path of length *k* starting from a given node in the graph

Problem definition

- Given a set I of start vertices, what is the best simple path of length *k*?
- Simple path:
  - each vertex is visited once, no cycles
- *Best* simple path
  - That is the most probable path
    - i.e., if edge weights show probabilities, the probability of a path can be computed by:

$$\prod_{\text{for every edge } e \in path} w(e_i)$$

Additive edge weights

- For an easier formulation of the problem (similar to shortest paths) it is better to work with additive edge weights rather than multiplicative ones. So convert each edge probability to:
  - new_weight (e) = -log weight(e)
  - probabilities between 0 and 1 → new weights positive values between 0 and Infinity
  - smaller probabilities will have larger weights and higher probabilities will have smaller weights → best path is the *shortest* path

Formal definition

- *Weight* of a path is the sum of the weights of its edges, and the *length* of a path is the number of vertices it contains.
- Given an undirected weighted graph $G=(V,E,w)$ with $|V|=n$, $|E|=m$ 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 with I and ends at v. If no such simple path exists, the algorithm should report this fact.
- Simple-path restriction makes the problem a difficult one.
  - without simple path restriction we can get the a shortest path of desired length by looping at smallest edges back and forth.

Dynamic programming

- The best simple-path of length k problem can be solved by dynamic programming.
- Define $W(v, S)$ as the minimum weight of a simple path of length $|S|$ which starts at some vertex in I, visits each vertex in S, and ends at v. Starting at smaller sets we can use the following recurrence function to fill in a table of $W(v, S)$ for all v and S.

$$W(v,S) = \min_{u \in S-\{v\}} W(u, S-\{v\}) + w(u,v), |S| > 1$$

$$W(v,\{v\}) = 0 \text{ if } v \in I \text{ and } \infty \text{ otherwise}$$

- Complexity: $O(kn^k)$

Color coding

- Idea: Instead of using vertex ids (resulting in n^k possible subsets of length k), let's assign random colors (out of k possible colors) to the vertices.
- Instead of searching for paths with distinct vertices, search for paths with distinct colors (*colorful* paths)
- This reduces the possible sets to look for to (2^k)

Color-Coding

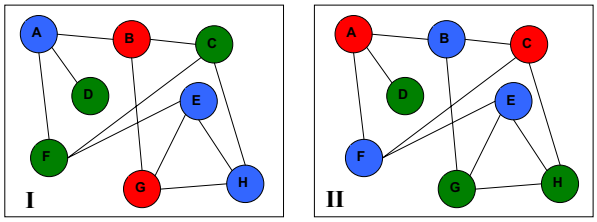
- Colorful paths can be found with dynamic programming
- Key point: a colorful path of length *k* contains a colorful path of length *k*-1.
- Store path information at each node for each subset of *k* colors
  - Only 2^k color subsets, rather than $O(n^k)$ node subsets

$$W(v,S) = \min_{u:c(u)\in(S-\{c(v)\})} W(u, S-\{c(v)\}) + w(u,v), |S|>1$$

$$W(v,\{c(v)\}) = 0 \text{ if } v \in I \text{ and } \infty \text{ otherwise}$$

- Runtime is $O(2^k km)$ << $O(kn^k)$ brute force
- Space is $O(2^k n)$ << $O(kn^k)$ brute force

Coloring Example



- Two different colorings on toy graph, *k*=3
- In coloring **I**, W(A,RGB) is built C->BC->ABC
- In coloring **II**, W(A,RGB) is built G->BG->ABG
- ABC is not colorful in coloring **II**

Final Exam

- May 24, Wednesday at 13:30 in BMB-2.
- Sequence Analysis
  - DP, Suffix Trees/Arrays, Multiple Alignment
- Phylogenetic trees
  - UPGMA, NJ
- Protein structure prediction
  - SS prediction, threading
- Protein structure comparison & classification
  - Iterative DP, Distance Matrices (DALI), Geometric hashing/Motif finding
- Microarrays
  - Clustering: k-means, hierarchical clustering, SOMs
- Biological networks
  - Gene Regulatory Networks: significant pattern detection
  - Construction of biological networks: Naive Bayes technique
  - Analysis of networks: network reliability (MC sampling), random walks