Introduction to Graphs

Wednesday, January 11, 2023 6:58 PM

Introductions and Early Motivation

Course Website: https://courses.math.rochester.edu/current/248
Office Location: Hylan 711
Office Hours: TBD
Book: Introduction to Graph Theory (2nd Edition) - Douglas B West

Questions about Student Backgrounds and Interest: (answers to be submitted to me)

- What are your names and what's a fun fact about you?
- Which of these topics have you studied in the past/feel comfortable with the basics of?

Math	Linear algebra, Set Theory, Combinatorics, Probability, Proof Writing
CS	Basic programming, algorithms or data structures

- What are your main topics of interest? What subjects motivate your interest most/what would you like to get out of this class?

Idea:

Graphs provide a mathematical way to encode the idea of *adjacency*

Example: (Koenigsberg Bridge Problem)

Can one cross all seven bridges in Koenigsberg without crossing any of them twice and end up back where they started?

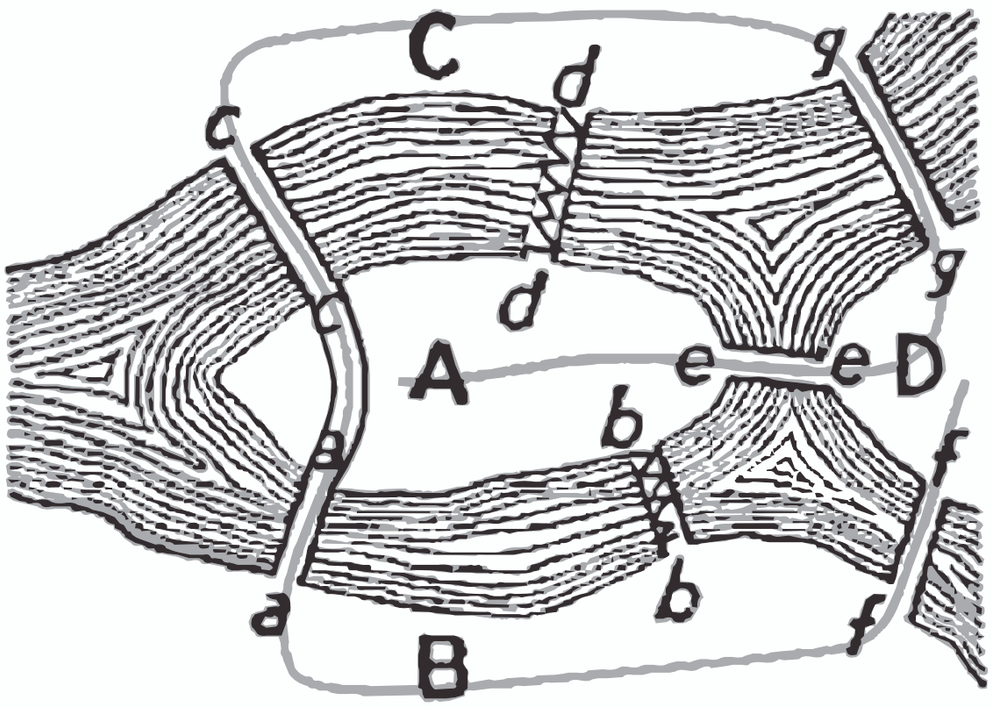
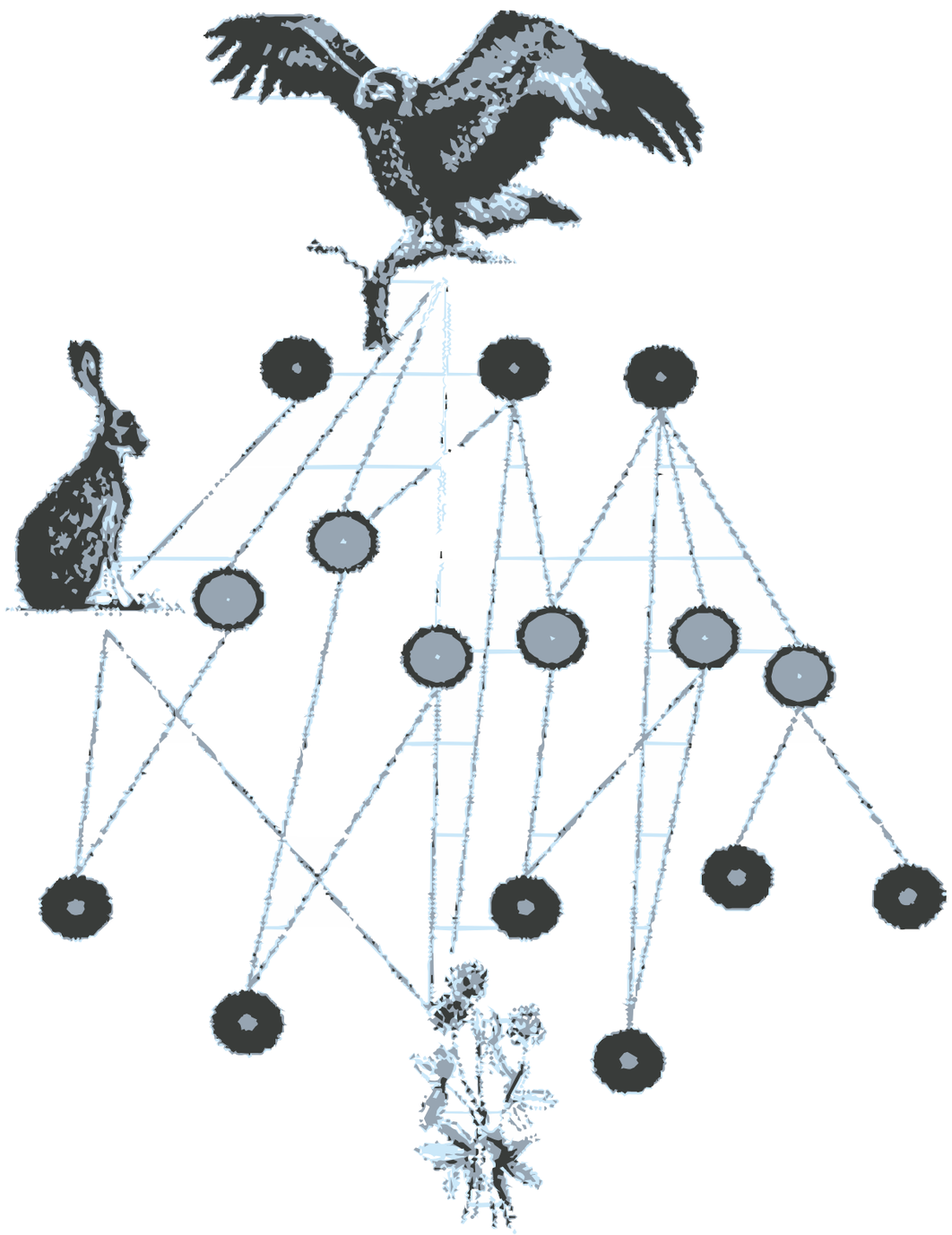


FIGURE 98. *Geographic Map: The Königsberg Bridges.*

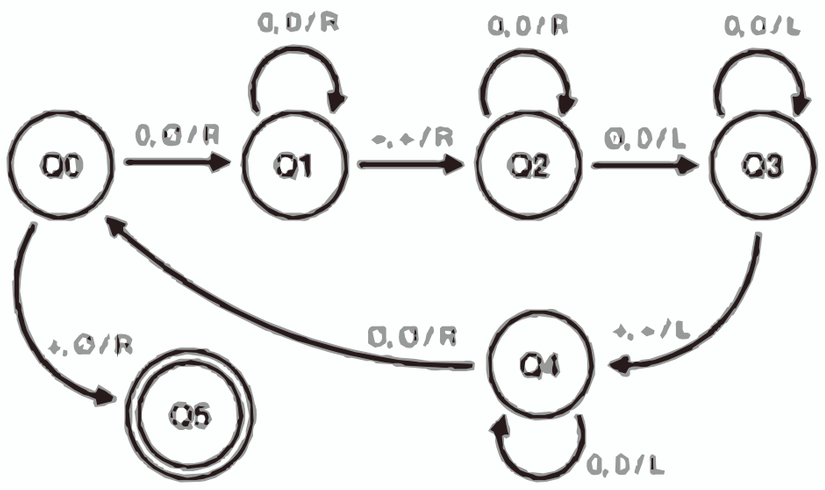
Example: (Biological, Ecological, Chemical Networks)

- How robust is a food web to the extinction of a single species?
- Which stages of a complex chemical process are rate-limiting?
- Given a large complicated system with many parts (many species, many chemical compounds, many proteins, etc.) how can we measure the complexity of the structure in a rigorous way?



Example: (Computational and Algorithmic Questions)

- How can we mathematize the idea of computation?



- How can we encode the intuitive idea of a network in a way computers can effectively process?

Definition:

A *graph* G is a triple consisting of
- a vertex set V or $V(G)$
- an edge set E or $E(G)$
- a relation from E to V associating each edge to either one or two vertices (called *endpoints*)

We will often write $G = (V, E)$ for convenient shorthand

Note:

One may wish to exclude the null graph with $V = \emptyset$ from consideration.

Review as Necessary:

- Basic set theory
  - equality of sets
  - set notation
  - empty set
  - subsets
  - cardinality
  - union, intersection, complement
  - disjointness of sets, partitions of sets
  - Cartesian product and tuples
  - relations, equivalence relations, equivalence classes
  - basic modular arithmetic as an example of the previous

- Proof theory
  - What is a proposition
  - conditional statements
  - contrapositives of statements
  - logical quantifiers, basic structure of proofs of them, negation of quantifiers
  - Direct proof, contrapositive proof, proof by contradiction
  - Induction
  - Recurrence relations
- Functions
  - Functions as mappings from domain to codomain
  - composition of functions
  - injectivity and surjectivity
  - bijections and inverse functions
  - growth rates of basic functions (bounded functions, logarithms, polynomials, exponentials, factorials, etc.)
- Combinatorics
  - summation notation over finite sets
  - permutations of finite sets
  - binomial coefficients, n choose k, the Binomial Theorem
  - Combinatorial proofs
  - Pigeonhole Principle

Graph Theory!

Definition:

A *loop* is an edge where both endpoints are the same

Definition:

Multiple edges are a collection of at least two edges all having the same endpoints as one another.

A Basic Classification:

	May Not Have Loops	**May Have Loops**
May Not Have Multiple Edges	Simple Graph	
May have Multiple Edges	Multigraph	Pseudograph (or sometimes Multigraph)

In what follows, we will in general assume that the term "graph" refers to a simple graph unless otherwise specified.

Definition:

A graph G is finite if both V and E are finite sets.

Definition:

If two vertices $u, v \in V$ are the endpoints of an edge in $e \in E$, we say that they are *adjacent*, that they are *neighbors*, that e *connects* u and v, and we write $u \leftrightarrow v$

Note:

For a simple graph, we can identify edges with their two endpoints, so we will often refer to "the edge uv" to denote the edge connecting these two vertices

Note:

We can think of a simple graph as a pictorial representation of a symmetric (but not reflexive!) relation

Motivating Question:

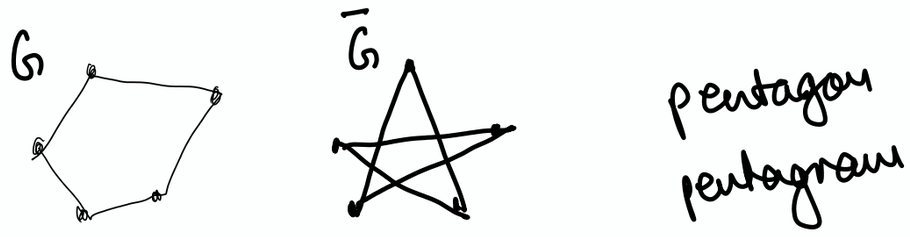
Does every group of six people contain a subset consisting of three people such that either
none of those people know either of the other two
each of those people knows both of the other two
?

(this is an example of a *clique finding problem*)

Let V be a set of 6 people

Define a graph with edges connecting pairs of people who know each other

We could also define a graph with edges connecting pairs of people who don't know each other



Definition:

Let $G = (V, E)$ be a simple graph. The *complement* of G is the graph $\overline{G}$ on the same vertex set V with edge set E such that $uv \in E$ if and only if $uv \notin E$

Definition:

A *clique* is set of vertices which are pairwise adjacent.

Carson will solve my clique finding problem.

Definition:

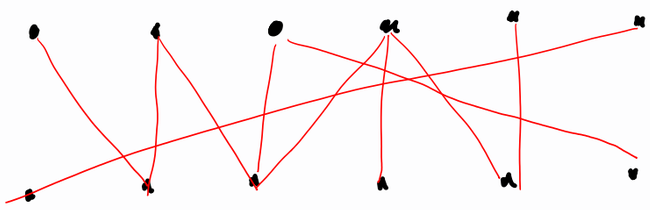
An *independent set* is a set of vertices which are pairwise nonadjacent

Motivating Question:

Suppose that you are attempting a complicated group project with several parts. Each person in your group is to do one part. Knowing that each person may be better at some tasks than others, how can you assign tasks to people?

(this is an example of a *matching problem*)

We could draw a graph with vertices consisting of tasks and people, connecting each person to all of the tasks they can do well.

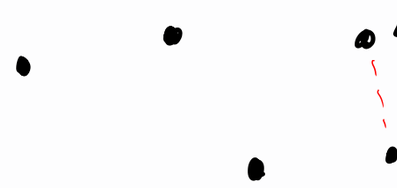


(Bipartite graph

Definition:

A graph $G = (V, E)$ is *bipartite* if $V = V_1 \cup V_2$ is the union of two disjoint independent sets.

Instead of Matthew's Sudoku puzzle, lets do assigning committees to timeslots:



← committees.

Committees are adjacent if they share have a common member.

Color the committees such that no 2 adjacent vertices have the same #.

Motivating Question:

How many colors do we need in order to draw a map of the world so that no adjacent countries have the same color?

Definition:

The *chromatic number* of a graph G is $\chi(G)$, given by the minimum number of distinct colors needed to label vertices so that all adjacent vertices receive different colors

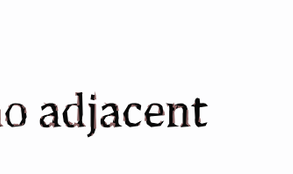
Definition:

A graph G is k-*partite* if $V(G)$ can be written as the union of k disjoint independent sets - called *partite sets*. (possibly empty disjoint union)

possibly empty

Proposition:

A graph G is k-partite if and only if $\chi(G) \leq k$



vertices given the same color must form an indep set.

key is that any indep. set can be broken up into more (possibly empty) indep sets

Motivating Question: (for paths)

What is the fastest route for me to travel to my home?

Let the vertex set represent road intersections and the edges the roads connecting them
Label each edge by distance or travel time

Definition:

A *path* is a simple graph whose vertices can be ordered so that two vertices are adjacent if and only if they are consecutive in the list.

→ implies vertices cannot be repeated (no loops)

Definition:

A *cycle* is a graph with an equal number of vertices and edges which can be drawn in a circle with consecutive edges in the circle connected.

Definition:

A *subgraph* of a graph G is a graph H such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ such that any edge in $E(H)$ has the same endpoints in H that it does in G

We say *G contains H* or *contains a copy of H*

Take $xy \in H$. $xy \in E(H)$ iff $xy \in E(G)$

Definition:

A graph G is *connected* iff each pair of vertices in G belongs to a path contained in G
Otherwise, G is *disconnected*.

Matrix Representation of a Graph

Definition:

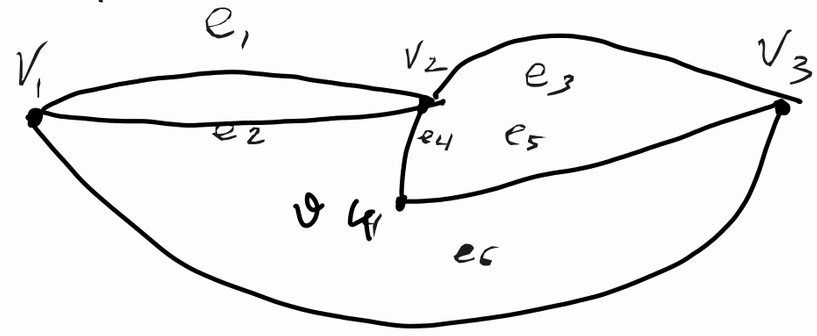
Let G be a graph without loops. Suppose that we fix the order of the vertices in $V(G)$ as $v_1, v_2, \ldots, v_n$

The *adjacency matrix* of G is the n by n matrix $A(G)$ with entries $a_{i,j}$ given by the number of edges in G with endpoints $\{v_i, v_j\}$

Now, let us also fix the order of the edges in $E(G)$ as $e_1, \ldots, e_m$.

The *incidence matrix* of G is the n by m matrix $M(G)$ with entries $m_{i,j}$ equal to 1 if v_i is an endpoint of edge e_j, and equal to 0 otherwise.

Ex:

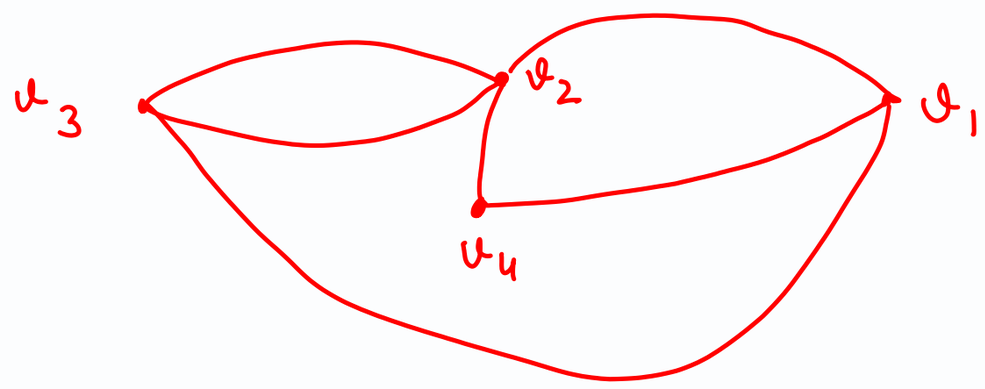


What are the adjacency matrix and incidence matrix of this graph?

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_1 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

$\sigma = 3214 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 4 \end{pmatrix}$



$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

↓ permute rows

$$\begin{pmatrix} 1 & 1 & 0 & 1 \\ 2 & 0 & 1 & 1 \\ 0 & 2 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

↓ permute columns

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Question:

Suppose we consider the square of the adjacency matrix $A^2(G) = A(G) \times A(G)$ for the above graph.

How can we interpret the entries of this matrix?

leave as open question if they don't figure it out.

$$A^2(G) = \begin{pmatrix} 0&2&1&0 \\ 2&0&1&1 \\ 1&1&0&1 \\ 0&1&1&0 \end{pmatrix} \begin{pmatrix} 0&2&1&0 \\ 2&0&1&1 \\ 1&1&0&1 \\ 0&1&1&0 \end{pmatrix}$$

$$= \begin{pmatrix} 5&1&2&3 \\ 1&6&3&3 \\ 2&3&3&1 \\ 3&3&1&2 \end{pmatrix}$$

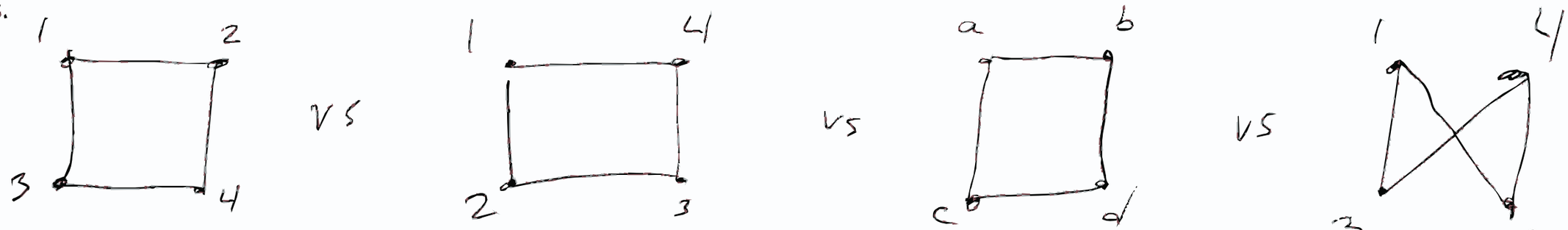
Question: What happens to the adjacency matrix/incidence matrix if we list the vertices/edges in a different order?

Question:

Construct a graph which has the following adjacency matrix

$$\begin{pmatrix} 0&2&1&0&0 \\ 2&0&0&3&1 \\ 1&0&0&1&0 \\ 0&3&1&0&0 \\ 0&1&0&0&0 \end{pmatrix}$$

What if we don't really care about the specific labels we've given to a vertices of a graph? We'll often care more about "structural properties" of a graph that would be the same no matter what we called the vertices.



What properties does a "relabeling function" of a graph have?

"bijection on vertices that preserves adjacency

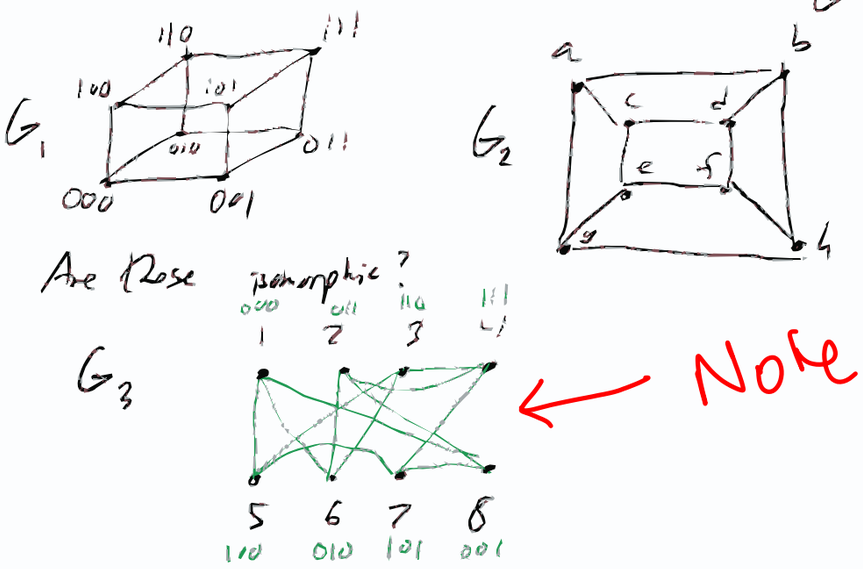
Definition:

Let G and H be simple graphs. An *isomorphism* or *graph isomorphism* is a bijection $f: V(G) \to V(H)$ such that $uv \in E(G)$ if and only if $f(u)f(v) \in E(H)$

If there exists a bijection between G and H, we write $G \cong H$

G is "isomorphic" to H

Ex



Note that 1234 are an indep set.

a h e d

c f b g

← edges have wrong degree

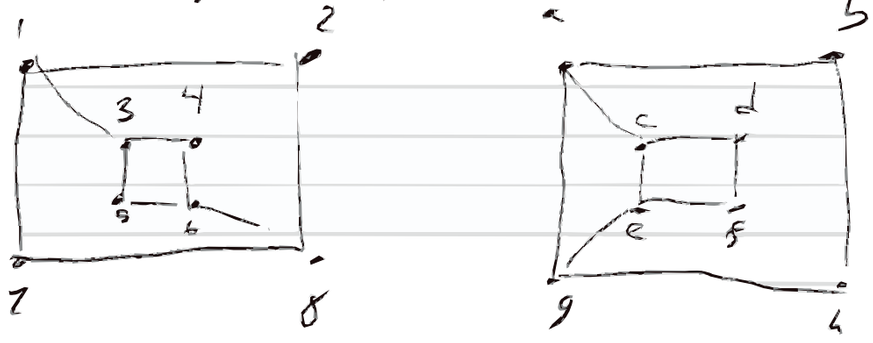
Ex: Are these isomorphic?

★ Show vertex degree are a graph invariant.

Idea: A graph invariant is a quantity that can be calculated for a graph s.t. if is always the same if 2 graphs are isomorphic

→ i.e. # of edges, # of vertices, connectedness, etc., girth (cycle length) minimal

Ex: Are these graphs isomorphic?



Note: Isomorphism is a relation on the set of graphs

Prop. Isomorphism is an equivalence relation. on the set of all graphs

What's a relation? Sets S & T

$R \subset S \times T$ is a relation.

1) Symmetry $(x,y) \in R \Rightarrow (y,x) \in R$
2) Reflexivity $(x,x) \in R$
3) Transitivity $(x,y) \in R, (y,z) \in R \Rightarrow (x,z) \in R$

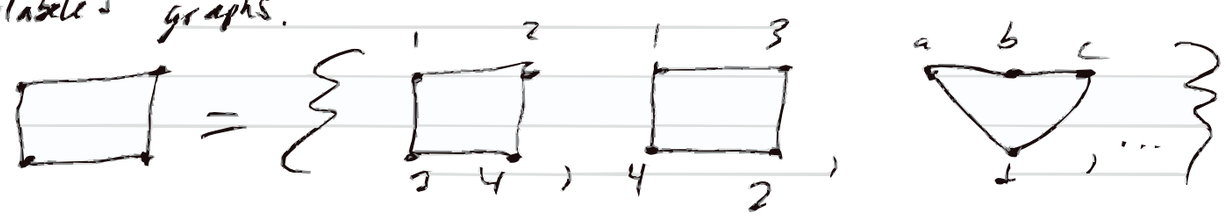
Pf: ① Reflexivity
② Symmetry
③ Transitivity

(end of lecture 2)

Note. Equivalence Relations & Equivalence Classes

Def. An isomorphism class of graphs is an equivalence class of graphs under isomorphism

We often denote such equivalence classes by writing unlabeled graphs.



Definition:

The *unlabeled path* and *unlabeled cycle* with n vertices are denoted P_n and C_n, respectively.

C_n is sometimes also called an n-cycle

Definition:

A *complete graph* on n vertices is a simple graph with all pairs of distinct vertices adjacent, and is denoted K_n

Question:

How many edges are there in a complete graph with n vertices?

Definition:

A *complete bipartite graph* or *biclique* is a simple bipartite graph such that two vertices are adjacent if and only if they lie in different partite sets.
If the first partite set contains r vertices and the second contains s vertices, the graph is denoted $K_{r,s}$

Question:

How many edges are there in $K_{r,s}$?

rs ← what if you construct a complete graph by breaking it up into a bipartite graph first.

$rs + \binom{r}{2} + \binom{s}{2} = \binom{r+s}{2}$

Note:

We will often refer to graphs without explicitly describing or labeling their vertices. In such cases, we are implicitly referring to an isomorphism class of graphs.

Sentences like "H is a subgraph of G" can be very carefully read to mean "There is a subgraph of G which is isomorphism to H" or "G contains a *copy* of H"

Questions like "Is this graph K_5?" should be understood as "Is this graph isomorphic to K_5?"

Question:

Suppose I fix a vertex set V with n elements, say $\{1, 2, 3, \ldots, n\}$
How many distinct labeled graphs can be made from these vertices?

If $n = 3$, how many isomorphism classes of graphs are there?

Proposition:

If two simple graphs H and G are isomorphic, then their complements are also isomorphic.

A complement of G is a graph $\bar{G}$ with the same vertices as G, and $uv \in E(\bar{G})$ iff $uv \notin E(G)$

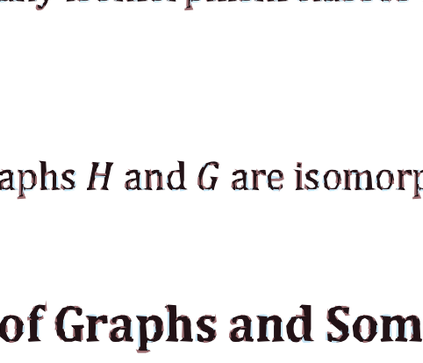
Decomposition of Graphs and Some Special Graphs

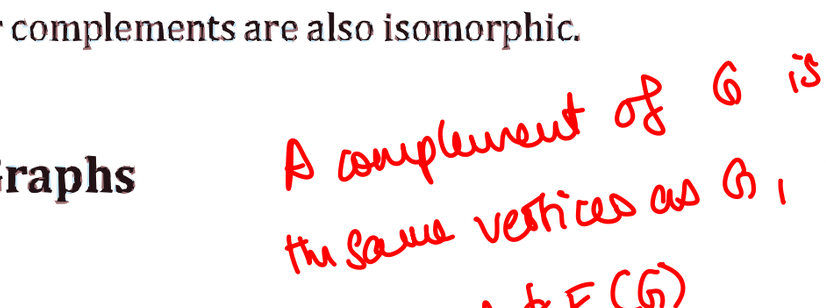
Question:

Consider the graph P_4. What is its complement?"

Definition:

We say a graph is *self-complementary* if it is isomorphic to its complement.





Definition:

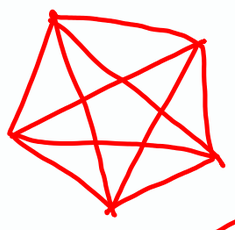
A *decomposition* of a graph G is a list of subgraphs of G such that each edge in $E(G)$ appears in exactly one subgraph in the list.

Proposition:
A graph H with n vertices is self-complementary if and only if K_n has a decomposition consisting of two copies of H.

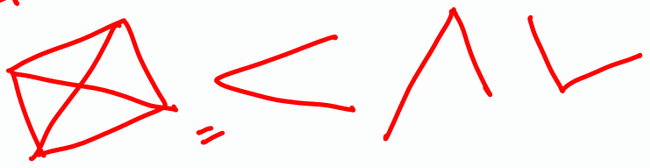
↳ If K_n: $G_1 \to G_2$, f_1, f_2, $\#$

G_1 and G_2 must have same # of vertices & edges ⇒ G_1 & G_2 must have n vertices. if $uv \in E(G_1)$ in fact $G_1 = \overline{G_2}$

Example:
K_5 is two copies of C_5



Example:
K_4 is three copies of P_3 →



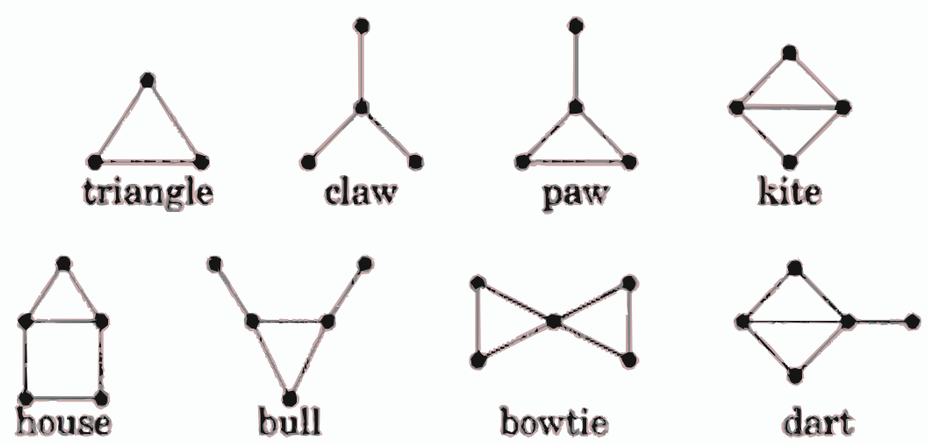
Note:
In many computational applications, we decompose complicated shapes into *triangulations*. Doing this is fundamentally about graph decompositions!

(Craig)

↳ really? → To ignore for now.

Note:
Lots of graphs have cute names, some of which are commonly used and others less so

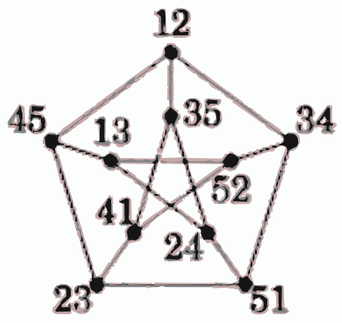
Constellation graphs? another project.



Which of these are self complementary?

There are myriad other specific graphs of interest.

Definition:
The *Petersen graph* is the simple graph G with $V(G)$ the 2-element subsets of a 5-element set with edges joining each pair of disjoint subsets.



$\{12345\} \to \binom{5}{2} = \frac{5 \cdot 4}{2} = 10$

$12 \leftrightarrow 34$ | 12 ... 23, 45

Can ignore argue directly

Proposition: (for the class)
Given any two distinct points in the Petersen graph, there exists a unique path of length either 1 or 2 (but not both) connecting them.

Definition:
The *girth* of a graph is the length of the shortest cycle contained in the graph. If the graph contains no cycle, the girth is said to be infinite.

Claim:
The Petersen graph has girth 5. → C_3: 12 34 ↑ only one vertex left

If no C_3, suppose C_4: 12 ... 45, 13. There is only one vertex connected to both.

Claim:
Any complete graph with at least three vertices has girth 3.

* EX

Ex

Claim:

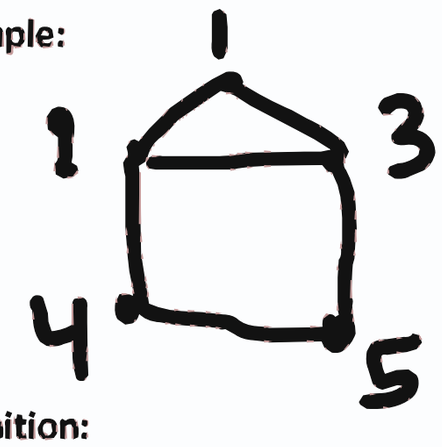
Any complete bipartite graph with at least 2 vertices in each partite set has girth 4.

One of the things that makes the Petersen graph really nice is that it has some nice symmetry properties. We can encode these properties really precisely.

Definition:

An *automorphism* of a graph G is an isomorphism from G to G.

Example:



Definition:

A graph G is *vertex-transitive* if, for every pair $u, v \in V(G)$, there exists an automorphism of G that maps u to v.

An easy example is the 4-cycle.

P_4 is not vertex-transitive.

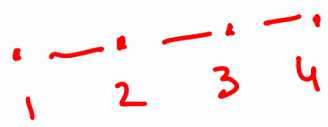
Claim:

The Petersen graph is vertex-transitive.

Note:

Suppose a graph G is vertex-transitive. If we prove a property of the graph is true for some specific vertex, it must be true for all vertices!

→ worth doing this example.



List of all possible automorphisms = S_4 (permutations)

σ is a transposition

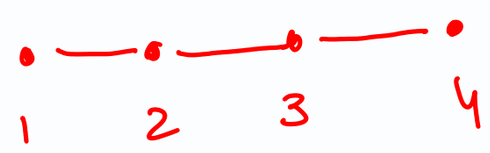
$\sigma(1)=3$
$\sigma(2)=2$
$\sigma(3)=1$
$\sigma(4)=4$

Is it an automorphism?

$E(P_4) = \{12, 23, 34\}$ $\quad E(\sigma P_4) = \{12, 23, 14\}$ ✗

and so on.

$\sigma = (4\,3\,2\,1)$



There are the only 2 automorphisms!

C_4

Isomorphism and Automorphism

A graph is a triple $G = (V, E, R)$

$R : E \longrightarrow V \times V / \sim$ is a map. Allows for multiple edges and loops.

↑ undirected case

In the simple case, we have

$E \subset \{B \subset V : |B| = 2\}$ a subcollection of the cardinality two subsets of V.

