CS 70 Discrete Mathematics and Probability Theory
Spring 2016 Rao and Walrand HW 3

Due Thursday February 11th at 10PM

1. **Homework process and study group** Who else did you work with on this homework? List names and student ID's. (In case of hw party, you can also just describe the group.) How did you work on this homework? Working in groups of 3-5 will earn credit for your "Sundry" grade.

2. **Stable Roommates Problem** (10 points)

   Suppose you are the head of the residential assistants in charge of assigning roommates. Currently you have four students who are going to move in: Kevin, Bryan, Jimmy and Michael. They have submitted their application with the following preferences:

Kevin	Bryan > Michael > Jimmy
Bryan	Jimmy > Kevin > Michael
Jimmy	Kevin > Bryan > Michael
Michael	Kevin > Bryan > Jimmy

   Consider an modified propose-and-reject algorithm which consists of each person x, one by one, proposing to the first person y on their list and executing as follows:

   - When y is proposed by x, y crosses off everyone below x on y's list.
   - If y holds 2 proposals, y rejects the person y prefers least (crosses off the person on y's list).
   - When x is rejected by y, x crosses off y on x's list and proposes to the next person immediately.

   This continues until everyone holds exactly one proposal. We start with the following proposals and produce the following table:

Kevin	→	Bryan, Bryan crosses off Michael
Bryan	→	Jimmy, Jimmy crosses off Michael
Jimmy	→	Kevin
Michael	→	Kevin, Kevin rejects/crosses off Jimmy, Jimmy crosses off Kevin
Jimmy	→	Bryan, Bryan rejects/crosses off Kevin, Kevin crosses off Bryan
Kevin	→	Michael, Michael crosses off Bryan and Jimmy

Kevin	~~Bryan~~ > Michael > ~~Jimmy~~
Bryan	Jimmy > ~~Kevin~~ > ~~Michael~~
Jimmy	~~Kevin~~ > Bryan > ~~Michael~~
Michael	Kevin > ~~Bryan~~ > ~~Jimmy~~

Since each person only has a list size of one, the algorithm terminates with the pairing: {(Kevin, Michael), (Bryan, Jimmy)}.

Now consider there are two more students who want to apply for housing: Joshua and Justin. Try the algorithm on the following table to find a pairing:

Kevin	Bryan	>	Joshua	>	Michael	>	Jimmy	>	Justin
Bryan	Joshua	>	Jimmy	>	Kevin	>	Justin	>	Michael
Jimmy	Kevin	>	Joshua	>	Bryan	>	Michael	>	Justin
Michael	Justin	>	Kevin	>	Bryan	>	Jimmy	>	Joshua
Joshua	Kevin	>	Justin	>	Bryan	>	Michael	>	Jimmy
Justin	Jimmy	>	Michael	>	Joshua	>	Kevin	>	Bryan

Note: The output of this example will be a stable pairing. However, for any instance, if it has a stable pairing, the algorithm cannot guarantee to find the stable pairing. In fact, the algorithm described above is only the Phase 1 of the Irving Algorithm. With the Phase 2, the Irving Algorithm can always find a stable pairing, if the given instance has one. For more information, please check
https://en.wikipedia.org/wiki/Stable_roommates_problem

Answer:

Kevin	→	Bryan	Bryan crosses off Justin and Michael.
Bryan	→	Joshua	Joshua crosses off Michael and Jimmy.
Jimmy	→	Kevin	Kevin crosses off Justin.
Michael	→	Justin	Justin crosses off Joshua, Kevin, and Bryan.
Joshua	→	Kevin	Kevin crosses off Michael and Jimmy; Kevin rejects Jimmy. Jimmy crosses off Kevin.
Jimmy	→	Joshua	Joshua rejects Jimmy. Jimmy crosses off Joshua.
Jimmy	→	Bryan	Bryan crosses off Kevin; Bryan rejects Kevin. Kevin crosses off Bryan.
Kevin	→	Joshua	Joshua crosses off Justin and Bryan; Joshua rejects Bryan. Bryan crosses off Joshua.
Bryan	→	Jimmy	Jimmy crosses off Michael and Justin.
Justin	→	Jimmy	Jimmy rejects Justin. Justin crosses off Jimmy.
Justin	→	Michael	Michael crosses off Kevin, Bryan, Jimmy, and Joshua.

