Lecture 6

Graphs & Coloring

Prep:

* Names in sex graph
* Get subgraph and induced subgraph into recitation notes.
* Get bipartite and bad greedy example into PS4
* Check course numbers in the exam graph
* Check Page rank in plan Lectures 8 – 11
* Find sex book

Take:

* ABC Sex Survey
* Sex Practice Book

Reminders \*\*

* PS4 due Monday at 7:30 pm
* Read sections 5.0 – 5.3

I once co-taught with a professor who was lamenting the fact that as the term progresses, attendance at lecture tends to drop off. It’s pretty dramatic by the end of the term when you’re lecturing but nobody is there. I asked him what he did about it. He said that there are only two things that can get students to attend lecture:

Candy and sex.

Now we’ve already tried candy **… ☺**

So today we’re going to talk about SEX.

In fact, we’re going to use graph theory to address a decades-old debate concerning the relative promiscuity of men vs women.

Graphs are incredibly useful structures in computer science and we’re going to be studying them for the next few weeks. They arise in all sorts of applications, including scheduling, optimization, communications, and the design & analysis of algorithms. In fact, we’ll even show how two Stanford students became gazillionaires because they used graph theory in a clever way. **\*\***

But first let’s talk about SEX.

The issue that we’ll address today is one of the most talked about & well-studied questions in human sociology: “on average, who has more opposite - gender partners, men or women?” By opposite-gender partnerships, I mean exactly one man and one woman. We won’t be counting same-sex or group activities. **☺** I’m not making a political or social statement here – it’s just that the math works out better when you count opposite-gender relationships.

Let’s start by taking a poll in class to get your opinion on this question.

* Raise your hand if you think men tend to have more opposite gender partners than women
* Raise your hand if you think women have more going on (i.e., if women have more opposite gender partners than men)
* Raise hand if equal
* Raise hand if no way of knowing.

In the popular literature, it seems that most people think that men have more opposite gender partners than women (for example, the leader of a harem is always a man, in polygamist culture – the man has multiple wives, not the other way around).

Not surprisingly, this issue has been the subject of many “scientific” studies. In one of the largest, researchers from University of Chicago interviewed a “random sample” of 2500 people over several years to try to get the answer once and for all.

Their study was over 700 pages long and was called “the Social Organization of Sexuality: sexual practices in the U.S.”

**Show Book**

Pretty famous when it was published in the late 90’s. On the cover, it quotes Time magazine as saying it is “The most important survey since the Kinsey report.”

Actually, there is a funny story with this book. When I first got it a few years ago, my daughter (who was 11 years old at the time) got her hands on it and said “Daddy, why do you have a sex book?” I grabbed it back and said it was for my class at MIT. This seemed to go over OK, & I thought I’d gotten out of trouble until the next day when I found out that she had texted all her friends with the breaking news that her Dad teaches sex-ed at MIT. **☺**

Anyway, as one of their central findings, the U Chicago study concluded that:

**Claim (U Chicago): on average, men have 74% more opposite – gender partners than women.**

This is in the USA. Now when you think about it, maybe 74% is about right. But not according to ABC News, who did a poll of 1500 adults in late 2004 and concluded that the percentage disparity is much greater.

**Show Study**

In particular, ABC News claims that the average man has 20 partners and the average woman has 6 for a percentage disparity of 233%. (I think that these numbers are over one’s lifetime.)

**Claim (ABC News): men have 233% more partners than women.**

The ABC News study claimed to be one of the most scientific ever done with only a 2.5% margin of error. We’ll study what margin of error means later in term. This sounds pretty conclusive.

The ABC News study – **show it** – is called “American Sex Survey: a peek between the sheets” -- hmm…doesn’t sound so scientific **☺.** It aired on Primetime Live in 2004. The Promo they ran for the study is really good:

“A groundbreaking ABC News Primetime Live survey finds a range of eye-popping sexual activities, fantasies and attitudes in this country, confirming some conventional wisdom, exploding some myths, and venturing where few scientific surveys have gone before”.

By the end of today, we will probably agree with that last statement!  **☺**

Anyway, who do you think is more right? U Chicago or ABC News? Is the disparity closer to 70% or 230%?

**Vote:** **U Chicago**

**ABC News**

**No way to tell.**

So how should we try to tackle such a problem? Now in theory, we could do our own 6.042 sex survey. But we probably wouldn’t get a meaningful answer and for sure, I would get fired! So I don’t think we are going to do that. ☺

Instead, we will slve this problem using graph theory. Of course, it might be more interesting to interview thousands of people about their sexual practices, but we can get the answer a lot more efficiently by modeling the problem as a graph and then doing a little analysis on the graph.