Given G, H an isomorphism is a (pair of bijections)

$f : V(G) \longrightarrow V(H)$

$f : E(G) \longrightarrow E(H)$

Satisfying $uv \in E(G) \iff f(u) f(v) \in V(H)$. ✯1

In the simple case enough to specify $f : V(G) \longrightarrow V(H)$ and condition ✯1.

Take $G = (\{1,3,5\}, \{13, 35\})$

$H = (\{pig, cat, tree\}, \{pig\ cat, cat\ tree\})$

Clearly G and H are isomorphic. So basically isomorphism is a relabelling of the vertices of G.

There are only many isomorphisms.

Automorphism $f : G \longrightarrow G$ bijection. Here

$f : V(G) \longrightarrow V(G)$ in the simple case.

The # of candidate automorphisms are finite! This is

because if $V(G) = \{1, \ldots, n\}$ the canonical labels. Any

bijection on $V(G)$ is a permutation on $[n]$!

Section 1.2 - Paths, Cycles & Trails

Thursday, January 19, 2023 4:06 AM

Quick review of strong induction...

Definition:

A *walk* is a list $v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k$ of vertices and edges such that, for $1 \leq i \leq k$ the edge e_i has endpoints v_{i-1} and v_i ← In simple graphs can just list vertices

A *trail* is a walk with no repeated edges

A u, v-*walk* or u, v-*trail* has initial vertex u and final vertex v

(no repeated vertices)

A u, v-*path* is a path whose degree 1 vertices are u and v with the others called *internal vertices*

The *length* of a walk is the number of edges present in the walk.

A walk or trail is *closed* if $v_0 = v_k$

Note:

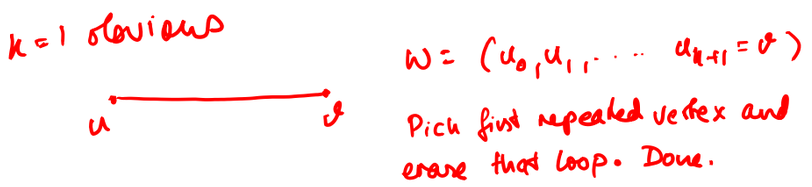
In a simple graph, it isn't really necessary to specify the edges explicitly for a walk.

Note:

We say one walk *contains* another if the latter is a sublist of the former

Lemma:

Every u, v-walk contains a u, v-path



Proof:

Induct on the length of the walk.

Definition:

A graph G is *connected* if it has a u, v-path for all $u, v \in V(G)$. Otherwise, it is *disconnected*.

The *connection relation* on $V(G)$ consists of those pairs of points for which there exists a u, v-path (we sometimes call such pairs of points *connected*, but it's a better term to avoid).

Claim:

The connection relation is an equivalence relation.

Definition:

The *components* or *connected components* of a graph are the equivalence classes of the connection relation.
Alternatively, they are maximal connected subgraphs.

A component is *trivial* if it has no edges, otherwise *nontrivial*

An *isolated vertex* is a vertex of degree 0

Claim:
A graph has trivial components if and only if it has isolated vertices.

Proposition:
Let G have n vertices and k edges. Then G has at least $n - k$ components.

Proof:
If $k = 0$, this is immediate. Otherwise, adding an edge potentially merges at most 2 components into 1.

$G \setminus \{e\}$ has at least $n-(k-1)$ components

Adding k^{th} edge can either keep components or merge 2 comp the same

Question:
What can happen to the number of components in a graph if we delete an edge in a subgraph?

What if we delete a vertex and take the induced subgraph?

Notation:
We write $G - e$ for the subgraph with a particular edge deleted, $G - M$ for the subgraph with a set of edges M all deleted.

We similarly write $G - v$ or $G - S$ for induced subgraphs with specified vertices deleted

We will write $G[T]$ to mean the induced subgraph on a set $T \subseteq V(G)$

Definition:
A *cut-edge* or *cut-vertex* of a graph is an edge or vertex, respectively, whose deletion increases the number of components.

Theorem:
An edge is a cut-edge if and only if it does not belong to a cycle.

Proof:
Suppose e has endpoints x, y. Consider deleting it and look at the multiplicity of paths.
On the other hand, suppose e is contained in a cycle. Then it's not a cut-edge.

Bipartite Graphs & Cycles

It can really be a pain in the neck to figure out if a given graph is bipartite. Being able to characterize this simply is very helpful.

Notation:

We say a walk is *odd* (*even*) if its length is odd (even).

Lemma:

Every closed odd walk contains an odd cycle.

Proof:

If the walk never repeats a vertex, done. Otherwise, we can examine the strictly shorter walk.

Definition:

A *bipartition* of G is a specification of two disjoint independent sets in G whose union is $V(G)$

"Let G be a bipartite graph with biartition X, Y" is a phrase we'll find ourselves saying requently.
Synonymously we'll call such a thing a X, Y-*bigraph*

Theorem: (Koenig 1936)

A graph is bipartite iff it has no odd cycle.

Proof:

Necessity is clear

Sufficiency → Fix a vertex, define an even-odd minimum path length bipartition. Show that two points in the same partite set can't be connected by an edge.

(The converse is useful here - if you can find an odd cycle in a graph, then it isn't bipartite)

Definition:

Given graphs $G_1, G_2, \ldots, G_n$, their union $G_1 \cup G_2 \cup \cdots \cup G_n$ is the graph with vertex set

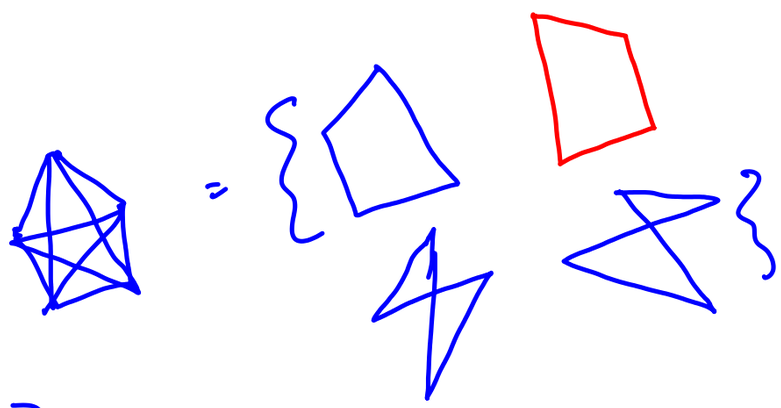
$$\bigcup_{i=1}^{n} V(G_i)$$

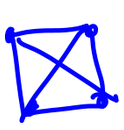
and edge set

$$\bigcup_{i=1}^{n} E(G_i)$$

Example for them:

Write K_5 as a union of 5-cycles. How many are required?
How many 4-cycles are required?



Better example : K_4  = {  }

union of bipartite graphs

Theorem:

The complete graph K_n can be expressed as the union of k bipartite graphs iff $n \leq 2^k$

Proof:

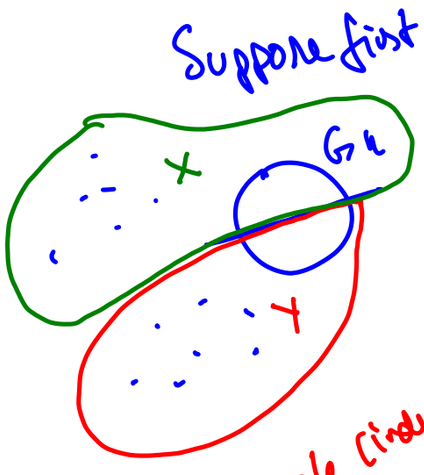
K_3 ∇ odd cycle

(induct on k)

- (k=1) K_n fails to be bipartite iff $n > 2$, so we have the desired result.
- Suppose that the desired statement is true for all smaller values of k

Let $K_n - G_1 \cup \cdots \cup G_k$ all bipartite, to show $n \leq 2^k$

Suppose first



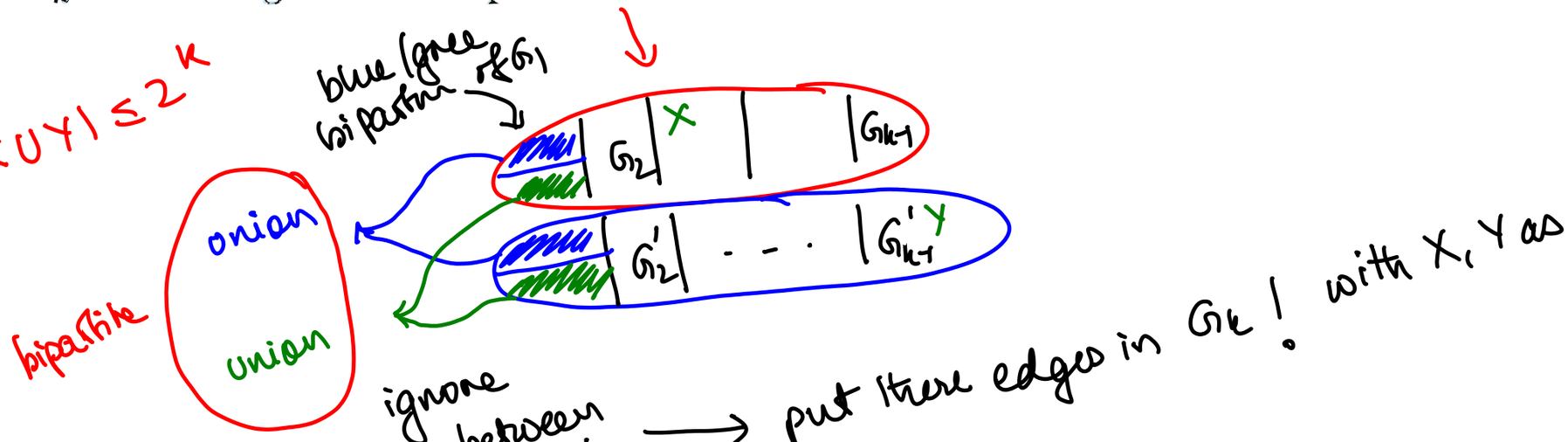
Define sets X, Y such that no edge in G_k connects two points in the same of these two sets

Take the induced subgraphs on X, Y. These are smaller graphs, and we can write each as the union of $k-1$ bipartite graphs.

Let $n \leq 2^k$. Define arbitrary subsets X, Y with cardinality no more than 2^{k-1}. They can be covered by $k-1$ bipartite graphs. Define G_i as the pairwise disjoint union of these, and let G_k contain edges between points of X and of Y

X is a complete subgraph and so is Y

X covered by $G_1 \ldots G_{k-1}$ $\Rightarrow |X| \leq 2^{k-1} \Rightarrow |X \cup Y| \leq 2^k$



ignore edges between blue & green sets. → put these edges in G_k! with X, Y as bipartition.

Eulerian Circuits

Definition:

(no repeated edges)

A graph is *Eulerian* if it has a closed trail containing all edges.

We call a closed trail a *circuit* if we don't specify the initial point, but write points cyclically.

Terminology:

We call a vertex *even* (*odd*) if it has even (odd) degree. A graph is *even* if all vertices are even.

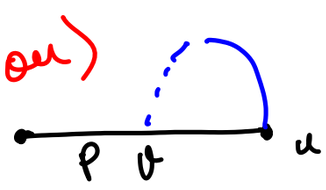
We call a path $P \subseteq G$ *maximal* if no strictly larger path containing it is contained in G

Lemma:

If every vertex of a graph G has degree at least 2, then G contains a cycle.

Proof:

Take a maximal path with endpoint u (no vertex repetitions)



Maximal ⇒ all its neighbors part of P since degree(u) ≥ 2, ⇒ ∃ an edge not in P from u to v

Note:

This argument only works with finite graphs - what goes wrong if the graph is infinite? Consider the integers.

Note:

The above proof is an example of an "*extremality argument*". These arguments rely on supposing a "maximal" example of some type of construction, using the extra information about maximality in a useful way.

Theorem:

A graph is Eulerian iff it is even and has at most one non-trivial component.

Proof:

Clearly having one component is a necessary condition, so we assume our graph is connected going forward.

Eulerian implies even is straightforward.

Even implies Eulerian:

Strong induction on # of edges. If $m=0$

G contains a cycle. Remove it, continue.

C ↙ Take $G' = G \setminus E(C)$

- Obvious

If $v \in C$, then it must have two edges removed from it. If not, then it has 0 edges removed $\Rightarrow$ G' is still even. (in fact all its components are even.

Corollary:

Every even graph decomposes into cycles.

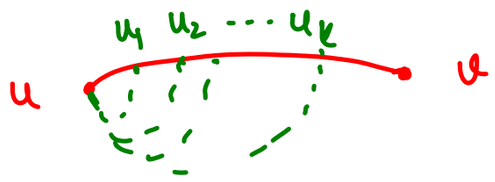
↙ Take G', every component is even.

Start removing cycles.

Will now show 2nd proof of Euler using extremality

Proposition:

If G is a simple graph in which every vertex has degree at least k, then G contains a path of length at least k. If $k \geq 2$, then G contains a cycle of length at least $k + 1$.



Proof:

(Extremality argument)

Consider a maximal path; its endpoint can only be adjacent to vertices in the path. Since it has degree k, the result follows.

For a cycle, take the furthest neighbor of the endpoint.

It's worth noting that not every vertex of a graph can be a cut-vertex.

Proposition:

Every graph with a non-loop edge has at least two vertices that are not cut-vertices.

(can the class figure out the counterexample with a loop edge?)

Proof:

↗ requires all neighbors to be on the path by maximality

The endpoints in a maximal path are not cut-vertices, since all of their neighbors are connected via the path.

(Common strategies with extremal proofs are to find maximal paths, vertices of maximal or minimal degree, maximal connected subgraphs, and so on)

Lemma:

In an even graph, every non-extendible trail is closed.

uses the fact that edges not repeated.

start at u

u — v

every time you arrive at a you've used up an odd # of edges incident at it

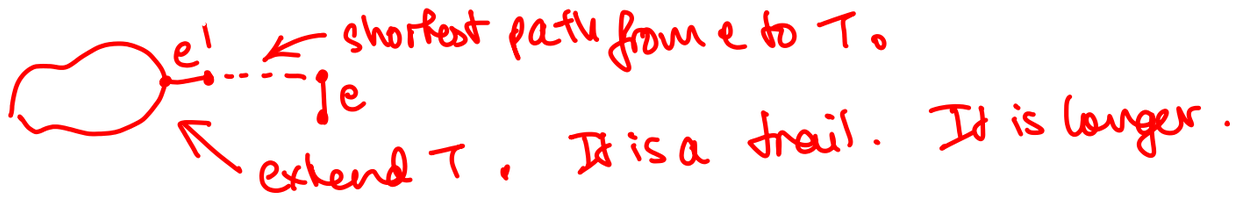
⇒ trail must continue. can only end at u!

Theorem:

A graph is Eulerian iff it is connected and even.

Proof:

Take a trail of maximal length. If it doesn't contain every edge, append that edge onto the trail. It is now longer.



Graphs that are Eulerian are "covered" by a single trail. What if we allowed more than 1? Can we figure out which graphs can be decomposed into two trails? Three trails? k trails?

Theorem:

For a connected nontrivial graph with exactly $2k$ odd vertices, the minimum number of trails that decompose it is $\max\{k, 1\}$

↑ for the all-even case when k=0.

Proof:

A trail must start and end at odd degree vertices. So if there are 2k vertices need at least k trails.

Arbitrarily pair up odd vertices and draw an extra edge between them. The resulting graph is even and connected. Take an Eulerian trail, and let it disconnect on the newly drawn edges.

(This proof also serves to demonstrate that we can prove pretty general theorems as simple corollaries of less general theorems.)

Gra

Section 1.3 - Vertex, Degrees, and Counting

Friday, January 27, 2023 12:00 AM

Definition:

The *degree* of a vertex v in a graph G is written $d_G(v)$ or $d(v)$ and is the number of edges incident to v, with loops counted twice.

Definition:

Given a graph G, we write $\Delta(G)$ as the maximum degree of vertices and $\delta(G)$ the minimum degree

Definition:

G is *regular* if $\Delta(G) = \delta(g)$
G is k-regular if it is regular with $\Delta(G) = k$

Definition:

The *neighborhood* of a vertex v is $N_G(v)$ or $N(v)$ the set of vertices adjacent to v

It's also useful for us to clarify our notation for the size of a graph.

Definition:

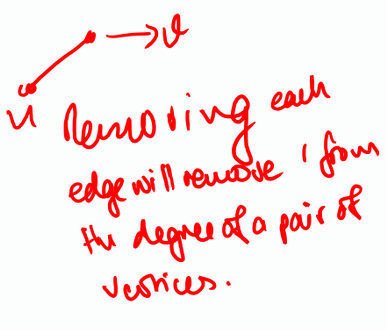
A graph has *order* $n(G)$ if it has n vertices.
A graph has *size* $e(G)$ if it has e edges.

Notation:

We'll sometimes refer to the set $[n] = \{1,2,3,\dots,n\}$

Proposition:

If G is a graph, $\sum_{v \in G} d(v) = 2e(G)$ ↳ degree sum formula



Corollary:

Every graph has an even number of vertices with odd degree. → clear

Corollary:

In a graph G, the average vertex degree is $\frac{2e(G)}{n(G)}$, and in particular

$$\delta(G) \leq \frac{2e(G)}{n(G)} \leq \Delta(G)$$

Question:

How many edges does an order n k-regular graph have? → $\frac{nk}{2}$

Proposition:

If $k > 0$, any k-regular bipartite graph has the same number of vertices in any partite set.

Proof:

All edges are between the partite sets. Count them based on edges in the first, then by edges in the second to establish equivalence. ↳ there are $k|X|$ many; there are $k|Y|$ many; $e(G) = k|X| = k|Y|$

Hypercubes

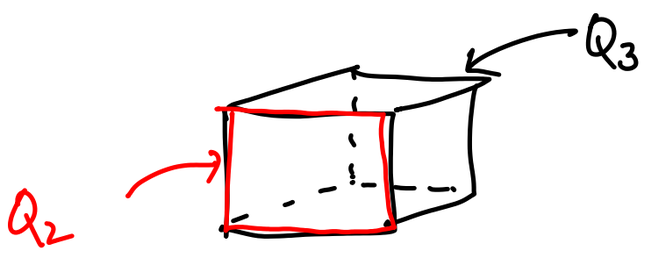
$V(G) = \{0,1\}^k$ $\quad xy \in E(G)$ iff $x - y = \pm e_i$

$e_1 = (1,0\ldots)$ $\quad e_2 = (0,1,\ldots)$

Is Q_k k-regular? You can flip any of the k coordinates to get a neighbor: YES.

$n(Q^k) = 2^k \Rightarrow 2e(G^k) = k\,2^k$

j subcube is a subgraph $\simeq Q_j$



Can keep any $k-j$ coordinates fixed and vary the others: (2^j many)

to obtain a subcube $\Rightarrow \binom{k}{k-j} 2^{k-j}$ many subcubes.

2^{k-j} options to keep fixed.

* Good HW problem. Exam

for $j=1$ $\quad \binom{k}{k-j} 2^{k-j} = \binom{k}{k-1} 2^{k-1} = k\,2^{k-1}$

* Is the hypercube bipartite?

worth skipping

Clever counting methods can be extremely helpful in analyzing graphs. (maybe exclude this example?)

Proposition:
The Petersen graph has 10 6-cycles.

Proof:
(Recall the Petersen graph)
Note that the Petersen graph manifestly contains 10 copies of the 'claw', one centered at each vertex

G has girth 5, so every 6-cycle is an induced subgraph - each point in such a cycle is adjacent to one external vertex. In Petersen graph, nonadjacent vertices have a unique common neighbor - take opposite points in the cycle.
Subtract cycle from G, 4 vertices remain, common neighbors have degree 1, last vertex has degree 3 - this is a claw.

To show each claw is obtained only once by this procedure, take a claw. Degree 1 vertices in it all have a common neighbor, so external vertices cannot coincide. Subtract claw from G, remaining graph is 2-regular, must be a 6-cycle.

These types of counting arguments have incredible power in graph theory.

(subgraphs of a graph with a single deleted vertex are sometimes called *vertex-deleted subgraphs*)

Proposition: (for them)
Let G be a simple graph with vertices $v_1, \ldots, v_n$ and $n \geq 3$. Then

$$e(G) = \frac{\sum_{v \in V(G)} e(G - v)}{n - 2}$$

Exercise for someone in class?

$$d_G(v_j) = \underbrace{\frac{\sum e(G - v_i)}{n-2}}_{e(G)} - e(G - v_j)$$

so given information about all vertex deleted subgraphs:

(all $e(G - v_j)$ for example) we can get

1) $e(G)$
2) $d_G(v_j)$ all vertex degrees.

It's reasonable to ask how well we can, given information about vertex-deleted subgraphs, figure out information about a graph itself.

Basically we are only given information like this

$\{C_3, C_3, P_3, P_3\}$

Conjecture: (Reconstruction Conjecture)

If G is a simple graph with at least 3 vertices, then G is uniquely determined by the list of (isomorphism classes of) its vertex deleted subgraphs.

Extremal Problems

We often wish to answer questions of the form: "How big (or small) of an example of ___ can be found among things of type ____?"
These are referred to as *extremal problems*

Proposition:

The minimum number of edges in a connected graph with n vertices is $n - 1$

Proof: The path satisfies this bound.
Fewer edges cannot connect this many points.

We show if k edges at least $n-k$ components.

Note:

Somewhat formally written, to show that β is the minimum value of $f(G)$ over some class of graphs $\boldsymbol{G}$, then we must show two things.

- $f(G) \geq \beta$ for all $G \in \boldsymbol{G}$
- $f(G) = \beta$ for some $G \in \boldsymbol{G}$

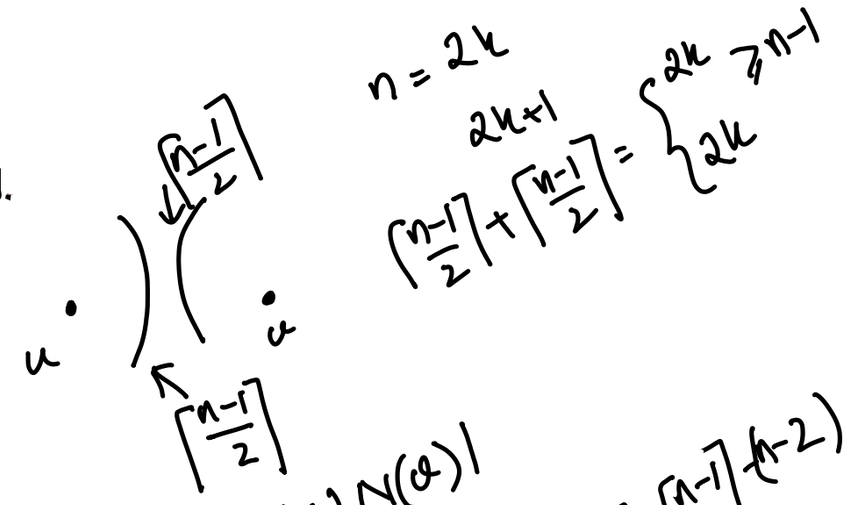
Also solved by pigeonhole

Proposition:

If G is a simple order n graph with $\delta(G) \geq \frac{(n-1)}{2}$, then G is connected.

Proof:

Any two vertices are adjacent or have a common neighbor



This proposition is actually part of an extremal problem in disguise.

Proposition:

The maximum value of $\delta(G)$ among simple graphs of order n is $\left\lfloor \frac{n}{2} \right\rfloor - 1$

disconnected

Proof:

Let G be an n-vertex graph with two components, $K_{\lfloor \frac{n}{2} \rfloor}$ and $K_{\lceil \frac{n}{2} \rceil}$

Use the previous proposition for the rest.

↑ has degree $\delta(G) = \lfloor \frac{n}{2} \rfloor - 1$

k $\quad$ $k+1$ (in the odd case)

$\lfloor \frac{n}{2} \rfloor - 1 = \begin{cases} \frac{n}{2} - 1 = \frac{n-2}{2} \\ k-1 < \frac{2k+1-2}{2} \end{cases}$

$\Rightarrow \lfloor \frac{n}{2} \rfloor - 1 \leq \frac{n-2}{2} < \frac{n-1}{2}$

Notice that in order to solve this extremal problem, we had to find an example for each value of n - a family of extremal solutions.

Definition:

The graph obtained by taking the union of graphs G and H with disjoint vertices is the *disjoint union* or *sum* $G + H$

The graph with m disjoint copies of G is mG

The previous proof used such a disjoint union.

Claim:

$K_n + K_m = \overline{K}_{m,n}$ ← complement of its bi clique

Note:

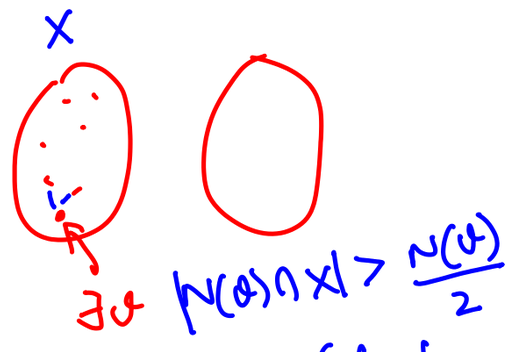
In a similar vein, we sometimes wish to find the largest example of some type within a single specified graph (largest clique, highest degree vertex, etc).

To distinguish these from extremal problems, we call them *optimization problems*

Solving such problems tends to be *very* complex. Proofs usually tend to describe an algorithm (sequence of steps) one could use to find such a optimum.

Theorem:

Every loopless graph G has a bipartite subgraph with at least $\frac{e(G)}{2}$ edges.



Proof:

Split $V(G)$ into two arbitrary sets X, Y. Take edges connecting these sets.
If this is not more than half the edges, there exists a vertex with more than half of adjacent vertices in the same set. Move it to the other set. Repeat. This process will terminate with a subgraph satisfying the desired property.

(if not $|N(v)\cap X| \le \frac{N(v)}{2}$ $\Rightarrow |N(v)\cap Y| \ge \frac{N(v)}{2}$

why must there exist such a vertex?

$\Rightarrow \sum_{v\in X} |N(v)\cap Y| \ge \sum_{v\in X} \frac{N(v)}{2}$ → degree sum formula

similarly $\sum_{v\in Y} |N(v)\cap X| \ge \sum_{v\in Y} \frac{N(v)}{2}$ $= e(G)$

2 · total # of edges between x and y

$\Rightarrow$ 2 · (# of edges between x and y) $\ge e(G)$

This question could be motivated by a military question or by various social group questions. Imagine you have a collection of armies each with their own enemy armies. If no two distinct have a common enemy, how many "enemy connections" can there be?

Question:

How many edges can be there be in an order n graph which contains no triangles?

Definition:

A graph G is *H-free* if G has no induced subgraph isomorphic to H

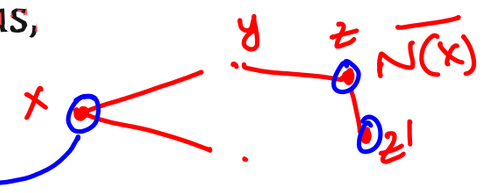
Theorem:

The maximum number of edges in a C_3-free simple graph of order n is $\left\lfloor \frac{n^2}{4} \right\rfloor$

Proof:

Let G be such a graph, and x its vertex of maximal degree, k.
No two neighbors of x are adjacent, so each edge is between an element of $N(x)$ and its complement, or between two elements of its complement. Thus,

$$\sum_{v\notin N(x)} d(v) \ge e(G)$$



includes x at least one each

The former sum is at most $(n-k)k$

$e(G) \le e(K_{n-k,k})$

The expression $(n-k)k$ is maximized when $k = \frac{n}{2}$

Let's try to prove this without calculus, though

(counts part of edge)

The expression $(n-k)k$ represents the number of edges in $K_{n-k,k}$

Move a vertex from size k set to size $n-k$ set gains $k-1$ edges, loses $n-k$, so change of $2k-1-n$ ($2k > n+1$ means change is positive)

To achieve this maximum, consider

$K_{\lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil}$ $\quad e\left(K_{\lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil}\right) = \left\lfloor \frac{n^2}{4} \right\rfloor$

• Does ∃ exist a graph with $\Delta(G)=k$, $e(G)=\frac{n^2}{4}$

what should k be to maximize $e(K_{n-k,k})$ (the edges in this biclique)

(You want $k \le \frac{n+1}{2}$, so $k=\left\lceil \frac{n}{2} \right\rceil$)

Note:

In principle, we could try to prove this kind of theorem by induction. However, it might be quite tricky and one has to be quite cautious. What goes wrong with the following argument for the previous theorem?

Base Case: $n = 2$ - this case is immediate
Induction Step:

Suppose the statement holds for $n = k$, so the complete graph $K_{\lceil \frac{k}{2} \rceil, \lfloor \frac{k}{2} \rfloor}$ is extremal

Even $k=2r$

$K_{r,r} \to K_{r+1,r}$

Odd $k=2r+1$

$K_{r,r+1} \to K_{r+1,r+1}$

in this case.
Add a new vertex to it to form a triangle-free graph with $k+1$ vertices
As long as new vertex is only adjacent to vertices from one partite set, this creates no triangles. This gives the new complete bipartite graph, completing the proof.

$K_{\lfloor (k+1)/2 \rfloor, \lceil (k+1)/2 \rceil}$

We are saying that if G contains $K_{\lceil \frac{k}{2} \rceil, \lfloor k/2 \rfloor}$ then adding a vertex shows the bound is not still satisfied.

Takeaway: If doing an inductive proof - make sure you start with an arbitrary thing satisfying the $k+1$-case and attempt to shrink it, rather than trying to grow the k-case.

Skip.

Schematically, if the induction proof is for the claim $A(n) \Rightarrow B(n)$ for all G of size n

G satisfies $A(n) \Rightarrow G'$ satisfies $A(n-1) \Rightarrow G'$ satisfies $B(n-1) \Rightarrow G$ satisfies $B(n)$

Graphic Sequences

We've previously used degrees as a way to think about graph isomorphism. But degree of a single vertex isn't a meaningful graph invariant. How can we phrase degree in such a way as to make it a graph invariant concept?

Definition:

The *degree sequence* of a graph is the list of vertex degrees, usually written in nonincreasing order

$$d_1 \leq d_2 \leq \cdots \leq d_n$$

Question:

Can any arbitrary degree sequence be realized by a graph?

2,2,3,4?

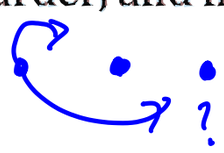
Proposition:

Iff $\sum d_i$ is even, then the given sequence is the degree sequence of some graph.

Proof:

(Necessity) Degree sum formula → the # of odd vertices must be even by degree sum.
(Sufficiency) Connect pairs of odd vertices. Then draw a whole bunch of loops.

This theorem is *much* harder, and not actually true, if we don't allow loops or multiple edges.
(Consider 2,0,0)



Definition:

A *graphic sequence* is a list of nonnegative numbers that is the degree sequence for a simple graph. A simple graph with a given degree sequence "realizes" that sequence

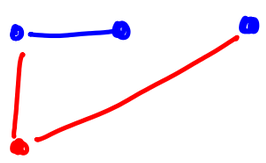
There are a lot of possible ways to characterize this. Multiple such characterizations can be found in Sierksma-Hoogeveen (1991)

2211 → remove 2 of its neighbors

101

→ 110

reorder



Idea:

How could we know if the sequence 3,3,3,3,3,2,2,1 is graphic?

Let's try to construct one - there's a vertex of degree 3 - could all of its neighbors also have degree 3?
If so, we could remove it from the graph, neighbors would now have degree 2. The resulting graph would have to have degree sequence

3,2,2,2,2,2,1

There's a vertex of degree 3, could all neighbors have degree 2?
If so, remove it, neighbors now have degree 1. Resulting graph would have degree sequence

2,2,1,1,1,1

Repeat

1,1,1,1,0
1,1,0,0
0,0,0
0,0
0

Draw each graph and build up

3 3 3 3 3 2 2 1

↓

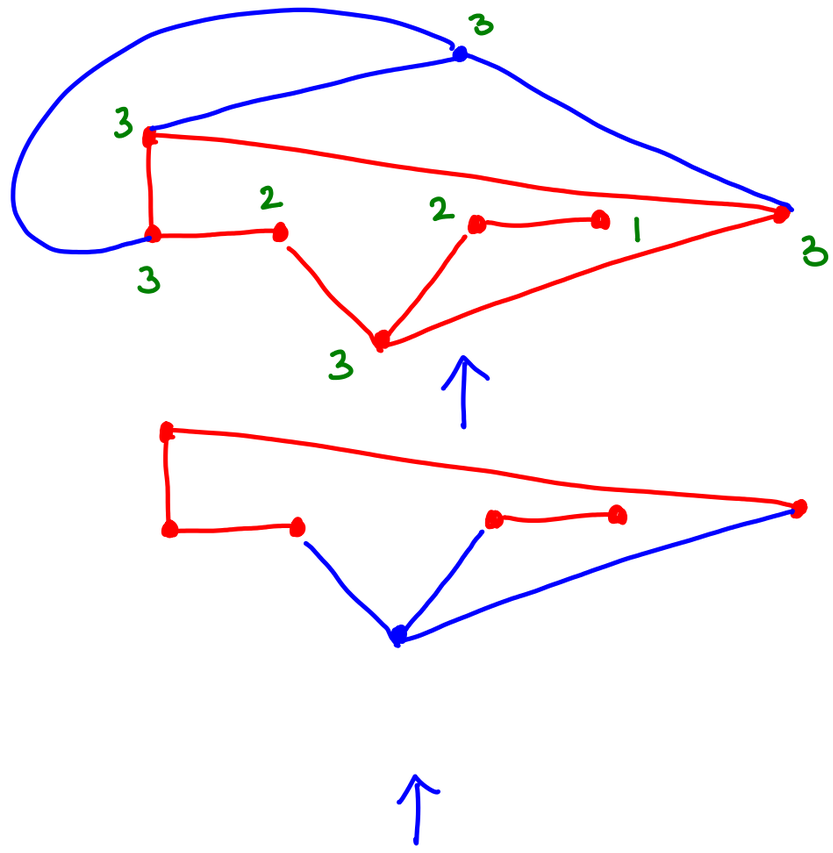
3 2 2 2 2 2 1

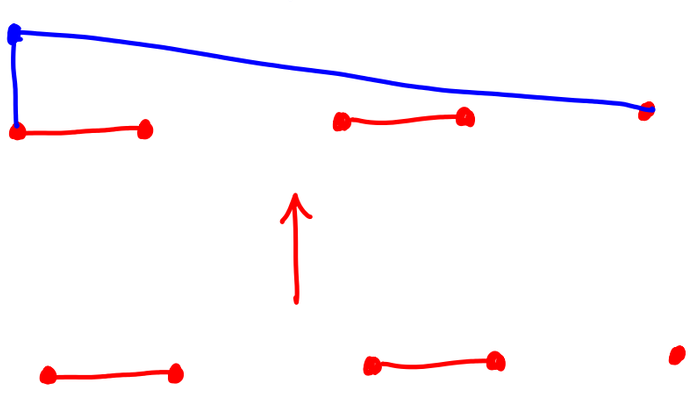
↓

2 2 1 1 1 1

↓

1 1 1 1 0





Theorem: (Havel [1955], Hakimi [1962])

For $n > 1$ an integer list d of size n is graphic if and only if d' is graphic, where d' is obtained from d by taking the largest element Δ of d and deleting it while decrementing the following Δ-largest elements by 1.

