Mathematics for Computer Science (30470023) 2020 Spring

Lecture 11 — May 11, 2020

Instructor: Andrew C. Yao *Scribes: Zuotong Wu, Siting Li*

1 Overview

In this lecture, we will see how graph algorithms can be applied to real examples in engineering and science. The first example comes from an area called operations research, and the second one is about bio-informatics.

2 Example 1: Radio Frequencies Allocation

2.1 Problem Description

Given n points $v_i = (x_i, y_i)$ and n numbers $r_i > 0$, there are n radio stations located at the n points. The communication radius of the station at point v_i is r_i. If two stations at v_i and v_j with $\|v_i - v_j\| \leq r_i + r_j$ use the same frequency, then their signals will interfere with each other. The task is to allocate a frequency to each radio station so that every pair of the stations does not interfere, and to make the number of different frequencies as few as possible.

2.2 Complexity

If we let $V = \{v_1, v_2, \cdots, v_n\}$ and $E = \{(v_i, v_j) \mid \|v_i - v_j\| \leq r_i + r_j\}$, then this problem can be considered as a Vertex-Coloring Problem on the graph $G = (V, E)$. It is hard to find an efficient solution to the Vertex-Coloring Problem, even in the special case that the graph is induced by the parameters v_i and r_i. It is proved that this special case is still NP-Complete, as hard as the Hamiltonian Circuit Problem.

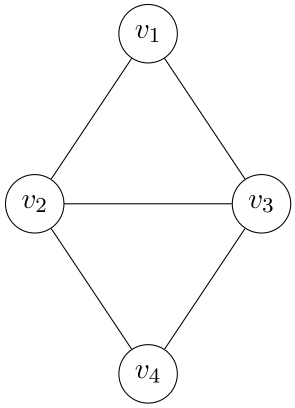
To deal with the NP-Hard problems practically, we usually use approximation algorithm to get a good solution to some extent.

2.3 The Greedy List Algorithm

One of the approximation algorithms is the Greedy List Algorithm. This algorithm finds an allocation function $f : V \to \mathbb{Z}^+$ in the following way:

Let $G = (V, E)$ be the conflict graph and $L = (v_1, v_2, \cdots, v_n)$. Then, allocate the minimum possible frequency to the vertices one by one ordered by L.

Here is an example:



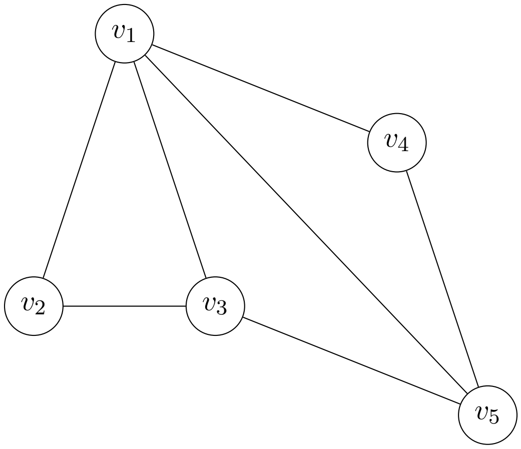
For convenience, we use "colors" from the Vertex-Coloring Problem instead of "frequencies".

$f(v_1)$ should be 1. If v_2 is also colored 1, then v_1 and v_2 will conflict, so v_2 should be colored 2. Since there is already a 1 and a 2 next to v_3, v_3 should be colored 3. v_4 should be colored 1 since there is no vertex with color 1 adjacent to v_4.

2.4 Analysis of Performance

2.4.1 Optimality

This algorithm does not always yield the best solution. Consider the following graph:



In this graph, the vertices are colored 1,2,3,2,4 respectively according to the Greedy List Algorithm. However, there exists a solution with fewer colors: $f(v_1) = f(v_5) = 1, f(v_2) = 3, f(v_3) = f(v_4) = 2$.

2.4.2 Performance in the unit-radius case

Although this algorithm may not give the best solution, it performs well on this frequency allocation problem.

In the unit-radius case, that is, for all $1 \leq i \leq n$, $r_i = 1$, the total number of colors used by the Greedy List Algorithm $A(G)$ is at most $5\chi(G) - 4$.

Here is the proof. Consider the moment ont of the largest number is allocated to vertex v_i. Let $G' = (N(v_i), E')$ be the subgraph induced by v_i's neighbors, that is, $N(v_i) = \{v \mid (v, v_i) \in E\}, E' = \{(u, v) \mid u, v \in N(v_i)\}$. Since the color of v_i must be different from that of all vertices in $N(v_i)$, $\chi(G) \geq \chi(G') + 1$.

In the last class, we proved that for any graph G, $\chi(G) \geq \frac{|V|}{\alpha(G)}$, so

$$\begin{aligned} \chi(G) &\geq \chi(G') + 1 \\ &\geq \frac{|V'|}{\alpha(G')} + 1 \\ &\geq \frac{A(G) - 1}{\alpha(G')} + 1 \\ A(G) &\leq \alpha(G')(\chi(G) - 1) + 1. \end{aligned}$$

Now we prove that $\alpha(G') \leq 5$.

If $\alpha(G') \geq 6$ then there must be two vertices v_j, v_k in the maximum independent set such that $\angle v_j v_i v_k \leq 60^\circ$, so

$$\|v_j - v_k\| \leq \max\{\|v_i - v_j\|, \|v_i - v_k\|\} \leq 1,$$

which conflicts with that v_j and v_k belong to the same independent set.