Let’s start by defining a graph. Informally, a graph is a bunch of dots (called nodes) and lines (called edges) where each line connects a pair of dots.

**x2**

**Draw in order: dots, lines, labels**

**Ex:**

**x4**

**x6**

**SAVE**

**x1**

**x7**

**x3**

**x5**

**Explain node labels or names.**

Now this is a math class, so we’ll be just a bit more formal with our definition.

**Def: A Graph G is a pair of sets (V, E) where**

**V is a non-empty set of elements called vertices (or nodes), and**

**E is a set of 2-element subsets of V called edges.**

Vertices correspond to dots in the picture and edges correspond to lines. We will go back and forth with the terms nodes and vertices—they are equivalent and you can use either.

**Ex:**

**V = x1, x2, x3, …, x7** 7 **-Node Graph**

**E = x1, x2, , x1, x3 , x2, x4 ,…,  x5, x7** 7 edges in this graph

For example, {x1, x2} is an edge connecting x1 and x2. You could also write it as **{x2, x1}** – the order does not matter.

**Edges also written x1 –x2, x1 – x3 …**

Later, when we talk about directed graphs, we’ll put an arrow on it and then the order will matter.

The definition of a graph is pretty simple but there are often small differences in the definitions from one and text to another. For example, some books allow a graph to be empty – or not have any nodes. We won’t allow that in this course because the empty graph is not so interesting and it is a counter example to some Thms we’ll want to prove. But we do allow graphs without edges. For example, this graph is OK:

**Ex:**

**x1**

**x2**

**x3**

**G = (V, E) V = x1, x2, x3 E = ∅** is the empty set.

When you do have edges, we say that:

**Def : Two nodes xi & xj are** said to be **adjacent if** they are connected by an edge, i.e.if **{ xi, xj}∈E.**

For example, x5 is adjacent to x7 but not x4 since there is no edge there.

There is a closely related notion called incident:

**Def : An edge e = {xi, xj} is** said to be **incident to xi and xj** (i.e. to the nodes that it joins).

**Draw e on x1 –x2 in saved graph & do example**

Then we can talk about the degree of a node:

**Def: the number of edges incident to a node is called the degree of the node.**

**Q/A Ex: deg (X5) = 3 deg (X7) = 1**

Show degree 0 node in empty graph

In this class, we will only look at simple graphs.

**Def: A graph is simple if it has no loops or multiple edges.**

A loop is an edge that has both endpoints at the same node.

**Loop**

Multiple edges are 2 or more edges with the same pair of end points.

**Multiple edges**

**AKA: Multiedges**

A graph with multiple edges is called a Multigraph. But we usually won’t allow that.

Any questions?

Ok, let’s use a graph to model the problem of opposite gender partners.

**Q:** What do you think the nodes will represent?

**A:** People and in this case there are two kinds of people: men & women.

**Draw Tom & Nicole first & work down**

**Draw edges later**

**Men Women**

**SAVE**

Tom Cruise Nicole Kidman

Ben Affleck Penelope Cruz

Jude Law Katie Holmes

Keith Urban Dozens more

**.**

**.**

**.**

Josh Hartnet

**Q:** What do edges represent?

**A:** Partners.

Draw in edges

Actually, this is a pretty hard graph to figure out. Can get a lot of data on famous people from [www.whosdatedwho.com](http://www.whosdatedwho.com). But that is just a small sampling of what is really going on out there. So we really will never know all the edges. Also, remember we are only drawing in the edges for opposite-gender relationships. If there is anything going on between Tom & Ben, we don’t want to know about it. **☺**

Now this graph is pretty big. In the USA, there about 300 million nodes. Of this, 50.8% are women and 49.2% are men.

**|V| = 300 Million**

**Label Vm and Vw on graph**

Cardinality notation

**|Vm| = 147.6 Million**

**|Vw| = 152.4 Million**

**Explain cardinality notation**

**Q:** Any idea of # edges? **|E| = ??**

**A:** Nope – I certainly don’t know—not even the surveys really know. But maybe we won’t need to…

What we are trying to figure out is the ratio of the average number of opposite gender partnerships for men to the average number of opposite gender partnerships for women.To figure this out, we:

**Def: Am = average number of opposite gender partners for men**

And we let Aw be the same thing for women

**Aw = average number of opposite gender partners for women**

Question to resolve: **What is Am/Aw ?** Is it **= 1.74?** or **3.33?**

**UChicago ABC News**

**N**ow we are going to figure out the real answer using a little graph theory. So let’s start by figuring out an expression for Am.