The only 1 element graphic sequence is $d_1 = 0$

Proof:

The 1 element case is trivial.

Suppose we have a sequence d of length more than 1 and a corresponding sequence d' (written in decreasing order) which is realized by a graph G'

Attach a new vertex to G' with Δ connections to vertices with degrees $d_2 - 1, d_3 - 1, \ldots, d_{\Delta+1} - 1$

Suppose d is realized by a graph G

Let $w \in V(G)$ have degree $d_1 = \Delta$

Let S be a set of vertices in G having degrees $d_2, d_3, \ldots, d_{\Delta+1}$

If $N(w) = S$, done

Else (modify G to increase $|N(w) \cap S|$)

There must exist $x \in S$ and $z \notin S$ with x not adjacent to w but z adjacent to w

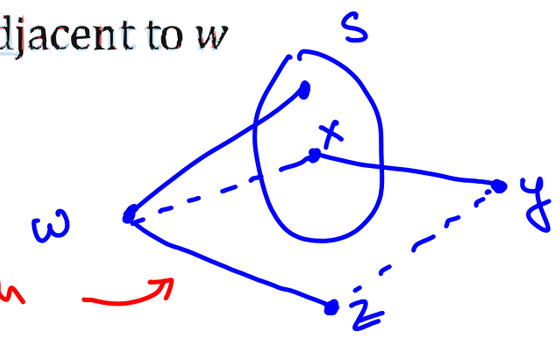
← By necessity, $d(x) \geq d(z)$

because it's in S and share degrees $d_2, \ldots d_{\Delta+1}$

⇒ ∃ vertex y connected to x but not z (since $z \sim w$)

Switch the connections up

Repeat previous step as necessary

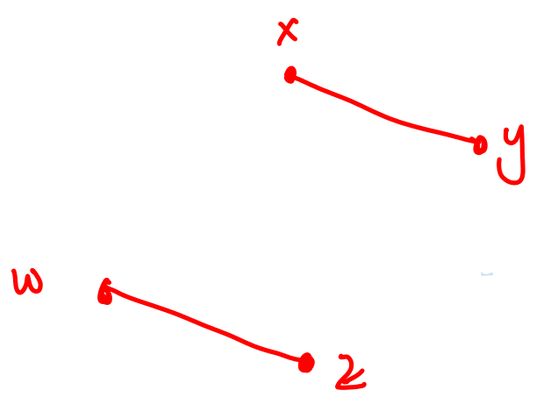


The procedure in the previous proof is kind of interesting, in that it codifies a particular way we can modify a graph without changing any vertex degrees.

Definition:

A *2-switch* is the replacement of a pair of edges xy and zw in a simple graph by edges yz and wx, given that neither of the latter were already in the graph

(draw)



Does not change vertex degrees.

These operations are surprisingly powerful!

same vertex set! so don't have to worry about relabeling.

Theorem: (Berge 1973)

If G, H are two simple graphs with vertex set V, then $d_G(v) = d_H(v)$ for every $v \in V$ if and only if there is a sequence of 2-switches that transforms G into H

Proof:

For sufficiency, 2-switches preserve vertex degree

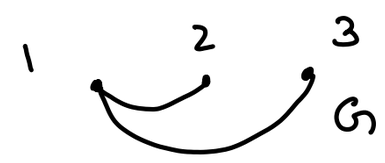
For necessity, go by induction with $n = 3$ base case (degree sequence is a complete invariant here and below)

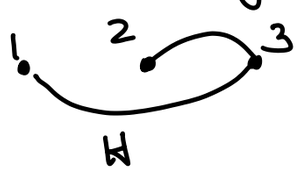
$d_1\ d_2\ d_3$

3 3 3
2 1 1
1 1 0
0 0 0

List of degree sequences

There is at most one simple graph with $d(\theta_i) = d_i$



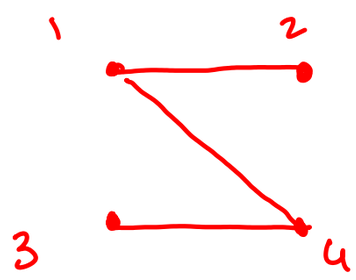


$d_G(1) \neq d_H(1)$

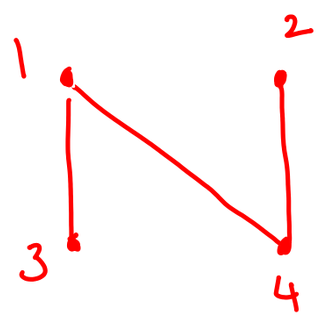
This might mislead you into thinking there are no 2 switches but these two graphs are isomorphic

Simplest nontrivial example.

$n=4$ case

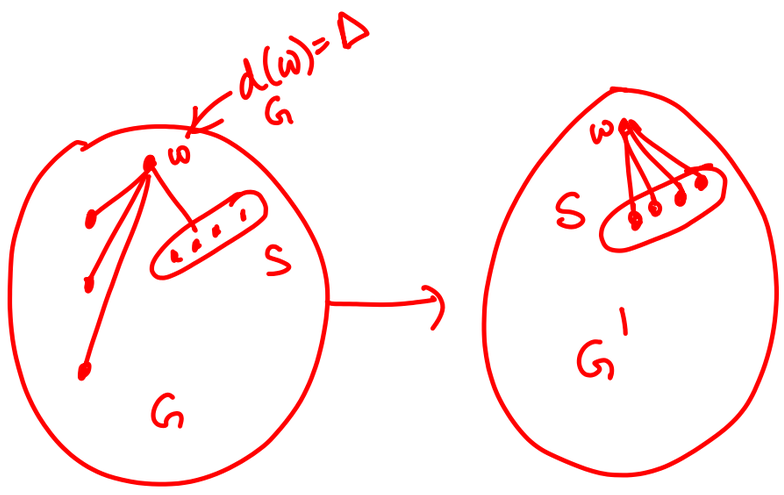


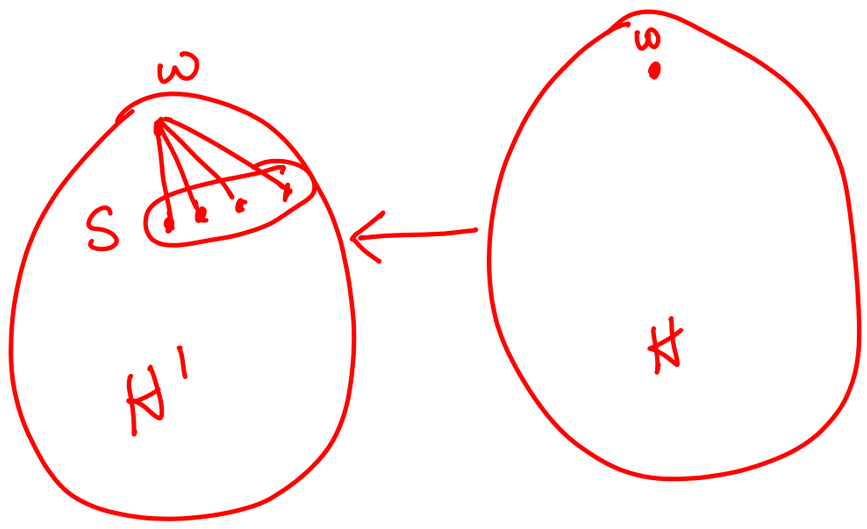
2 1 1 1 →



For $n \geq 4$

Transform both G and H using the previous strategy to ones where the highest degree vertex connects to the next highest degree vertices
Delete highest degree vertex from both, use induction.





Section 1.4 - Directed Graphs

Monday, February 6, 2023 5:27 PM

Graphs as we've discussed them are pictorial representations of symmetric relations on a set V

Of course, in general relations need not be symmetric. Can we generalize our discussions of graphs to incorporate this case? How much of what we previously introduced remains? What changes?

Example:

Suppose we consider the relation on $\{0,1,2,3,4,5,6\}$ given by xRy if $x^2 \equiv y \bmod 7$ skip.

Draw this as a directed graph

Definition:

A *directed graph* or *digraph* G is a triple consisting of a vertex set $V(G)$, an edge set $E(G)$ and a function assigning to each edge an ordered pair of vertices.
If $e \mapsto (x, y)$ we say x is the *tail* and y is the *head* of the edge, and we say e is an edge from x to y

We will often identify an edge with this ordered pair of vertices, calling xy an edge
We also say that y is a *successor* of x, and that x is a *predecessor* of y
We write $x \to y$ as "there is an edge from x to y"

Definition:

In a digraph, a *loop* is an edge for which the tail and head coincide.
Multiple edges are edges having the same ordered pair of endpoints.
A digraph is *simple* if there are no multiple edges

Each vertex can have one single loop

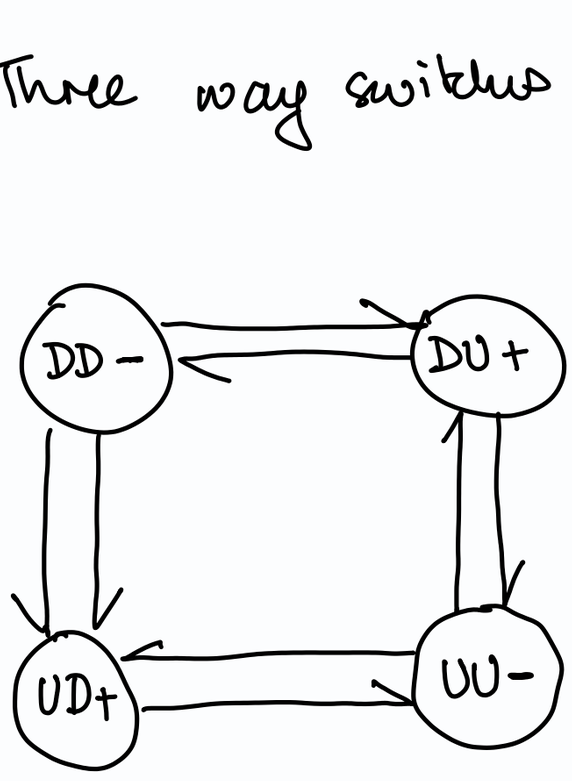
Definition: (tangential)

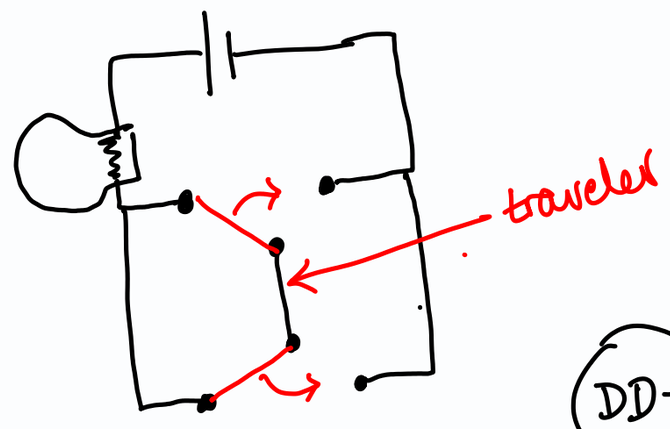
Finite state machines

Collection of states is vertices
Edges represent possible transitions (often labeled with causative descriptions)

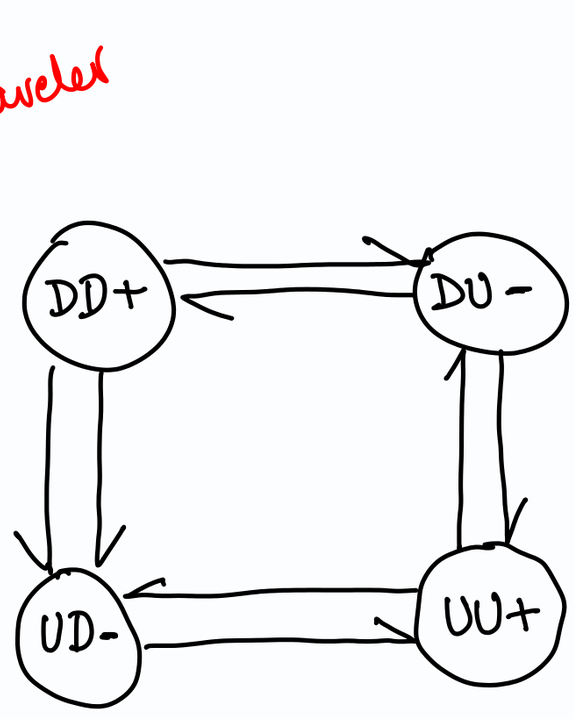
Turing Machines (if desired)

Three way switches





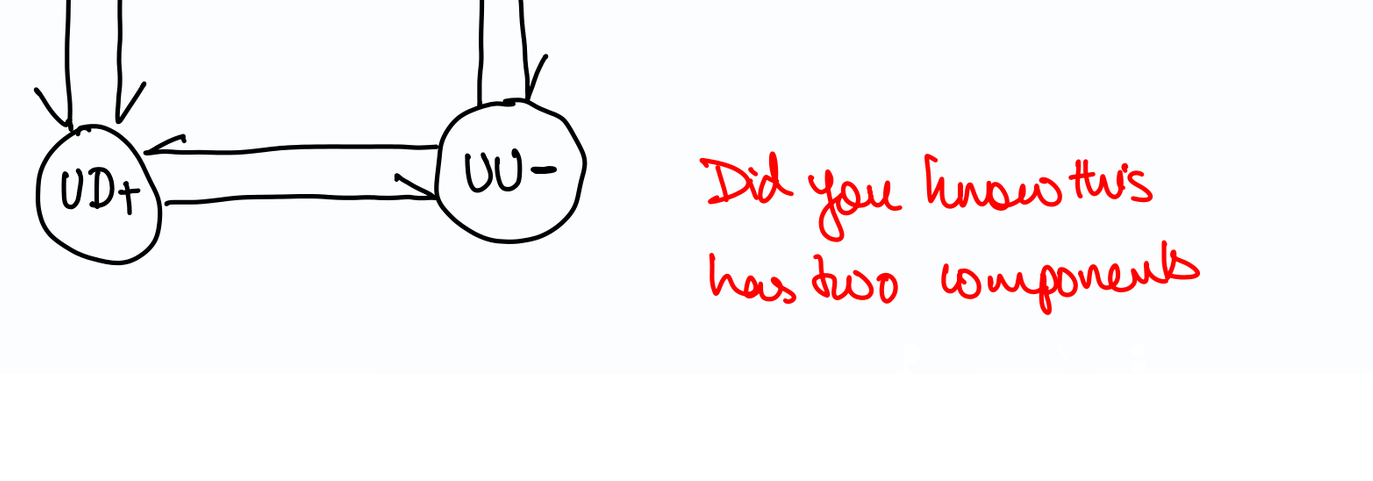
Did you know this has two components



Definition: (tangential)

Markov Chains

Collection of states is vertices
Edges are labeled by their probability, such that the total probability out of each vertex is 1
(explain this - simple example of weather or such?)



weather

We can define a few similar kinds of graphs in this case as we did in the undirected graph case)

Definition:

A digraph is a *path* if it is a simple digraph whose vertices can be linearly ordered so that $u \to v$ if and only if v immediately follows u in the order.

A *cycle* is similarly defined with an ordering of the vertices as points on a circle.

Note:

Digraphs give us a general way to encode relations

Any arbitrary function $f: A \to A$ induces a relation Γ_f on A given by $\{a, f(a) | a \in A\}$

(This relation is called the *graph* of f)

This, in turn, gives us a digraph, the *functional digraph* of f! (This is the very basics of the simplest case of dynamical systems)

$f(1) = 2$ $\quad$ $f(4) = 1$ $\quad$ $f(7) = 7$

$f(2) = 4$ $\quad$ $f(5) = 3$

$f(3) = 6$ $\quad$ $f(6) = 5$

Proposition: (decide whether or not this is worth including)

If A is a finite set and $f: A \to A$, then the functional digraph consists of disjoint pieces consisting of cycles each with a finite number of attached 'predecessor paths'

Note:

Some people use "node" and "arc" instead of "vertex" and "edge" for digraphs.

Now, we'd like to relate this slightly more general case to what we already know, as much as possible.

Definition:

The *underlying graph* of a digraph D is the graph G with the same vertex set obtained by treating the edges of D as unordered pairs

When discussing the underlying graph of a digraph we use D & G. Otherwise just use G.

Definition:

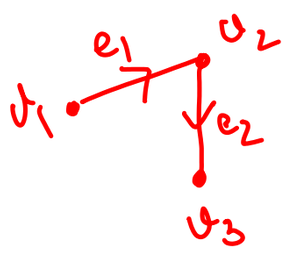
We can define *subgraphs, isomorphisms, decompositions, unions* of digraphs just as with graphs.

Definition:

The *adjacency matrix* of a digraph G is the matrix with the entry at i, j giving the number of edges in G from v_i to v_j

Is the adjacency matrix symmetric? No.

In the *incidence matrix* $M(G)$ of a loopless digraph G we set the entry at i, j to $+1$ if v_i is the tail of e_j, to -1 if v_i is the head of e_j, or 0 otherwise.



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{cc} e_1 & e_2 \end{array} \\ \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 0 & -1 \end{bmatrix} \end{array}$$

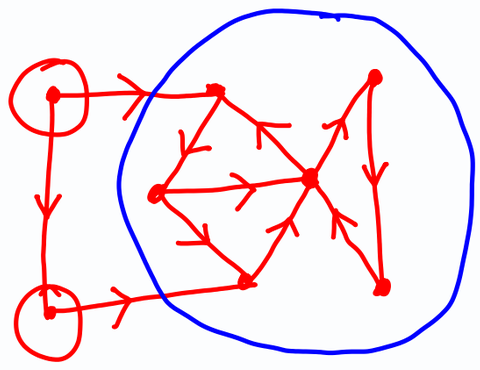
Question:

What can we say about connectedness?
(Draw a picture to illustrate why this is a nuisance)

Definition:

A digraph is *weakly connected* if its underlying graph is connected.
A digraph is *strongly connected* or *strong* if there is a path from u to v for each pair of vertices u, v

Strong components of a digraph are maximal strong subgraphs.



Note:

For regular graphs, a graph was the union of its components.

Is a digraph necessarily the union of its weak components?
What about its strong components?

NOPE

Application: (Basic game theory)

Two player games can be written as finite state machines
Vertex sets are states of the game with edges connecting states which are connected by a single move.
Some states are *winning states* for player 1, some others are *losing states*.

This framework is incredibly useful for asking questions about optimal strategies in a game, or in developing AI frameworks for such tasks

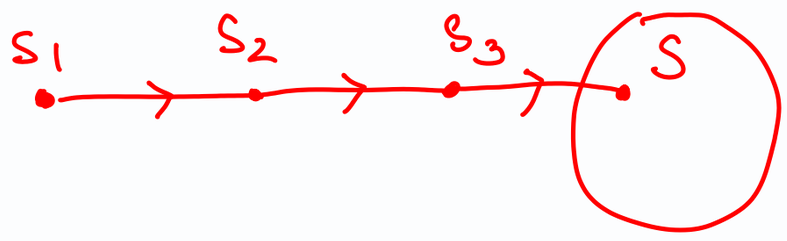
Suppose that we have a game such that the win states W are such that whichever play makes a move which enters the set W wins the game, with no edges leaving W

(Nim [that game with picking 1 or 2 pebbles] strategy idea here)
Suppose you can identify a set of states S containing W satisfying two properties

- No two vertices in S are adjacent.
- Every vertex in $\overline{S}$ has an outgoing edge to a vertex in S ← you can win!

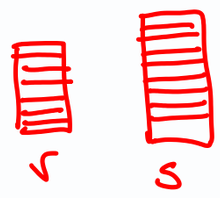
If you can make a move that brings the game to a state in S, you will win (without any subsequent mistakes)

Player 1 makes a move



S = set of states that once you move to will force a win for you.

Ex: 2 piles of pennies



Can remove any # of pennies from one pile. Whoever removes the last coin wins.

$W = \{(0,0)\}$ $S = \{(r,r) : r \geq 0\}$. No edges within S.

Definition:

A *kernel* in a digraph D is a set $S \subseteq V(D)$ such that S induces no edges and every vertex outside S has a successor in S. → Note: if S contains W, then we have a winning strategy for our game

Question:

When can we guarantee the existence of a kernel in a digraph? Are there good techniques to finding them?

Theorem: (Richardson 1953)

Every digraph having no odd cycle has a kernel

Pf: Suppose D is strong (=y connected)

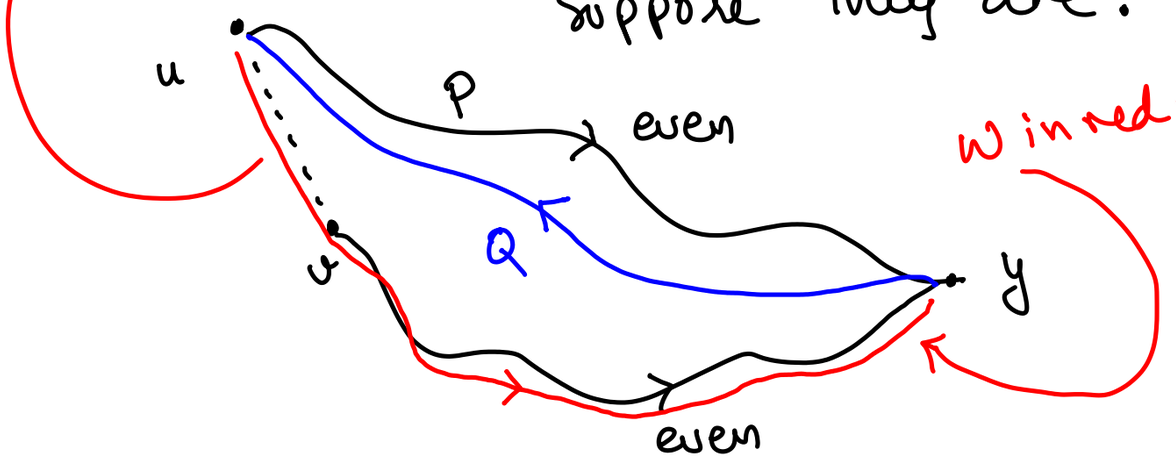
Fix $y \in V$. Let $S = \{x : d(x,y) \text{ is even}\}$ → $x \xrightarrow{\text{to}} y$ distance is length of shortest path from x to y.

Suppose $z \notin S$, then since D is strong $z \to y \Rightarrow d(z,y)$ is odd. Take

$z \xrightarrow{z'} y$ path $d(z',y)$ is even $\Rightarrow z' \in S$. So every $z \notin S$ has a successor in S.

Left to show if $u,v \in S$, they cannot be connected.

Suppose they are.



w in red is odd length walk.

$|Q|$ can be odd or even

$\Rightarrow |QUP|$ or $|QUW|$ is odd.

$\Rightarrow \exists$ closed odd walk.

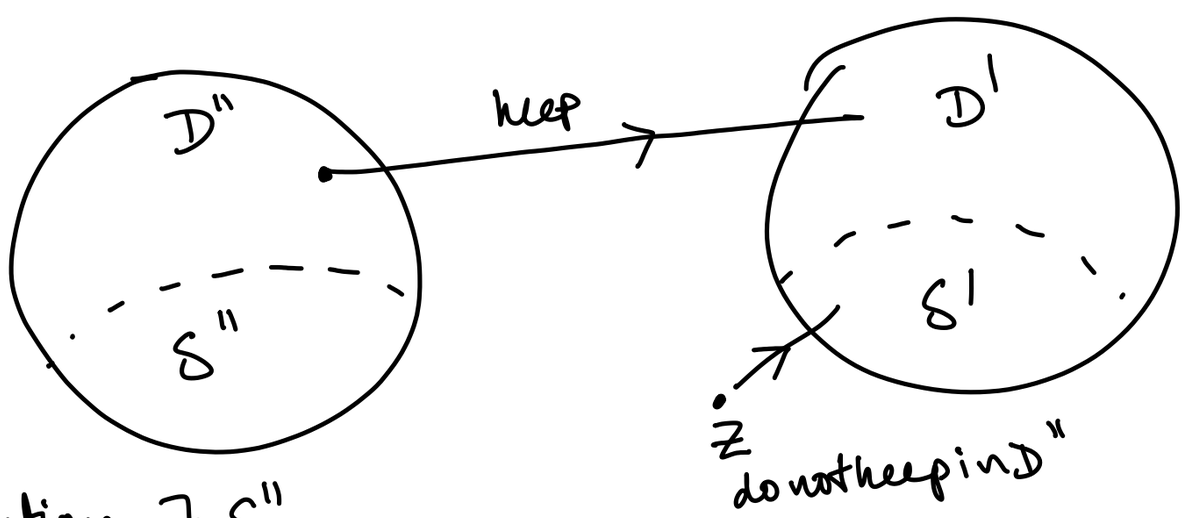
But every closed walk contains an odd cycle!

↑ good strategy in general.

The rest of the proof is by induction.

Ex: $\exists$ a strong component D' which does not have edges from $V(D')$ to do $(V(D')^c)$. D' has a kernel! (subgraph induced by) S'.

Find D'' by deleting S' and and all predecessors of S' in D'.



by induction $\exists S''$.
a kernel of D''.

$S'' \cup S'$ is a kernel!

One of the most useful concepts for graphs was the idea of degree. How is this altered in the directed case?

Definition:

Let v be a vertex in a digraph. The *outdegree* $d^+(v)$ is the number of edges with tail v. The *indegree* $d^-(v)$ is the number of edges with head v

The *out-neighborhood* or *successor set* $N^+(v)$ is the set of successors.
Likewise, *in-neighborhood* or *predecessor set* $N^-(v)$

For a digraph G, we have minimum and maximum indegrees and outdegrees $\delta^{\pm}(G)$ and $\Delta^{\pm}(G)$

Proposition:

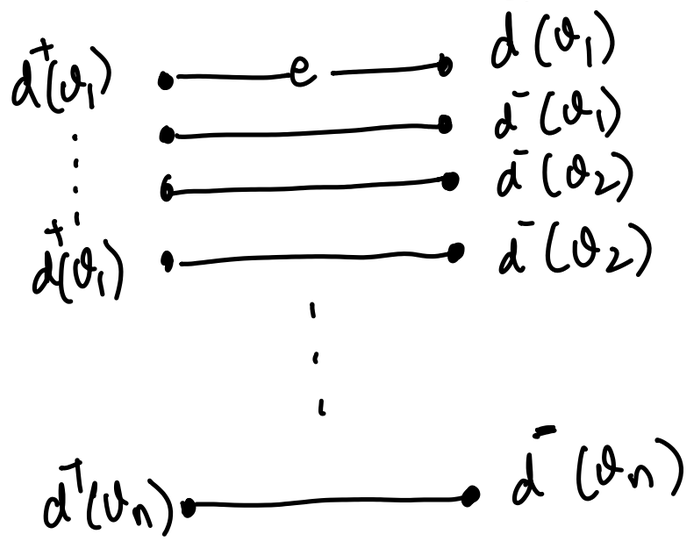
In a digraph, the sum of indegrees and outdegrees are both equal to the number of edges.

We can define the equivalent of degree sequences, too!
In this case, we would be interested in a list of "degree pairs" $\left(d^+(v_i), d^-(v_i)\right)$

Proposition:

A list of pairs of non-negative integers is realizable as the degree pairs of a digraph if and only if the sum of the first coordinates equals the sum of the second coordinates.

↳ consistent with degree sum formula.

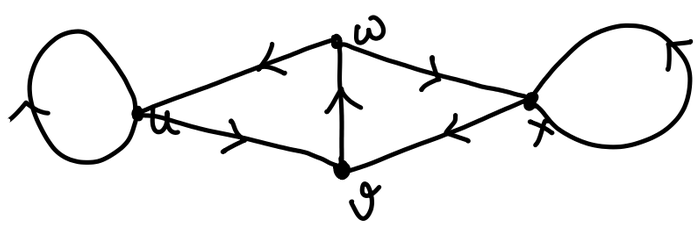


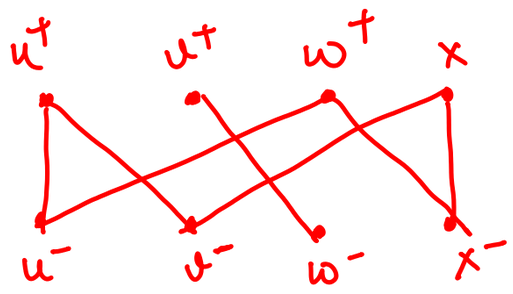
Of course such a digraph is not simple.

Is there an equivalent to the Havel-Hakimi criterion?
Stating such things tends to be a pain in the neck. Typical techniques involve transforming digraphs into graphs in some way, then using that transformation to translate results.

Definition:

The *split* of a digraph D is a bipartite graph G whose partite sets V^+, V^- are each copies of $V(D)$, consisting of points v^+, v^- for each $v \in V(D)$
An edge connects u^+ to v^- in G if $u \to v$ in D





Note:

When one defines such a transformation, it's often helpful to ask what properties are and are not translated over in a convenient way
How does degree in the split of a digraph relate to indegree/outdegree? (in a reasonably clear way)
How does connectedness relate? (not really at all)

This is one more reason to allow splits

Can one invert the construction of the split? Namely, given a X, Y-bigraph with $|X| = |Y| = n$, can one construct a directed subgraph from it whose split is the original graph? (Yes, just use a bijection between partite sets and done)

You can provide a recursive test for such a digraph a provide a Havel-Hakimi style condition (Ex 32)

Can we ask questions about Eulerian cycles on directed graphs?

Definition:

We can define *trails*, *walks, circuits,* the *connection relation* in the same way as for graphs, with the only caveat being that motion along an edge must go from the tail to the head

Definition:

An *Eulerian trail* in a digraph is a trail containing all edges. An *Eulerian circuit* is a closed trail containing all edges. A digraph is *Eulerian* if it has an Eulerian circuit.

Proving which digraphs are Eulerian is mostly the same as in the graph case. First, prove a lemma.

Lemma:

If G is a digraph with $\delta^+(G) \geq 1$ then G contains a cycle. The same can be said if $\delta^-(G) \geq 1$

Proof:

(maximal path argument)

Theorem:

A digraph is Eulerian if and only if $d^+(v) = d^-(v)$ for each vertex v and the underlying graph has at most one nontrivial component.

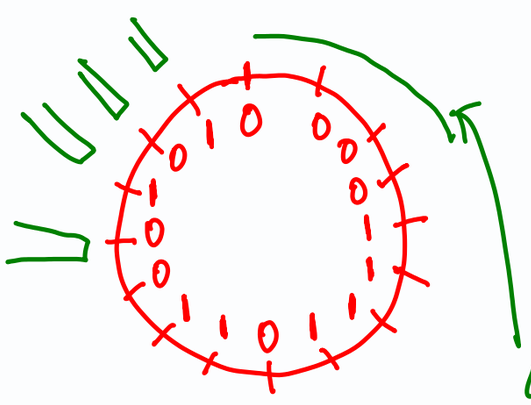
Pf: Ex 19 or 20 or in class.

Application: (De Bruijn Cycles)

Question:

Is it possible to write 2^n binary digits in a cyclic order such that all substrings of length n are different binary strings?

n=4
$2^4 = 16$



0000
0001
0011
0111
1111
1110
1101
1011

determine where you are by reading just n digits.] Crank position sensor? Encoder.

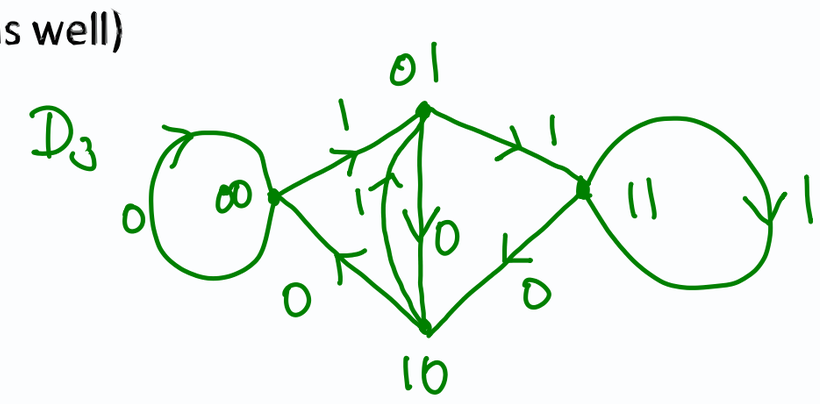
Keep track of where you are by just reading n digits.

Solution:

Define a digraph D_n with vertices given by $n-1$ binary strings, and directed edges between s_1 and s_2 if s_2 is obtained from s_1 by deleting the first digit and appending a new digit to the end. (edges are usually labeled, as well)

corresponds to the shift operation.

(Draw D_2)



01110100

011 010
111 100
110 000
101 001

Claim:

D_n is strong (there's actually always a path of length n)

Claim:

D_n is Eulerian

These graphs are called *de Bruijn graphs* of order n on an alphabet of size 2.
(One can define similar things for larger alphabets as well)

Every vertex has out degree 2. (can always append 0 or 1)
" " in " "

aa → ab
ba → b

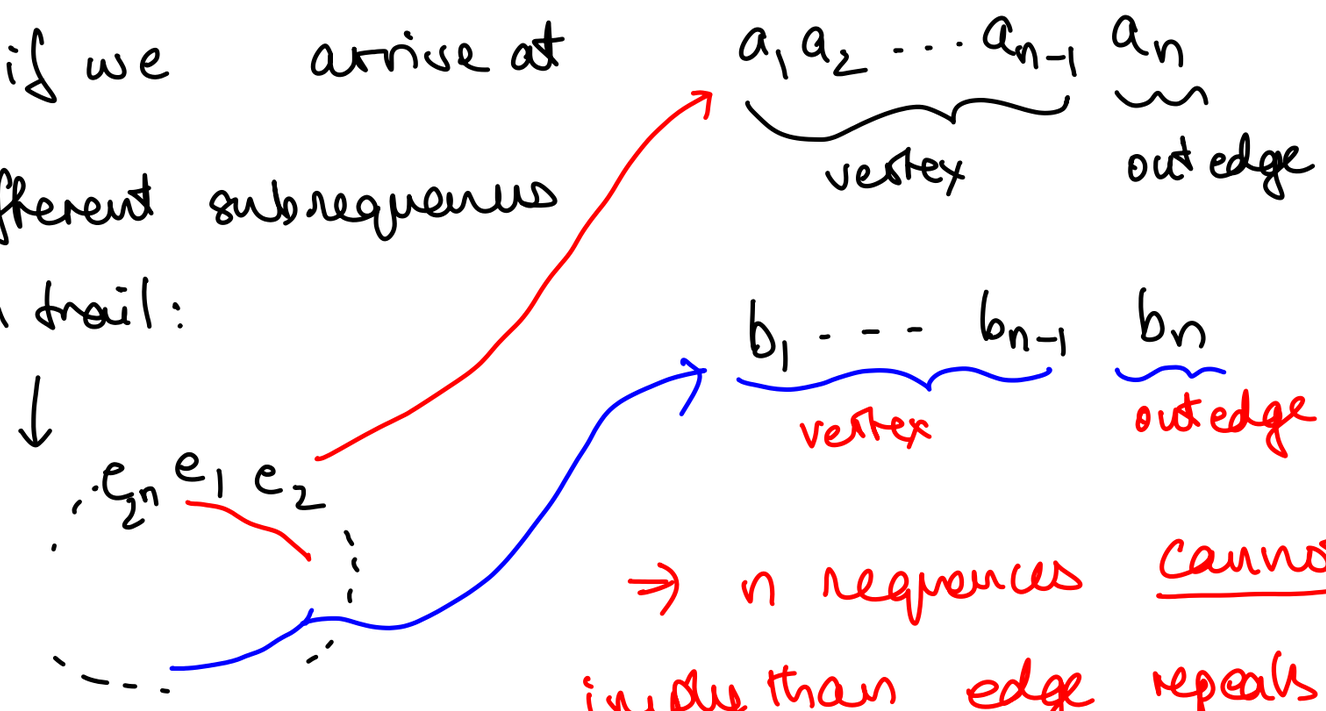
⇒ it is Eulerian.

