

CSE 203A: Randomized Algorithms

Spring 2026

Lecture 14: Random Walks on Graphs; Perfect Matchings; Hit, Cover, and Commute Times

Date: 04/29/2026

Instructor: Daniel M. Kane

Scribe: Stefan Gadiant

Lecture Overview

In this lecture, we continued our analysis of random walks in graphs and estimated the return times for random walks to vertices. We then explored random walks in finding perfect matchings in regular bipartite graphs via augmenting paths. Finally, we introduced hitting time, commute time, and cover time, and showed some examples on different graphs.

1 Introduction

Building on the general framework of Markov chains introduced in the previous lecture, we now narrow our focus to processes where the states are vertices of a graph and transitions occur along edges. This specialized structure allows us to relate the physical properties of the graph (connectivity, degrees) to the long-term behavior of the random walk.

2 Brief Reminders about Previous Topics

A Markov chain consists of a sequence of states where the next state depends only on the current one. For a chain to converge to a unique steady state distribution π , it must be:

1. **Finite:** The state space is not infinite.
2. **Connected:** Any state can reach any other state.
3. **Aperiodic:** The gcd of all return times to any state is 1.

Then the result we proved last time is that there exists a steady state π , and $q^t \rightarrow \pi$ as $t \rightarrow \infty$.

3 Main Lecture Content

3.1 Random Walks on Undirected Graphs

For a random walk on an undirected graph $G = (V, E)$, the transition probability from u to v is $P_{uv} = 1/\deg(u)$ if $(u, v) \in E$, and 0 otherwise.

Theorem 3.1. *For an undirected graph, the steady state distribution for a random walk is $\pi_v = \frac{\deg(v)}{2|E|}$.*

Proof. We check the steady-state condition $\pi = \pi P$:

$$\begin{aligned} \sum_{u \in V} \pi_u P_{uv} &= \sum_{u \sim v} \frac{\deg(u)}{2|E|} \cdot \frac{1}{\deg(u)} \\ &= \sum_{u \sim v} \frac{1}{2|E|} \\ &= \frac{\deg(v)}{2|E|} = \pi_v \end{aligned} \tag{1}$$

where $u \sim v$ denotes that u is a neighbor of v . □

Let the **hitting time** be $h_{uv} = \mathbb{E}[\text{time to reach } v \mid \text{currently at } u]$.

Then the **return time** is $h_{uu} = \mathbb{E}[\text{time to reach } u \mid \text{currently at } u] = \mathbb{E}[\text{time to return to } u]$.

Let r_1, r_2, r_3, \dots be a sequence of independent and identically distributed (i.i.d.) random variables with mean h_{uu} representing the intervals between consecutive visits to vertex u .

To compute $h_{uu} = \mathbb{E}[r_1]$, let n be the number of times the walk visits u during $r_1 + r_2 + \dots + r_n$ steps. By law of large numbers, the average time between visits is:

$$\lim_{n \rightarrow \infty} \frac{\sum_{i=1}^n r_i}{n} = \mathbb{E}[r_1] = h_{uu}$$

Since $r_1 + r_2 + \dots + r_n \approx n \cdot h_{uu}$, we have $\frac{n}{T} \approx \frac{1}{h_{uu}}$. Then the fraction of time we spend at u is $\frac{n}{nh_{uu}} = \frac{1}{h_{uu}}$.

Then if we want to compute $\mathbb{E}[\text{number of visits to } u \text{ in the first } n \text{ steps}]$, we have

$$\begin{aligned} \mathbb{E}[\text{number of visits to } u \text{ in the first } n \text{ steps}] &= \sum_{t=1}^n P(X_t = u) \\ &= \sum_{t=1}^n q_n^t \rightarrow n\pi_u \\ &= \sum_{t=1}^n \pi_u = n\pi_u \end{aligned} \tag{2}$$

So the fraction of time at u in expectation is π_u , which means that $\frac{1}{h_{uu}} = \pi_u$.

3.2 Finding Perfect Matchings in Bipartite Graphs

Let $G = (L \cup R, E)$ be a k -regular bipartite graph with $|L| = |R| = n$. Every k -regular bipartite graph contains a perfect matching.

- **Deterministic Baseline:** The Hopcroft-Karp algorithm finds a perfect matching in $O(m\sqrt{n})$ time. For k -regular graphs, $m = nk$, yielding $O(kn^{3/2})$.
- **Random Walk Claim:** We can find a perfect matching in $O(n \log n)$ expected time using a random walk strategy to find augmenting paths.