Kevin	~~Bryan~~	>	Joshua	>	~~Michael~~	>	~~Jimmy~~	>	~~Justin~~
Bryan	~~Joshua~~	>	Jimmy	>	~~Kevin~~	>	~~Justin~~	>	~~Michael~~
Jimmy	~~Kevin~~	>	~~Joshua~~	>	Bryan	>	~~Michael~~	>	~~Justin~~
Michael	Justin	>	~~Kevin~~	>	~~Bryan~~	>	~~Jimmy~~	>	~~Joshua~~
Joshua	Kevin	>	~~Justin~~	>	~~Bryan~~	>	~~Michael~~	>	~~Jimmy~~
Justin	~~Jimmy~~	>	Michael	>	~~Joshua~~	>	~~Kevin~~	>	~~Bryan~~

Pairing: (Kevin, Joshua), (Bryan, Jimmy), (Michael, Justin).

3. **Induction on Graphs** (5 points)

   What is wrong with the following "proof"?

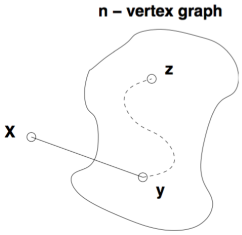
   False Claim: If every vertex in an undirected graph has degree at least 1, then the graph is connected.

Proof: We use induction on the number of vertices $n \geq 1$.

Base case: There is only one graph with a single vertex and it has degree 0. Therefore, the base case is vacuously true, since the if-part is false.

Inductive hypothesis: Assume the claim is true for some $n \geq 1$.

Inductive step: We prove the claim is also true for $n+1$. Consider an undirected graph on n vertices in which every vertex has degree at least 1. By the inductive hypothesis, this graph is connected. Now add one more vertex x to obtain a graph on $(n+1)$ vertices, as shown below.



All that remains is to check that there is a path from x to every other vertex z. Since x has degree at least 1, there is an edge from x to some other vertex; call it y. Thus, we can obtain a path from x to z by adjoining the edge $\{x,y\}$ to the path from y to z. This proves the claim for $n+1$. □

Answer: The mistake is in the argument that *"every $(n+1)$-vertex graph with minimum degree 1 can be obtained from an n-vertex graph with minimum degree 1 by adding 1 more vertex."* Instead of starting by considering an arbitrary $(n+1)$-vertex graph, this proof only considers an $(n+1)$-vertex graph that you can make by starting with an n-vertex graph with minimum degree 1. As a counterexample, consider a graph on four vertices $V = \{1,2,3,4\}$ with two edges $E = \{\{1,2\},\{3,4\}\}$. Every vertex in this graph has degree 1, but there is no way to build this 4-vertex graph from a 3-vertex graph with minimum degree 1.

More generally, this is an example of *build-up error* in proof by induction. Usually this arises from a faulty assumption that every size $n+1$ graph with some property can be "built up" from a size n graph with the same property. (This assumption is correct for some properties, but incorrect for others, such as the one in the argument above.)

One way to avoid an accidental build-up error is to use a *"shrink down, grow back"* process in the inductive step: start with a size $n+1$ graph, remove a vertex (or edge), apply the inductive hypothesis $P(n)$ to the smaller graph, and then add back the vertex (or edge) and argue that $P(n+1)$ holds.

Let's see what would have happened if we'd tried to prove the claim above by this method. In the inductive step, we must show that $P(n)$ implies $P(n+1)$ for all $n \geq 1$. Consider an $(n+1)$- vertex graph G in which every vertex has degree at least 1. Remove an arbitrary vertex v, leaving an n-vertex graph G' in which every vertex has degree... uh-oh! The reduced graph G' might contain a vertex of degree 0, making the inductive hypothesis $P(n)$ inapplicable! We are stuck —— and properly so, since the claim is false!

4. Graphs

(a) (8 points) Suppose we have n websites such that for every pair of websites A and B, either A has a link to B or B has a link to A. Prove or disprove that there exists a website that is reachable from every other website by clicking at most 2 links. (*Hint: Induction*)

Answer: We prove this by induction on the number of websites n.

Base case For $n = 2$, there's always a link from one website to the other.

Induction Hypothesis When there are k websites, there exists a website w that is reachable from every other website by clicking at most 2 links.

