Chapter 1

Random Walks IV

By Sariel Har-Peled, April 2, 2024[①]

> "Do not imagine, comrades, that leadership is a pleasure! On the contrary, it is a deep and heavy responsibility. No one believes more firmly than Comrade Napoleon that all animals are equal. He would be only too happy to let you make your decisions for yourselves. But sometimes you might make the wrong decisions, comrades, and then where should we be? Suppose you had decided to follow Snowball, with his moonshine of windmills-Snowball, who, as we now know, was no better than a criminal?"
>
> Animal Farm, George Orwell

1.1. Cover times

We remind the reader that the cover time of a graph is the expected time to visit all the vertices in the graph, starting from an arbitrary vertex (i.e., worst vertex). The cover time is denoted by $\mathcal{C}(\mathsf{G})$.

Theorem 1.1.1. *Let G be an undirected connected graph, then $\mathcal{C}(\mathsf{G}) \leq 2m(n-1)$, where $n = |V(\mathsf{G})|$ and $m = |E(\mathsf{G})|$.*

Proof: (Sketch.) Construct a spanning tree T of G, and consider the time to walk around T. The expected time to travel on this edge on both directions is $\mathbf{CT}_{uv} = \mathtt{h}_{uv} + \mathtt{h}_{vu}$, which is smaller than $2m$, by Lemma 1.3.1. Now, just connect up those bounds, to get the expected time to travel around the spanning tree. Note, that the bound is independent of the starting vertex. ■

Definition 1.1.2. The ***resistance*** of G is $\mathbf{R}(\mathsf{G}) = \max_{u,v \in V(\mathsf{G})} \mathbf{R}_{uv}$; namely, it is the maximum effective resistance in G.

Theorem 1.1.3. $m\mathbf{R}(\mathsf{G}) \leq \mathcal{C}(\mathsf{G}) \leq 2e^3 m\mathbf{R}(\mathsf{G}) \ln n + 2n$.

Proof: Consider the vertices u and v realizing $\mathbf{R}(\mathsf{G})$, and observe that $\max(\mathtt{h}_{uv}, \mathtt{h}_{vu}) \geq \mathbf{CT}_{uv}/2$, and $\mathbf{CT}_{uv} = 2m\mathbf{R}_{uv}$ by Theorem 1.3.2. Thus, $\mathcal{C}(\mathsf{G}) \geq \mathbf{CT}_{uv}/2 \geq m\mathbf{R}(\mathsf{G})$.

As for the upper bound. Consider a random walk, and divide it into ***epochs***, where a epoch is a random walk of length $2e^3 m\mathbf{R}(\mathsf{G})$. For any vertex v, the expected time to hit u is $\mathtt{h}_{vu} \leq 2m\mathbf{R}(G)$, by Theorem 1.3.2. Thus, the probability that u is not visited in a epoch is $1/e^3$ by the Markov inequality. Consider a random walk with $t = \ln n$ epochs. We have that the probability of not visiting u is $\leq (1/e^3)^{\ln n} \leq 1/n^3$. Thus, all vertices are visited after $\ln n$ epochs, with probability $\geq 1 - \binom{n}{2}/n^3 \geq 1 - 1/n$. Otherwise, after this walk, we perform a random walk till we visit all vertices. The length of this (fix-up) random walk is $\leq 2n^3$, by Theorem 1.1.1. Thus, expected length of the walk is $\leq 2e^3 m\mathbf{R}(\mathsf{G}) \ln n + 2n^3(1/n^2)$. ■

1.1.1. Rayleigh's Short-cut Principle.

Observe that effective resistance is never raised by lowering the resistance on an edge, and it is never lowered by raising the resistance on an edge. Similarly, resistance is never lowered by removing a vertex.

Interestingly, effective resistance comply with the triangle inequality.

Observation 1.1.4. *For a graph with minimum degree d, we have $\mathbf{R}(\mathsf{G}) \geq 1/d$ (collapse all vertices except the minimum-degree vertex into a single vertex).*

Lemma 1.1.5. *Suppose that G contains p edge-disjoint paths of length at most ℓ from s to t. Then $\mathbf{R}_{st} \leq \ell/p$.*

1.2. Graph Connectivity

Definition 1.2.1. A ***probabilistic log-space Turing machine*** for a language L is a Turing machine using space $O(\log n)$ and running in time $O(\text{poly}(n))$, where n is the input size. A problem A is in RLP, if there exists a probabilistic log-space Turing machine M such that M accepts $x \in L(A)$ with probability larger than $1/2$, and if $x \notin L(A)$ then $M(x)$ always reject.

Theorem 1.2.2. *Let* USTCON *denote the problem of deciding if a vertex s is connected to a vertex t in an undirected graph. Then* USTCON $\in$ RLP.

Proof: Perform a random walk of length $2n^3$ in the input graph G, starting from s. Stop as soon as the random walk hit t. If u and v are in the same connected component, then $\mathrm{h}_{st} \leq n^3$. Thus, by the Markov inequality, the algorithm works. It is easy to verify that it can be implemented in $O(\log n)$ space. ■

Definition 1.2.3. A graph is ***d-regular***, if all its vertices are of degree d.

A d-regular graph is ***labeled*** if at each vertex of the graph, each of the d edges incident on that vertex has a unique label in $\{1, \ldots, d\}$.