# of edges = $2 \cdot \underbrace{2^{n-1}}_{\text{vertices}} = 2^n$ (by degree-sum)

Let's take a trip of n steps; $a_1 a_2 \cdots a_{n-1}$ (edge labels) $\Rightarrow$ we are vertex $(a_1, \ldots, a_{n-1})$

So if we arrive at

2 different subsequences of the trail:



$\Rightarrow$ n sequences cannot match since it would imply than edge repeats. But the path is Eulerian.

$\Rightarrow$ Only distinct n sequences appear. There are 2^n different starting points $\Rightarrow$ 2^n different n-sequences appear.

Orientations and Tournaments

Quick Question:

How many distinct simple digraphs are there on a given vertex set?

2^{n^2} n^2 ordered pairs.

It's important for us to have a number of ways to transition between graphs and digraphs. One way to go from a graph to a digraph is to just assign directions to all the edges.

Definition:

An *orientation* of a graph G is a digraph obtained from G by picking an orientation ($x \to y$ or $x \leftarrow y$) for each edge $xy \in E(G)$

An *oriented graph* is an orientation of a simple graph.

A *tournament* is an orientation of a complete graph.

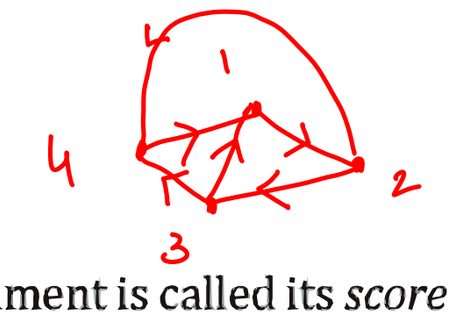
$3^{\binom{n}{2}}$ (in/out/no edge)

$2^{\binom{n}{2}}$

(It's called a tournament by analogy with a collection of sports teams playing games or such in a "round-robin tournament")

Idea:

One thinks of $x \to y$ as imagining "x defeats y"



Note:

Following this analogy, the outdegree sequence of a tournament is called its *score sequence*

Question:

In a tournament, how can one determine the indegrees if given the outdegrees?

$\bar{d}(i) = (n-1) - d^+(i)$

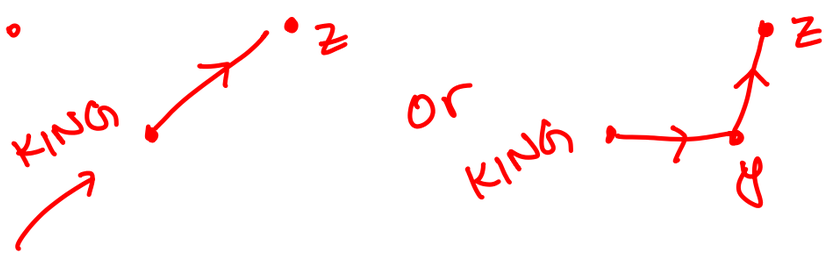
Question:

How many oriented graphs are there on a given vertex set?

Question:

How many tournaments are there on a given vertex set?

Tournaments may not necessarily have a single "winner", but we can nonetheless characterize those teams which did the "best" in some sense. Multiple teams may have the same # of wins.



Definition:

In a digraph, a *king* is a vertex from which every vertex is reachable by a path of length at most 2

Note:

In the context of tournaments, a king is a vertex v such that for every other vertex w, either $v \to w$

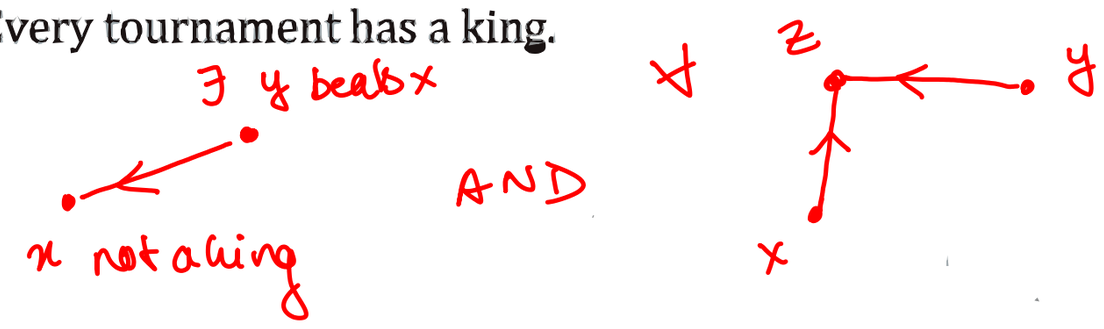
or there is a third vertex u such that $v \to u$ and $u \to w$

(Kings need not be unique!)

Proposition: (Landau 1953)

Every tournament has a king.

Pf:



$\Rightarrow \deg(y) > \deg(x)$ Its clear from the picture.

$\Rightarrow$ Every vertex of maximal degree is KING.

Section 2.1 - Basic Properties of Trees and Distance

When we study graphs as networks, we're often interested in one of two features.

Efficiency - how well does a graph connect a collection of vertices?
Resiliency - often determinable by looking at the cycle structure of graphs

Trees will let us study the former

Definition:

A graph with no cycle is *acyclic*, or a *forest*

A *tree* is a connected acyclic graph.

A *leaf* (or *pendant vertex*) is a vertex of degree 1

↓ a leaf is a vertex

Examples:

Paths are trees
Claws are trees

Proposition:

All trees and forests are bipartite.

We will often be particularly interested in finding large trees as subgraphs of a given graph.

Definition:

A *spanning subgraph* of G is a subgraph with vertex set $V(G)$

A *spanning tree* is a spanning subgraph that is a tree.

Trees have *many, many* properties which are extremely useful. We'll want to establish that all of these properties are equivalent, so that if we find one of them to be true then we know all are.

Lemma:

Every tree with at least two vertices has at least two leaves. Deleting a leaf from an n-vertex tree produces a tree with $n-1$ vertices.

Proof: A connected graph with at least 2 vertices has an edge.

Take a maximal path to get leaves.

Take G, delete leaf get $G' = G - w$. This is connected and acyclic.

Note:

This lemma has the useful implication that all trees with n vertices can be built by attaching a new vertex to a tree with $n-1$ vertices.

Theorem:

For an n-vertex graph G with $n \geq 1$, the following are equivalent.

A) G is connected and has no cycles
B) G is connected and has $n-1$ edges
C) G has $n-1$ edges and no cycles
D) G has no loops and has, for each $u, v \in V(G)$, exactly one u,v-path

Proof:

First prove that any two of {connected, acyclic, $n-1$ edges} implies the third.

A→(B,C) by induction on n. Take a graph with n-vertices, remove a leaf

enough to prove n-1 edges: prev lemma says take G, $n(G)=n$, delete an leaf. Get $G' = G - v$ acyclic connected, $n(G')=n-1$. By induction $e(G')=n-2$

from cycles. But deleting edges will not make it disconnected.

B→(A,C) Delete edges until G is acyclic. The resulting graph is connected, so must have at least $n-1$ edges. (prev. lemma said at least n-k components. ⇒ $k \geq n-1$

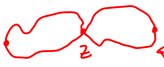
C→(A,B) Split into components, each component must satisfy B sum vertices → $G_1 \cdots G_k$ $e(G_i) = n(G_i) - 1$ ⇒ $\sum e(G_i) = n - k$ ⇒ $k = 1$

Now we prove equivalence of D with first three

A → D Take two different paths, find a cycle.

D → A If G has a cycle, that's two paths between a pair of vertices.

In fact find 2 paths st $|P_1| + |P_2|$ is minimal. Take inf over u, v as well ⇒



← not possible ⇒ this is a cycle.

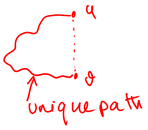
no cycles ⇒ no loops (cycles of length 1)

obvious.

Corollary:

Every edge of a tree is a cut edge ← an edge is a cut edge iff it belongs to a cycle (lemma)

Adding one edge to a tree forms exactly one cycle [draw a picture] ←



Every connected graph contains a spanning tree. ← find cycles, delete one edge.

The set of spanning trees of a graph will tell us a lot about the structure of the graph. We'll need a few basic propositions in this direction.

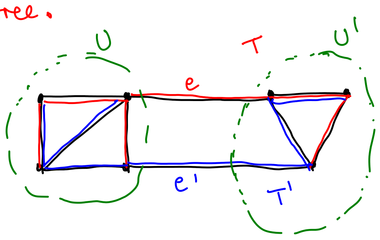
Proposition:

Suppose T, T' are spanning trees of a connected graph G and suppose $e \in E(T) \setminus E(T')$
Then there is an edge $e' \in E(T') \setminus E(T)$ such that

$T - e + e'$

is a spanning tree of G

Suppose e unique to T ∃ an e' unique to T' you can add to keep T a spanning tree.



e is a cut edge of T. ⇒ 2 components U, U'

Must exist $e' \in T'$ since T' is connected. It cannot be in T

T - e + e' is connected and has n-1 edges ⇒ it's a tree.

This proposition is really similar, but we do the addition and subtraction in the other order

Proposition:

Suppose T, T' are spanning trees of a connected graph G and suppose $e \in E(T) \setminus E(T')$
Then there is an edge $e' \in E(T') \setminus E(T)$ such that

$T' + e - e'$

is a spanning tree of G

Again fix any edge unique to T'. ⇒ ∃ an e' you can remove from T' and add e to it & T' + e - e' is again a spanning tree.

T' + e has a cycle. Remove any edge from that cycle!

T' + e - e' has n-1 edges!

Note:

One can actually find e' to satisfy both propositions simultaneously. (Ex 37)

Every graph which is 'connected enough' contains a great variety of different trees.

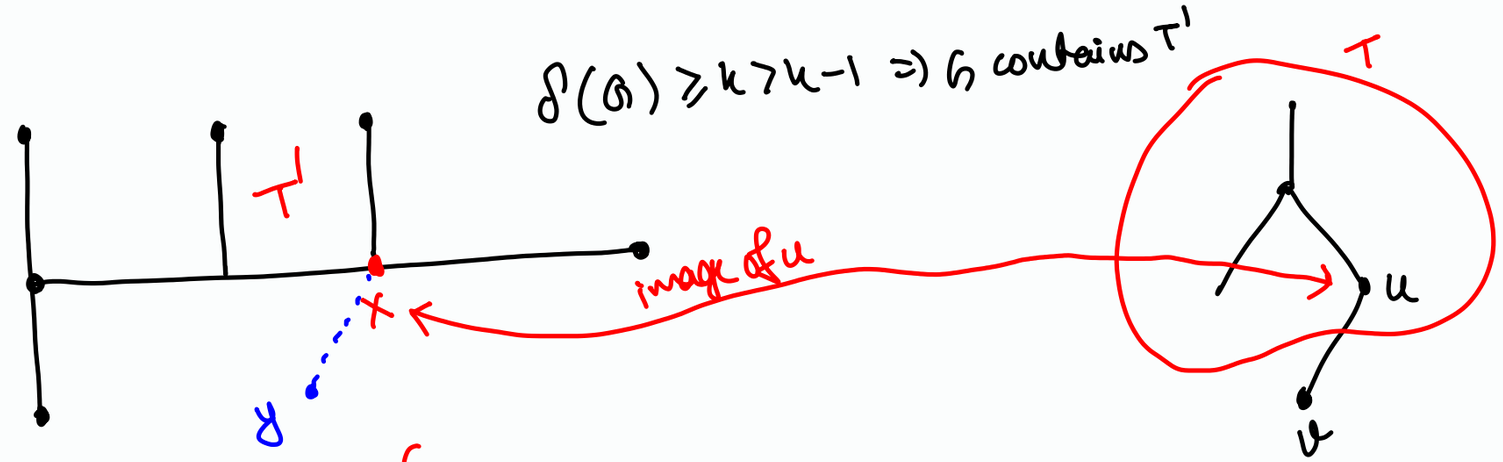
Proposition:

If T is a tree with k edges and G is a simple graph with $\delta(G) \geq k$, then T is a subgraph of G

Proof:

Induction on k. $k = 0$ is trivial - just a vertex

Take a leaf v in T, let u be its neighbor. G contains $T' = T - v$ since $\delta(G) \geq k > k - 1$
Let $x \in V(G)$ correspond to $u \in V(T')$. Necessarily, T' cannot contain all G-neighbors of x



Why can't T' contain all neighbors of x? Because T' has at most $k-1$ edges.
But $d(x) \geq k$. Map y to v to get T.

Note:

The graph K_k has $\delta = k - 1$ but contains no tree with k edges.

(Since a tree on k vertices must have $k-1$ edges)

One might wish to prove an equivalent proposition based on the number of edges in G.
The *conjectured* bound is that if $e(G) > n(k-1)/2$ then G contains all trees of order k

(Erdös (1964) can tell them alot Erdös magic.)

Measuring Distance in Graphs

Definition:

If G has a u, v-path, the *distance* from u, v is $d_G(u, v)$ or $d(u, v)$ is the least length of a u, v-path.

If G has no u, v-path, we set $d(u, v) = \infty$

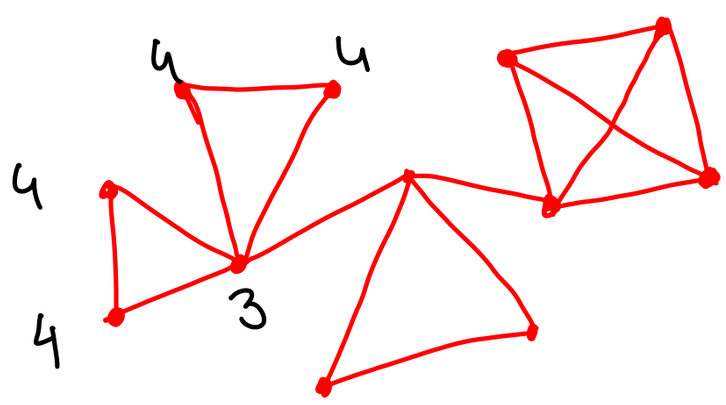
diam. circumscribed to enclose set

The *diameter* of G is $diam\ G = \max_{u,v \in V(G)} d(u, v)$

The *eccentricity* of a vertex u is $\epsilon(u) = \max_{v \in V(G)} d(u, v)$

The *radius* of a graph G is $rad\ G = \min_{u \in V(G)} \epsilon(u)$

minimal eccentricity



Note:

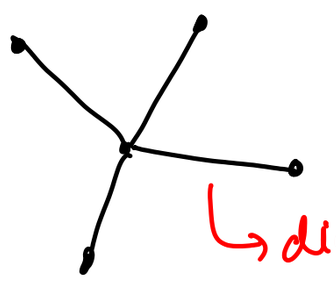
The diameter is the maximum vertex eccentricity.

Question:

What can we say about radius/diameter if a graph is disconnected?

Question:

On $n \geq 3$ vertices, what is the tree with smallest diameter?



↳ diam 2

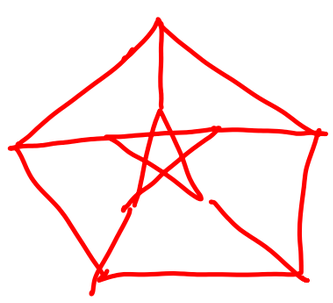
↳ radius 1

We can think of diameter of a graph as reflective of two things

The order of the graph - many vertices means it is possible for paths to be longer

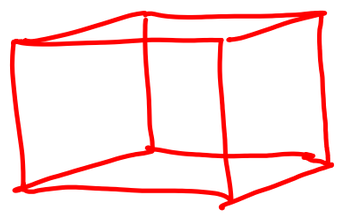
The number of missing edges - the fewer edges present, the more circuitous paths will have to be

↑ from a complete graph



diam (Petersen) = 2

radius (Petersen) = 2



Q^k $\quad diam(Q^k) = k$ (flip all coordinates)

Question: Tree with largest diam on n vertices?

Theorem:

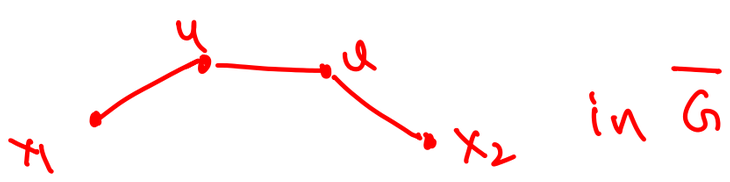
If G is a simple graph, then $diam\ G \geq 3$ implies $diam\ \overline{G} \leq 3$

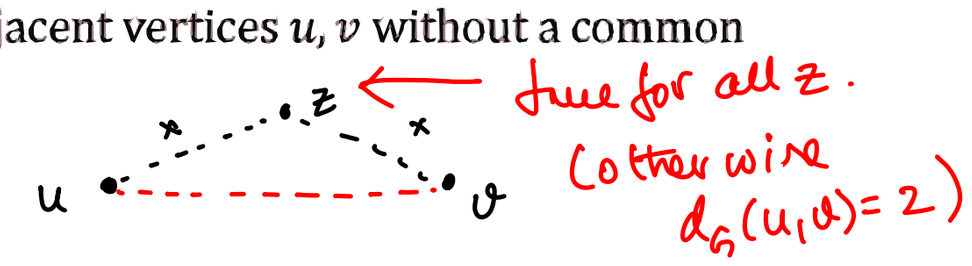
Proof:

Since G does not have diameter 1 or 2, there are nonadjacent vertices u, v without a common neighbor.

Every vertex in $\overline{G}$ must be connected to u and v ← that is not u or v

Worst case, from x_1 to u to v to x_2 (draw)





There are *many, many* ways to measure the idea of *centrality* in a graph. (There will be some homework about this eventually.)

Here's one way.

Definition:

The *center* of a graph G is the subgraph induced by the vertices of minimum eccentricity.

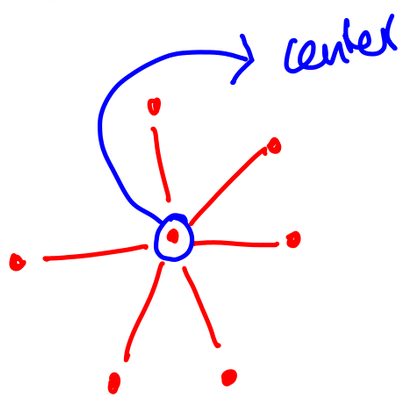
not degree.

Question:

Under what conditions is the center of a graph the whole graph?

- All vertices have same eccentricity - radius = diameter

For trees, we can characterize the center pretty well.



Theorem: (Jordan 1869)

The center of a tree is a vertex or an edge.

Proof:

By induction on number of vertices.

If $n \leq 2$ then the center is the whole tree.

Every vertex at maximum distance from $u \in T$ is a leaf! Delete all leaves

$e_{T'}(u) = e_T(u) - 1 \quad \forall\ u \in V(T)$ (Paths maximizing distance must end at a leaf)

If v is a leaf in T and w is its neighbor, $e_T(v) > e_T(w)$. But the center is the vertex of minimal e_T $\Rightarrow \arg\min_u e_T(u) = \arg\min_v e_T(v)$

By induction the center of T' is a vertex or a tree.

If we're treating a graph as a network, then we'll be quite interested in asking how short paths that can get around the graph are. We can quantity this, in an average sense, by looking at all possible shortest paths.

Definition:

The *Wiener index* of a graph G is $D(G)$ (or $W(G)$) given by the formula

$$D(G) = \sum_{u,v \in V(G)} d_G(u,v)$$

This kind of quantity is really important if we take a sequence of finite sets growing to an infinite set - discontinuities in the (appropriately normalized) limit can be used to tell us about phase transitions of materials [Wiener did it with paraffin] — ★ Good project.

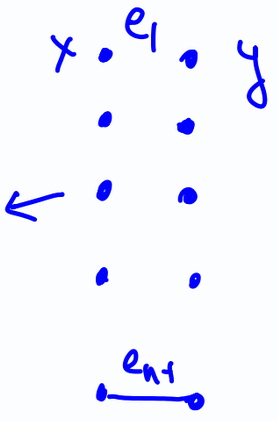
For finite graphs, we can characterize the extremal cases of the Wiener index without too much trouble.

Theorem:

Among trees with n vertices, the Wiener index $D(T)$ is minimized by stars and maximized by paths, in both cases uniquely.

Proof:

Tree has $n-1$ edges, so $n-1$ pairs of vertices are distance 1, and all other pairs of vertices are more distant.
Star achieves 2 for all others, which is minimal.



Suppose another tree achieves this: $(n-1)$ distance 1 and $\binom{n}{2} - (n-1)$ distance 1.

Pick a pair not in this set $\Rightarrow$ distance ≥ 2.

Pick x — v where x is a leaf.

Pick all other $n-2$ vertices. If they all have dist 2 then

* Compute $D(K_{1,n-1})$
* Show $D(P_n) = \binom{n+1}{3}$

For Wiener index of paths, note that

$$D(P_n) = D(P_{n-1}) + \binom{n}{2}$$

$D(P_n) = D(P_{n-1}) + \sum_{i=1}^{n-1} i$ ← distances from n^{th} vertex

Since $D(P_1) = 0$ and $D(P_2) = 1$, we have $D(P_n) = \binom{n+1}{3}$

We prove maximization by induction on n. $n = 1$ is immediate.
Take a leaf u in a tree T.

$$D(T) = D(T-u) + \sum_{v \in V(T) \setminus u} d(u,v)$$

Certainly $D(T-u) \leq D(P_{n-1})$ by induction. Enough to show $\sum_{u \in V(T) \setminus u} d(u,v)$ is maximized when $T = P_n$ and u is a leaf of T.

If T is P_n then the list of distances from u is $1, 2, \ldots, n-1$. Let P' be a maximal path from u in T. Then this contains the distances $1, 2, \ldots, n(P')-1$. Therefore if $n(P') < n$, some distances must repeat and reduce $\sum d(u,v)$

Thinking of the Wiener index as a measure of "the most connected graph", the next proposition is intuitive.

Proposition:

Out of connected n-vertex graphs, the complete graph minimizes D

Here's a useful little lemma, with a similar ideas as the previous proposition.

Lemma:

If H is a subgraph of G, then $d_H(u,v) \geq d_G(u,v)$

Corollary:

If G is a connected n-vertex graph, then $D(G) \leq D(P_n)$

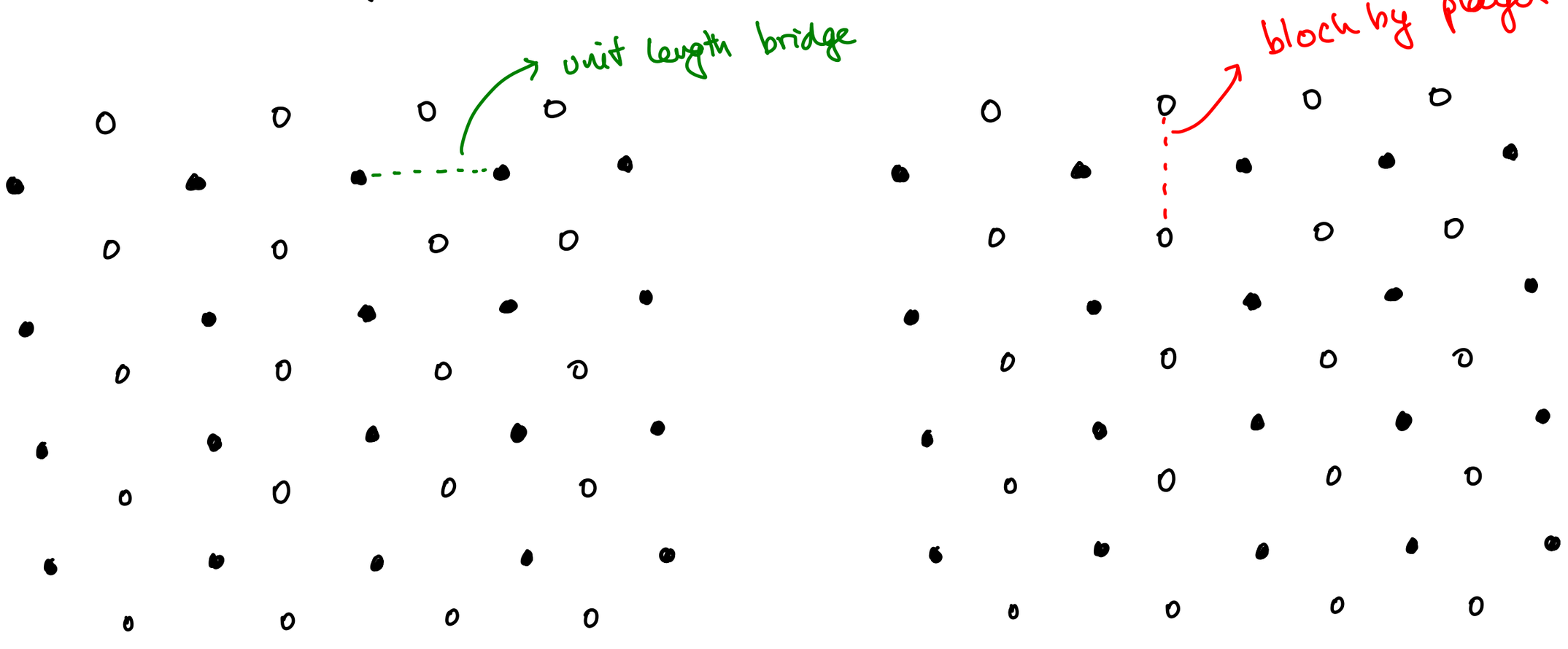
Proof:

Take a spanning tree of G. $T \subset G$. Then $D(G) \leqslant D(T) \leqslant D(P_n)$

If more time remains, talk about rooted trees, levels of trees, regular or balanced trees, and infinite trees. You could define the free monoid if desired.

Brid-It

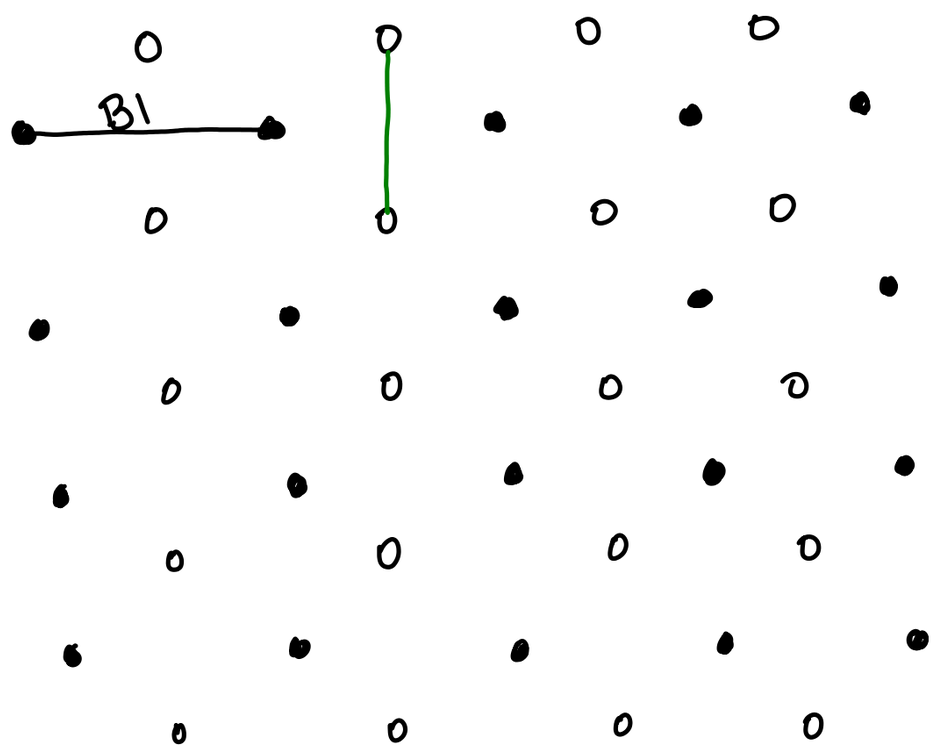
I like this example because it has a connection



Objective: Player 1 forms a bridge from East to West.
" 2 " " North to South.

Claim: Player 2 cannot have a winning strategy.

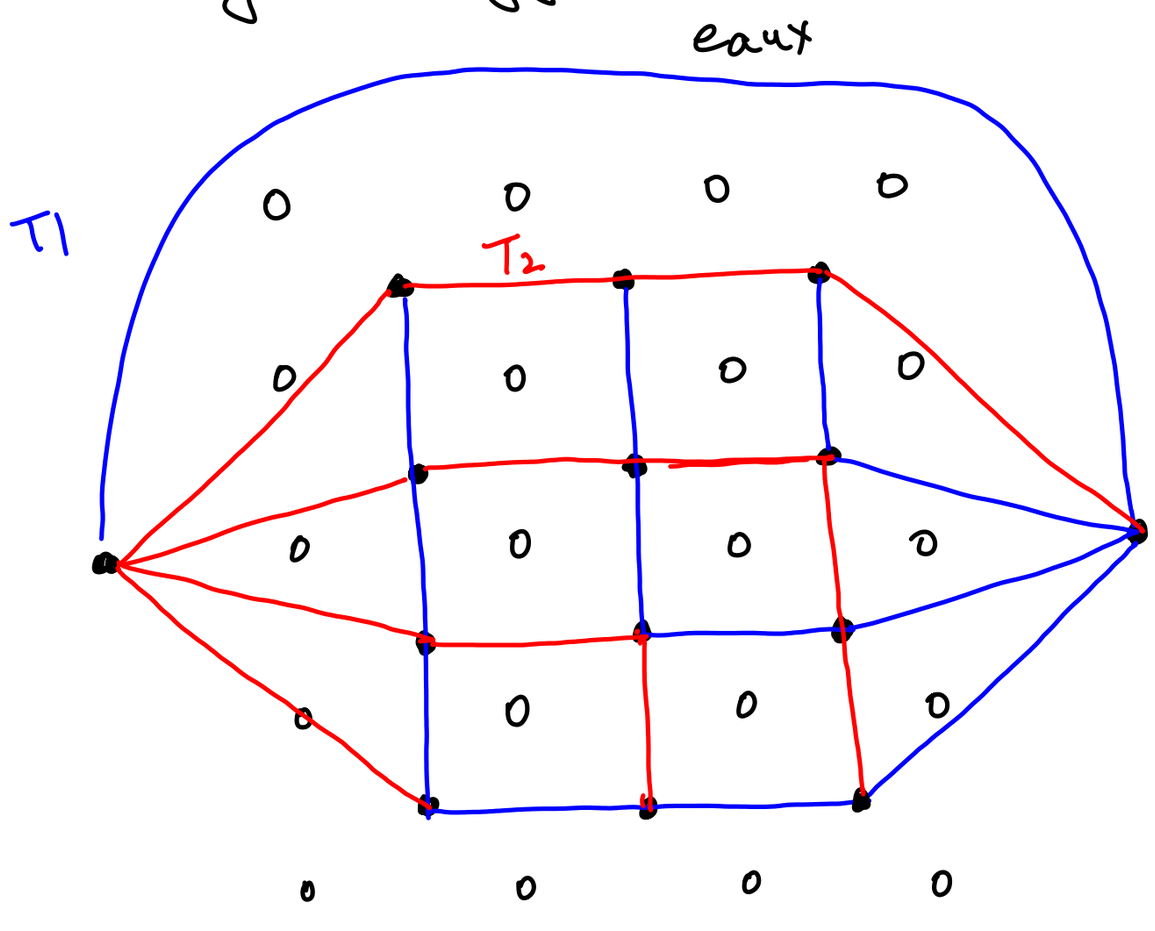
Pf: Since the board is symmetric, if player 2 did, then player 1 can just follow the strategy of player 2. But the board is not symmetric to parity! I don't buy it.



Now look at what P2's strategy calls for. If it's, then B1 is already played, so play an arbitrary move. And continue. Well I kinda buy it.

→ ☆ Can you turn this into a proof?