Induction Step Let A be the set of websites with a link to w, and B be the set of websites two links away from w. The induction hypothesis states that the set of k websites $W = \{w\} \cup A \cup B$. Now suppose we add another website v. Between this website and every website in W, there must be a link from one to the other. If there is at least one link from v to $\{w\} \cup A$, w would still be reachable from v with at most 2 clicks. Otherwise, if all links from $\{w\} \cup A$ point to v, v will be reachable from every website in B with at most 2 clicks, because every website in B can click one link to go to a website in A, then click on one more link to go to v. In either case there exists a website in the new set of $k+1$ websites that is reachable from every other website by clicking at most 2 links.

(b) (8 points) We have shown in the lecture (or you have read Lecture Note 5) that a connected undirected graph has an Eulerian tour if and only if every vertex has even degree.

Prove or disprove that if a connected graph G on n vertices has exactly $2d$ vertices of odd degree, then there are d walks that *together* cover all the edges of G (i.e., each edge of G occurs in exactly one of the d walks; and each of the walks should not contain any particular edge more than once).

Answer: We split the $2d$ odd-degree vertices into d pairs, and join each pair with an edge, adding d more edges in total. Notice that now all vertices in this graph are of even degree. Now by Euler's theorem the resulting graph has an Eulerian tour. Removing the d added edges breaks the tour into d walks covering all the edges in the original graph, with each edge belonging to exactly one walk.

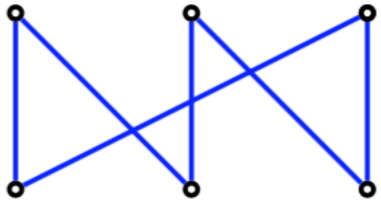
5. **Another Problem on Graphs**

In this problem, we are given a bipartite graph: $G = (L, R, E)$ where there are two sets of vertices, L and R, and $E \subseteq L \times R$, or each edge is incident to a vertex in L and a vertex in R. We also know that every vertex has degree *exactly* d.

We wish to partition the edges into d perfect matchings: a perfect matching is a set of edges where every vertex is incident to exactly one edge in the matching. Another view is that each vertex is matched to another vertex; similar to a pairing in stable marriage except that the pair must correspond to an edge in the graph. A matching is a set of edges where the number of edges incident to any vertex is at most 1 (as opposed to equal to 1 for a perfect matching.)

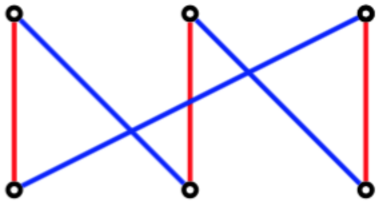
(a) (5 points) Draw a 6 vertex example graph that for $d = 2$ that meets the conditions above for an instance.

Answer:



(b) (5 points) Indicate two matchings in your graph that cover the edges.

Answer:



The red and blue edges each form a perfect matching.

(c) (6 points) Prove that for any instance of this problem that $|L| = |R|$. (Remember every vertex has degree d for any instance.)

Answer: Each edge is incident to one vertex in $|L|$ and in $|R|$, thus the total number of edge incidences with L and $|R|$ is $|E|$.

The edges incidences for L (R) is $d|L|$ ($d|R|$) by definition of degree. Thus,

$$d|L| = |E| = d|R| \implies |L| = |R|.$$

(d) (6 points) Prove that the length of any cycle in an instance of this problem is even.

Answer: Proof: Take a walk along a cycle, since each edge goes between V_0 and V_1, at each step, the set of the resulting vertex alternates in each step. Thus, to to return to the starting point as a cycle must, the alternation must occur an even number of times, and the cycle must have an even number of edges.

(e) (6 points) Prove that you can partition the edges in a simple cycle in this graph into exactly two perfect matchings with respect to the vertices in the cycle.

Answer: Proof: Take a walk along the cycle, and color each edge alternately 1 and 2. Each middle node is adjacent to exactly one edge colored 1 and one edge colored 2. The starting edge is colored 1 and the ending edge is colored two, thus the starting/ending vertex is also incident to only one edge of each color. The colors partition the edges into two sets, and each vertex has degree 1 in each set. Thus each set is a matching on the vertices in the cycle. □

(f) (6 points) Assume d is a power of 2; $d = 2^k$ for some natural number k. Give an efficient algorithm to compute a partition of the edges into perfect matchings. (Note that trying all possible partitions is not efficient. The algorithm should not take exponential time.)