Any sequence of symbols $\sigma = (\sigma_1, \sigma_2, \ldots)$ from $\{1, \ldots, d\}$ together with a starting vertex s in a labeled graph describes a ***walk*** in the graph. For our purposes, such a walk would almost always be finite.

A sequence σ is said to ***traverse*** a labeled graph if the walk visits every vertex of G regardless of the starting vertex. A sequence σ is said to be a ***universal traversal sequence*** of a labeled graph if it traverses all the graphs in this class.

Given such a universal traversal sequence, we can construct (a non-uniform) Turing machine that can solve USTCON for such d-regular graphs, by encoding the sequence in the machine.

Let $\mathcal{F}$ denote a family of graphs, and let $U(\mathcal{F})$ denote the length of the shortest universal traversal sequence for all the labeled graphs in $\mathcal{F}$. Let $\mathbf{R}(\mathcal{F})$ denote the maximum resistance of graphs in this family.

Theorem 1.2.4. *Let $\mathcal{F}$ be a family of d-regular graphs with n vertices, then $U(\mathcal{F}) \leq 5m\mathbf{R}(\mathcal{F}) \lg(n\,|\mathcal{F}|)$.*

Proof: Same old, same old. Break the string into *epochs*, each of length $L = 2m\mathbf{R}(\mathsf{G})$. Now, start random walks from all the possible vertices, for all graphs in $\mathcal{F}$. Continue the walks till all vertices are being visited. Initially, there are $n^2\,|\mathcal{F}|$ vertices that need to visited. In expectation, in each epoch half the vertices get visited. There are $n\,|\mathcal{F}|$ walks, each of them needs to visit n vertices. As such, the number of vertices waiting to be visited is bounded by $|\mathcal{F}|\,n^2$. As such, after $1 + \lg_2\!\left(n^2\,|\mathcal{F}|\right)$ epochs, the expected number of vertices still need visiting is $\leq 1/2$. Namely, with constant probability we are done. ■

Let $U(d,n)$ denote the length of the shortest universal traversal sequence of connected, labeled n-vertex, d-regular graphs.

Lemma 1.2.5. *The number of labeled n-vertex graphs that are d-regular is* $(nd)^{O(nd)}$.

Proof: Such a graph has $dn/2$ edges overall. Specifically, we encode this by listing for every vertex its d neighbors – there are $N = \binom{n-1}{d} \leq n^d$ possibilities. As such, there are at most $N^n \leq n^{nd}$ choices for edges in the graph[2]. Every vertex has $d!$ possible labeling of the edges adjacent to it, thus there are $(d!)^n \leq d^{nd}$ possible labelings. ■

Lemma 1.2.6. $U(d,n) = O\left(n^3 d \log n\right)$.

Proof: The diameter of every connected n-vertex, d-regular graph is $O(n/d)$. Indeed, consider the path realizing the diameter of the graph, and assume it has t vertices. Number the vertices along the path consecutively, and consider all the vertices that their number is a multiple of three. There are $\alpha \geq \lfloor t/3 \rfloor$ such vertices. No pair of these vertices can share a neighbor, and as such, the graph has at least $(d+1)\alpha$ vertices. We conclude that $n \geq (d+1)\alpha = (d+1)(t/3-1)$. We conclude that $t \leq \frac{3}{d+1}(n+1) \leq 3n/d$.

And so, this also bounds the resistance of such a graph. The number of edges is $m = nd/2$. Now, combine Lemma 1.2.5 and Theorem 1.2.4. ■

This is, as mentioned before, not a uniform algorithm. There is by now a known log-space deterministic algorithm for this problem, which is uniform.

1.2.1. Directed graphs

Theorem 1.2.7. *One can solve the* $\overrightarrow{\text{STCON}}$ *problem, for a given directed graph with n vertices, using a log-space randomized algorithm, that always output NO if there is no path from s to t, and output YES with probability at least* $1/2$ *if there is a path from s to t.*

Proof: (Sketch.) The basic idea is simple – start a random walk from s, if it fails to arrive to t after a certain number of steps, then restart. The only challenging thing is that the number of times we need to repeat this is exponentially large. Indeed, the probability of a random walk from s to arrive to t in n steps, is at least $p = 1/n^{n-1} \geq n^n$ if s is connected to t.

As such, we need to repeat this walk $N = O((1/p)\log \delta) = O(n^{n+1})$ times, for $\delta \geq 1/2^n$. If have all of these walks fail, then with probability $\geq 1-\delta$, there is no path from s to t.

We can do the walk using logarithmic space. However, how do we count to N (reliably) using only logarithmic space? We leave this as an exercise to the reader, see Exercise 1.2.8, ■

Exercise 1.2.8. Let N be a large integer number. Show a randomized algorithm, that with, high probability, counts from 1 to M, where $M \geq N$, and always stops. The algorithm should use only $O(\log \log N)$ bits.

1.3. Tools from previous lecture

Lemma 1.3.1. *For any edge* $(u \to v) \in E$, $\mathrm{h}_{uv} + \mathrm{h}_{vu} \leq 2m$.

Theorem 1.3.2. *For any two vertices u and v in* G, *the commute time* $\mathbf{CT}_{uv} = 2m\mathbf{R}_{uv}$.

[2] This is a callous upper bound – better analysis is possible. But never analyze things better than you have to - it usually a waste of time.