It is just the average degree of male nodes.

**Sum of degrees**

****

**Number of men**

**Q:** Is there a simpler form for the sum?

**A:** Yes

= 

**Explain **

This is where we use opposite gender partners – since every edge gets counted once, not twice or zero times. Explain.

Next, let’s figure out the average number of partners for women. Same idea as for men:

**Aw** = ** =** 

So let’s look at Am/Aw.

 **=** 

Wow! This is nice! We don’t know |E| but it doesn’t matter since the |E| term cancels and we don’t need to know it.

**=** 

**= 1.0325**

This means that the average man has about 3% more opposite gender partnerships than the average women. So U Chicago study was way off! ABC News study is completely nuts! Huge effort to get a totally wrong answer!

The amazing thing is that the answer really has nothing to do with the promiscuity of men vs. women –it just has to do with the relative number of men and women.

So why is this true? What is going on? **Collectively**, men and women have the same number of opposite gender partnerships since it takes two to tango. You have one man and one woman in each partnership – but there are fewer men so they must have a higher rate of relationships.

So it’s really a stupid question to do a study on since it can be easily answered using math. It has nothing to do with behaviors – just the population count!

As it turns out, there have been lots of studies like this and they always miss the same underlying issue. For example, a few years ago, The Boston Globe ran an explosive story on a survey of the study habits of students on Boston area campuses. Their survey showed that on average, minority students tended to study with non-minority students a lot more than the other way around (i.e., than non-minority students study with minority students). They went on at great length about why this “**remarkable phenomenon**” might be true, and they came up with all sorts of explanations as to why minority students study with non-minority students compared to the other way around.

**Q:** Now can anyone tell me why it is certainly true and not so surprising that the average minority student has more non-minority study partners than the other way around?

**A:** Because they are a minority! By definition, there are fewer minority students than non-minority students. End of story!

This is about as basic as it gets! We don’t need a study or a Sociology PhD from down the street to explain it to us. **☺**

We’ll see lots of other bogus studies later, especially when we get to probability at the end of the term.

OK, that’s all I’m going to say about sex today. Any questions?

In the partner example we have been discussing, we used the notion of an edge to denote a partnership or affinity that exists between two nodes. There are many examples in computer science where edges are used to denote just the opposite - namely, where an edge is used to represent a conflict between two nodes.

For example consider the problem of scheduling final exams at MIT. Every class with a final needs to get a slot in the schedule but you don’t want to schedule two exams at the same time if a student is taking both courses.

For instance, suppose we need to schedule exams for 6.041, 6.042, 6.002, 6.003 and 6.034.

Draw on board as shown below.

We can model the scheduling constraints as a conflict graph where there is a node for each exam and an edge joining two nodes if the corresponding exams cannot be held at the same time (since a student is in both courses).

For example, 6.041 & 6.042 cannot have their exams at the same time since there are students in both courses ⇒ edge. Draw on graph. But 6.042 and 6.034 can be at the same time since don’t have students in both – so no edge. Overall, the conflict graph might look something like this: Draw all edges:

**6.042**

**6.041**

**6.002**

**6.003**

**6.034**

**SAVE**

Ex:

Now suppose the allowed time slots for the exams are:

**Slots**

**Wednesday 5:00 – 7:00 pm**

**“ 7:00 – 9:00 pm**

**SAVE**

**“ 9:00 – 11:00 pm**

**“ 11:00 – 1:00 am**

**“ 1:00 – 3:00 am**

Your job is now to assign a slot to each exam (or node) so that adjacent nodes get different slots (e.g. 6.002 & 6.042). It’s ok to have several exams in a row – after all this is MIT – not that school down the street – but even at MIT, you don’t have to take two exams at once. **☺**

If we’re willing to have exams in the middle of the night, this is an easy problem to solve – every exam gets its very own slot. No conflict that way – but hard on faculty who have to stay up to give exam. **☺**  Much better if we can squeeze all the exams into a few slots, but without creating conflicts. This turns out to be a well-known problem in graph theory called the

**Graph Coloring Problem: Given a graph G and k colors, assign a color to each node so adjacent nodes have different colors.**

Then the minimum number of colors you need to color the graph is called the chromatic number of the graph.