Answer: Algorithm:: Find an eulerian tour in each connected component of the graph. Walk along the path coloring each edge with color 1 and color 2. Now, we recurse on the two degree $d/2$ graphs of color 1 and color 2 edges, and union the partitions of the edges in the two subgraphs.. If the graph has degree 1, we return all the games as the 1 week solution.

(g) (6 points) Prove your algorithm from the previous part is correct.

Answer: Proof: Each intermediate vertex is incident to $d/2$ edges of color 1, and $d/2$ edges of color 2. The starting vertex is also incident to $d/2$ of each color; when it is in the middle of the tour the incoming is one color, the outgoing is another, and the start/end edges are differently colored since any tour has even length.

We can inductively assume that the procedure produces a feasible partition into matchings on the two subgraphs as their degrees are a power of 2. Thus, we get a total of d perfect matchings.

The base case is degree 1, we return a single set of edges which clearly induces degree 1 on the vertices. □

6. **Trees** (10 points)

Show that the edges of a complete graph on n vertices for even n can be partitioned into $\frac{n}{2}$ edge disjoint spanning trees.

Recall that a complete graph is an undirected graph with an edge between every pair of vertices. The complete graph has $\frac{n(n-1)}{2}$ edges. A spanning tree is a tree on all n vertices — so it has $n-1$ edges. So the complete graph has enough edges (for even n) to create exactly $\frac{n}{2}$ edge disjoint spanning trees (i.e. each edge participates in exactly one spanning tree). You have to show that this is always possible.

Answer: We proceed by induction.

Base Case: Consider a complete graph on 2 vertices; this is simply •—•. This can clearly be partitioned into $\frac{2}{2} = 1$ edge disjoint spanning tree, because the graph is already a tree.

Inductive Hypothesis: Assume that the edges of a complete graph on k vertices (for k even) can be partitioned into $\frac{k}{2}$ edge disjoint spanning trees.

Inductive Step: We need to partition the edges of a complete graph G_{k+2} on $k+2$ vertices into $\frac{k}{2}+1$ edge disjoint spanning trees.

To do this, label the vertices of G_{k+2} as $v_1, v_2, \ldots, v_{k+2}$. Remove the vertices v_{k+1} and v_{k+2} (and associated edges) to form a complete graph G_k with k vertices $v_1, \ldots, v_k$. By the inductive hypothesis, G_k has $\frac{k}{2}$ edge disjoint spanning trees; call these trees $T_1, \ldots, T_{k/2}$.

Add the vertices v_{k+1} and v_{k+2} back into G_k to once again form the graph G_{k+2}. These vertices come with $2k+1$ extra edges, connecting (v_i, v_{k+1}) and (v_i, v_{k+2}) for each $i = 1, 2, \ldots, k$, and also (v_{k+1}, v_{k+2}). These edges must be included into spanning trees.

We wish to extend the trees $T_1, \ldots, T_{k/2}$ to include the new vertices v_{k+1} and v_{k+2}. To do this, for each tree T_i, attach two new edges (v_i, v_{k+1}) and $(v_{i+k/2}, v_{k+2})$. This extends each tree T_i to be a spanning tree.

The remaining edges form one additional spanning tree. These edges are $(v_{i+k/2}, v_{k+1})$ and (v_i, v_{k+2}) for $i = 1$ to $k/2$, along with the connecting edge (v_{k+1}, v_{k+2}). These edges connect each of the vertices v_{k+1} and v_{k+2} to half the remaining vertices, and together with the edge between v_{k+1} and v_{k+2} this gives the desired spanning tree.

Therefore, we have covered the graph in $\frac{k}{2}+1$ edge disjoint spanning trees. This completes the induction.

Remark: The key idea here is the following:

Take a graph with k vertices that is partitioned into $\frac{k}{2}$ spanning trees. In the inductive step, we want to add two vertices (with associated edges). To maintain a partitioning into spanning trees, we must expand the preexisting $\frac{k}{2}$ trees to the new vertices, but this is a bit subtle!

We need to add the two new vertices to each of the preexisting $\frac{k}{2}$ trees, which takes $2 \cdot \frac{k}{2} = k$ edges connecting the preexisting k vertices to the two new vertices. *It's really important that we use only one edge out of each of the original vertices!* This is because otherwise, we would use up both new edges out of one of the vertices v_j, but then our final new spanning tree wouldn't be able to reach v_j, so the remaining $k+1$ edges wouldn't be able to form a spanning tree!