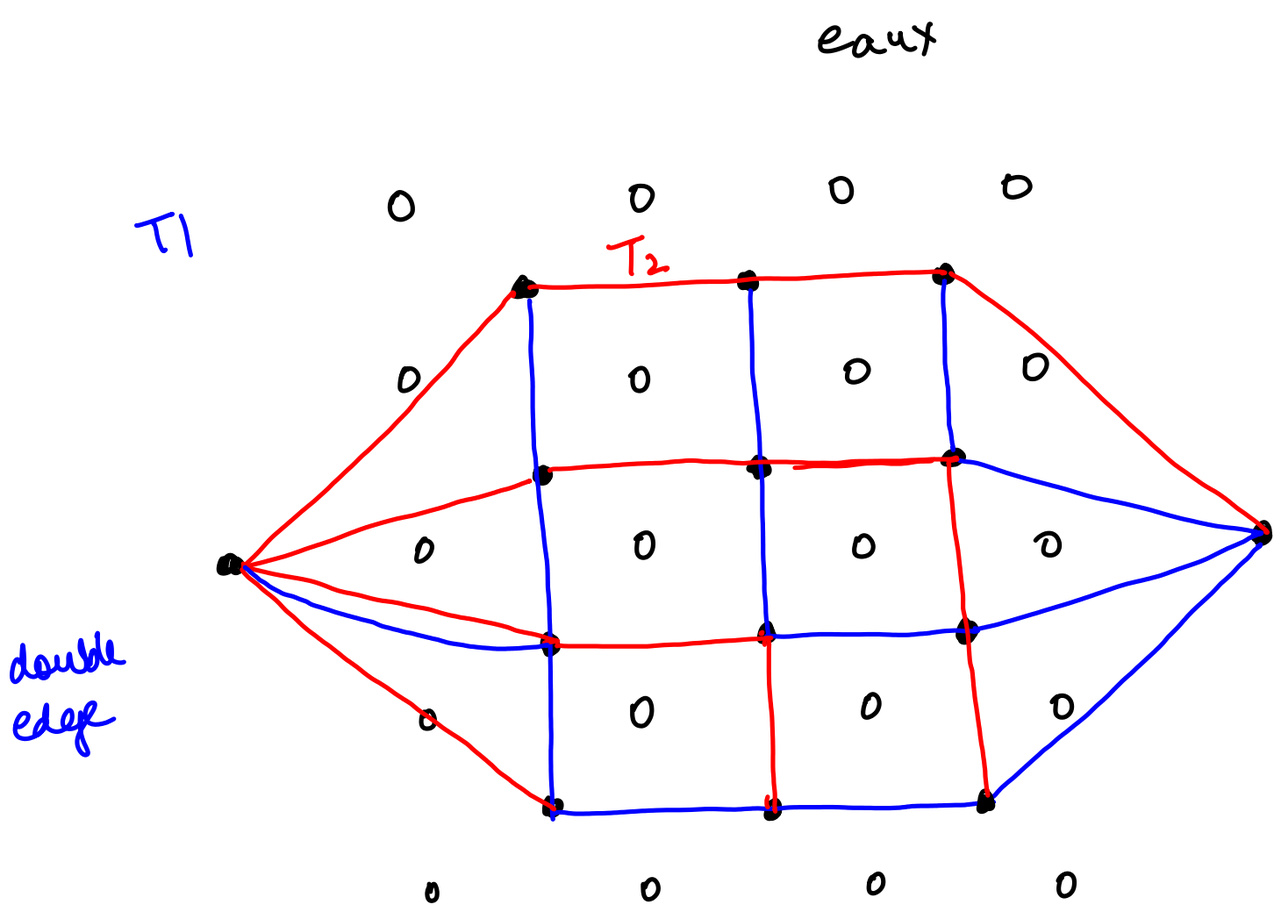
Player 1's winning strategy:



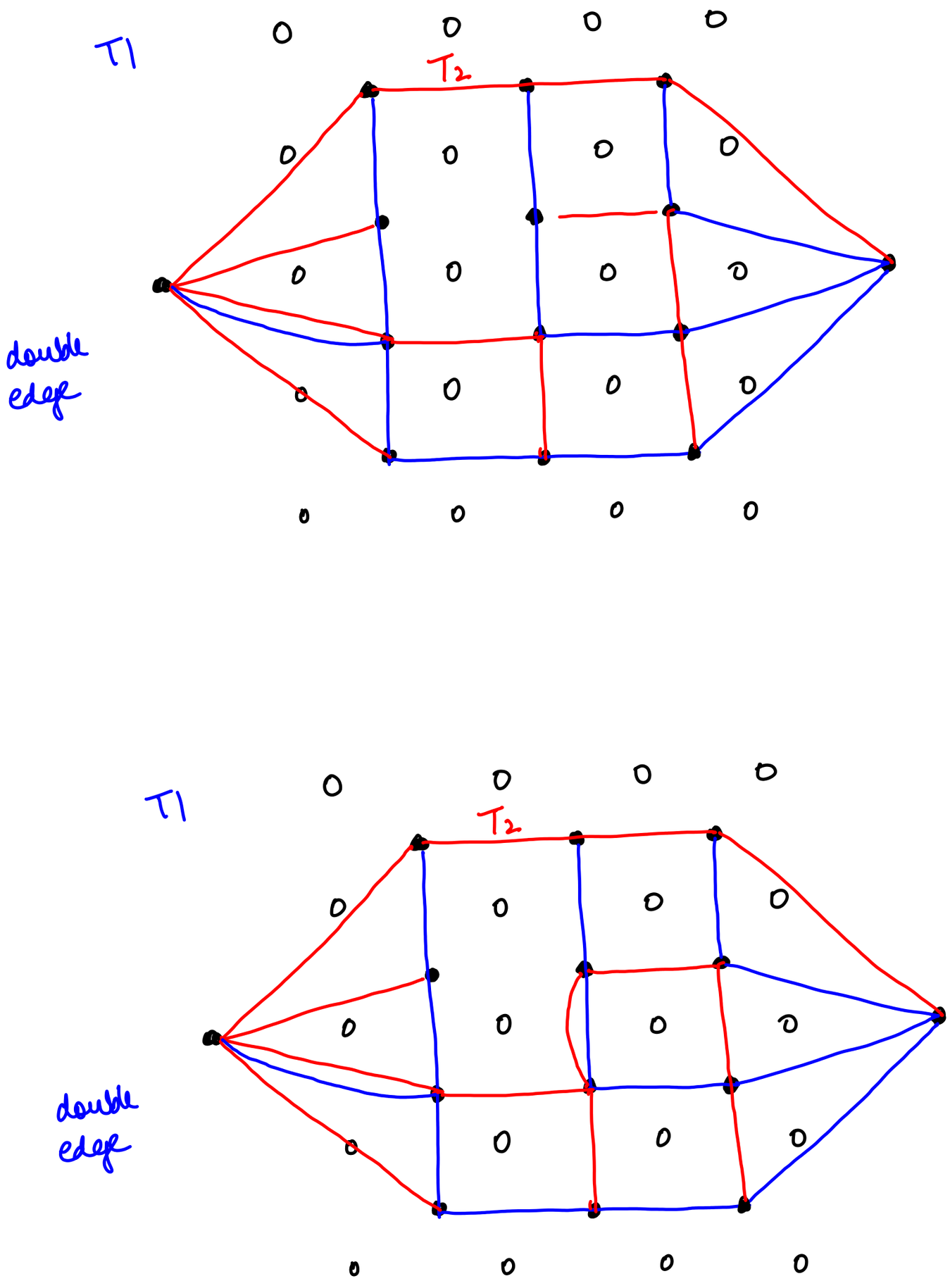
← can make this replacement.

→ P2 plays 1st and takes e_{aux}.

→ This disconnects T2 into 2 components. $\exists\ e \in T_1$ that reconnects T_2



Player 2 cuts T_1



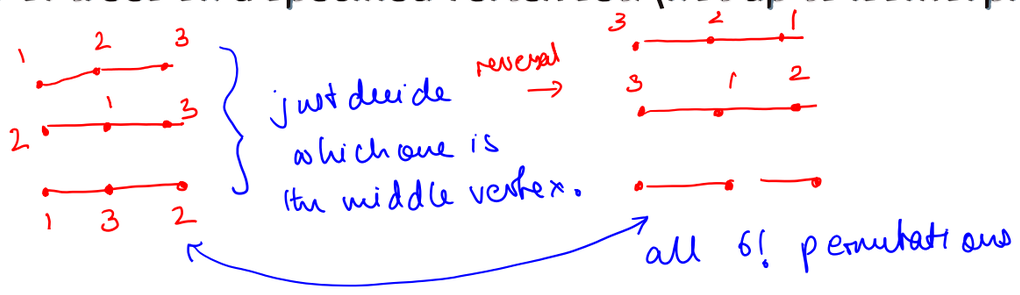
The process stops when no single edges remain to be cut.

Section 2.2 - Spanning Trees and Enumeration

Monday, February 20, 2023 11:49 PM

Today we're concerned with counting the number of trees on a specified vertex set. (not up to isomorphism)

2 vertices (1 and 2) - 1 tree
3 vertices (1, 2, 3) - 3 trees
4 vertices (1, 2, 3, 4) - 16 trees



Question: Is there an isomorphism between  and  ?

YES

Question: Is there an automorphism between  and  ?

NO

Remember: 2 graphs are automorphic iff they have the same adjacency matrix.

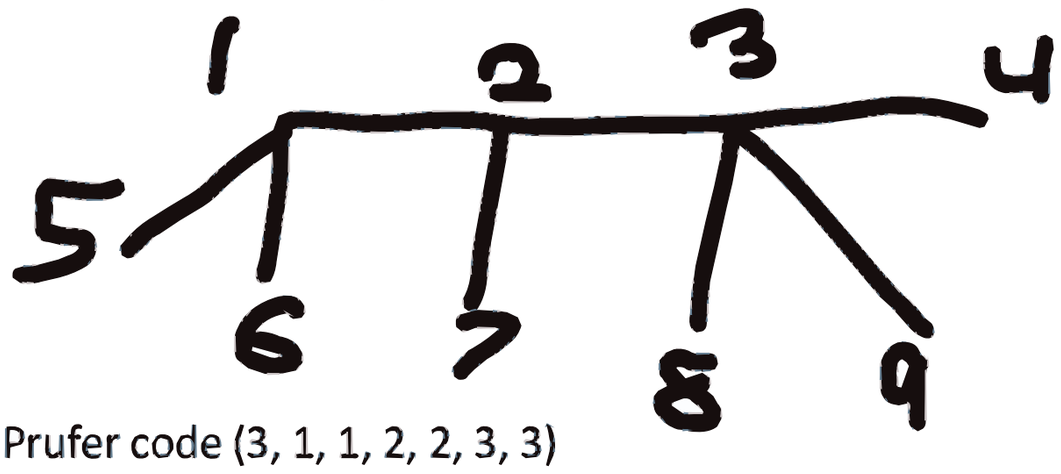
So we want to count trees upto automorphism (but not upto isomorphism)

To figure out a clever way to count trees, we'll use an algorithm to represent a tree on a given vertex set $S \subset N$

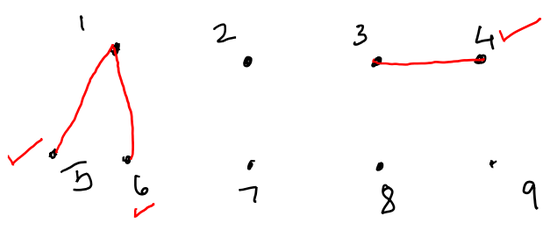
Algorithm: (Prufer Codes)

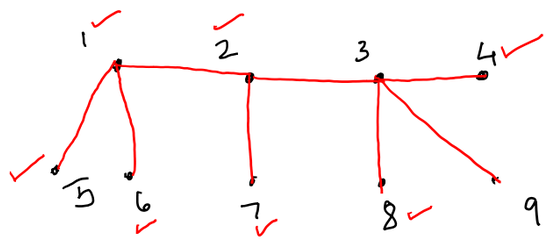
Input: a tree T with a given vertex set $S \subset N$
Output: $f(T) = (a_1, \dots, a_{n-2})$ a list of $n-2$ values in S

At the i-th step of the algorithm, delete the leaf of the current tree with smallest label (in S).
Let a_i be the label of the neighbor of that leaf.



This graph has Prufer code (3, 1, 1, 2, 2, 3, 3)

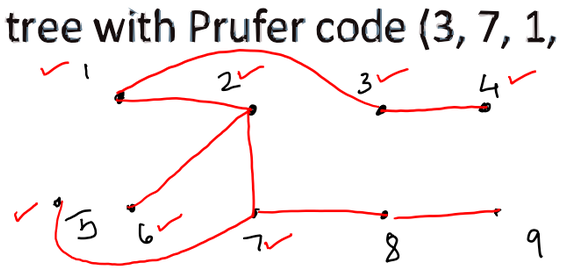




Question:

What does the tree with Prufer code (3, 7, 1, 2, 2, 7, 8) with $S = [9]$ look like?

what is the smallest thing that doesn't appear there? That must have been a leaf



Technique for Finding a Graph from a Code:

Start with a graph on vertex set S with no edges.
(idea - we'll go through a loop, each time adding an edge and marking one point)
At the i-th step:
Consider the portion of the Prufer code starting with a_i (ignoring all to the left of this)
Not all unmarked vertex labels can occur in this list (too short), let x be the smallest missing unmarked label
Include the edge xa_i
Mark x
At the end, two unmarked vertices remain. Connect them.

Theorem: (Cayley's Formula [1889])

For a set $S \subset N$ of size n, there are n^{n-2} trees with vertex set S

#of automorphism classes of labeled trees.

$n=3 \Rightarrow 3^{3-2} = 3$ trees.

- we consider labeled trees.
- if there is a relabeling that leaves the adjacency matrix invariant, then it will not be considered a separate tree. Unique up to automorphism.

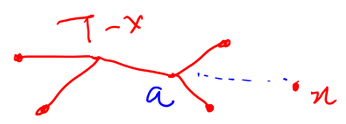
Proof: (Prufer 1918)

We'll show that our f is a bijection.

We proceed by induction on n. The $n = 1$ case is obvious.
The $n = 2$ case is the real base case, having an empty Prufer code.

For $n \geq 3$

Any leaf of T does not appear in the Prufer code $f(T) \equiv a = (a_1 \cdots a_n)$
In the process of determining the a, the first vertex not appearing in a was some $x \in S$, the first deleted, and was connected to a_1
Thus, for any T for which $f(T) = a$, x is a leaf of T connected to a_1
Then $T - x$ is a tree which must have Prufer code $(a_2, a_3, \ldots a_{n-2})$ on set $S \setminus \{x\}$
Use induction hypothesis to show that $T - x$ is uniquely defined.
There's only one way to attach x, so we're done.
(it must be attached to x)



This technique of proof gives us an easy way to count the number of trees after specifying each vertex's degree on a specified vertex set.

Corollary:

Given positive integers $d_1, \ldots, d_n$ summing to $2n - 2$, there are exactly $\frac{(n-2)!}{\prod(d_i-1)!}$ trees with vertex set $[n]$ such that vertex i has degree d_i for all i

$d(T) = n-1$ (Ah then, how do we know it's a tree)

$\sum(d_i - 1) = 2n - 2 - n = n - 2$

Proof:

If $x \in V(T)$ has degree d, it appears $d - 1$ times in the Prufer code for T

Then we are counting lists of length $n - 2$ with $(d_i - 1)$ indistiguishable entries for each i

(i.e. just the vertex label i)

Use division rule of combinatorics.

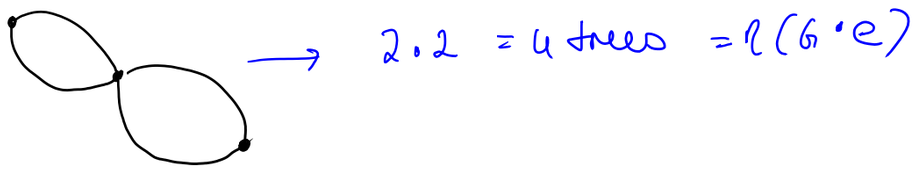
Note:

This counting stuff brings us pretty naturally to our main topic for this section - counting the number of spanning trees in a graph.
This is an *incredibly* important task for understanding properties of a graph.
Applications in circuit design, social network analysis, clique-finding, and so on

Each distinct tree on a specified vertex set S is a spanning tree in the complete graph on S
So we've solved this counting problem for complete graphs!

What about other graphs?

those that involve e

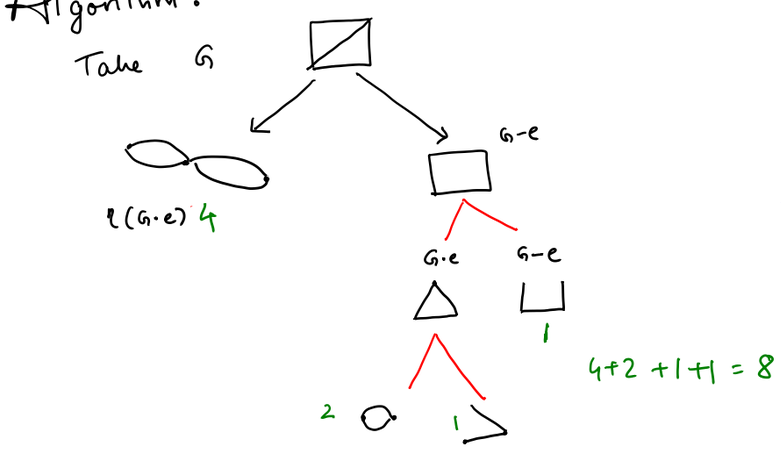


$\rightarrow$ 2·2 = 4 trees $= \ell(G \cdot e)$

$\rightarrow$ 4 trees (any 1 of the 4 edges in a cycle can be counted)

Algorithm:

Take G



So you can always produce a path (with multiple edges) at the end for which it is easy to count spanning trees.

Ex: For C_n, the # of spanning trees is n.

Definition:

In a graph G, the *contraction* of edge e with endpoints u, v is the graph denoted $G \cdot e$, with u, v amalgamated into a single vertex. For each edge other than e incident on u or v (with multiplicity) draw an edge incident on that ne

Note:

The contraction of a graph is NOT necessarily a subgraph!

Proposition:

Let $\tau(G)$ denote the number of spanning trees of the graph G. For $e \in E(G)$ not a loop,

$$\tau(G) = \tau(G - e) + \tau(G \cdot e)$$

Proof:

Spanning trees that omit e are spanning trees of $G - e$ ← no vertices are omitted.

We define a bijection from spanning trees of G including e to spanning trees of $G \cdot e$

G·e has one fewer edge so does T·e

For any spanning tree T of G including e, the contraction of e in T yields a spanning tree of $G \cdot e$ (it is spanning with right number of edges)

All other edges maintain their identity and labels under contraction, so no two distinct spanning trees of G can map to the same spanning tree of $G \cdot e$

This can be undone by expanding the contracted edge back out, so the described function is a bijection.

Note:

This turns the task of finding spanning trees for a graph with e edges into the task of finding all spanning trees of two graphs with $e - 1$ edges

Very inefficient algorithm → exponential in e

Can be improved slightly with the following remark.

Remark:

If G is a connected loopless graph with no cycle of length at least 3, then $\tau(G)$ is the product of the edge multiplicities. A disconnected graph has no spanning trees.

← but it allows for multiple edges

← because if you replace multiple edges by a single edge it is already a tree.

This algorithm is still terrible. Making this into a computationally tractable task requires some additional cleverness.

Definition:

Given a loopless graph G, the *Graph Laplacian* is the matrix $L = D - A$ with A the adjacency

matrix of G and D the diagonal matrix of vertex degrees.
(some people might term this the negative Laplacian, it depends on who you ask)

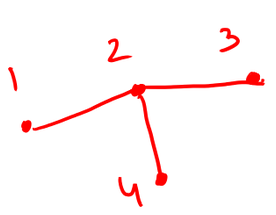
This matrix contains an *incredible* amount of information about a graph, and is one of the main conduits for linear algebra techniques into graph theory.

Note:

Why is this related to the second derivative?

$\delta f(x) = f(x+1) - f(x)$

$\delta^2 f(x) = \delta f(x) - \delta f(x-1) = f(x+1) - 2f(x) + f(x-1)$



$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} = A$$

$$D - A = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & & & \\ & 3 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} = D$$

Q: Show that D is an eval of the adjacency matrix.

Theorem: (Matrix Tree Theorem)

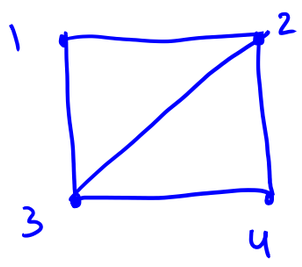
Given a loopless graph G with vertex set $v_1, \dots, v_n$. Let Q be the graph Laplacian of G. Then $\tau(G)$ is given by any entry in the cofactor matrix of G.

That is to say, if Q^* is the matrix formed from Q by deleting row s and column t, then

$$\tau(G) = (-1)^{s+t} \det Q^*$$

Let's do at least one example before we delve into the fairly complicated proof.

Do the graph on 4 vertices with only one edge missing (the kite). (degree sequence 3 3 2 2)
Should get 8 spanning trees.



$$D - A = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}$$

$$\begin{vmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 2 \end{vmatrix} = 3(5) + 1(-3) - (4)$$

$$= +15 - 3 - 4 = 8$$

Proof:

Lemma 1:

If M is a matrix such that the entries of M in each row sum to 0, then the cofactors of M are constant in each row.

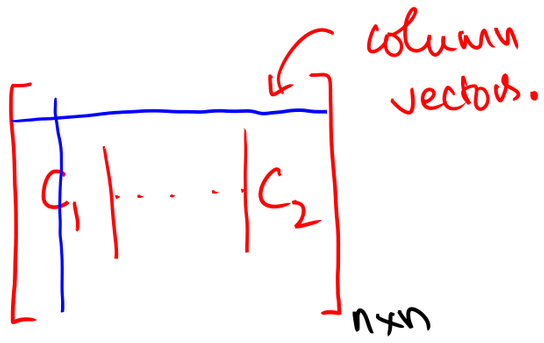
Its not it's obvious.

[this is a fairly technical little linear algebraic lemma, the strategy to prove it is to note that the cofactor matrix is related to the determinant of a matrix to the matrix inverse if it exists]
We use that lemma to argue that it suffices to consider $s = t$

$$A^{-1} = \begin{bmatrix} C_{11} & C_{12} & \cdots \end{bmatrix}^T \frac{1}{\det(A)}$$

$$A^{-1}A = I \Rightarrow \det(A)\,\delta_{ij} = \sum_k C_{ik}^T a_{kj} = \sum_k C_{ki}\, a_{kj}$$

cofactor



Delete column j

$$\begin{bmatrix} \hat{C}_1 & \cdots & \hat{C}_k \end{bmatrix} \xrightarrow{j} B_j \qquad \det(B_j)(-1)^j \text{ is a constant.}$$

Will show $\det(B_1) = (-1)\det(B_2)$

$$\det(B_2) = \left| \hat{C}_1\ \hat{C}_3 \cdots \hat{C}_k \right| = \left| -\sum_{j=2}^{k} C_j \quad \hat{C}_3 \cdots \hat{C}_k \right|$$

Multilinearity →

$$= -\sum_{j=2}^{k} \left| \hat{C}_j\ \hat{C}_3 \cdots \hat{C}_k \right| = -\left| \hat{C}_2 \cdots \hat{C}_k \right|$$

only $j=2$ term survives.

Lemma 2:

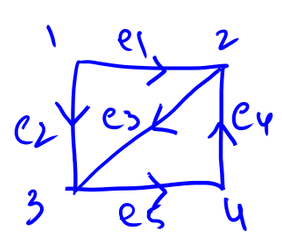
If D is an orientation of G and M the incidence matrix of D, then $Q = MM^T$

Proof:

(recall with edges $e_1, \ldots, e_m$ the incidence matrix consists of values m_{ij} which are 1 if v_i is the tail of e_j, -1 if its the head, and 0 else)

→ definition of M for digraph.

The entry at index i, j in MM^T is the dot product of rows i and j of M
If $i \neq j$, this product includes a -1 for each edge between v_i and v_j
If $i = j$, this includes a $+1$ for each incident edge in G



$$M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{pmatrix} -1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix} \end{array}$$

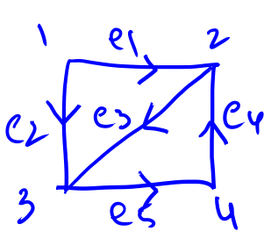
$(MM^T)_{23}$

MM^T = dot product of rows.

$(MM^T)_{ii} = d(i)$

Lemma 3:

If D is an orientation of G and M is the incidence matrix of D, let B be a $(n-1)$ by $(n-1)$ submatrix of M. Then $\det B = \pm 1$ if the corresponding $n-1$ edges form a spanning tree of G. Otherwise $\det B = 0$



$$M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{pmatrix} -1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix} \end{array}$$

$n = 4$

$$\begin{vmatrix} -1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & 1 \end{vmatrix} = -1$$

You can sort of see this

Proof:

In the case where the edges form a spanning tree, we proceed by induction on n.
If $n = 1$, we have a 0x0 submatrix which we'll definitionally take to have determinant 1.
If $n > 1$

Let T be the spanning tree. It has at least two leaves, such as x, at least one of which is represented in the matrix. (because it has n-1 entries)
The row corresponding to x has only one non-zero entry in B.
Compute determinant by expanding along this row.
Only relevant $(n-2)$ by $(n-2)$ matrix is a submatrix of the adjacency matrix for $D - x$
By inductive hypothesis, the determinant is ± 1 if $T - x$ is a spanning tree (which it is)

Again see the leaf pruning operation. amazing.

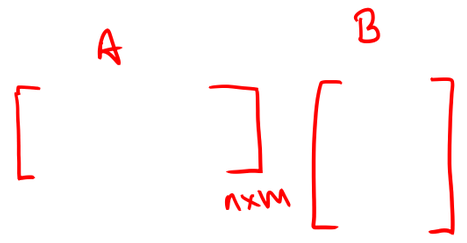
On the other hand, if edges corresponding to columns of B do not form a spanning tree, they contain a cycle C
Take a linear combination of columns c_i by
$a_1c_1 + a_2c_2 + \cdots a_{n-1}c_{n-1}$
with $a_i = 0$ if the corresponding edge is not in C, $+1$ if followed in the right direction by C, -1 if followed in reverse direction by C
Result must be 0 at each vertex, hence columns are linearly dependent, so $det = 0$

(We need one more tool from linear algebra)

Theorem: (Binet-Cauchy Formula)
For A an n by m matrix and B an m by n matrix with $m \geq n$, then

$$\det AB = \sum_S \det A_S \cdot \det B_S$$

where the sum runs over all subsets of size n in $[m]$, and A_S, B_S are the submatrices consisting of rows with indices in S



Now, we use these lemmas to determine $\det Q^*$

effectively deleting a vertex

Let M as in the above lemmas, and M^* the result of deleting row t of M
Note $Q^* = M^*(M^*)^T$ by Lemma 2
If $m < n - 1$, then there are no spanning trees and need at least n-1 edges.

M^* is $(n-1)\times(n-1)$

$Q^* = [\quad][\quad]$ ($n-1 \times m$)

$\det(Q^*) = \det(R_{n-1\times n-1}\, M^*(M^*)^T)$ (R: row operations)

$= \det(R)\det(M^*(M^*)^T) = \det\left(\begin{bmatrix}0000\end{bmatrix}(M^*)^T\right)$ (since rank $\leq m$)

$= \det(\text{all zeros row}) = 0.$

So assume $m \geq n-1$

Apply Binet-Cauchy to Q^* (can apply since $m \geq n-1$). In this case, the two matrices (run over S of size $n-1$ that appear in the sum) are transposes of one another, so have equal determinants.

Apply Lemma 3 to see that the sum reduces to a sum over edge sets which are spanning trees.

$$\det(Q^*) = \sum_{S,\,|S|=n-1} \det(M_S)\det((M^T)_S) = \sum_{S,\,|S|=n-1} \det(M_S)^2$$

Theorem: ± 1 iff S corresponds to a spanning tree!

Decompositions and Graceful Labelings

with $\neq$ edge

We can decompose any graph into a union of edges, that is to say, trees of size 2. When can we decompose a graph G into copies of a larger tree T?

Certainly we need $e(T)$ to divide $e(G)$, and certainly $\Delta(T) \leq \Delta(G)$

$\Delta(T) = e(T)+1$

↖ could be a small graph with lots of edges and very few vertices.

Conjecture: (Ringel 1964)
If T is a fixed tree with m edges, then K_{2m+1} decomposes into $2m+1$ copies of T

↗ $\frac{2m\cdot(2m+1)}{2}$ edges $\Rightarrow m \mid e(K_{2m+1})$ $\quad m+1 \leq v(K_{2m+1})$ $\quad \frac{e(K_{2m+1})}{e(T)} = 2m+1$

This conjecture is hard to approach directly, so work focuses on a stronger conjecture.

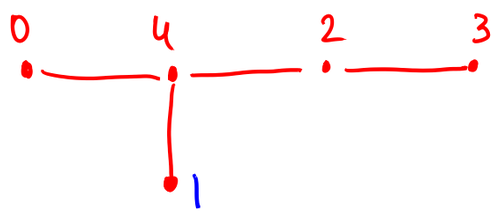
A *graceful labeling* of a graph G with m edges is a function (just put #s on labels)
$f: V(G) \to \{0, \dots, m\}$
such that distinct vertices are assigned distinct numbers and

$$\{|f(u) - f(v)| : uv \in E(G)\} = \{1, \dots, m\}$$

A graph that admits a graceful labeling is called *graceful*

(This property amounts to requiring that the difference in labels across any edge in the graph is distinct

↳ It sounds a lot like the Erdös distinct distances problem.

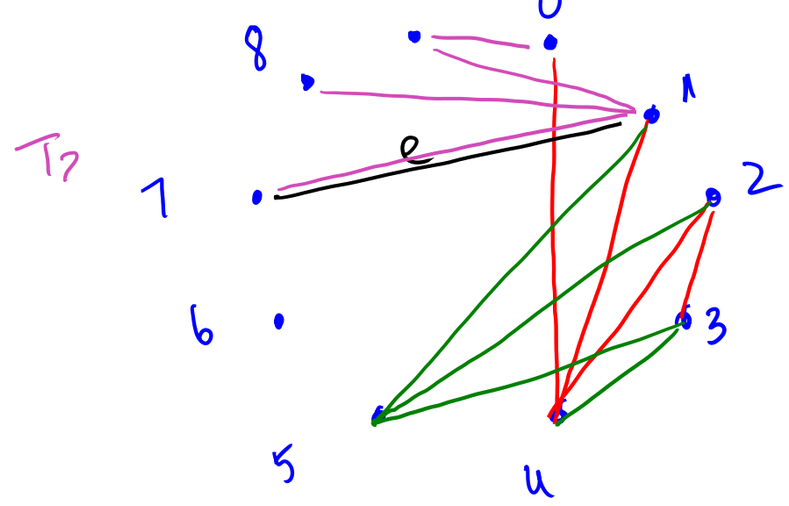
$T =$ 

distances $= \{4, 2, 1, 3\} = \{1, \dots, 4\}$

Conjecture: (Graceful Tree Conjecure - Kotzig, Ringel 1964)
Every tree has a graceful labeling.

While obviously we can't prove any open conjectures in class, we can prove that this conjecture would imply the previous one.

Note that $K_9 = K_{2\cdot 4+1}$ decomposes into 9 copies of T $(m=4)$



The difference class of an edge in K_{2m+1} is the # of intermediate edges. Take the black edge e between 7 and 1. It is in difference class 3

You can view the distance as either 5 or 3. Always we will have one of them being $\leq m$. $d_1 + d_2 + 2 = 2m+1 \Rightarrow d_1 + d_2 = 2m-1 \Rightarrow$ either d_1 or d_2 will be $\leq m$. It is clear that e appears in T_7.

This can be turned into Rosa's theorem.

Theorem: (Rosa 1967)

If a tree T with m edges has a graceful labeling, then K_{2m+1} admits a decompositioninto $2m + 1$ copies of T

Proof:

Consider vertices of K_{2m+1} as congruence classes modulo $2m + 1$

Define subgraphs $T_0, \dots, T_{2m}$ by

$V(T_i) = \{i, i+1, \dots, i+m\}$

$E(T_i)$ contains an edge between $i + r$ and $i + s$ iff the graceful labeling of T contains an edge between labels r and s

Every edge of K_{2m+1} is contained in some T_i because:

There exists vertices in the labeling of T with any specified difference, so just translate the values until they match

No edge mn of K_{2m+1} is contained in T_i and T_j for $i \neq j$ because:

then $m - i$ and $n - i$ are adjacent in T

and $m - j$ and $n - j$ are adjacent in T

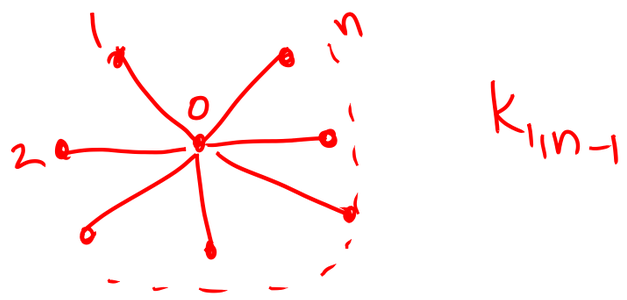
This violates the property of a graceful labeling, that all label differences occur only once.

↓

The same difference appears twice.

Note:

While it is hard to find graceful labelings for any arbitrary tree, there are a lot of trees where a strategy is known.

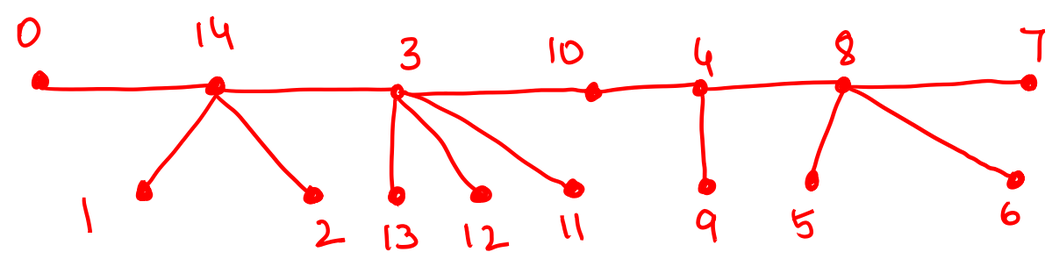


$K_{1,n-1}$

The next kind of graph generalizes paths and stars.

Definition:

A *caterpillar* is a tree in which a single path (the *spine*) is incident to (or contains) every edge

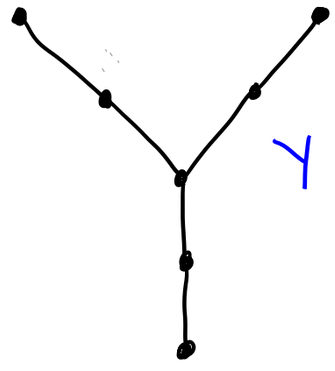


Pick an intermediate value x like 7.

Start with 0 → Then follow the rule: every edge must have a vertex above 7 and one below 7.

Theorem:

A tree is a caterpillar iff it does not contain the tree Y

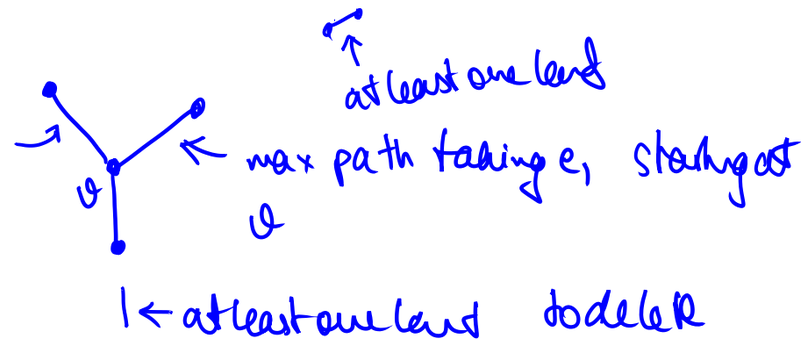


Proof:

Let T be a tree, and T' the tree with all leaves deleted.
T contains Y iff T' is has a vertex of degree 3 (thus not a path).

So if $\Delta(T') \leq 2$, and it's a tree

$\Rightarrow$ T' is a path.



Pretty amazing consequence: K_{2m+1} can be divided up into copies of arbitrary caterpillars!

Spanning Trees in Digraphs

We can talk about spanning trees in digraphs as well - there's a more general that applies to digraphs.

Definition:

A *branching* or *out-tree* is an orientation of a tree having a root of indegree 0 and all other vertices of indegree 1.
An *in-tree* is an out-tree with reversed orientation.

Theorem: (Directed Matrix Tree Theorem - Tutte 1948)

Given a loopless digraph G, let $Q^- = D^- - A^T$ and $Q^+ = D^+ - A^T$ with D^- (D^+) the diagonal matrix of indegrees (out-degrees) of vertices of G and

$$A = (a_{ij})$$

the matrix with a_{ij} = the number of edges from v_i to v_j
The number of spanning out-trees (in-trees) of G o ed at v_i is the value of any cofactor in the ith row of Q^- (ith column of Q^+)

In the textbook A is written as A^T, the transpose of the adjacency matrix.

We won't discuss a proof of this, though similar ideas are involved as in our proof for graphs.

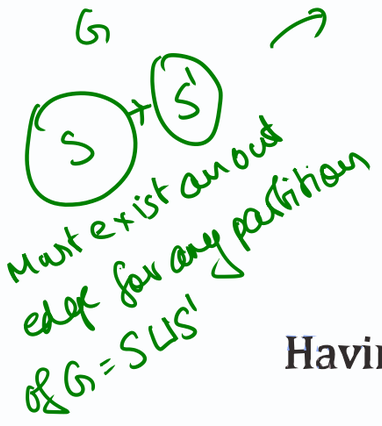
There are some nice results about our ability to produce search algorithms that are related to this theorem.

Lemma:

If G is a strong digraph, then every vertex is the root of an out-tree (and an in-tree)

Proof:

Fix a vertex, iteratively add edges to produce an out-tree on a growing set S, there must exist an edge leaving S by strong connectivity.