**Def: The minimum value of k for which such a coloring exists is the Chromatic Number (G) of G.**

**Q:** What do the colors represent in our exam scheduling problem?

**A:** Time slots.

**Go to slots chart and add color**

**C1**

**C2**

**C3**

**C4**

**C5**

Now we could color the graph with 5 colors by using a different color for each node but that would have a different slot for every exam and we would be up all night, so let’s see if we can do better than 5 colors in this example.

**6.034 C1**

**6.041 C1**

**6.003 C2**

**6.002 C3**

**6.042 C4**

Yikes – 6.042 got the 11 – 1 am slot. Not so good.

**Q:** Can we do better? Fewer slots?

**Take vote on whether possible to do with 3 colors.**

**Q:** Anybody see how to do it?

**A:** **Do with three colors**

Good news – don’t need the 11 pm slot on Wednesday night!

**Q:** Can anybody do it in 2 colors?

**A:** No!

**Q:** Can anybody say why 2 colors is not possible?

**A:** Can’t do 2 since in all three must be different.

So 3 is optimal and for this graph, the chromatic number is 3.

**Questions?**

It turns out that figuring out the chromatic number for large graphs is usually really hard to do. No one knows of a fast algorithm that works for every graph. The weird thing is that given a coloring, it is easy to check whether or not it is valid – you just look at every edge & check that the colors on either end are different, but the only way known to find a coloring with the minimal number of colors is to essentially try all possibilities and that takes exponential time. In other words, if the graph has n nodes, the time is exponential in n & that is huge when the graph has 100 or more nodes.

In fact, even trying to figure out whether or not the graph can be colored with 3 colors is hard. This is called the 3-coloring problem. Sometimes it is easy to determine if a graph can be colored with 3 colors, as it was in this case, but in general it is very hard, and the best algorithms known take exponential time.

Graph coloring is a classic example of what’s called an NP – complete problem. How many people ***don’t*** know what it means to be an NP –complete problem? OK, so it’s worth explaining this concept.

It turns out that there are thousands of optimization problems like graph coloring where it is easy to check whether or not a proposed solution is valid but where no one knows a fast algorithm that always finds a solution quickly. And it turns out that many of these problems are equally hard in the sense that if you found a fast algorithm to solve one of them, you could turn it into a fast algorithm to solve the other problems. Such problems are said to be NP-Complete. This means that either every NP-complete problem can be solved quickly, where quickly means in polynomial time—for example in n^2 or n^3 time where n is the size of the problem instance. Or none of them can be solved in polynomial time.

So if you solve one NP-complete problem, you solve them all! Even better, you win one of those $1 million Millennium prizes that we talked about in the first lecture.

This is known as the P = NP ? problem. P and NP are classes of problems. Roughly speaking, P is the class of problems solvable in polynomial time and NP is the class of problems where you can check a solution in polynomial time. If P=NP, then there is an algorithm that quickly solves all the problems like graph coloring. If P =/ NP, then there is no such algorithm.

Most everyone thinks polynomial time algorithms do not exist for NP-complete problems (i.e., that P =/ NP) but no one has been able to prove it. Even if you prove that algorithms do not exist for an NP-complete problem, you still win $1 Million, since then you would have shown that no NP-complete problem has a fast solution.

Lots of people have claimed to solve the problem. A few years ago, there was a lot of buzz when a well-known scientist at HP Labs claimed to have found a solution – but his proof was found to have a fatal error a few weeks later.

Questions?

Because graph coloring comes up so often in practice, we need to find some solution, even if it is not optimal. In fact, you will often face this kind of challenge in your careers when you encounter an NP-complete problem. It’s unlikely that you will come up with the optimal answer, but you’ll still need to come up with something. You can tell your boss that the problem is NP-complete but he or she will still want you to do something.

As an example, suppose your future boss gives you a graph and asks you to color it with a small number of colors. How would you go about it? Your approach doesn’t have to always use the minimum number of colors –if it did, you could win a million dollars and quit your job—but it can’t take 2^100 steps. What would you do?

**Explore ideas from class**

**Settle on following ALG**

**“Basic”** (Graph) **Coloring Alg for G=(V,E):**

1. **Order the nodes v1, v2, …, vn**
2. **Order the colors C1, C2, …**
3. **For i = 1, 2 …, n:**

**Assign the lowest legal color to vi.**

In fact, this is exactly how we colored the exam graph.