So to do this, we need to split the original k vertices into two equal subsets of $\frac{k}{2}$ vertices each, and connect each half to one of the two new vertices. Once we do that, we can then justify forming a new spanning tree from the remaining edges, which allows us to complete the argument.

7. Another Problem on 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 learn from lecture note based on these properties. Let's start with the properties:

(a) (6 points) Prove that any pair of vertices in a tree are connected by exactly one (simple) path.

Answer: 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) (6 points) Prove that adding any edge to a tree creates a simple cycle.

Answer: 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.

Now you will show that if a graph satisfies either of these two properties then it must be a tree:

(c) (6 points) Prove that if every pair of vertices in a graph are connected by exactly one simple path, then the graph must be a tree.

Answer: Assume we have a graph with the property that there is a unique simple path between every pair of vertices. We will show that the graph is a tree, namely, it is connected and acyclic. First, the graph is connected because every pair of vertices is connected by a path. Moreover, the graph is acyclic because there is a unique path between every pair of vertices. More explicitly, if the graph has a cycle, then for any two vertices x, y in the cycle there are at least two simple paths between them (obtained by going from x to y through the right or left half of the cycle), contradicting the uniqueness of the path. Therefore, we conclude the graph is a tree.

(d) (6 points) 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.

Answer: 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.

8. Hypercubes

(a) (10 points) Prove that any cycle in an n-dimensional hypercube must have even length.

Recall that a cycle is a closed (simple) path and its length is the number of vertices (edges) in it. The n dimensional hypercube is a graph whose vertex set is the set of n-bit strings, with an edge between vertices u, v iff they differ in exactly one bit (Hamming distance $= 1$).

Answer: Here are three ways to solve this problem: argue via bit flips, parity of Hamming distance, or induction on n. In each case we try to give credit to solutions according to how clearly they expressed the main idea. However, induction on n is more difficult and prone to build-up error.

Answer 1: Bit flips

Main idea: moving through an edge in a hypercube flips exactly one bit, and moreover each bit must be flipped an even number of times to end up at the starting vertex of the cycle.

Here are a sequence of four proofs roughly based on this idea, starting with the most convincing and ending with the least convincing. Also included is a critique saying what is missing in the later proofs.

Proof 1: Each edge of the hypercube flips exactly one bit position. Let E_i be the set of edges in the cycle that flip bit i. Then $|E_i|$ must be even. This is because bit i must be restored to its original value as we traverse the cycle, which means that bit i must be flipped an even number of times. Since each edge of the cycle must be in exactly one set E_j, the total number of edges in the cycle = $\sum_j |E_j|$ is a sum of even numbers and therefore even.

Proof 2: Let C be a cycle in an n-dimensional hypercube. As we go along the edges of C we must end up where we started. Because traversing an edge in a hypercube flips exactly one bit, this means every flipped bit must eventually be flipped back. This means that the number of edges in C must be even. □

Proof 3: Each edge of the cycle flips one bit. Let the starting point be x, and let the farthest the cycle goes from x be y which is at a Hamming distance of k. Then the cycle must flip all those k bits back to return to x. Therefore the total number of edges in the cycle is $k+k=2k$, an even number.

Proof 4: By induction on n, the dimension of the hypercube. For the induction step, we know that the $(n+1)$-dimensional hypercube is made up of two n-dimensional hypercubes, where every vertex in one n-dimensional hypercube has an edge connected to their 'twin' vertex in the other n-dimensional hypercube. Any cycle in the $(n+1)$-dimensional hypercube has to go back and forth from one n-dimensional hypercube to the other an even number of times, since otherwise it will start in one n-dimensional hypercube, and end in the other, and cannot be a cycle. So each edge in the cycle is either in n-dimensional hypercube or the other or goes between the two n-dimensional hypercubes. The number of edges of each of the first two types is even by the induction hypothesis and the last number is even as shown earlier. Therefore the total number of edges in the cycle is even. □

Critique of Proofs 2–4:

i. Proof 2 is almost correct, and got full credit. However, if one were to be picky one might say that the proof does not make it clear that each bit can be flipped back and forth, and must indeed flip an even number of times (which might be greater than 2) to return to its original value. The proof is also not very explicit about stating that the total number of edges in the cycle is even because the number of edges in the cycle corresponding to each bit position is even.

