1 Touring Hypercube

In the lecture, you have seen that if G is a hypercube of dimension n, then

- The vertices of G are the binary strings of length n.
- u and v are connected by an edge if they differ in exactly one bit location.

A *Hamiltonian tour* of a graph is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that:

- Each vertex appears exactly once in the sequence.
- Each pair of consecutive vertices is connected by an edge.
- v_0 and v_k are connected by an edge.

(a) Show that a hypercube has an Eulerian tour if and only if n is even. (*Hint: Euler's theorem*)

(b) Show that every hypercube has a Hamiltonian tour.

Solution:

(a) In the n-dimensional hypercube, every vertex has degree n. If n is odd, then by Euler's Theorem there can be no Eulerian tour. On the other hand, the hypercube is connected: we can get from any one bit-string x to any other y by flipping the bits they differ in one at a time. Therefore, when n is even, since every vertex has even degree and the graph is connected, there is an Eulerian tour.

(b) By induction on n. When $n = 1$, there are two vertices connected by an edge; we can form a Hamiltonian tour by walking from one to the other and then back.

Let $n \geq 1$ and suppose the n-dimensional hypercube has a Hamiltonian tour. Let H be the $n+1$-dimensional hypercube, and let H_b be the n-dimensional subcube consisting of those strings with initial bit b.

By the inductive hypothesis, there is some Hamiltonian tour T on the n-dimensional hypercube. Now consider the following tour in H. Start at an arbitrary vertex x_0 in H_0, and follow the tour T except for the very last step to vertex y_0 (so that the next step would bring us back to x_0). Next take the edge from y_0 to y_1 to enter cube H_1. Next, follow the tour T in H_1 backwards from y_1, except the very last step, to arrive at x_1. Finally, take the step from x_1 to x_0 to complete the tour. By assumption, the tour T visits each vertex in each subcube exactly once, so our complete tour visits each vertex in the whole cube exactly once.

To build some intuition, here are the first few cases:

- $n = 1$: 0, 1
- $n = 2$: 00, 01, 11, 10 [Take the $n = 1$ tour in the 0-subcube (vertices with a 0 in front), move to the 1-subcube (vertices with 1 in front), then take the tour backwards. We know 10 connects to 00 to complete the tour.]
- $n = 3$: 000, 001, 011, 010, 110, 111, 101, 100 [Take the $n = 2$ tour in the 0-subcube, move to the 1-subcube, then take the tour backwards. We know 100 connects to 000 to complete the tour.]

The sequence produced with this method is known as a Gray code.

2 Trees

Recall that a *tree* is a connected acyclic graph (graph without cycles). In the note, we presented a few other definitions of a tree, and in this problem, we will prove two fundamental properties of a tree, and derive two definitions of a tree we learned from the note based on these properties. Let's start with the properties:

(a) Prove that any pair of vertices in a tree are connected by exactly one (simple) path.

(b) Prove that adding any edge (not already in the graph) between two vertices of a tree creates a simple cycle.

Now you will show that if a graph satisfies this property then it must be a tree:

(c) Prove that if the graph has no simple cycles and has the property that the addition of any single edge (not already in the graph) will create a simple cycle, then the graph is a tree.

Solution:

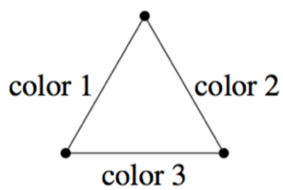
(a) Pick any pair of vertices x, y. We know there is a path between them since the graph is connected. We will prove that this path is unique by contradiction: Suppose there are two distinct paths from x to y. At some point (say at vertex a) the paths must diverge, and at some point (say at vertex b) they must reconnect. So by following the first path from a to b and the second path in reverse from b to a we get a cycle. This gives the necessary contradiction.

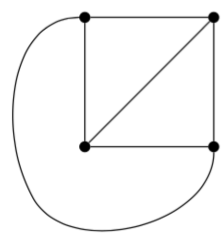
(b) Pick any pair of vertices x, y not connected by an edge. We prove that adding the edge $\{x,y\}$ will create a simple cycle. From part (a), we know that there is a unique path between x and y. Therefore, adding the edge $\{x,y\}$ creates a simple cycle obtained by following the path from x to y, then following the edge $\{x,y\}$ from y back to x.

(c) Assume we have a graph with no simple cycles, but adding any edge will create a simple cycle. We will show that the graph is a tree. We know the graph is acyclic because it has no simple cycles. To show the graph is connected, we prove that any pair of vertices x, y are connected by a path. We consider two cases: If $\{x,y\}$ is an edge, then clearly there is a path from x to y. Otherwise, if $\{x,y\}$ is not an edge, then by assumption, adding the edge $\{x,y\}$ will create a simple cycle. This means there is a simple path from x to y obtained by removing the edge $\{x,y\}$ from this cycle. Therefore, we conclude the graph is a tree.

3 Edge Colorings

An edge coloring of a graph is an assignment of colors to edges in a graph where any two edges incident to the same vertex have different colors. An example is shown on the left.





(a) Show that the 4 vertex complete graph above can be 3 edge colored. (Use the numbers $1,2,3$ for colors. A figure is shown on the right.)

(b) How many colors are required to edge color a 3-dimensional hypercube?

(c) Prove that any graph with maximum degree d can be edge colored with $2d-1$ colors.

(d) Show that a tree can be edge colored with d colors where d is the maximum degree of any vertex.

Solution:

(a) Three color a triangle. Add the fourth vertex, notice that each edge has a different color available from the set of three colors.

(b) 3. Recall that edges connect vertices that differ in a dimension. And each vertex is incident to exactly one edge for each dimension. Thus, the entire set of edges for a specific dimension can be colored with a single color.

(c) By induction on the number of edges. We will use a set of $2d-1$ colors. Remove an edge and $2d-1$ color the remaining graph from our set. This can be done by the induction hypothesis as the remaining graph's degree is no bigger than d and the graph has fewer edges. The edge is incident to two vertices each of which is incident to at most $d-1$ other edges, and thus at most $2(d-1) = 2d-2$ colors are unavailable for edge e. Thus, we can color edge e without any conflicts.

(d) By induction on the number of vertices. Base case is a single vertex, which has no edges to color, and thus can be colored with 0 colors. For the inductive step, we start by removing any leaf v from the tree. We can then color the remaining tree with d colors. Note that vertex v's neighboring vertices has degree at most $d-1$ without the edge to v and thus its incident edges use at most $d-1$ colors. Thus, there is a color available for coloring the edge incident to this vertex.