Having an in-tree is also really helpful for finding Eulerian circuits!

recall digraph Eulerian iff strongly connected and $d^+(v)=d^-(v)$

Algorithm: (Eulerian circuit in a digraph given a spanning in-tree)

Input: Eulerian digraph with no isolated vertices and a spanning in-tree T consisting of paths to a vertex v

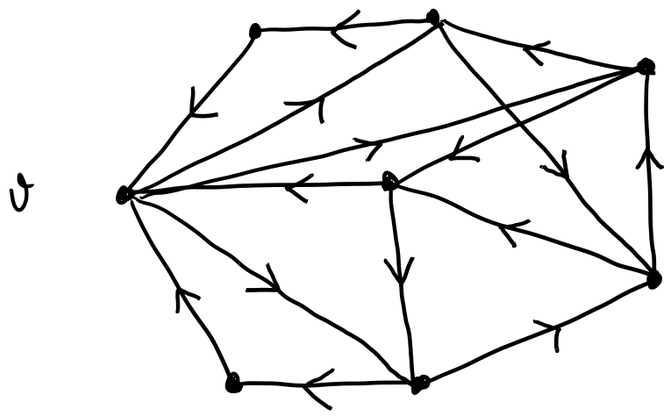
Output: An Eulerian circuit

First, for each vertex u, list the edges leaving u in an arbitrary order, with the one leaving it that is contained in T listed last (for v, all are arbitrary)

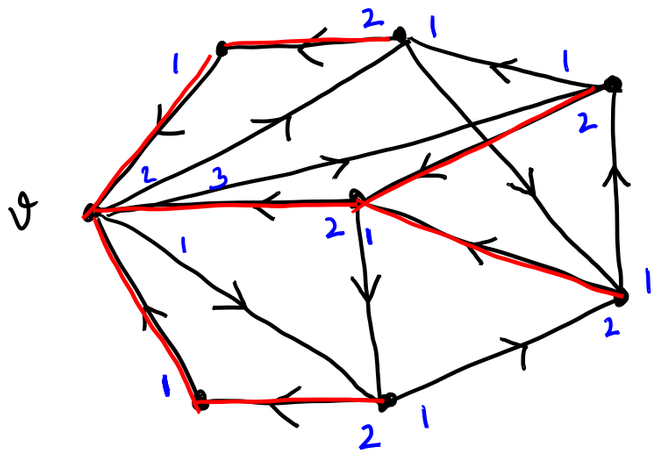
Begin at v. At each step, traverse the first not-already-traversed edge leaving the current vertex listed on that vertice's outward edge list.
Eventually you will end at v with no more edges to traverse.

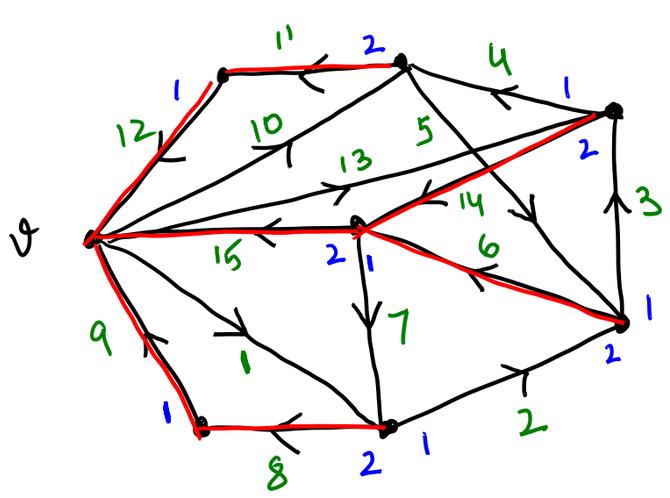
Claim: (I won't prove this)

The algorithm above always produces an Eulerian circuit.



Given a spanning in-tree





Leads to

Theorem (van Aardenne-Ehrenfest): In an Eulerian digraph the # of Eulerian circuits is (de Bruijn 1951)

$$\#\{\text{intrees to any vertex}\} \times \prod_{v \in G} (d(v)-1)!$$

Note $d^+(v) = d^-(v)$

Corresponds to # of ways to label out edges.

Pf: The above algorithm provides all Eulerian circuits.

Ex: can somebody provide a proof?

Section 2.3 - Optimization and Trees

In applications, the structure of a graph is often insufficient for purposes at hand.

Definition:

A *weighted graph* is a graph with numerical labels on edges.

Labeling edges and vertices in different ways is a great way to make the structure of a graph relevant to a problem at hand.

We often interpret such weights as distances, in which case we must require that they be *non-negative* (or sometimes strictly *positive*)

Optimization problems can be asked about various features of weighted graphs.

Task: Find the spanning tree of a connected weighted graph with minimum weight

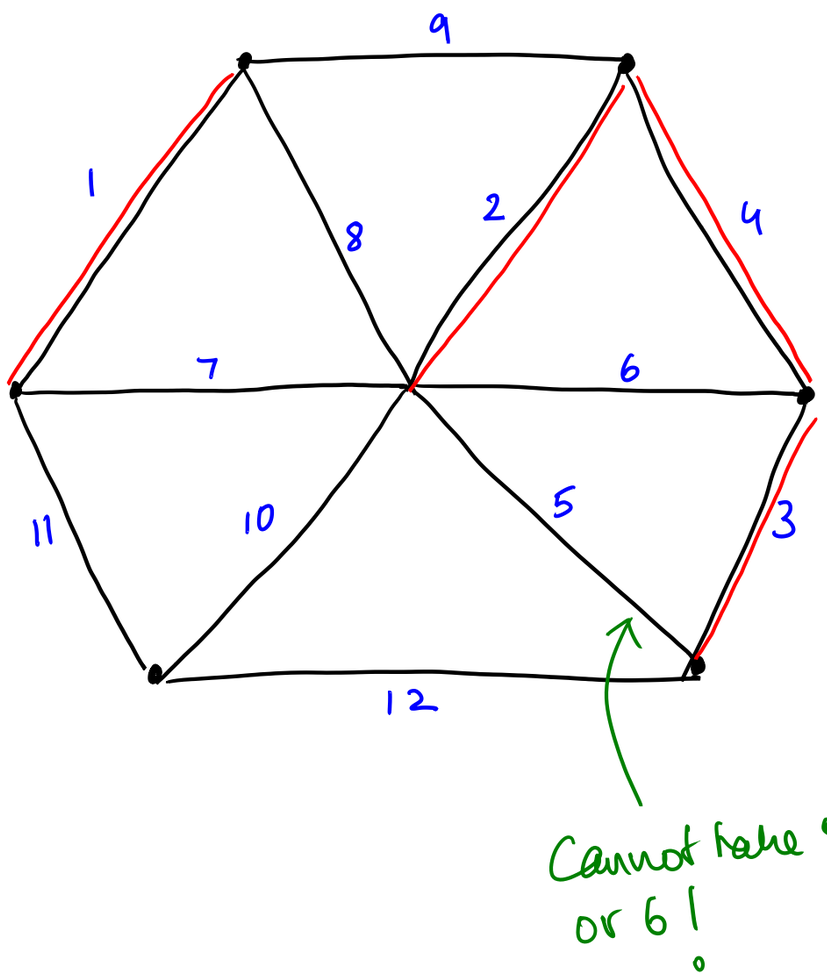
Algorithm: (Kruskal's Algorithm - for minimum spanning trees [a greedy algorithm])

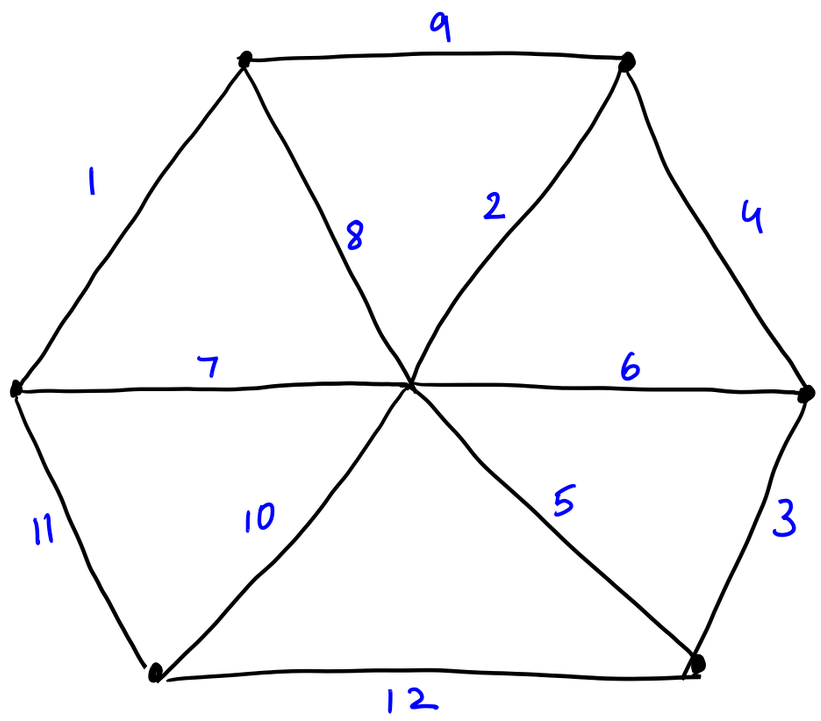
Input: a weighted connected graph G

A spanning subgraph has vertex set $V(G)$

Idea: Work with an acyclic spanning subgraph and expand it piecemeal with low weight edges.

Set H as the empty graph on our vertex set.
Loop:
Among edges of G between two distinct components of H, pick the one of lowest weight
Add it to H





This is a pretty naïve algorithm, but it happens to be optimal and really easy to implement, which is awesome

Theorem: (Kruskal 1956)

The above algorithm produces a minimum weight tree.

Proof: → If the final graph has more than one component, there must have been an edge connecting them so T is connected.

The algorithm does construct a tree, since G is connected and H is always acyclic.

Let T be the produced tree and T' a minimal tree. If $T \neq T'$ then there is some first edge e picked in the construction of T not present in T'

$T' + e$ contains a cycle with some e' not in T ← because if all e' in the cycle are in T

Consider $T' + e - e'$

At the step of the construction of T that picked e, e and e' were both available

Hence $w(e) \leq w(e')$

So $T' + e - e'$ has either equal (if so, repeat the argument - having a spanning tree which contains even more early vertices of T) or strictly less weight than T', a contradiction.

Task:

Find the shortest path between two points.

This is a particularly famous algorithm, built on the idea that if the shortest path from a to b goes through c, then necessarily it must be the case that the subpath from a to c is also shortest.

Definition:

In a weighted graph, the *distance* $d(a,b)$ between two points is the sum of the weights of the shortest path between two points.

Algorithm: (Dijkstra's Algorithm)

Input: A graph (or digraph) with non-negative edge weights and an initial vertex u

Weights $w(xy)$ for vertices xy with weight understood as ∞ if xy is not an edge in the graph

Idea: S will be the set of vertices whose smallest path is known

$t: V(G) \to R$ will be the shortest path from u to other vertices known so far

Set $S = \{u\}$, $t(u) = 0$, $t(z) = w(uz)$ for all $z \neq u$ (if not connected to u $w(u,z) = +\infty$)

Loop:

Select v outside S with $t(v)$ minimized, add v to S

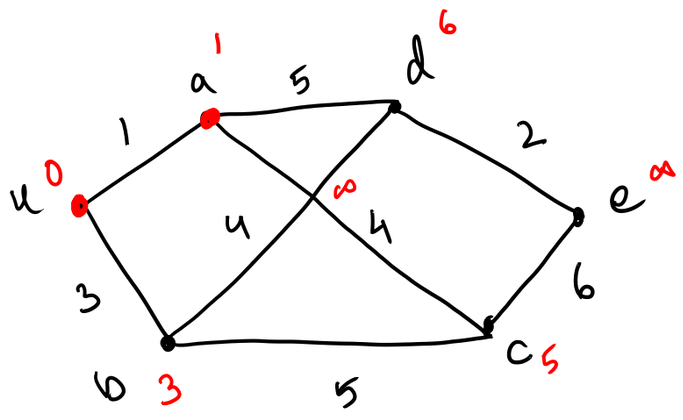
For each edge vz between v and a vertex $z \notin S$

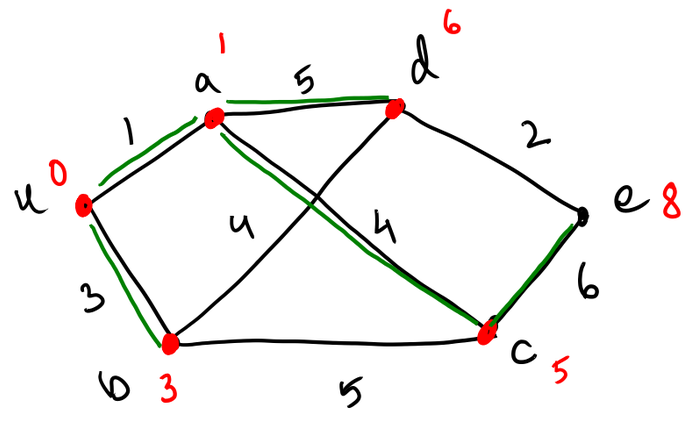
Update $t(z) = \min\{t(z), t(v) + w(vz)\}$

Stop if S contains all vertices or $t(z) = \infty$ for all $z \notin S$

Set $d(u,v) = t(v)$ for all v

(go through an example of this algorithm operating, just draw an arbitrary graph to do it)



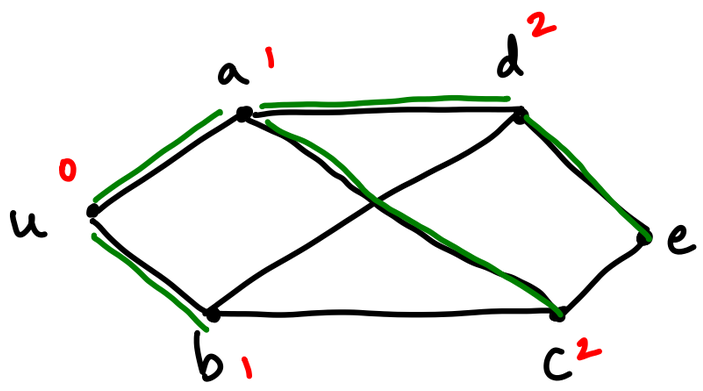


Dijkstra generates a tree. Only need to remember the green edges. : the vertex from which the final value was found. Following paths in the tree generates the geodesics.

BFS: All weights are $= 1$ on the graph.

R = FIFO queue of vertices to search.

S = vertices already searched.



$R = \{u\}$ $d(u,u) = 0$

$S = \emptyset$

$R = \{b\ a\}$ $S = \{u\}$ $\longrightarrow$ $R = \{c\ d\ b\}$ $S = \{ua\}$

$R = \{cd\}$ $S = \{uab\}$ $\longrightarrow$ $R = \{ec\}$ $S = \{uabd\}$

Theorem:

Given a graph or digraph G and a vertex $u \in V(G)$, Dijkstra's Algorithm computes $d(u, z)$ for every $z \in V(G)$

Proof:

We claim two things are true while the algorithm is operating

1) If $z \in S$, then $t(z) = d(u, z)$
2) If $z \notin S$, then $t(z)$ is the least length of a u, z path reaching z directly from S

← without going to vertices in $(S \cup \{z\})^c$

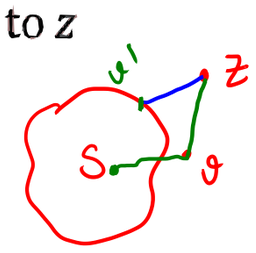
or in other words, through an edge from S.

We prove these by induction on the size of S.
Both immediately true when size is 1.
Inductive step:

Let v be the vertex not in S with smallest $t(v)$.
The algorithm will choose v
By inductive hypothesis, the shortest path to v from S is $t(v)$ by minimality of v

Then after updating t, $d(u, v) = t(v)$

For all other vertices $z \notin S$, we must now consider paths through S that go to v and then directly on to z



$t'(z) = \min\{t(z), t(v) + w(vz)\}$

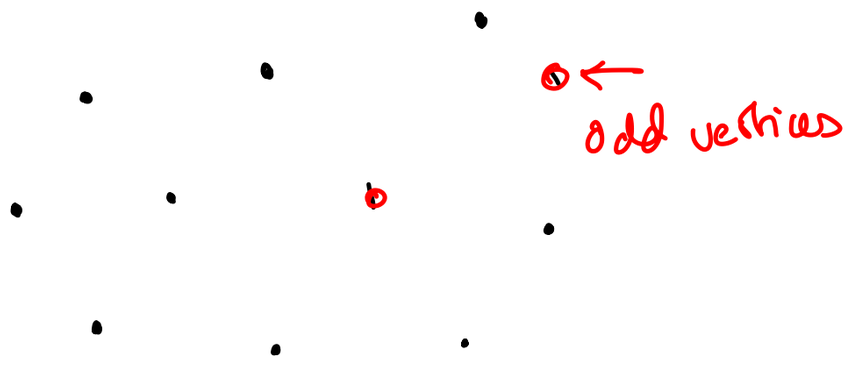
If the optimal path goes through v', by induction $t(z)$ is the shortest path from S directly to z. If it goes through v, then we know $t(v) = d(u,v)$

$\Rightarrow t'(z) = d(u,v) + w(uv)$

Note:

If we apply Dijkstra's algorithm to an unweighted graph, the result is known as a *Breadth-First Search* algorithm

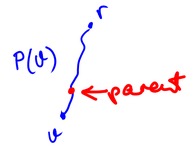
Guan's Postman Problem (Guan 1962)



If all vertices even degree then just find Eulerian circuit.

If odd degree vertices exist, recall that the # of trails is equal to the # of vertices of odd degree.

Rooted Trees and Their Uses



Definition:

A *rooted tree* is a tree with one vertex r chosen as *root*.
For each vertex v, let $P(v)$ be the unique path from r to v
The *parent* of v is its neighbor in $P(v)$
The *children* of v are all other neighbors
The *ancestors* are the vertices of $P(v) - v$
The *descendants* are vertices u such that $P(u)$ contains v
The *leaves* are vertices with no children

A *rooted plane tree* or *planted tree* is a rooted tree with a left-to-right ordering specified for the children of each vertex

(CS)

Though all such trees are important mathematically, in applications binary trees are often most significant.

Definition:

A *binary tree* is a rooted plane tree where each vertex has at most two children (denoted *left child* or *right child*)
The subtrees rooted at the children of the root are the *left subtree* and *right subtree*

A k-ary tree has at most k children from each vertex

Application - Data Compression:

Suppose I am sending you a message, built out of symbols from some alphabet S
To transmit it to you, I need to convert it into binary.

I want to send the message using as few bits as possible (compression).
But, I need you to be able to read the message, so whatever I do to it must be reversible

Strategy:

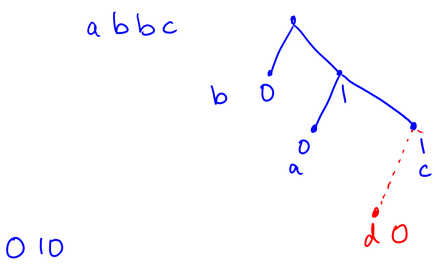
For each symbol in S, replace it with a binary string (its *code*).
If the symbol appears a lot in the message, make the code short. If the symbol appears infrequently, the code can be longer.
Send a table listing these conversions along with the message. (the message should be much bigger than the table for this to effective)

For this strategy to be coherent, the recipient needs to be able to break up a long binary string into segments each corresponding to individual symbols. Since the segments may be different lengths, this can be tricky.

Idea:

If no code for a symbol is the initial part of another symbol's code, we'll be able to undo the compression.
This gives us a *prefix-free* condition.

Equivalently, this lets us draw the codes as leaves of a rooted binary plane tree (draw an example)



a - 1	10
c - 1	11
b - 2	0
d	110

won't know when to stop reading.

OTHER encodings: ASCII. 8 bit (256 characters) a = 97

1	2	4	8	16	32	64	128	256
1	0	0	0	0	1	1	0	0

8 × 4 = 32 bits of data.

Generally $8k$. Here: table, tree, bit length, actual message.

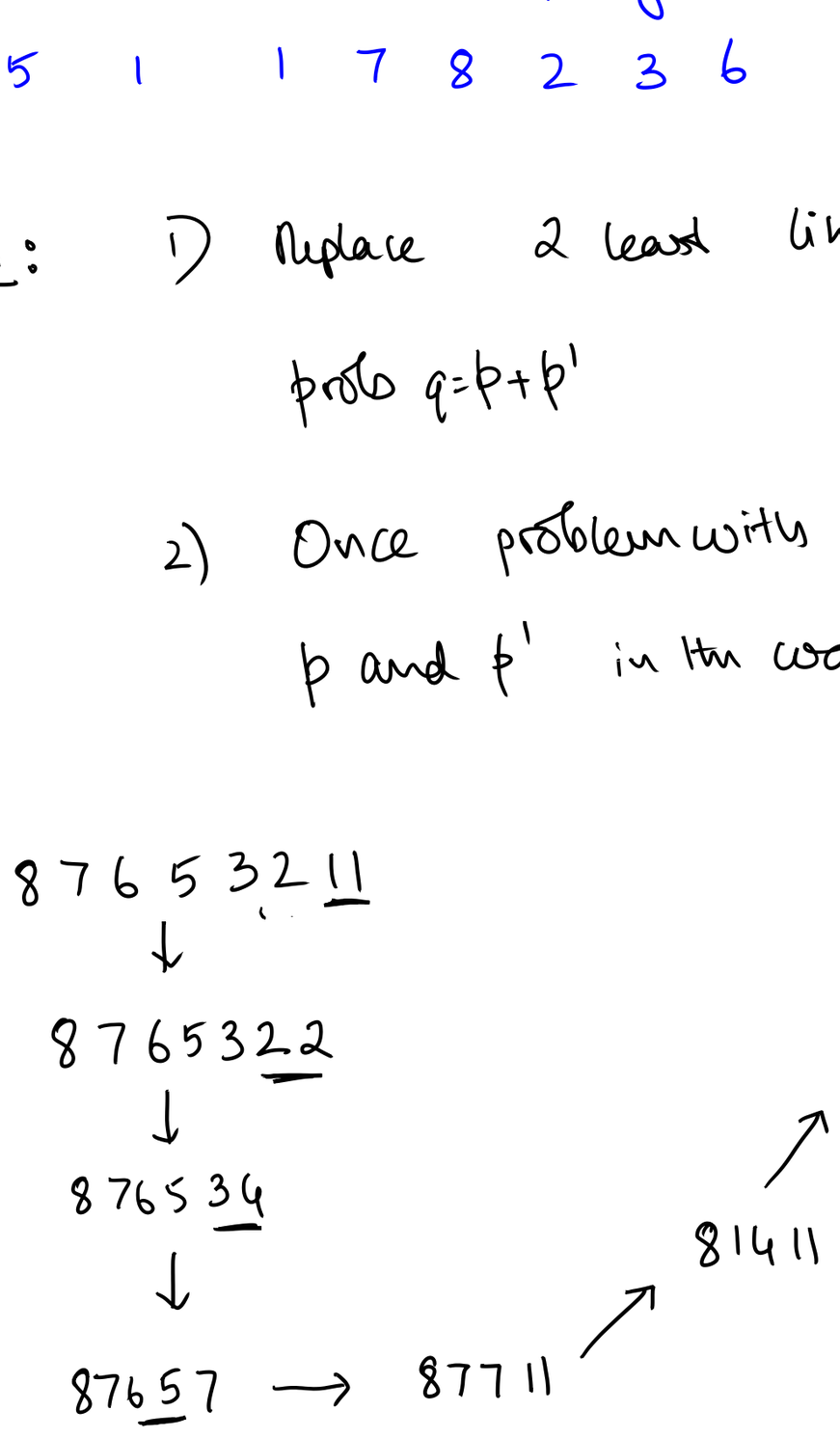
Here if code has k characters, with distribution $p_1 \cdots p_k$. Then avg code length:

$= \sum p_i c_i$

	a	b	c	d	e	f	g	h	
Frequency:	5	1	1	7	8	2	3	6	(= sum 33)

Algorithm: 1) Replace 2 least likely items with one combined item with prob $q = p + p'$

2) Once problem with $n-1$ items solved, give values 0 or 1 to p and p' in the code.



Algorithm: (Huffman Coding 1952)

Input: Weights (frequencies or probabilities) $p_1, \ldots, p_n$ of each symbol

Output: Prefix-free codes (in the form a binary tree with n leaves)

View each weight as a vertex.
If $n = 1$, we're done
If $n \geq 2$, pick the two smallest weights and make them both children of a single vertex.
Delete them from the list of weights and place their sum in the list of weights.

Theorem:

Given a probability distribution $\{p_i\}$ on n items, Huffman's algorithm produces the prefix-free code with minimum expected length.

Proof: (maybe don't prove this?)

Note:

It is not always the case that prefix-free codes give the best possible compression of a message. Sometimes, different compression schemes are better.
There's some really fascinating mathematics involved in this subject.
One can look to Shannon 1948 for the really interesting result

Theorem:

Given a probability distribution $\{p_i\}$ on n items, any code for those items has expected average length at least

$$H(p) = -\sum p_i \log_2 p_i$$

This value is called the *entropy*.

Section 3.1 - Matchings and Covers

In this section, we'll be translating a couple of common types of questions into graph theoretic terms. This will lead us to some interesting properties of graphs.

Definition:

A *matching* M in a graph G is a set of non-loop edges with no shared endpoints. The vertices incident to the edges of a matching are said to be *saturated* by M. Other vertices are *unsaturated*. A *perfect matching* is a matching for which all vertices are saturated.

Question:

What can we say about the incidence matrix of a matching?

→ each row has only one 1.

Question:

How many perfect matching are there in a complete graph on $2n$ vertices?

$(2n-1)(2n-3)\cdots(3)(1)$

Let f_n = # of perfect matchings of $\{1,\dots,n\}$.

For 1, there are $2n-1$ choices of partner. For each such there are f_{n-1} on the remaining set $\Rightarrow$ $f_n = (2n-1)f_{n-2}$

2nd way: Order $\underset{\text{pair}}{\bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet}$ $2n!$

But ordering of pair doesn't matter. Order in which pair appears doesn't matter $\Rightarrow$ $\dfrac{2n!}{2^n n!}$

Note:

Not all graphs have perfect matchings!

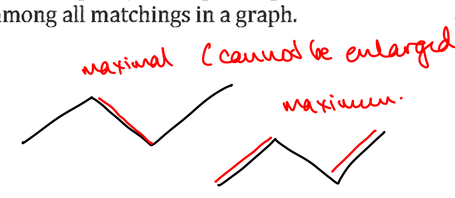
In general, we'll be interested in asking how large of a matching we can find for a given graph

Definition:

A *maximal matching* in a graph is a matching that cannot be enlarged by adding an edge.
A *maximum matching* is a matching with the largest size among all matchings in a graph.

Note:

A maximum matching is maximal.
A maximal matching need not be a maximum.



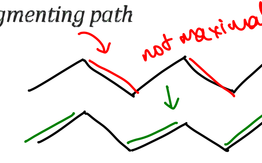
This example illustrates a bit of how we may enlarge matchings.

Definition:

An *M-alternating path* for a matching M is a path that alternates between edges in M and edges not in M
An M-alternating path whose endpoints are unsatured by M is an *M-augmenting path*

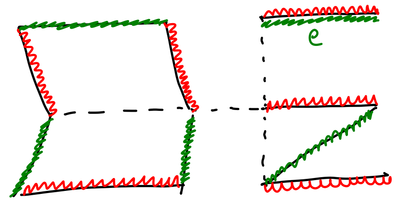
Claim:

If we have an M-augmenting path, then M is not maximal.



Definition:

Given graphs G and H, the symmetric difference is the subgraph of $G \cup H$ with edge set $E(G)\Delta E(H)$



$M \cap M' = e$

$M \Delta M' =$ alternating cycles
alternating paths!

Lemma:

Every component of the symmetric difference of two matchings is a path or an even cycle.

Proof:

F can have at most one edge from M and one from M' (since they're matchings)

Let $F = M \Delta M'$. No vertex in F has degree larger than 2, so F is a disjoint union of paths and cycles.

Cycles must alternate between elements of the two matchings, so even length.

Theorem: (Berge 1957)

A matching M in a graph G is a maximum matching in G if and only if G has no M-augmenting path.

Proof:

We already know one direction. → if it has an M augmenting path then it not maximal.

Suppose M is not a maximum matching, so M' is a strictly larger matching. Consider $F = M \Delta M'$

Each cycle in F has the same number of elements from each matching, so there must be a path. This is M-augmenting.

↳ if $|M'| > |M|$

Broad Setting:

Suppose you are running a hiring committee to hire several new staff members in an organization. X is the set of open jobs. Y is the set of applicants. Each applicant is well-suited to some subset of the jobs. As the hirer, you'd like each job opening to be filled by a single, distinct, suitable applicant.

The setup indicates a "suitability" graph which is X, Y-bipartite and indicates that we wish to find a matching that saturates X

(Alas, as the hirer, we don't really care whether or not it saturates Y - some candidates may need to apply elsewhere.)

Note:

In this setting, any such matching is necessarily a maximum.

Theorem: (Hall's Theorem - P. Hall 1935)

An X, Y-bigraph G has a matching that saturates X if and only if $|N(S)| \geq |S|$ for all $S \subset X$ (*Hall's Condition*)

Proof:

This is clearly necessary, for any such matching must only touch neighbors of elements of X

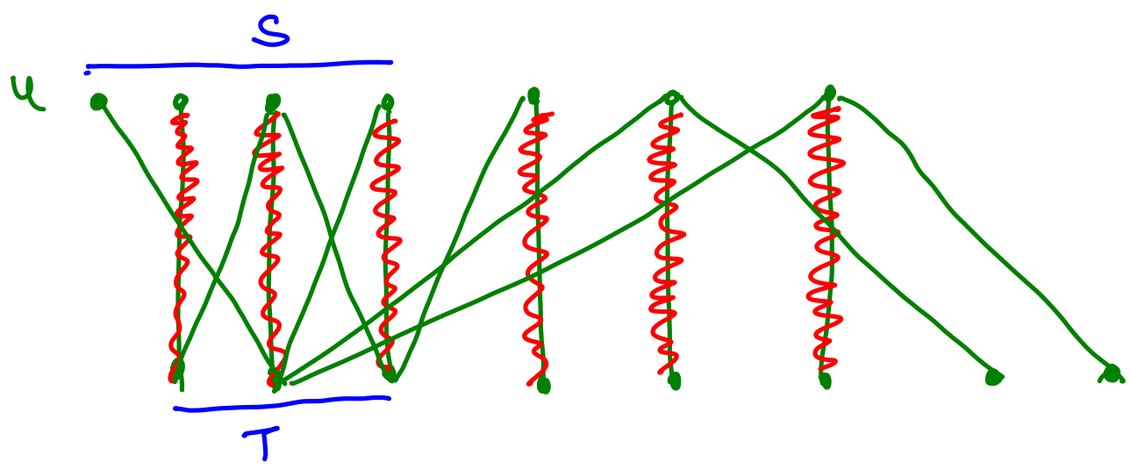
Let's suppose that M is a maximum matching in G and M does not saturate X

Want to show that there is a set S such that $|N(S)| < |S|$

Fix $u \in X$ unsaturated by M.

Let Z be the set of vertices reachable from u by M-alternating paths.

$S = Z \cap X$ $T = Z \cap Y$



u not adjacent to any M edge

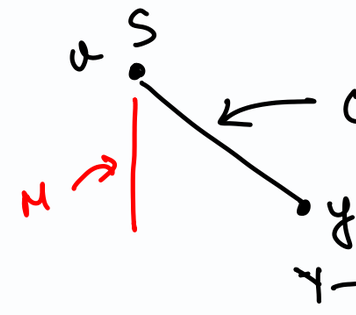
paths from u go to X along M^c edges and return to X along M edges.

$S - \{u\}$ is reached from T via an M edge. Every vertex of T must be saturated, otherwise there is an M augmenting path.

This gives a bijection from T to $S - u$ (is one-one since M is matching, and onto since $S - \{u\}$ must be reached via T).

$\Rightarrow$ $T \subset N(S)$.

In fact $T = N(S)$ since if $y \in N(S)$ and $y \in Y - T$



can't be in M since every v already matched to T. $\Rightarrow$ M alternating path so y cannot be in $Y - T$.

But $|T| = |S - \{u\}| = |S| - 1 < |S|$

$|T| = |N(S)|$

Corollary:

For $k > 0$, every k-regular bipartite graph has a perfect matching.

Proof:

If G is an X, Y-bigraph, regularity implies $|X| = |Y|$. Thus any matching satisfying Hall's condition is a perfect matching.

Take an $S \subset X$

Edges from S to $N(S)$ = $k|S|$

Edges from $N(S)$ = $k|N(S)|$

$\Rightarrow k|S| \leq k|N(S)|$

Note:

This is a pretty special case where we know about perfect matchings. Can we get at maximum matchings more easily?

Getting a lower bound on the size of a maximum matching is easy - just find a matching.
Getting an upper bound is quite hard!
Likewise, if I give you a matching, you likely don't want to find out if it is maximal by looking for an M-augmenting path.
We need a better way of reasoning about such things.

Definition:

A *vertex cover* of a graph G is a set $Q \subseteq V(G)$ that contains at least one endpoint of every edge.
Vertices in Q are said to *cover* $E(G)$

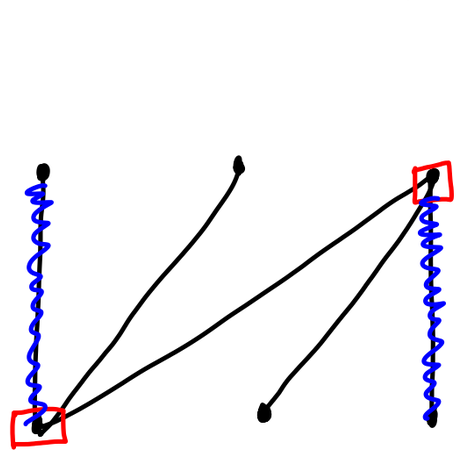
Idea:

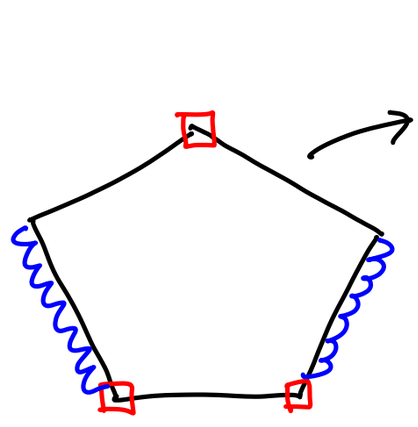
Suppose I give you a vertex cover of a graph and a matching of a graph.

Each edge in the matching has two distinct endpoints, at least one of which is in the vertex cover.

Hence, the cardinality of the vertex cover is at least that of the matching

-- *every vertex cover is at least as large as every matching of a graph*





minimal |vertex cover| > |maximal matching|

difference can be very large.

Theorem: (Koenig 1931, Egervary 1931)

If G is bipartite, then the maximum size of a matching in G equals the minimum size of a vertex cover of G

Proof:

G an X, Y-bigraph

Let Q be a minimum size vertex cover of G. We already know every matching is at most the size of Q.

To construct a matching with size Q. Going to use Hall.