An **augmenting path** is a path starting and ending at unmatched vertices where edges alternate between the current matching M and $E \setminus M$. Adding such a path increases $|M|$ by 1.

To find these paths, we construct a directed graph $G' = (V', E')$. Let M be the current matching.

- **Vertices:** $V' = \{\text{matched vertices in } L\} \cup \{S\}$, where S is a super-node representing all currently unmatched vertices in L .
- **Edges:**
 1. For S , for each unmatched $u \in L$ and each neighbor $v \in R$: If v is matched to $u' \in L$, add edge $S \rightarrow u'$. If v is unmatched, this represents an augmenting path (a transition back to S).
 2. For a matched $u \in L$, for each neighbor $v \in R$ (excluding its match): If v is matched to u' , add edge $u \rightarrow u'$. If v is unmatched, add edge $u \rightarrow S$.

In this directed graph G' , every node has out-degree k . The state S has an effectively higher weight proportional to the number of unmatched vertices.

We now analyze the hitting time h_{ss} . Let t be the number of currently unmatched vertices in L (and thus t in R). The state S contains t unmatched vertices, each with k edges. The total number of edges in G' is nk . Since G' is regular in its out-degree, the steady state distribution is uniform over the vertices of L if we treated S as t individual nodes.

The probability of being in state S in the steady state of the walk is $\pi_s = \frac{t}{n}$. Using our result from 3.1:

$$h_{ss} = \frac{1}{\pi_s} = \frac{n}{t}$$

Finding a full perfect matching requires n augmentations. The total expected time is:

$$\mathbb{E}[\text{Total Time}] = \sum_{t=1}^n h_{ss} = \sum_{t=1}^n \frac{n}{t} = n \sum_{t=1}^n \frac{1}{t}$$

Recalling that the harmonic sum $\sum_{t=1}^n \frac{1}{t} \approx \ln n$, the total complexity is:

$$\mathbb{E}[\text{Total Time}] = O(n \log n)$$

This is significantly faster than the $O(n^{1.5})$ deterministic bound for dense regular graphs.

3.3 Commute and Cover Times

We define two other metrics of the random walk:

1. **Commute Time:** $C_{uv} = h_{uv} + h_{vu}$.
2. **Cover Time:** $C_u(G) = \mathbb{E}[\text{time starting at } u \text{ to hit every vertex at least once}]$

Examples:

- For a complete graph K_n , the cover time is $O(n \log n)$ by the stamp collector's problem.
- For a path P_n , the cover time is $O(n^2)$.

- Consider a Lollipop graph L_n consisting of a clique $K_{n/2}$ and a path $P_{n/2}$ attached at a junction vertex j . Let v be the leaf at the end of the path. To compute the cover time, we consider the hitting time h_{jv} from the junction to the leaf.

On the path $P_{n/2}$, the walk behaves like a 1D random walk, but at the junction j , the degrees are asymmetric, since the degree of j is roughly $n/2$ (from the clique) plus 1 (from the path). The probability of stepping into the path from j is $P_{j,v_1} \approx \frac{1}{n/2}$. Because the walk is $n/2$ times more likely to retreat into the clique than to advance along the path, and a 1D walk on a path of length n already takes $O(n^2)$ steps, the total hitting time becomes:

$$h_{jv} = \Theta(n \cdot (n/2)^2) = \Omega(n^3)$$

Since cover time $C(G) \geq h_{jv}$, the cover time of the lollipop graph is $\Omega(n^3)$.

Theorem 3.2. For any edge $(u, v) \in E$, the commute time $C_{uv} \leq 2|E|$.

Proof. Consider a Markov chain where the states are the $2m$ directed edges of the graph. For an edge (u, v) , a transition occurs to an edge (v, w) with probability $1/\deg(v)$. The steady state distribution for this edge-walk is uniform: $\pi_{(u,v)} = \frac{1}{2m}$ for all directed edges.

The expected return time to a directed edge $e = (u, v)$ is $1/\pi_e = 2m$. A return to the directed edge (u, v) starting from u involves moving from $u \rightarrow v$ (1 step), then moving from v back to u through some path, and then traversing the edge (u, v) again. The time taken for the walk to go from v back to u and then traverse (u, v) is the hitting time from v to the edge (u, v) . This process covers the commute from u to v and back. Thus, $C_{uv} \leq 2m$. \square

References

- [1] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.