15-750 Algorithms (Spring 2002) — Prof. Manuel Blum

Homework #7 — **Solutions**

Problem 1: Graph coloring (20 points)

(this is Problem 6.5 from M&R; you should solve it without using the reference given in the book)

Let G be a 3-colorable graph. Consider the following algorithm for coloring the vertices of G with 2 colors, so that no triangle of G is monochromatic. The algorithm begins with an arbitrary 2-coloring of G. While there is a monochromatic triangle in G, it chooses one such triangle, and changes the color of a randomly chosen vertex of that triangle. Derive an upper bound on the expected number of such re-coloring steps before the algorithm finds a 2-coloring with the desired property.

(Solution due to Ashwin Bharambe, Jonathan Moody, and Susmit Sarkar)

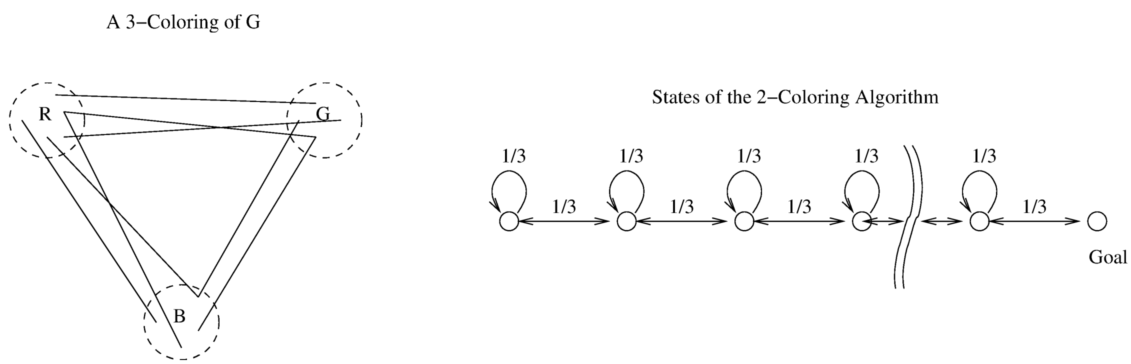


Figure 1: Random walk on 2-colorings

Let $\mathcal{C}$ be any 3-coloring of the graph G. Let V_b, V_r and V_g be the vertex sets colored by blue, red and green colors, respectively. Clearly, each of them is an *independent set*. Hence any *triangle* in the graph must be of the form shown in Figure 1.

Let $\mathcal{C}'$ be the arbitrary 2-coloring of G from which the algorithm starts. Suppose that $\mathcal{C}'$ uses blue and red colors only. Let's compare $\mathcal{C}'$ with $\mathcal{C}$. If the algorithm reaches a state starting with $\mathcal{C}'$ such that all vertices $\in V_b$ are colored blue and all vertices $\in V_r$ are colored red, then clearly, no triangle can be monochromatic irrespective of the color of the vertices $\in V_g$.

Thus, we can measure the distance of the *goal* coloring from the present one using the number of miscolored blue or red vertices, i.e., $v \in V_b$ such that $\text{color}(v) = \text{red}$ and $v' \in V_r$ such that $\text{color}(v') = \text{blue}$.

Now, at each step, the algorithm chooses a monochromatic triangle at random. It can either be a `rrr` or `bbb` triangle. If it flips the color of a vertex $\in V_g$, the distance from the *goal* coloring remains the same.

In the case when it is a `rrr` triangle, if it flips the color of a vertex $\in V_b$, the number of miscolored vertices decreases. On the other hand, if the flip occurs for a vertex $\in V_r$, this number increases. Similar situation exists in the `bbb` case.

The algorithm, thus, can be represented as performing a random walk on a n vertex *line*, where $n = |G|$ with the probabilities shown in Figure 1. Hence, the expected time to complete is the expected time to reach the end vertex starting from any vertex in between. In the worst case, it can start from the other end of the line when all vertices are miscolored.

The recurrence for the expected time is,

$$\begin{aligned} h_{in} &= \frac{1}{3}\left(h_{in} + h_{i-1,n} + h_{i+1,n}\right) \\ \Rightarrow h_{in} &= \frac{1}{2}\left(h_{i-1,n} + h_{i+1,n}\right) \end{aligned}$$

From this, we get $h_{0n} = \mathcal{O}(n^2)$.[1] Hence, the expected running time of the algorithm is $\mathcal{O}(n^2)$.

Problem 2: A long directed random walk (10 points)

(this is Problem 6.16 from M&R)

Show that the expected time for a random walk to visit every vertex of a strongly connected directed graph is not bounded above by any polynomial function of n, the number of vertices. In other words, construct a directed graph that is strongly connected and where the expected cover time is super-polynomial.

