Note:

(1) Try to give the best bound you can for the quantities to be calculated in the problems below.

(2) The next page has hints for some of the problems.

1. Consider the random walk on the complete graph K_n. This is the graph on n vertices in which each vertex is connected to every other vertex.

   (10 points) What is the expected time for the walk to visit *all* n vertices of the graph at least once?

   The expected time to visit all vertices is known as the *cover time* of a graph. In the case of K_n, this does not depend on the start vertex, by symmetry. For a general graph, the cover time is the maximum, over all vertices v, of the expected time to visit all vertices in a walk starting at v.

2. Let L_n be the n-vertex *lollipop graph*, which consists of a clique (complete graph) on $n/2$ vertices $n, n-1, \dots, n/2$, and a path from $n/2$ to 1:

   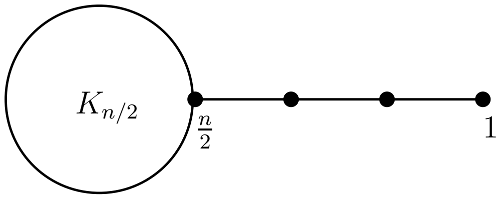

   Figure 1: *The n-vertex lollipop graph L_n.*

   (10 points) What is the expected time to reach 1 in a walk starting at $n/2$, i.e., the hitting time $h_{\frac{n}{2},1}$? Is this equal to the hitting time $h_{1,\frac{n}{2}}$?

3. Consider the random walk on the circle C_n, starting at 0. C_n is the graph on the vertices $0, 1, \dots, n-1$, in which vertex i is connected to vertices $i-1 \pmod n$ and $i+1 \pmod n$.

   (a) **(5 points)** Which vertex has the highest probability of being visited last?

   (b) **(5 points)** Which vertex i has the highest hitting time $h_{0,i}$?

4. Let G_n be the *directed* graph on the vertices $1, 2, \dots, n$, with edges (j, i), for all $j > i$. Consider the following walk on it that starts at n. At every step, we choose a random edge (k, ℓ) leaving the current vertex k, and move to the vertex ℓ. We stop when we reach vertex 1.

   (a) **(5 points)** What is the probability that vertex i is visited during the walk?

   (b) **(5 points)** What is the expected time to reach 1?

   (c) **(5 points)** What is the probability that two different vertices i and j are *both* visited during the walk?

   Let G'_n be the graph obtained from G_n by adding self-loops (i, i) at every vertex, and consider the analogous walk on G'_n.

   (d) **(5 points)** How do the answers to the above questions change?

   Remarkably, this simple randomised process can be used to efficiently generate a number uniformly at random from $1, 2, \dots, n$, along with its prime factorisation. Contrast this with the fact that there is no known efficient randomised algorithm to factor a given number.

Hints for the problem set:

1. Suppose the walk has visited m different vertices so far. What is the expected time (from then on) to visit a new vertex—a vertex different from the m visited so far?

2. Collapse all the vertices in the clique except $n/2$ into one "super-vertex", and consider the resulting walk on the line.

3. (a) Take any vertex i. If i is visited last, then *both* its neighbours are visited before it. Condition on the event that $i-1$ visited before $i+1$ (and vice-versa). How does the probability that i is visited last, given this event, depend on i?

   (b) Think of C_n as a chain of n beads. For the hitting time of i, hang the chain by the ith bead so that it looks like a line. Consider the resulting walk on the line.

4. (a) One approach: try to derive a recurrence for the probability.

   A second approach: Imagine that every random choice is made in two stages: at every time step, if we are at a vertex $j > i$, we choose to move to a vertex from $1, 2, \ldots, i$ with probability $i/(j-1)$ and to a vertex from $i+1, \ldots, j-1$ with probability $(j-1-i)/(j-1)$, and *then* pick a random vertex from the corresponding set. What is the probability that the walk visits i now?

   (b) The time taken to hit 1 is the number of vertices visited, i.e., the sum over i of $I(i \text{ is visited})$, the indicator of the event that i is visited by the walk.