ii. Proof 3 claims the total length of the cycle is twice the Hamming distance from the starting point x to the farthest point on the cycle. But this claim is false. Starting from x the cycle can move farther from and closer to x many times, and moreover if y is the farthest point in Hamming distance from x, then the number of edges from x to y in the cycle does not need to be equal to the number of edges from y back to x.

iii. Proof 4 is even farther from being a proof. It correctly points out that the number of times the cycle crosses back and forth between the two n-dimensional cubes must be even (this is just saying that the number of times bit 1 is flipped is even). But then it appeals to the inductive hypothesis, and this is quite meaningless, since the intersection of the cycle with the n-dimensional hypercube will not in general look anything like a cycle.

Answer 2: Parity

Main idea: Parity of Hamming distance is equal to parity of number of edges traversed, so to get a cycle we need an even number of edges.

Sample proof: Let C be a cycle in an n-dimensional hypercube, and let x be any vertex in C. Argue (using induction on k or other techniques) that if we start from x and walk for k edges to another vertex y, then the Hamming distance $H(x,y)$ is even if and only if k is even. Each edge traversal brings about a change in vertex binary representation at one bit. Therefore, the Hamming distance changes by 1 and its parity flips when we traverse an edge.

But as we go along the edges of C we must eventually get back to x. At this point the Hamming distance is $H(x,x) = 0$, which is even. This shows that the number of edges in the cycle must be even as well. □

Answer 3: Induction on n

The most common answer made the mistake of a pretty serious build-up error. Here is an example of such a proof:

Proof by induction on n, the dimension of the hypercube.

Base case: For $n = 2$, there is only one cycle and it has length 4, which is even.

Induction Hypothesis: Any cycle in a n-dimensional hypercube has even length.

Induction Step: Let C be a cycle in the $(n+1)$-dimensional hypercube. The $(n+1)$-dimensional hypercube is made up of two n-dimension hypercubes. There are two cases:

i. C lies in one of the two n-dimensional hypercubes. In this case we are done by the induction hypothesis.

ii. C crosses between the two n-dimensional hypercubes. In this case there is an even cycle in each of the two n-dimensional hypercubes. To connect them, we must remove an edge from each of the two cycles in the n-dimensional hypercubes and connect each of the endpoints to their twin vertex in the other n-dimensional hypercube. Now the number of edges in the cycle is $\text{odd} + \text{odd} + 2 = \text{even}$, where $\text{odd} = \text{even cycle} - \text{one edge}$.

□

There are many problems with this proof. First, it completely ignores the fact that the cycle can go back and forth between the two n-dimensional hypercubes a number of times. But even if we were to focus on the special case where it goes back and forth just once, the proof is still seriously wrong. This is because even in this case, the part of the cycle in each n-dimensional hypercube is just a path between two possibly distant vertices, i.e., it need not look anything like a cycle with one edge deleted. In particular this path could be of even or odd length.

There is nonetheless a way of writing down a correct proof by induction. This involves a couple of ideas, including strengthening the induction hypothesis to any tour in the n-dimensional hypercube:

Main idea: A tour in an n-dimensional hypercube can be decomposed into some components in the two $(n-1)$-dimensional subcubes plus an even number of crossing edges. The components in both subcubes can be superimposed to form a tour in the $(n-1)$-dimensional hypercube, allowing us to apply the inductive hypothesis.

Sample proof: We use induction on n.
Base case: For $n = 2$, it is easy to show that every tour has even length.
Inductive hypothesis: Any tour in the $(n-1)$-dimensional hypercube has even length.
Inductive step: Let C be a tour in the n-dimensional hypercube. Consider the decomposition of the n-dimensional hypercube into two $(n-1)$-dimensional subcubes. We decompose C into three parts: the edges that lie in the first subcube, the edges in the second subcube, and the edges crossing the subcubes. Argue that because C is a tour, the number of crossing edges must be even (but not necessarily 2). The edges of C in each subcube do not have to be a tour; in fact they can be collections of disjoint paths. The components in one subcube also don't have to be equal or symmetrical to the components in the other subcube. But argue that when you superimpose them (superimpose the vertices of one subcube with the corresponding vertices in the other subcube), you get a tour in the $(n-1)$-dimensional hypercube. Now apply the inductive hypothesis to conclude that the total number of edges of C in both subcubes must be even. To get the total number of edges in C we need to add the number of crossing edges, which is also even. This completes the inductive step. □

(b) (10 points) A *Hamiltonian path* in an undirected graph $G = (V, E)$ is a path that goes through every vertex *exactly once*. A *Hamiltonian cycle* (or *Hamiltonian tour*) is a cycle that goes through every vertex exactly once. Note that, in a graph with n vertices, a Hamiltonian path consists of $n-1$ edges, and a Hamiltonian cycle consists of n edges.