As a result,

$$\begin{aligned} A(G) &\leq \alpha(G')(\chi(G) - 1) + 1 \\ &\leq 5\chi(G) - 4. \end{aligned}$$

2.5 The General Problem

How does this algorithm work in the general graphs? Unfortunately, this is not the case. There exist graphs such that $\frac{A(G)}{\chi G} \geq n^\epsilon$, where $\epsilon > 0$. This is left as homework.

Deep theory work showed that this is unavoidable. That is, if there is an algorithm that achieves $\frac{A(G)}{\chi(G)} \leq n^{1-\epsilon}$ in polynomial time for some $\epsilon > 0$, then P = NP. Hence, if you try to find such an algorithm, then you are trying to attack the P = NP problem on the wrong side of the expectation.

3 Example 2: DNA Fragment Assembly

3.1 A Problem in Bio-informatics

Take a segment of σ (a genome sequence). How do we determine σ?

Fortunately, we have chemical method to cut σ into fragments, each of length l (not precisely). So we can take a genome σ, replicate it by chemical or biological method, put the copies into a particular solution with enzymes, and let the enzymes cut them into shorter sequences. After this process, all substrings with length l are available. Let $S(\sigma)$ be the set of substrings of σ with length l. It is called the "spectrum". If there is no repetition of substrings, then $|S(\sigma)| = n - l + 1$, where $n = |\sigma|$.

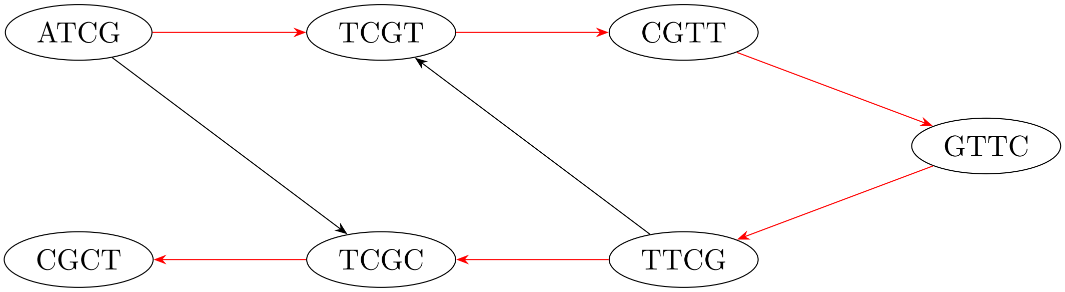
Next, we use a biochip with 4^l units. Pour the solution of spectrum over this chip. Each of the units somehow "attracts" a certain kind of substring. Observe the biochip in an ultraviolet light (i.e. fluorescent illumination) and you will find that some units are lit up. In this way, we can determine $S(\sigma)$.

Now comes the computer scientists' part: given $S(\sigma)$, how to determine σ?

3.2 One Early Solution

Definition 3.1. For a directed (simple) graph $G = (V, E)$, a Hamiltonian Path P is a path in G that contains each vertex exactly once. ♢

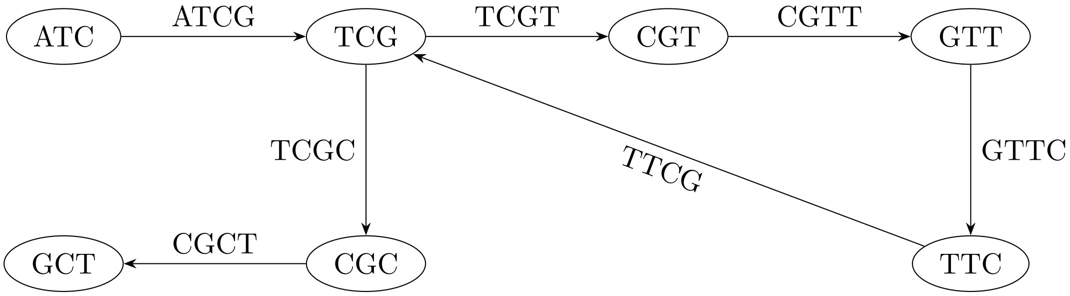
Given a spectrum $S(\sigma)$, define a directed graph $G = (V, E)$, where $V = S(\sigma)$ and $(v_i, v_j) \in E$ iff $v_i = x_1x_2 \ldots x_l$ and $v_j = x_2x_3 \ldots x_lx_{l+1}$. If we find a Hamiltonian Path in G, then it leads to a possible σ. (There may be a few possible solutions, but perhaps most of them do not make biological sense. We only need to provide all possible solutions here.) Suppose σ = ATCGTTCGCT and $l = 4$, the graph is illustrated below:



However, Pevzner, being a computer scientist, was aware that the Hamiltonian Path Problem was not a suitable formulation of this problem because it has no efficient solution.

3.3 Another Solution

Pevzner ([**?**, **?**]) tried to formulate it instead as a Eulerian Path Problem. He reconstructed the graph in this manner: regard $(l-1)$-substrings as nodes and l-substrings as edges. Now we have $G' = (V', E')$, where $V' = \{x_1x_2\ldots x_{l-1} | x_1x_2\ldots x_{l-1}\cdot \in S(\sigma) \text{ or } \cdot x_1x_2\ldots x_{l-1} \in S(\sigma)\}$ and E' contains all the (v, v') such that $v = x_1x_2\ldots x_{l-1}$, $v' = x_2x_3\ldots x_l$ and $x_1x_2x_3\ldots x_l \in S(\sigma)$. The graph G' of the same example is showed below:



Therefore, a consistent σ gives rise to an Eulerian Path in G. So we can find out all possible solutions by graph algorithms for Eulerian Path Problem, which is way more efficient than the early solution.