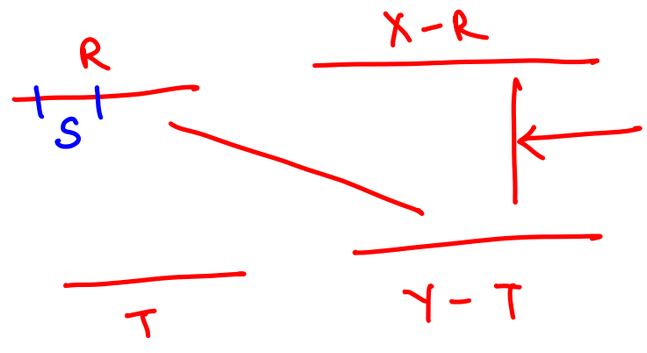
Let $R = Q \cap X$ and $T = Q \cap Y$

Consider subgraphs induced by $R \cup (Y - T)$ and $T \cup (X - R)$, call them H and H'

Claim:

H has a matching that saturates R into $Y - T$ and H' a matching that saturates T into $X - R$

If we have that, since $R \cup T = Q$, we have a matching of size $|Q|$



edge not possible, since Q is a cover

$N_H(S) \subset Y - T$ (by defn of H)

If $|N_H(S)| < |S|$ then we can replace S by $N_H(S)$ since the edges from S to T are already covered by T. This would give a smaller cover.

Hall's condition $\Leftarrow$

If a matching that saturates R in Y−T and one that saturates T into X−R $\Rightarrow |M| = |R| + |T|$!

$\beta_v(G)$ = size of minimal vertex cover

$\alpha_M(G)$ = size of maximal matching

$\alpha_M = \beta_v$

Note:

Observe that for bipartite graphs, this turns the problem of finding a maximum matching into an equivalent question about minimizing vertex covers. Finding a lower bound of size is easy (just find a matching). Finding an upper bound of size is easy (just find a vertex cover). This makes it a *great* optimization problem.

This kind of setup in an optimization problem is called a *min-max relation.*

In some generality, we may have a maximazation problem **M** and a minimization problem **N** on the same class of objects (like graphs) such that for every candidate solution M to **M** and every candidate solution N to **N** the value of M is at most that of N

Such problems are often called *dual optimization problems*

If we have dual problems, finding candidate solutions of equal size guarantees that both are optimal, and gives a min-max relation.

Definition:

The *independence number* of a graph is the maximum size of an independent set of vertices.

no 2 are adjacent.

Definition:

An *edge cover* of G is a set L of edges such that every vertex of G is incident to some edge of L
Vertices of G are said to be *covered* by the edges of L

Edge matching cover | vertex cover indepset

Note:

A perfect matching is an edge cover with $n(G)/2$ edges
Any matching is an edge cover of the subgraph of saturated vertices

This problem is ultimately quite related to the previous, so we'll introduce some terminology for convenience.

Definition:

The maximum size of an independent set is $\alpha(G)$ $\max_I$
The maximum size of a matching is $\alpha'(G)$ $\max_M$
The minimum size of a vertex cover is $\beta(G)$ $\min_V$
The minimum size of an edge cover is $\beta'(G)$ $\min_E$

These quantities tend to be fairly interrelated.

$\max_M = \min_V$

Note:

Koenig-Egervary theorem says $\alpha'(G) = \beta(G)$ for bipartite graphs.

We'll want to show that $\alpha(G) = \beta'(G)$ for bipartite graphs without isolated vertices

$\max_I = \min_E$

Notation:

For a subset of $V(G)$, we'll use an overline to indicate the complement within $V(G)$

Lemma:

In a graph G, $S \subseteq V(G)$ is an independent set if and only if $\overline{S}$ is a vertex cover.
In particular, $\alpha(G) + \beta(G) = n(G)$

$\max_I + \min_V = n$ (since $|S| + |\overline{S}| = n$)

Proof:

Let S be an independent set. Then every edge has at least one-endpoint on a vertex in $\overline{S}$. Then $\overline{S}$ is a vertex cover.

If $\overline{S}$ is a vertex cover it is incident to every edge $\Rightarrow$ no 2 vertices in S can have an edge.

Taking a maximum independent set gives the desired result.

Theorem: (Gallai 1959)

If G is a graph without isolated vertices, then $\alpha'(G) + \beta'(G) = n(G)$ $\max_I + \min_E = n$

Proof:

Let M be a maximum matching. For each unsaturated vertex, add one edge to M to obtain an edge cover L
The number of vertices covered by L is 2 for each edge in M, and 1 for each edge not in M, so
$2|M| + (|L| - |M|) = n$
Thus, $|L| = n - |M|$
Hence, $\beta'(G) \leq n(G) - \alpha'(G)$ ← obvious $\min_E \leq n - \max_M$

Let L be a minimum edge cover. For any edge $e \in L$, if both endpoints are incident on other edges in L then $e \notin L$ by minimality

Hence, any component formed by edges in L has radius 1 (only one vertex can have degree bigger than 1), so is a star
Let k be the number of components. Each non-central vertex in a star is a leaf, so $|L| = n - k$
Choose one edge from each component to form a matching M with $|M| = n - |L|$
Hence $\alpha'(G) \geq n(G) - \beta'(G)$

form a path between the degree 2 vertices and we will form

constructive

subtract out center vertices

$\max_M \geq n - \min_E$

$n(C_i) - 1$ edges in each component.

Corollary: (Koenig 1916)

If G is a bipartite graph with no isolated vertices, then $\alpha(G) = \beta'(G)$

Proof:

Use the previous two results and the Koenig-Egervary relation

no isolated vertices $\Rightarrow$

$\max_I + \min_V = n$
$\max_M + \min_E = n$

bipartite $\Rightarrow \max_M = \min V$

$\Rightarrow \max_I = \min_E$

We use no isolated vertices here: every component has at least one edge.

Section 3.2 - Algorithms and Applications

Problem:

How do we actually find a maximum matching in a bipartite graph?

Idea: Look for augmenting paths

Algorithm: (Augmenting Path Algorithm)

Input: An X, Y-bigraph G, a matching M in G, the set U of M-unsaturated vertices in X

Idea: Explore all M-alternating paths from U. For each vertex we reach, record how we got there.

$S \subseteq X$ and $T \subseteq Y$ the explored sets

First, set $S = U$ and $T = \emptyset$

Loop:

If all vertices in S are *marked*, stop. $T \cup (X \setminus S)$ is a minimum cover, M is a maximum matching

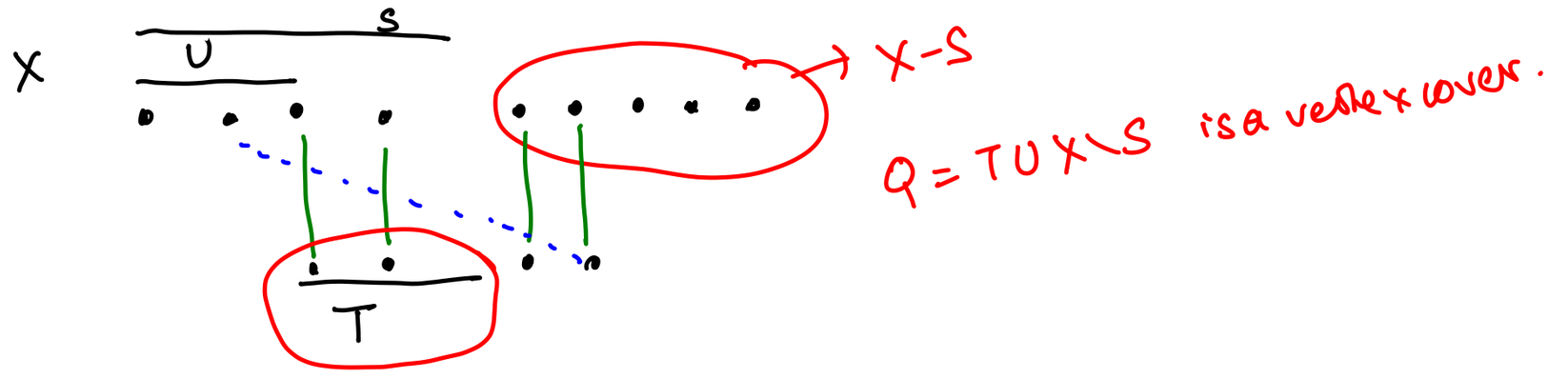
Select an unmarked $x \in S$. Consider all $y \in N(x)$ with $xy \notin M$

If y is unsaturated, we have an augmenting path.

If y is saturated, there exists $w \in X$ such that $wy \in M$

Add y to T and w to S

After all such y are explored, *mark* x



Theorem:

Repeatedly applying the Augmenting Path algorithm to a bipartite graph produces a matching and a vertex cover of equal size.

Proof:

If algorithm produces an augmenting path, we can use it to produce a strictly larger matching. This can only happen finitely many times.

If algorithm outputs a set $Q = T \cup (X \setminus S)$ and a matching M, we need to show that the former is a vertex cover, the latter is a matching, and they have equal size. M is certainly a matching.

To show vertex cover, we need to argue that no edge connects $Y \setminus T$ and S

If one did, it would connect some $x \in S$ to some $y \in Y \setminus T$. If y is unsaturated, augmenting path, if saturated the algorithm should have explored it before marking x

S∪U is explored from T via M ⇒ every vertex in T is matched to S-U. There can be no M from S to Y-T: They cannot be connected to U. And as argued before S-U is already matched.

Also by construction, when exploring $x \in S$, if ∃ an unmarked edge from x to $y \in Y-T$, then y would have been added to T!

Every X-U must be saturated by definition. They cannot be connected to T. ⇒ $|M| = |T| + |X-S|$

Also no edges from S to Y-T ⇒ $|M| = |T| + |X-S|$.

Note:

Define big O notation

Definition:

The *running time* of an algorithm is the max number of computational steps used expressed as a function of the size of the input.

- max over all inputs of that size

Definition:

A *good algorithm* is one with polynomial running time.

Remark:

If G is an X, Y-bigraph with $n(G) = n$ and $e(G) = m$, then since $\alpha'(G) \leq \frac{n}{2}$, we need only run the algorithm above at most $n/2$ times.

Each time we run the algorithm, it examines each vertex at most once, then marks it. Thus we traverse each edge at most once.

That means the number of edge explorations we have to do to find a max matching is

$O(nm)$

There are better algorithms.

Problem:

What if our graph is weighted, and we're interested in finding a matching of maximum total weight?

(It suffices to consider $K_{n,n}$ by adding vertices and edges of weight 0, and it suffices to consider non-negative weights since we may simply set negative weights to 0 then solve the problem then delete edges with 0 weight to solve the original.)

On $K_{n,n}$ with non-neg weights, some maximum weighted matching is perfect, so we need only find a maximum perfect matching.

Much as with the standard maximum matching problem, we can dualize this one, which will help.

Side Note:

If we can solve this problem, we can also solve the problem of finding a perfect matching of minimum weight. To do so, just pick M really really large, and compute M minus the weights, then maximize.

Example:

A farmer owns n farms (X) and n processing plants (Y)

Each plant is capable of processing the amount of crops grown on one farm.

The profit from sending the crops from farm x_i to plant y_j is w_{ij}

We have an X, Y-bigraph with weights w_{ij}

Maximizing profit is a weighted maximum matching problem.

The government thinks too much corn is being produced, so offers payments to farmers in exchange for not growing and processing corn.

Government will pay u_i in exchange for farmer not using farm x_i and will pay v_j in exchange for farmer not using plant y_j

If $u_i + v_j < w_{ij}$, then the farmer is incentivized to use the edge to make more money

If $u_i + v_j > w_{ij}$, then the farmer is incentivized to take the government payout.

The government thus wants to make sure that $u_i + v_j \geq w_{ij}$ for all i, j

But, to do so for the least cost, the government wants to minimize $\sum u_i + \sum v_j$

Definition:

A *transversal* of an n-by-n matrix consists of n positions in the matrix, one in each row and

each column
Finding a transversal with maximum sum is the *Assignment Problem*
This is just a matrix way of writing the maximum weighted matching problem for non-negative weights
Maximizing the total weight $w(M)$ is the goal

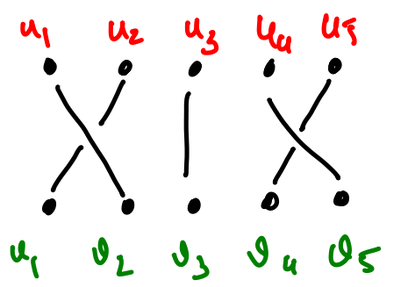
A *weighted cover* is a choice of labels $u_1, \dots, u_n$ and $v_1, \dots v_n$ such that $u_i + v_j \geq w_{ij}$ for all i, j.
The *cost* $c(u, v)$ of a cover is $\sum u_i + \sum v_j$
The *minimum weighted cover* problem is finding a cover with minimum cost.

Lemma: (Duality of weighted matching and weighted cover problems)
For a perfect matching M and weighted cover (u, v) in a weighted bipartite graph G
$c(u, v) \geq w(M)$
Moreover, $c(u, v) = w(M)$ if and only if M consists of edges $x_i y_j$ such that $u_i + v_j = w_{ij}$
In this case, both M and (u, v) are optimal

Proof:
M saturates all vertices, so for edge each $x_i y_j \in M$, we have $u_i + v_j \geq w_{ij}$.
vertices. Every vertex is accounted for

Equality is only obtained if each of these inequalities is an equality.

Each edge accounts for two



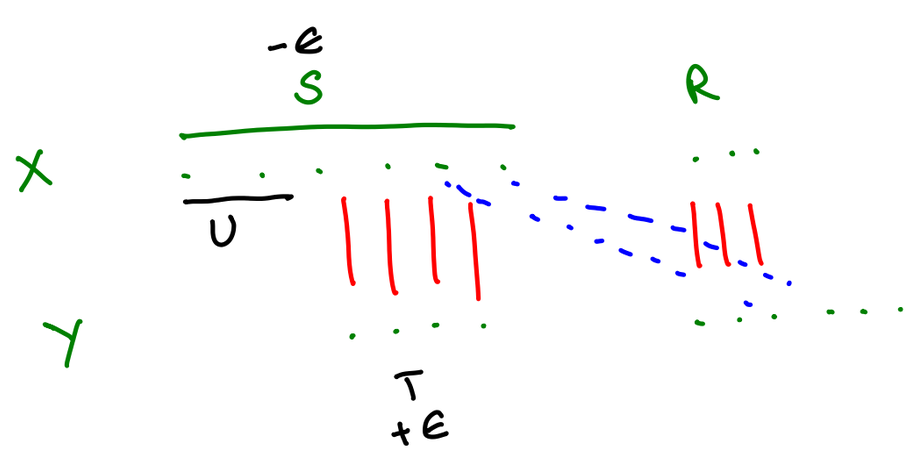
Fix perfect matching. Then all covers $\geq \sum_{ij \in M} w_{ij}$
s.t. $u_i + v_j = w_{ij}$ $ij \in M$ $u_i' + v_j'$
by choosing the edges of the matching.

Definition:
The *equality subgraph* $G_{u,v}$ for a weighted cover (u, v) is the spanning subgraph of $K_{n,n}$ whose edges are the pairs $x_i y_j$ such that $u_i + v_j = w_{i,j}$
In the cover, the *excess* for i, j is $u_i + v_j - w_{i,j}$

Idea:
We want to find a cover (u, v) such that $G_{u,v}$ has a perfect matching. If $G_{u,v}$ has a perfect matching, then so does $K_{n,n}$. The weight of this matching is $\sum u_i + \sum v_j$ and it must thus be optimal

Aug path
If $G_{u,v}$ has no perfect matching, find a maximum matching M in $G_{u,v}$ and a minimum vertex cover Q
Set $R = Q \cap X$ and $T = Q \cap Y$
Matching has $|R|$ edges from R to $Y - T$ and $|T|$ edges from T to $X - R$

Change (u, v) to preserve weight equality on all edges in M, but to cause zero excess on an edge from $Y - T$ to $X - R$
We now have a strictly larger maximum on this new $G_{u,v}$
To do this change:
Call ϵ the min excess of all edges from $Y - T$ to $X - R$
For all $x_i \in X - R$, reduce u_i by ϵ
For all $y_j \in T$, increase v_j by ϵ



For at least one blue edge $u_i + v_j - \epsilon = w_{ij}$, so the equality subgraph expands.

Algorithm: (Hungarian Algorithm - Kuhn 1955 and Munkres 1957)
Input: A matrix of weights on the edges of $K_{n,n}$ with bipartition X, Y

Can skip.

Let (u, v) be a cover with $G_{u,v}$ spanning (one can take $u_i = \max_j w_{ij}$ and $v_j = 0$)
Loop:
Find a maximum matching M in $G_{u,v}$
If M is perfect, stop. M is a maximum weight matching

Otherwise, let Q be a minimum vertex cover in $G_{u,v}$, let $R = Q \cap X$ and $T = Q \cap Y$
Let $\epsilon = \min\{u_i + v_j - w_{ij} : x_i \in X - R, y_j \in Y - T\}$
Decrease u_i by ϵ for all $x_i \in X - R$. Increase v_j by ϵ for $y_j \in T$
Use these new values as a new cover that has less cost.

(You can think of this entire algorithm using matrices, by writing the matrix of excesses. The equality subgraph corresponds to entries in the matrix which are equal to 0.)

(Consider doing an example, just write out an arbitrary 5x5 or 4x4 matrix of non-negative weights, it should work out.)

Theorem:

The Hungarian Algorithm finds a maximum weight matching and a minimum cost cover.

Proof:

If the algorithm terminates, we're done.

Denote (u, v) the current cover and suppose we have no perfect matching in $G_{u,v}$ yet. Denote (u', v') the modified cover. Necessarily $\epsilon > 0$, so $(u, v) \neq (u', v')$
For any edges between R and $Y - T$, no excess has changed.
Likewise for edges between $X - R$ and T
For edges between $X - R$ and $Y - T$, $u'_i + v'_j = u_i + v_j - \epsilon$, so by the choice of ϵ the weight is still covered
For edges between R and T $u'_i + v'_j = u_i + v_j + \epsilon$, so the weight is still covered
Thus (u', v') is still a cover.

We need only argue now that the algorithm terminates in finite time. (This is actually fairly tricky)

Easier Case with a more Straightforward Argument:

If I can assume the weights are rational, then WLOG I can assume they're integers by clearing denominators.
Then $\epsilon \geq 1$ in every step, and the cost of the cover is reduced in every step by an integer amount.

So the matching edges are unchanged. Moreover the cost of the cover is $\sum u_i' + \sum v_j' = \sum_{i \in X-R} u_i - \epsilon|X-R| + \sum_{i \in R} u_i + \sum_{i \in T} u_i + \epsilon|T| + \sum_{i \in Y-T} v_i = C(u,v) - \epsilon(|X-R| - T)$

Since T matches into $X-R$ $|T| \leq |X-R|$

So if the difference decreases by at least 1 we're good!

The cost started out finite, and is bounded below by the weight of a matching.
Thus, the algorithm terminates after finitely many steps.

Worth skipping. Is therein the exercises

Harder Case:

If the weights are real numbers, we need to work harder.
The issue is that the matching doesn't have to *strictly* increase in size every iteration of the loop
Instead, what we have to show is that if we keep running the loop, it *eventually* increases

Claim:

If we run the loop of the Hungarian algorithm n times, the matching will increase in size at least once.

Proof of Claim:

Let M be a maximum matching at some step with corresponding minimum vertex cover Q

The augmenting path algorithm gives us this cover by exploring M-alternating paths from U the set of unsaturated vertices

Let S denote the reachable set in X and T the reachable set in Y
The vertex cover was $R \cup T$ with $R = X \setminus S$

If we apply an iteration of the Hungarian algorithm loop using this vertex cover, equality is maintained for all edges in M
Edges from T to R vanish from $G_{u,v}$, but no M-alternating paths traversed those edges so that changes nothing
A new edge from S to $Y - T$ appears.

If it creates an M-augmenting path, we have a strictly larger matching.

If not, U is unchanged but T is now larger (we can reach the new point through the new edge)

T can only increment in size at most n times, so eventually we will be forced to hit the other case

The stable matching is removed and must be discussed.

Problem:

Stable matchings

Suppose n applicants are applying for n job positions.

Each applicant has an ordered preference list of the positions they want
Each job position has an ordered ranking of the best candidates for that position.

Suppose I gave you a matching

Specify an applicant x and a position a, supposing they aren't matched
Suppose x prefers a over their current match and a prefers x over their current match
Then x and a might renege on the agreement and match each other.

We say (x, a) is an *unstable pair*

Definition:

A perfect matching is a *stable matching* if it has no unstable unmatched pair.

Does a stable matching necessarily exist, and can we find it?

Algorithm: (Gale-Shapley Proposal Algorithm)
(this is actually a great paper, called "College admissions and the stability of marriage")
Input: Preference rankings for each applicant and for each job position.

Loop: men
Each applicant selects the position highest on their preference list who has not rejected them.
If all positions women get one selecting applicant, use that as the matching. men
Else, for each position with more than one selecting applicant
Reject all of them except the one highest on the position's preference ranking
Everyone says "maybe" to all remaining selected applicants.

Theorem: (Gale-Shapley 1962)
The proposal algorithm produces a stable matching.

Proof:
The algorithm definitely produces a matching if it terminates. It must terminate, since the total length of remaining proposals decreases.

Suppose the result is not stable, so (x, a) is an unstable pair for some x, a with $x \sim b$ and $y \sim a$
During the algorithm, x first selected their favorite position, doing so repeatedly until that job received a better applicant and rejected x
In particular, over iterations, applicants selects nonincreasing quality matches every round positions receive nondecreasing quality offers every round

Since x ended with a worse match than a, they must have selected a during the algorithm
Since a ended with a worse match than x, they must have never received a proposal from x

The key observation also says that x would not have proposed to b without proposing to a first and then being rejected!

Disgustingly simple.

Asymmetry:

Males propose first and if they all get their first choice, then done!

So males come out on top here.

You can also have women doing the proposing.

Residents | Hospitals

NRMP National Resident Matching Program. Discovered this algorithm 10 years before Gale-Shapley.

Originally hospitals did the proposing! They were happier.

In the job-candidate version, the applications are equivalent to women saying, hey we're available for marriage. The job offers, are proposals!

→ There are many alternatives and variants. Men can give women weights and vice versa, and you get a weighted graph to which the Hungarian algorithm can be applied.

Section 4.1 - Cuts and Connectivity

We will be pursuing the analogy of thinking of graphs as networks. We'd like to ask how robust we can make these networks.

In a network, loops are pretty irrelevant, since they don't affect connectivity at all.
We will thus universally assume that graphs have *no loops*

Definition:

A *separating set* or *vertex cut* of a graph G is a set $S \subseteq V(G)$ such that $G - S$ has more than one component.

The *connectivity* of G is $\kappa(G)$, the minimum size of a vertex set S such that $G - S$ is disconnected or has only one vertex

A graph is k-connected if its connectivity is at least k

Note:

K_n has connectivity $n - 1$

For any graph not containing K_n, having connectivity $\kappa(G)$ means there is a separating set of size $\kappa(G)$ and no smaller such sets.

Question:

What is the connectivity of $K_{m,n}$?

$K_{3,3}$ has connectivity 3. It is 1-connected, 2-connected, 3-connected by not 4 connected.

K_1 has connectivity 0 even though it is connected.

↳ Question.

Claim:

The n-dimensional hypercube Q_k has connectivity k

Proof:

Take the neighbors of a fixed vertex in Q_k to get $\kappa(Q_k) \leq k$

We prove the other direction by induction.

K_1 = Q_0 and Q_1 are complete graphs ↳ K_2

Note that Q_k can be written as two copies Q, Q' of Q_{k-1} linked by a matching between corresponding vertices

Let S be a vertex cut of Q_k

Suppose first that $Q - S$ and $Q' - S$ are both connected.

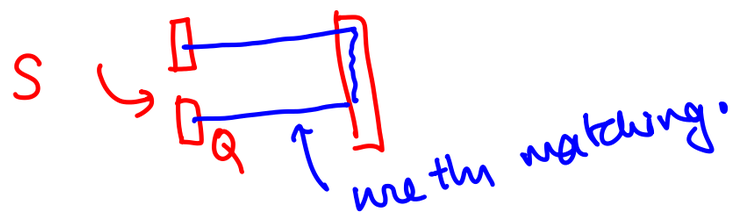
Then S contains one endpoint of each matched pair, so $|S| \geq 2^{k-1} > k$

Then $Q_k - S$ is also connected unless S contains one vertex from every pt of the matching

Then WLOG we may assume $Q - S$ is disconnected, so $|S \cap Q| \geq k - 1$ by Inductive hypothesis

But $S \cap Q' \neq \emptyset$ ← if not then Q' is connected & you can use the matching to go from one part of $Q - S$ to another

Hence $|S| \geq k$, so $\kappa(Q_k) \geq k$



Note:

For any vertex $v \in V(G)$, $N(v)$ is a vertex cut.

Thus $\kappa(G) \leq \delta(G)$

We've actually discussed how many edges a graph needs to have in order to have $\delta(G) \geq k$, with a lower bound of $\left\lceil \frac{kn}{2} \right\rceil$ edges - achieved by the hypercubes with $n = 2^k$ (n vertices) ← for given k

At least $\frac{nk}{2}$ edges by degree sum. It may not be an integer so need $\left\lceil \frac{nk}{2} \right\rceil$ edges.

This will be achieved by

Hence we have a lower bound on the number of edges needed for a graph on n vertices to have $\kappa(G) \geq k$

We'll show that this is sharp if $k < n$ $\forall n,k$ $\Rightarrow \exists$ a graph G st $K(G)=k$ and $\lceil \frac{nk}{2} \rceil$ edges

Example: (Harary Graphs)

Fix $2 \leq k < n$, we define a graph $H_{k,n}$

Place n vertices around a circle at equally spaced points.

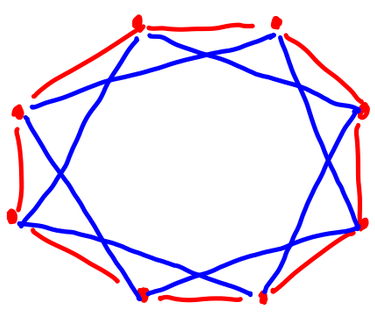
If k is even, make each vertex adjacent to the nearest $\frac{k}{2}$ vertices in each direction around the circle

If k is odd and n is even, make each vertex adjacent to the nearest $\frac{k-1}{2}$ vertices in each direction, and adjacent to the opposite vertex

ignore

If k and n are both odd, index vertices by integers mod n. Take $H_{k-1,n}$ and add the edges $i \leftrightarrow i + \frac{(n-1)}{2}$ for $0 \leq i \leq \frac{n-1}{2}$

$H_{4,8}$



$n_2 = 2$ Each edge has degree k.

Theorem: (Harary 1962)

$\kappa(H_{k,n}) = k$, so the minimum number of edges in a k-connected graph on n vertices is $\lceil \frac{kn}{2} \rceil$

Proof:

(The case where $k = 2r$ is even)

Certainly $\delta(H_{k,n}) = k$, so we need only show $\kappa(H_{k,n}) \geq k$ (since by previous $K(H_{k,n}) \leq k$)

Let $S \subseteq V(G)$ with $|S| < k$

Fix $u, v \in V(G) - S$

There is a clockwise u, v-path and a counterclockwise u, v-path along the circle, let the interior points of those paths be A and B

By Pigeonhole, either $S \cap A$ or $S \cap B$ contains fewer than $\frac{k}{2}$ vertices

Each vertex is connected to the next $\frac{k}{2}$ vertices in each direction, so there's still a u, v −path in that direction

Note:

Going for a direct proof of $\kappa(G) \geq k$ requires either

Consider a vertex cut S and show $|S| \geq k$

Consider a set S with $|S| < k$ and show $G - S$ is connected

Indirect proofs are usually by contradiction.

There is some art to knowing which will be easier to figure out and which will be easier to write up for any given example

We now know that there is a required number of edges for a graph to even possibly be k-connected.

Is there a number of edges that would force it? (This is a question I'll intentionally leave open)
(Note that multiple edges wouldn't really affect anything, so one need consider simple graphs only.)

Vertices | Separating set / Vertex cut
Edges | Disconnecting set | Edge cut.

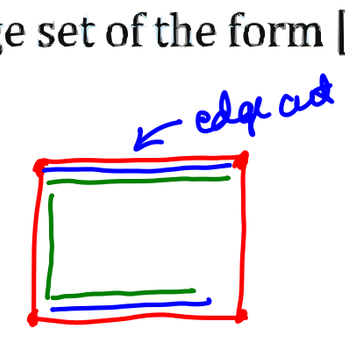
Definition:

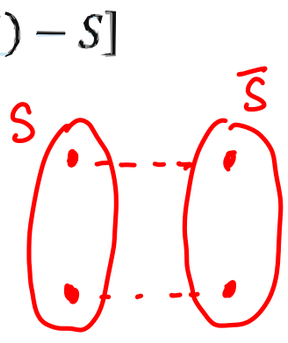
A *disconnecting set* of edges is a set $F \subseteq E(G)$ such that $G - F$ has more than one component.
A graph is *k-edge-connected* if every disconnecting set has at least k edges.
The *edge-connectivity* of G is $\kappa'(G)$, the minimum size of a disconnecting set.

Definition:

Given $S, T \subseteq V(G)$, we write $[S, T]$ as the set of edges with one endpoint in S and the other endpoint in T

An *edge cut* is an edge set of the form $[S, V(G) - S]$





vs

disconnecting sets can contain more edges than necessary.

Question:

Every edge cut is a disconnecting set. The opposite is false.

Every *minimal disconnecting set* is an edge cut.

Question: Where is the difference coming from? Is coming from the phrase "more than one component".

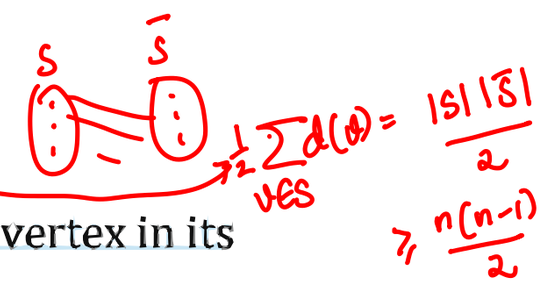
$\kappa(K_1)$

Theorem: (Whitney 1932) "Need fewer vertices than edges to cut a graph"
If G is a simple graph, then $\kappa(G) \leq \kappa'(G) \leq \delta(G)$

Proof: Assume $n \geq 2$

Edges incident to any given vertex form a disconnecting set, so $\kappa'(G) \leq \delta(G)$

Note that $\kappa(G) \leq n(G) - 1$ (duh)
Take a minimal edge cut $[S, V(G) - S]$. If every vertex in S is adjacent to every vertex in its complement, then this edge cut is bigger than $n(G) - 1$ $\Rightarrow \kappa'(G) \geq \kappa(G)$



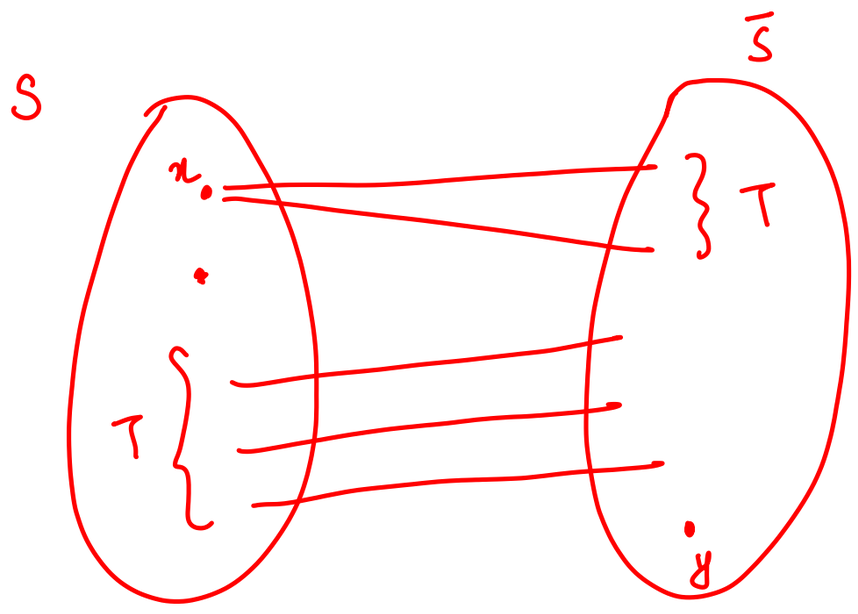
Idea: There is a vertex cut leading inside an edge cut (with less than $n-1$ edges)

Else, fix $x \in S$ and $y \in \bar{S}$ nonadjacent. Let T be set of neighbors of x in $\bar{S}$ and vertices in $S - \{x\}$ adjacent to something in $\bar{S}$ $\geq n-1$
$\Leftrightarrow n \geq 2$
T is definitely a separating set, since all x, y-paths pass through T ← vertex cut

Pick all edges from x to $T \cap \bar{S}$ and one edge for each point in $T \cap S$
This is a subset of $[S, \bar{S}]$, so

$$\kappa'(G) = \left|\left[S, \bar{S}\right]\right| \geq |T|$$

A path $x \to y$ must get to $\bar{S}$ somehow and it must pass through T.



Q: Can you find a regular graph where $\kappa(G) < \delta(G)$

Note:

If $\kappa(G) = \delta(G)$, then necessarily $\kappa'(G) = \delta(G)$
This includes complete graphs, bicliques, hypercubes, and Harary graphs

There is a rich subject exploring relationships between these topics. Flexibility is an interesting question.

Theorem:

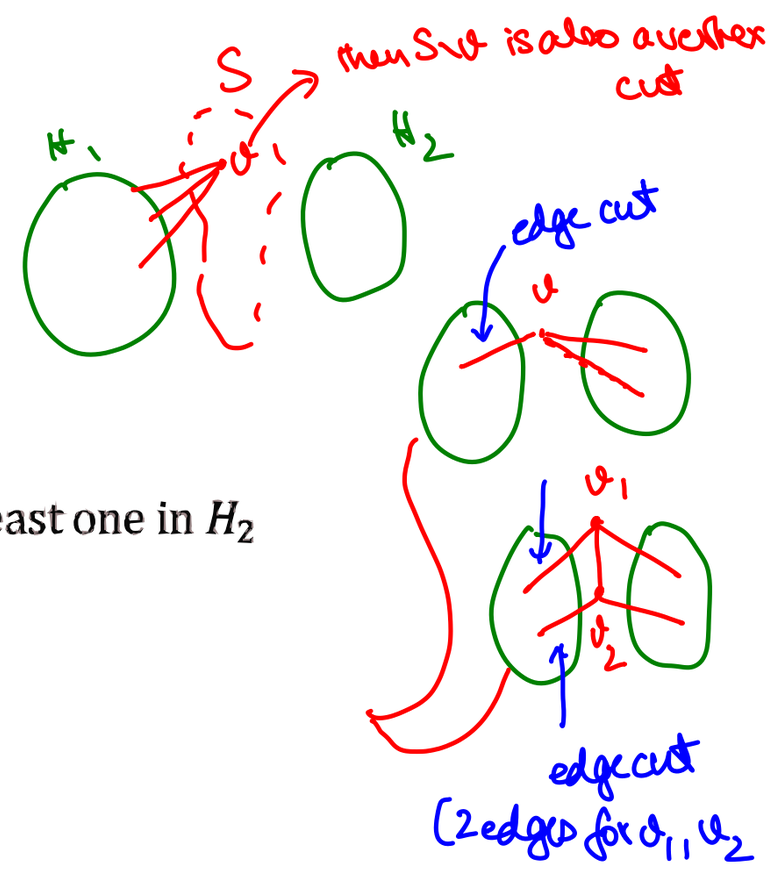
If G is a 3-regular graph, then $\kappa(G) = \kappa'(G)$

→ demonstrate ∃ an edge cut of same size.