Prove that for every $n \geq 2$, the n-dimensional hypercube has a Hamiltonian cycle.

Answer:

We proceed by induction on n. In the base case $n = 2$, we have the 2-dimensional hypercube, which is a square graph on $V = \{00, 01, 10, 11\}$. Here we have a Hamiltonian cycle $00 \to 01 \to 11 \to 10 \to 00$.

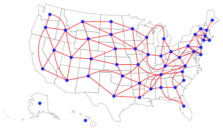
Suppose now that the $(n-1)$-dimensional hypercube has a Hamiltonian cycle. Let $v \in \{0,1\}^{n-1}$ be a vertex adjacent to 0^{n-1} (the notation 0^{n-1} means a sequence of $n-1$ zeroes) in the Hamiltonian cycle. By removing the edge $\{0^{n-1}, v\}$ from the cycle, we obtain a Hamiltonian path in the $(n-1)$-dimensional hypercube that starts at 0^{n-1} and ends at v.

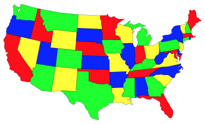
We now want to construct a Hamiltonian cycle in the n-dimensional hypercube. Recall the decomposition of the n-dimensional hypercubes into 0-subcube and 1-subcube, where the 0-subcube (respectively, the 1-subcube) is the $(n-1)$-dimensional hypercube with vertices labeled by $0x$ (respectively, $1x$) for $x \in \{0,1\}^{n-1}$, and every vertex $0x$ in the 0-subcube is connected to the corresponding vertex $1x$ in the 1-subcube.

Then the following is a Hamiltonian cycle in an n-dimensional hypercube: have a path that goes from $0^n \in \{0,1\}^n$ to $0v$ by passing through all vertices in the 0-subcube (this is simply a copy of the Hamiltonian path in dimension $(n-1)$ from 0^{n-1} to v), then an edge from $0v$ to $1v$, then a path from $1v$ to 10^{n-1} that passes through all vertices in the 1-subcube (this is another copy of the Hamiltonian path in dimension $(n-1)$ traveled in reverse), and finally an edge from 10^{n-1} to 0^n. This completes the proof.

9. **Four Colorable?** (10 points)

In the lecture, we have shown that every planar graph can be colored with five colors. We have also shown that it can also be colored with only four colors. The coloring example of U.S. map is shown below. In this question, prove the following: any planar graph of maximum degree 4 has a four coloring.





Answer: We proceed with a proof by induction on the number of vertices n. $P(n)$ denotes that any planar graph of maximum degree 4 has a four coloring.

Base case: $P(n)$ is trivially true for $n <= 4$.

Inductive Step: Assume $P(n1)$ is true: planar graph with $n1$ vertices of maximum degree 4 has a four coloring. Now consider a planar graph with n vertices of maximum degree 4.

Remove an arbitrary vertex v and its incident edges from the graph. The graph with $n1$ vertices still has maximum degree 4 so it's four colorable by hypothesis. Now we try to add vertex v back to the colored graph with $n-1$ vertices. If there is an unused color from all of the neighbors of v, then we can assign an unused color to v and the graph is still four colorable.

Now let's consider the case where v has degree four and every other color is used for its neighbors. For each neighbor of v, we label them clockwise in order v_1,v_2,v_3,v_4, containing corresponding colors 1, 2, 3, 4. Let's consider v_1 and v_3. Say we attempt to change the color of v_1 from 1 to 3. If these two vertices aren't connected, then we can recolor v_1 to color 3 and assign v to color 1. We are done. In the case where v_1 and v_3 are in fact connected, let's consider vertices v_2 and v_4. These two vertices cannot be connected because if a path between those two vertices would have to cut through v_1 and v_3. It can't happen due to the planar graph. We can then assign the color 2 or 4 to vertex v and make v_2 and v_4 have the same color. This makes $P(n)$ true, completing the inductive step.