**Show example – label nodes 6.034, 6.041, 6.002, 6.003, 6.042 as v1, …, v5 to make 4 colors.**

**Explain what it means to be legal**

Questions?

By using different orders, we can get different numbers of colors – in the exam graph, this order gave 4 colors but a different order can get 3 colors. **Do order 6.034, 6.003, 6.002, 6.041, 6.042.**

A lot of work has gone into finding good orders for coloring graphs in this way – in fact, my junior thesis when I was an undergrad was to figure out good orders.

**Tell Story—Recursive Largest First**

**Students wanted exams before holidays-faculty after. Registrar said exams wouldn’t fit in the week before break.**

**Algorithm did it in 4.5 days—much better than 2 weeks.**

**Caused big stir—reported in campus paper.**

**Faculty had to come clean—but didn’t change.**

The “Basic” algorithm is an example of what is known as a Greedy Algorithm. It is greedy in that we just plow through the graph, always choosing the lowest (or best) possible color for each node. We never go back and try to do things better, or do recursion or anything that is locally sub-optimal in the hopes of getting a better eventual, global result.

Greedy algorithms often work well in practice. Like this algorithm, they are simple to code and they often give a good result. Because they are simple, you can also sometimes analyze their performance, at least for certain classes of graphs.

For example, we can prove that the Basic Graph Coloring Algorithm does very well if the graph does not have any nodes with high degree. This is true no matter what ordering is used. In particular.

**Theorem: For all d, if every node in G has degree ≤ d, then Basic Alg uses at most d + 1 colors for G.** (No matter what ordering is used for the nodes, you will never do worse than d+1 colors).

**SAVE**

**Q.** For example, what is the maximum degree of our exam graph?

**A.** d = 3

So this means that the Basic Alg was guaranteed to use at most 4 colors, which it did – it even did a little better with the second ordering.

This is a very useful result to prove since you could have a graph with thousands of nodes and max degree 3—then the theorem says you can quickly get a coloring with at most 4 colors.

**Proof: ideas?**

**By induction**

**What is the next step?**

**I.H. P(d)**

Unfortunately, if you go down this path, it will lead to disaster.

Tell story of nightmare on exam

Base case works but when you try inductive step, you’ll need to convert a graph with maximum degree d+1 into a graph with maximum degree d by removing lots of edges, then color it with d + 1 colors then add back edges and patch up coloring using only 1 more color. The proof is possible but very hard and would take a ton of time—killed class on exam.

Now we tell you if IH doesn’t work, then try something stronger. But it turns out that that approach doesn’t work here either since it turns out that any stronger hypothesis won’t be true. (There are graphs where you can’t use less than d+1 colors.) So this looks like trouble.

It turns out that when you are doing induction on graphs, you should always try to induct on the number of nodes in the graph. And if that doesn’t work, then try inducting on the number of edges. In this case, we were inducting on the maximum node degree and that ends up making the proof very hard.

This is really important to remember on exams. We always ask you a graph induction problem on first test and often students will try to induct on wrong variable (say d) and it inevitably causes disaster. So be sure to look for alternatives if having trouble with Induction Step in proof.

So let’s try that here. Start by modifying IH to include n as the parameter:

**P(n): For all d, if every node in an n-node graph G has degree ≤ d, then Basic Alg uses at most d+1 colors for G.**

Note: this is not a strengthening of I.H. like we saw in second week of class – just inducting on a different parameter (nodes instead of colors).

**Q:** What’s next?

Q/A

**d = 0**

**1 color = d + 1 colors**

**A:** **Base Case: n = 1 ⇒ 0 edges**

**Q: What’s next?**

**A: Ind Step: Assume P(n) for purposes of induction.**

**Let G = (V, E) be any (n+1) – node graph and d be the max degree in G. Need to show Basic Alg uses d + 1 colors for G. Let’s see what happens. First step is to order the nodes: v1, v2, … vn, vn+1, any way at all.**

**Q:** Any ideas how to use the IH? I know that for any n-node graph, d+1 colors works when d is the max degree of the n-node graph.

**A:** Apply it to subgraph on first n nodes. In other words: **Remove vn+1 from G to create G/ = (V/, E/)**

Now when we remove a node from a graph, we also have to remove all the edges incident to the node too. For example, if we remove V5 = 6.042 from the exam graph, must pull out 3 edges incident to 6.042 – can’t have edges without endpoints.