Consider a graph G_n with n vertices $v_1, \ldots, v_n$, and directed edges (v_{i-1}, v_i) and (v_i, v_1), for $i = 2 \ldots n$. G_n is a strongly connected directed graph (without self-loops or multiple edges), where v_1 has out-degree 1, and every other vertex has out-degree 2. Let h_{ij} denote the hitting time in a random walk starting at v_i and ending upon first reaching of v_j. We will show that h_{1n} is super-polynomial, which directly implies that $C(G_n)$, the expected cover time of G_n, is super-polynomial.

We get the following set of equations:

$$\begin{aligned} h_{1n} &= 1 + h_{2n} \\ h_{2n} &= 1 + h_{3n}/2 + h_{1n}/2 \\ h_{3n} &= 1 + h_{4n}/2 + h_{1n}/2 \\ &\vdots \\ h_{(n-2)n} &= 1 + h_{(n-1)n}/2 + h_{1n}/2 \\ h_{(n-1)n} &= 1 + h_{1n}/2\ , \end{aligned}$$

[1] This is the same as the recurrence for hitting times in the 'normal' straight line graph.

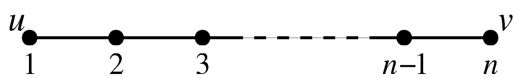
which yield directly

$$
\begin{aligned}
h_{1n} &= 1 + (2 + h_{1n})(1/2 + 1/4 + 1/8 + \ldots + 1/2^{n-2}) \\
h_{1n} &= 1 + (2 + h_{1n})(1 - 1/2^{n-2}) \\
&\ldots \\
h_{1n} &= 2^{n-2}(3 - 1/2^{n-3}) = \Omega(2^n) .
\end{aligned}
$$

□

Problem 3: Take a walk on a … straight line (20 points)

(a) What is the *exact* value for the expected time for a random walk to go from u to v, where u and v are opposite ends of the n-node straight line graph:



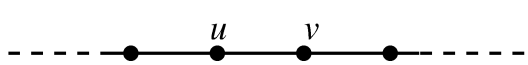
Compute this quantity both directly and using the method of resistive networks. Run a simulation to determine the expected time empirically: perform a series of experiments for $n = 11$ with $t = 1, 10, 100, \ldots$ trials (where each trial is a complete random walk from u to v), and take the arithmetic mean from each experiment. How many trials do you need to get the mean within 1% of the exact computed value (i.e. two decimal digits correct)?

A direct computation of h_{uv} was described in the solution to Problem 2(a) of Homework #6. The method of resistive networks yields by symmetry

$$h_{uv} = C_{uv}/2 = (n-1)^2 .$$

With $n = 11$ the expected hitting time is 100 steps. The empirical estimation of h_{uv} converges nicely with the growing number of trials. About 10^5 trials are sufficient to get the mean within 1% of the exact computed value.

(b) What is the *exact* value for the expected time for a random walk to go from u to v, where u and v are adjacent vertices of the 2-way infinite straight line graph:



Compute this quantity however you like. Run a simulation, as in part (a). What do you observe? Understand and explain.

Since the graph is infinite, the method of resistive networks yields directly $h_{uv} = \infty$. The empirical estimation of h_{uv} diverges with the growing number of trials — although many trials reach v pretty fast, some trials manage to walk quite far away left of u, and need many steps to come back to v. As we increase the number of trials, the probability that at least one trial takes many steps grows, and so does the arithmetic mean.

Problem 4: A "universal" universal sequence (extra credit)

(this problem is due to Grant Reaber)

Recall from the lecture on universal sequences, that for each n there exists a universal sequence of length polynomial in n, which traverses every labeled graph[2] with n vertices.

Is it true, that there exists a single (infinite) universal sequence S such that for each n the initial portion of S, of length polynomial in n, traverses every labeled, connected graph with n vertices. Prove your answer.

Yes, there exists a "universal" universal sequence. Let S_n denote a universal sequence (of polynomial length) for graphs with n vertices, and let $|S_n|$ denote length of S_n. Wlog. assume that $|S_i| \geq |S_j|$ if $i > j$. Consider an infinite sequence $S = S_1 S_2 S_3 \ldots$, i.e. a concatenation of all S_i's. For a given n define S'_n as an initial portion of S consisting of all universal sequences up to S_n, i.e. $S'_n = S_1 S_2 \ldots S_n$. Since S'_n contains S_n as a subsequence, S'_n is also a universal sequence for graphs with n vertices. Moreover, the length of S'_n is at most n times $|S_n|$, hence it is also polynomial in n. $\square$.

[2] The graphs considered are connected, undirected, without self-loops or multiple edges.