* **Show on example –**

The resulting graph G’ has n nodes.

**Q:** What is max degree of G/?

**A:** ≤d. since degree does not rise when you remove nodes & edges. Might go down but can’t go up.

**Q:** So what can we say about number of colors used by Basic Alg on G**/** for ordering v1, v2, … vn.

**A:** ≤d + 1

**G/ has max degree ≤ d and n nodes so P(n) says that Basic Alg uses ≤ d+1 colors for v1, v2, …, vn.**

Use coloring of G/ to color G.

**Show on 6.042 graph**

All we have left to do now is color vn+1. Hopefully, we won’t need color d+2 or our proof won’t work—we need to color vn+1 with one of first d+1 colors. Lets see how to do this.

**u1**

≤ d neighbors. A neighbor is a node that you are adjacent to.

**vn+1**

**u2**

****

****

****

**ud**

**vn+1 has ≤ d neighbors ⇒**

**explain <=d colors ruled out, leaving at least 1 color left**

**∃ color in C1, C2, …, Cd + 1 not used by any neighbor. Give vn+1 that color.**

**⇒ Basic Alg uses ≤ d+1 colors on G ⇒ P (n+1) is T**

Questions?

This proof is very typical of induction proofs on graphs and when you need to prove something about an algorithm.

Ok – so we know how to color any graph with max degree d using at most d + 1 colors.

**Q:** Could we have improved this bound in general?

**A:** No. Sometimes d + l colors is the best you can do.

**Q:** Can anyone think of a graph for which this bound is optimal?

**A:**

**Kn**: **n – node graph with all possible edges**

**d = n – 1**

**χ(Kn)= n = d + 1**

since every node must have diff color.

This is called **complete graph** Kn

**aka clique** – just like clique

of friends (edge = friends)

Sometimes, the d+1 bound is very far from opt. Can anyone think of example where it is far from Opt?

**A: Ex:**

**n – node Star**

**d = n – 1 Q/A**

**# colors = 2**

**Q:** How well would Basic Alg do here?

**A:** 2 Colors. Optimal. Doesn’t matter how you order nodes.

**Explain**

**Questions?**

So the upper bound in the Theorem is way high, but the Basic Algorithm still did ok. Turns out that basic algorithm often does well. In fact, you might start to think that it is always good and try to collect the $1M prize. ☺

Unfortunately, they won’t give it to you since there are examples of large-degree graphs where the Basic Algorithm performs terribly. In fact, you will see one in PS4.

Coloring problems come up in all sorts of applications.

For example, at Akamai, we run a network of 150,000 **\*\*** servers to distribute content on the Internet and we deploy a new version of the software running on the 150,000 **\*\*** servers every few days. Now we can’t deploy on every server at the same time since we have to take down server to deploy software and if we took down every server at once – that would be a big problem. We also can’t do each server one at time since it would take forever (well not forever, but it would take over ten years, since each server takes about an hour).

Moreover, certain pairs of servers can’t be taken down at the same time since they have common critical functions.

In fact, we solve this problem by making a 150,000 – node conflict graph and coloring it. There is a node for each server and an edge between any pair of servers that cannot be installed at the same time.

8 colors ⇒ 8 waves of install.

At a much smaller scale, the same problem exists with register allocation for variables. Here, you need to assign every variable to a register & you can’t have variables that are active at same time assigned to the same register. # colors = # registers needed.

The most famous example of graph coloring is the map coloring problem. In this case, a country or state on a map corresponds to a node and an edge joins two nodes if the corresponding territories on the map share a border. The question is how many colors are needed so that adjacent territories get different colors (need different colors for adjacent countries so you can tell they are different countries). As we mentioned in the first lecture, it was ultimately shown that 4 colors suffice – very famous result.

Do if time

As one last example of a coloring problem, suppose we need to assign frequencies to radio stations or cell towers. If two towers have an overlap in their broadcast area, they can’t be given the same frequency.

Frequencies are limited and a very expensive commodity (companies pay billions of dollars to the gov’t for them) … and so we want to minimize the number of frequencies are needed.

**Q.** For example: how many frequencies are needed in this case?

**A**

**A**

**B**

**C**

**E**

**D**

**B**

**E**

**C**

**D**

**Explain 4, 3, not 2**

1. **X (G) = 3** frequencies necessary & sufficient

Graph coloring comes up all over the place – you will encounter several of them in your career.