Proof:

Take S a minimum vertex cut and H_1, H_2 two components of $G - S$
Since S is minimal, each $v \in S$ has at least one neighbor in H_1 and at least one in H_2
Cannot have 2 of each.

There are 2 possible situations $\forall\ v \in S$.

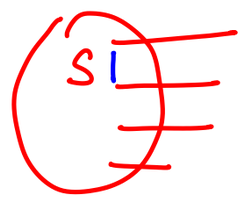


This breaks all paths from H_1 to H_2 and deletes exactly one edge for each $v \in S$, so $\kappa(G) = \kappa'(G)$

What if I have a graph where $\kappa'(G) < \delta(G)$. Then edge cuts are pretty small compared to vertex degrees, so we aren't just isolating vertices. Can we express how big the components we're cutting off are?

Proposition:

If $S \subseteq V(G)$, then $\left|\left[S, \overline{S}\right]\right| = \left|\sum_{v \in S} d(v)\right| - 2e(G[S])$



Proof:

Edges in $G[S]$ are counted twice in the sum $\sum_{v \in S} d(v)$ if you remove them, you're left with the edges from S to $\overline{S}$.

Corollary:

If G is a simple graph and $\left|\left[S, \overline{S}\right]\right| < \delta(G)$ for some nonempty proper subset S of $V(G)$, then $|S| > \delta(G)$ ← what does this mean? You can perform an edge cut by simply choosing $S = \{v\}$ where $d(v) = \delta(G)$.

Proof: are given

We $\delta(G) > \left|\sum_{v \in S} d(v)\right| - 2e(G[S])$ $= |[S, \overline{S}]|$

Use:

$d(v) \geq \delta(G)$ ⟶ $\Rightarrow \delta(G) > \delta(G)|S| - 2e(G(S))$

$\Rightarrow 2e(G(S)) > \delta(G)(|S| - 1)$

Use $e(G(S)) \leq \frac{|S|(|S|-1)}{2}$ (complete graph formula)

$\Rightarrow |S|(|S|-1)) > \delta(G)(|S|-1)$

Resulting inequality requires $|S| > 1$, (otherwise $0 > 0$) which gives the inequality.

We may often find ourselves breaking graphs into many smaller pieces. Removing many edges may give large edge cuts containing smaller edge cuts.

Definition:

A *bond* is a minimal nonempty edge cut

Proposition:

If G is connected, then an edge cut F is a bond if and only if $G - F$ has exactly two components.

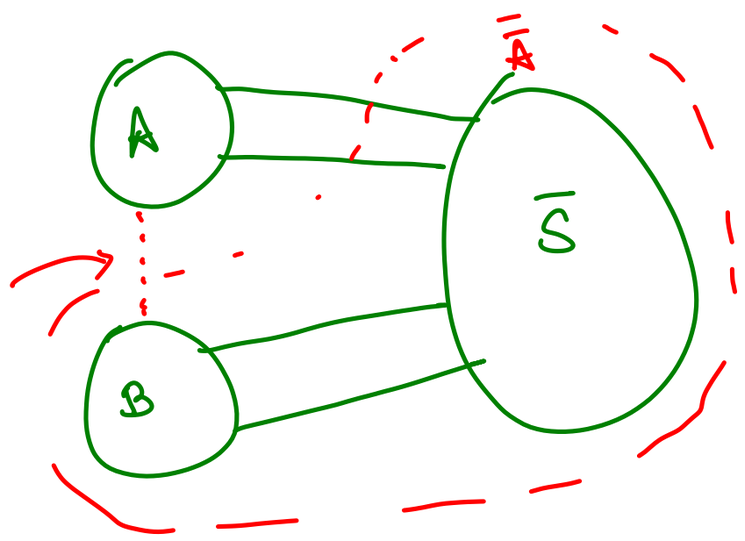
Proof:

If $F = [S, \overline{S}]$ is an edge cut with $G - F$ having exactly two components, take a strict subset $F' \subset F$

$G - F'$ contains both components of $G - F$, but must contain an edge between S and $\overline{S}$

this is because $[S,\overline{S}]$ is all the edges between S and $\overline{S}$, and if you leave one out, S and $\overline{S}$ are connected $\Rightarrow$ F is minimal!

Converse:



if it exists it is part of F

Suppose $F = [S, \overline{S}]$ is st $G-F$ has more than one component.

$[A, \overline{A}]$ does not include the edges from B to $\overline{S}$ so it is strictly smaller than $[S, \overline{S}]$

So this tells us that maybe the disconnecting set definition is a bit unnecessary.

BLOCKS :

Blocks are analogs of strong components of digraphs.

We know how to break a graph into components. However, when thinking about robustness, we might like to break connected graphs into pieces such that each piece is "very connected" with more tenuous links existing between pieces.

Definition:

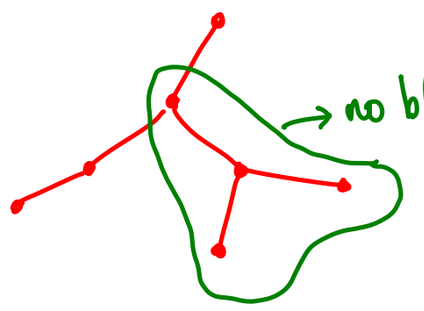
A *block* of a graph G is a maximal connected subgraph of G that has no cut-vertex. If G itself is connected and has no cut-vertex, G is a block.

↳ meaning of "very connected"

Can cut edge replace cut vertex?

Question:

Suppose G is a tree. What are the blocks of G?



→ no block can have more than 3 vertices (because it will have a non leaf vertex that is a cut-vertex)

- The graph of an edge is a block iff it is a cut-edge. (Because if it is not a cut edge, it must be part of a cycle, and cycles have no cut-vertices)
- If a block has more than 2 vertices, it must be 2-connected! (since it has ≥ 3 vertices and no cut-vertex)

⇒ Every edge of a tree is a cut edge

Note:

Blocks of a loopless graph consist of
- isolated vertices
- cut-edges ← the graph of the cut-edge
- maximal 2-connected subgraphs (with at least 3 vertices)

Note:

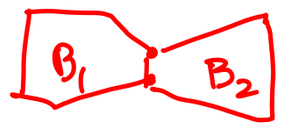
Blocks in a graph form a decomposition of a graph. This can have really great properties, not unlike strong components of a digraph.

Proposition:

Two blocks in a graph have at most one vertex in common.

Proof:

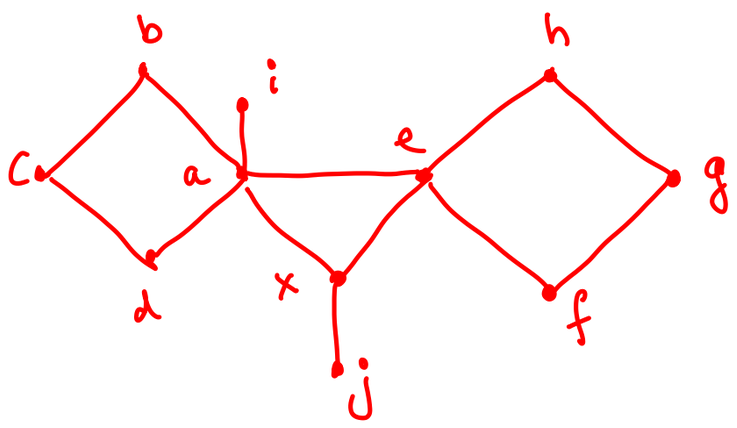
Proof by contradiction, let B_1, B_2 have two vertices in common. There can be no cut-vertex between them (paths can go through either)

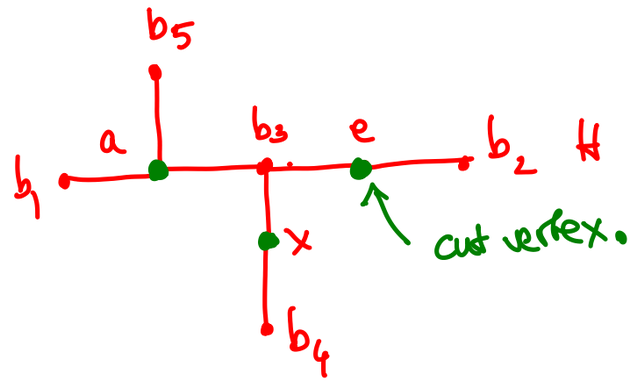


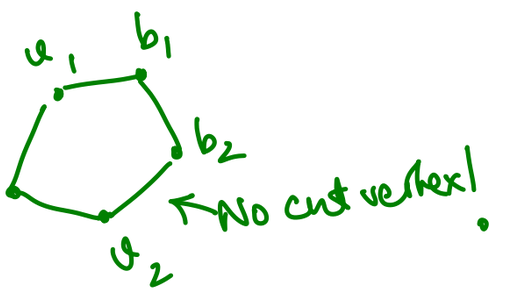
$B_1 \cup B_2$ has no cut vertex either so is a larger block.

Definition:

The *block-cutpoint graph* of a graph G is a bipartite graph H in which one partite set consists of cut-vertices of G, and the other partite set has one vertex b_i for each block B_i of G. For each cut vertex v, Include vb_i as an edge of H iff $v \in B_i$ (in G)







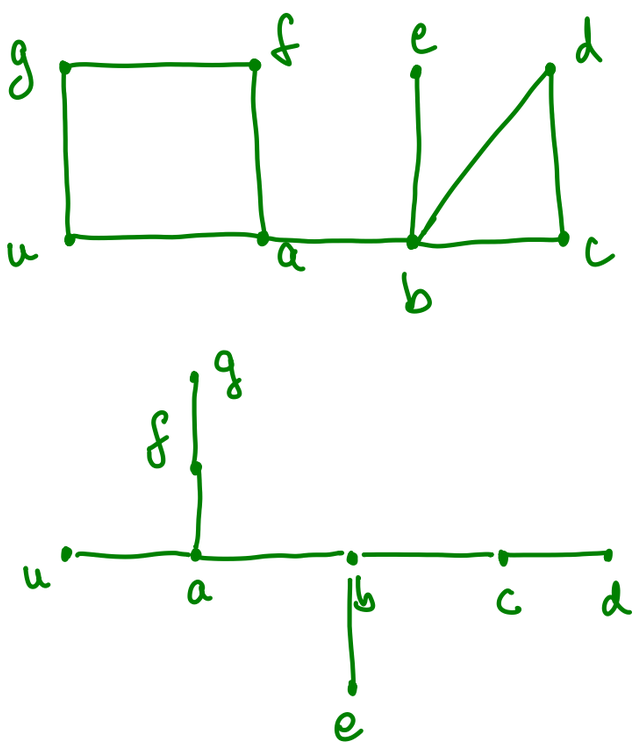
Block-cutpoint graphs are always trees!

Question:

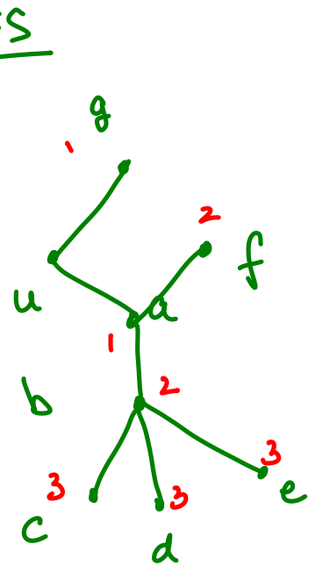
What does this graph look like when G is connected?

Finding blocks in a graph algorithmically may be done using depth first search. (It may be worth briefly talking about the difference between depth first search and breadth first search.

DFS



BFS



(use the picture from the book to talk about exploring graphs by DFS [and maybe BFS], and growing trees by doing so)

When discovering a new vertex v from an old vertex x, include xv

Lemma:

If T is a spanning tree of a connected graph G grown by DFS from u, then every edge of G not in T consists of two vertices v, w such that v lies on the u, w-path in T

Proof:

Let $vw \in e(G)$, with v encountered before w

Since vw is an edge, we can't have finished with v before w is added to T

Thus w appears in the subtree rooted at v, so the path from u to w passes through v

Algorithm: (Finding the Blocks of a graph)

Input: Connected graph G

Idea: Build a DFS tree T, discard portions of T as blocks are found. Maintain an "active" vertex

Fix a root $x \in V(H)$, make x ACTIVE, set $T = \{x\}$

Loop:

Let v denote the current ACTIVE vertex

If v has an unexplored edge vw

If $w \notin V(T)$ add vw to T, mark vw explored, make w ACTIVE

If $w \in V(T)$, then w is ancestor of v, mark vw explored

If v has no more unexplored incident edges

If $v \neq x$

w the parent of v Make w ACTIVE

If no vertex in the subtree T' rooted at v has an explored edge to an ancestor above w

basically saying v is a cut vertex.

$V(T') \cup \{w\}$ is the vertex set of a block

Record that info, delete $V(T')$ from T

If $v = x$

End program

Do an example of this algorithm, but don't prove that it works.

Section 4.2 - k-Connected Graphs

Intuitively, being "very connected" means that there should be a lot of different paths that go between any two points.

If a network is to be fault tolerant, it should be at least 2-connected. We'll explore this case first.

Definition:

Two u, v-paths are *internally disjoint* if they have no common internal vertices.

Theorem: (Whitney 1932)

If G has at least three vertices, G is 2-connected if and only if for each pair $u, v \in V(G)$ there exist internally disjoint u, v-paths in G.

Proof:

(<=) Deletion of any point cannot disconnect any pair of other points, so this is immediate.

(=>)

Induction on $d(u, v)$

Base case - remove edge between u and v, since $\kappa' \geq \kappa$ the resulting graph is still connected.

Let $k = d(u, v)$

Pick the shortest u, v-path, let w be last point on it before v

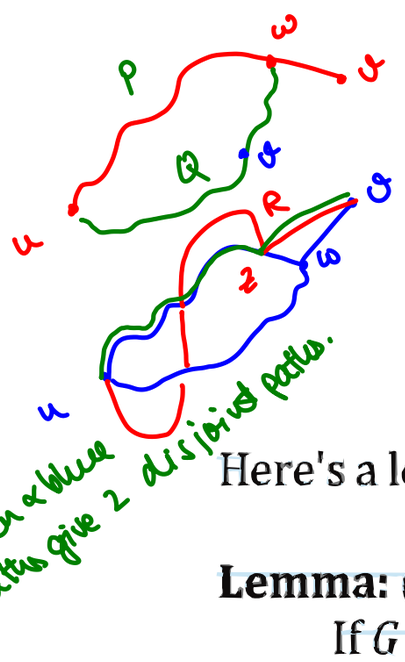
By induction hypothesis, two paths to w from u. If one contains v, we're done, by looking at the formed cycle.

Otherwise, may assume neither contains v. Consider $G - w$, there's a u, v-path R.

If this avoids previous paths, done.

Otherwise, let $z \in R$ be the last point intersecting previous paths (draw picture).

May come up with internally disjoint paths.



Here's a lemma which is useful for building up bigger k-connected graphs from smaller ones.

Lemma: (Expansion Lemma)

If G is k-connected and G' is G with an added vertex y with at least k neighbors, then G' is k-connected.

Proof:

Let S be a separating set. If $y \in S$, then $S - \{y\}$ must separate G

If $y \notin S$, then either $N(y) \subseteq S$ or $G' - \{\text{component containing } y\}$ is disconnected, and $S - \{y\}$ separates G again

Theorem:

If G has at least three vertices, the following conditions are equivalent

A') G is 2-connected

A) G is connected and has no cut-vertex

B) For all $x, y \in V(G)$, there are internally disjoint x, y-paths

C) For all $x, y \in V(G)$, there is a cycle through x and y

D) $\delta(G) \geq 1$, and every pair of edges in G lies on a common cycle. Surely we could take $\delta \geq 2$

Proof:

A' = A is a definition

A = B is the previous theorem

B = C is obvious

D => C

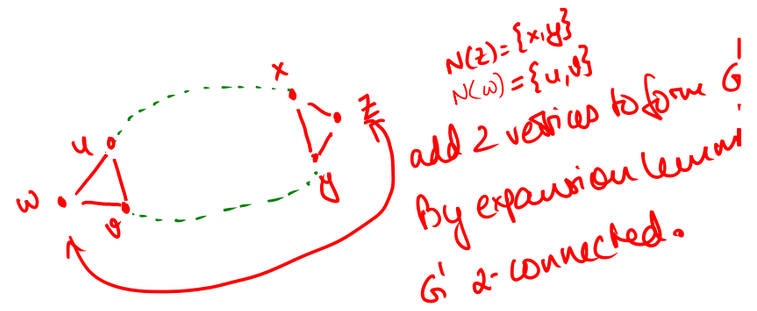
No edge is isolated, pick an edge incident on x and y

(A,B,C) => D

G is connected, so $\delta(G) \geq 1$ certainly

Take edges uv and xy. Use expansion lemma to add a vertex incident only on u, v and another incident only on x, y

Take a cycle between these two new vertices.



Definition:

In a graph G, a *subdivision* of an edge uv is the replacement of uv with a path u, w, v between u, v through a new vertex w

Lemma:

If G is 2-connected, then G' obtained by subdividing an edge of G is 2-connected.

(without adding 3 vertices)

Proof: Since it involves edges, think of using the cycle criterion through edges in the previous theorem. Take uw and any other edge, just use the cycle in G

As it turns out, combining the expansion lemma and the subdivision lemma actually lets us build up all possible 2-connected graphs (ears are just subdivided degree 2 vertices)

Definition:

An *ear* of a graph G is a path in G that is contained in a cycle and is maximal (all of whose internal vertices have degree 2.

An *ear decomposition* of G is a decomposition $P_0, \dots, P_k$ such that P_0 is a cycle and P_i for $i \geq 1$ is an ear of $P_0 \cup \cdots \cup P_i$

(draw picture)

Theorem: (Whitney 1932)

A graph is 2-connected iff it has an ear decomposition. Moreover, every cycle in a 2-connected graph is the initial cycle in some ear decomposition.

Proof:

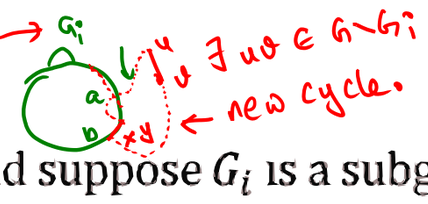
Graphs with ear decompositions are 2-connected by lemmas.

each time you add a ear between u,v. Now G+uv is 2 connected since uv is an edge.

The subdivide uv into P and use the previous lemmas!

Let C be a cycle in 2-connected G, set $G_0 = C$, and suppose G_i is a subgraph of G obtained by adding i ears successively to G_0

Inductive step



a = first time the cycle touches Gi from u

b = first time cycle touches Gi from v

Suppose $G_i \neq G$

Take uv an edge in $G - G_i$ and xy an edge in G_i.

Form a cycle containing both, this cycle contains a path with endpoints the only intersections with G_i. This is an ear.

→ Process must terminate with $G_i = G$ eventually

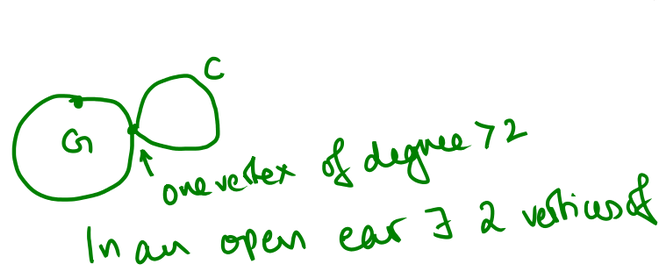
Note that this edge xy has 2 endpoints in Gi and this ensures ∃ an OPEN ear.

Question: What about edge connectivity? Can we characterize it in any simple ways?

Definition:

A *closed ear* in a graph G is a cycle C such that all vertices of C except one have degree 2 in G.

A *closed ear decomposition* of a graph G is a decomposition $P_0, \dots, P_k$ such that P_0 is a cycle and P_i is an ear or a closed ear in $P_0 \cup \cdots \cup P_k$



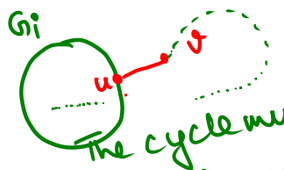
THEOREM

A graph is 2-edge-connected iff it has a closed ear decomposition, and every cycle in a 2-edge-connected graph is the initial cycle in some closed ear decomposition.

Proof:

Cut edges cannot be in any cycles, so 2-edge-connected iff every edge is in a cycle.
If G has a closed ear decomposition, this is immediately true.

If G is 2-edge-connected, let P_0 be a cycle in G.
Suppose $G_i \subset G$ has a closed ear decomposition $P_0, \dots, P_i$, pick uv not in G_i with $u \in V(G_i)$.
It is contained in a cycle, which must eventually return to G_i, forming an ear or a closed ear.



The cycle must return to G_i - (since it must come back to u)
Note the difference between this and the previous.

In the 2-connected case we use the characterization that $\forall$ 2 edges, $\exists$ a cycle containing them.

Digraphs

Definition:

Given a digraph D:

A *separating set* or *vertex cut* is a set $S \subseteq V(D)$ such that $D - S$ is not strongly connected
The connectivity $\kappa(D)$ and k-connectedness are defined in identical ways as the undirected case

For $S, T \subseteq V(D)$, denote by $[S, T]$ the set of edges with tails in S and heads in T
An *edge cut* is the set $[S, \bar{S}]$ for some $\emptyset \neq S$
k-edge-connectedness and edge-connectivity $\kappa'(D)$ are defined identically as the undirected case

Note:

The digraph case is fairly convenient in that $[S, \bar{S}]$ may now be thought of precisely as "the set of edges leaving S". This leads to a nice fact

D is k-edge-connected if and only if for all nonempty proper vertex subsets S, there are at least k edges in D leaving S

The following proposition will give us some convenient intuition about strong digraphs, relating them to 2-connected undirected digraphs.

Proposition:

Adding a (directed) ear to a strong digraph produces a strong digraph.

Proof:

Pretty straightforward, show that every set has an edge departing it.

Question:

(draw an undirected graph which is not 2-edge-connected)
If this is a road network and all of the roads were suddenly made one-way, would you be able to go from any point in this graph to any other point?

Theorem: (Robbins 1939)

A graph has a strong orientation if and only if it is 2-edge-connected.

Proof:

Obviously connectedness is necessary, and the absence of cut edges is necessary

Take G 2-edge-connected. Take a closed ear decomposition. Arrange the initial cycle cyclically, and then as each ear is added, make it a consistent direction. The orientation will remain strong.

This theorem can actually be generalized pretty substantially.

Theorem: (see Frank 1993)

A graph G has a k-edge-connected orientation if and only if it is $2k$-edge-connected.

We have seen 2-connected $\Leftrightarrow$ 2 disjoint paths between x, y.

$\Updownarrow$

ear decomposition

2 edge-connected $\Leftrightarrow$ closed/open ear decomp.

k-Connectedness

Can we come up with an easier equivalent condition for being k-connected?

We'll first formulate some 'local' properties which are easier to think about.

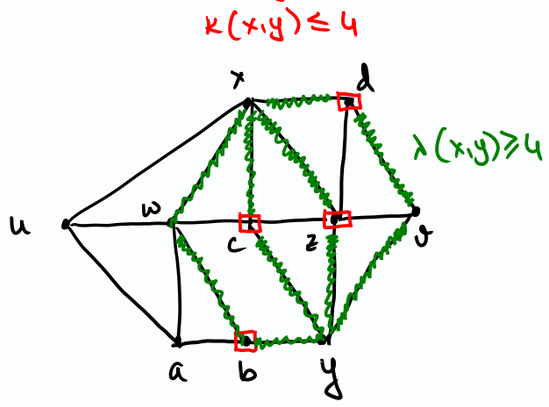
Definition:

Take $x, y \in V(G)$. A set $S \subseteq V(G) - \{x, y\}$ is an x, y-*separator* or an x, y-*cut* if $G - S$ has no x, y-path.
Let $\kappa(x, y)$ be the minimum size of an x, y-cut
Let $\lambda(x, y)$ be the maximum size of a set of pairwise internally disjoint x, y-paths
For $X, Y \subseteq V(G)$ an X, Y-path is a path with first vertex in X and last vertex in Y

Note:

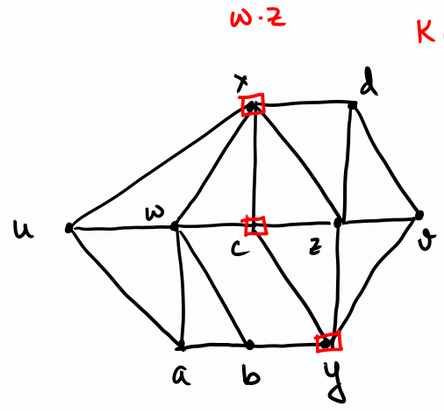
An x, y-cut must contain an internal vertex of each x, y-path, so $\kappa(x, y) \geq \lambda(x, y)$
(this gives a duality relationship and an easy way to check optimality)

x, y: $\kappa(x,y) \leq 4$

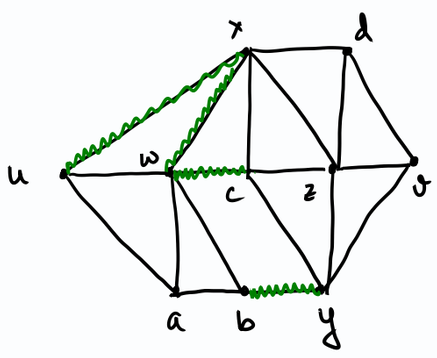


$\lambda(x,y) \geq 4$

w, z: $\kappa(w,z) \leq 3$



$\lambda(w,z) \geq 3$ (easy to see)



Edge connectivity:
$\kappa_e(w,z) \leq 4$
$\lambda_e(w,z) \geq 4$ (easy to see)

3d sierpinsky.

Theorem: (Menger 1927) [Same guy as the sponge]

If x, y are vertices of a graph G and $xy \notin E(G)$, then the minimum size of an x, y-cut equals the maximum number of pairwise internally disjoint x, y-paths.

important condition.

Proof:

$\kappa_v(x,y) \geq \lambda_v(x,y)$

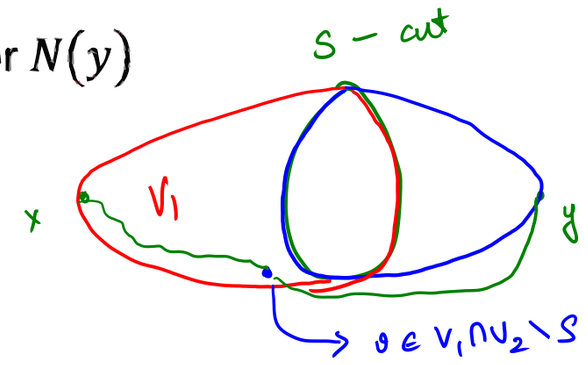
We already have one direction of the equality. Need to show the other direction.

We work by induction on $n(G)$. If $n(G) = 2$, $xy \notin G$ so the graph is trivial
Let $k = \kappa_G(x, y)$, want to find k disjoint paths
$N(x)$ and $N(y)$ are both x, y-cuts, so no minimum x, y-cut properly contains them.

In other words $\kappa(x,y) \leq \min\{|N(x)|, |N(y)|\}$

Case 1:

G has a minimum x, y-cut S which is not $N(x)$ or $N(y)$



1st vertex x_1, last vertex S, no internal vertices in x or S.

Combine x, S-paths and S, y-paths

Let V_1 = vertices on x, S-paths

V_2 = vertices on S, y-paths

To show $V_1 \cap V_2 = S$, certainly $S \subseteq V_1 \cap V_2$

Take $v \in V_1 \cap V_2 \setminus S$, there exists an x, S-path containing v and a y, S-path containing v

Traverse pieces of them to avoid S with an x, y-path, this is impossible since S is an x-y cut

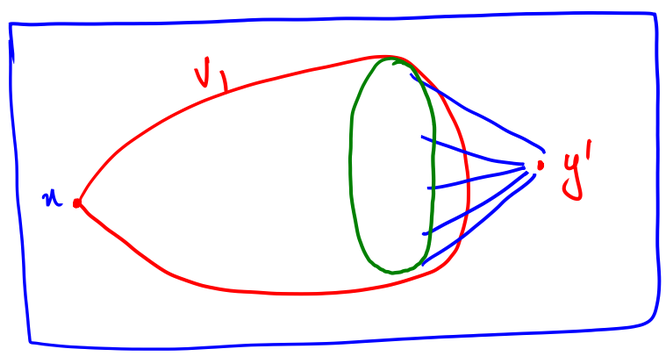
Similarly, any $v \in N(y) \setminus S$ is not in V_1 by same argument, ← if it is go from x to $v \to y$ since $v \in N(y)$.

and any $v \in N(x) \setminus S$ is not in V_2

Take induced graph on V_1, add vertex y' adjacent to all points in S to get graph H_1

Take induced graph on V_2, add vertex x' adjacent to all points in S to get graph H_2

Every x,y path in G starts with x,S path which is also in H_1



So if R is (x,y') cut, it must cut (x,S) because if you can get to S you can get to y'. ⇒ R is also an x,y cut in G!

remove S from H_1

$\Rightarrow k = \kappa_G(x,y) \leq \kappa_{H_1}(x,y') \leq k$

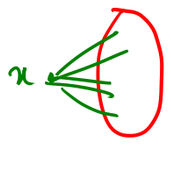
But $H_1 \subsetneq G$ ⇒ By induction ∃ k disjoint (x,y') paths ⇒ ∃ k disjoint (x,S) paths. Do the same with V_2 and join paths together to get k disjoint x,y paths in G.

Case 2:

Every minimum x, y-cut is $N(x)$ or $N(y)$. Why can we not repeat previous construction?

$H_1 = G$ ⇒ cannot use induction since $n(H_1) = n(G)$

If G has a vertex v which is not x, y nor in $N(x)$ or $N(y)$ then $\kappa_{G-v}(x, y) = k$, so inductive hypothesis gives desired paths. $(n(G-v) = n(G) - 1)$



v cannot be in any minimal x,y cut if $S = N(x)$ (say)

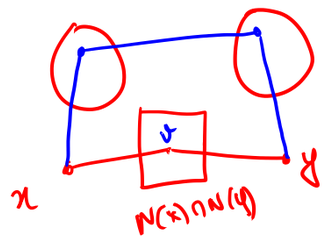
So $\kappa_{G-v}(x,y) = k$ as well,

$N(x)-v$ (say) is a cut ⇒ $\kappa_{G-v}(x,y) \leq k-1$.

If G has a vertex $v \in N(x) \cap N(y)$, then v is in every x, y-cut, so $\kappa_{G-v}(x, y) = k - 1$ and inductive hypothesis gives $k - 1$ internally disjoint paths. Include x, u, v as the last path.

If $S = N(x)$ then every vertex in $N(x)$ is connected to some $N(y)$

Remove v. If ∃ $z \in N(x)$ that only connected to v, then z can be removed from the cut S ⇒ z must be connected to some vertex other than v in $N(y)$.



Assume $\exists$ a cut S' in $G-v$
st $|S'|<k-1$. $S' \cup v$ cannot be a cut. $\exists$ a path

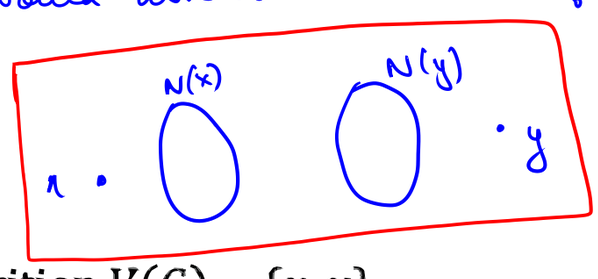
$x z_1 z_2 y$ or $x z y$

where $z_1 \in N(x) \setminus (S' \cup \{v\})$ $z_2 \in N(y) \setminus S' \cup \{v\}$

Or $z \in N(x) \cap N(y) \setminus (S' \cup \{v\})$

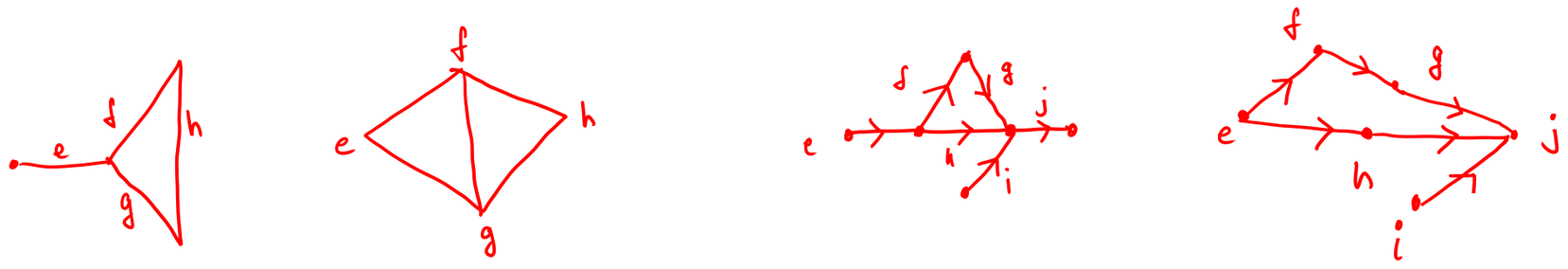
But this would also be a cut in $G-v$! Contradiction.

$\Rightarrow \kappa_{G-v}(x,y) \geq k-1$



Thus, may now assume $N(x)$ and $N(y)$ partition $V(G) - \{x, y\}$
Take G' a bigraph with bipartition $N(x)$ and $N(y)$ and edges $[N(x), N(y)]$
All x, y-paths cross from $N(x)$ to $N(y)$, and vertex cuts break all paths, so x, y-cuts in G are vertex covers in G', so $\beta(G') = k$ ($\min_V(G') = k$)
By Koenig-Egervary Theorem, G' has matching of size k $\min_V(G') = \max_M(G') = k$
Combining matched edges with edges to x, y, we get desired paths.

This statement is inherently about k-(vertex)connectivity. To get a similar statement about k-edge-connectivity, we translate our graph somewhat.

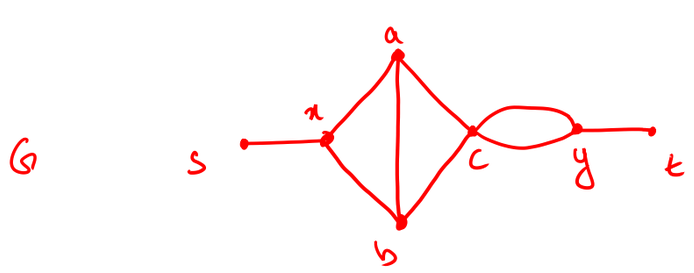


Definition: (This is discussed explicitly on homework)
The *line graph* of a graph G, $L(G)$ is the graph with $V(L(G)) = E(G)$ and $ef \in E(L(G))$ if $e, f \in E(G)$ are incident on a common vertex (or for digraphs, head of e is tail of f)

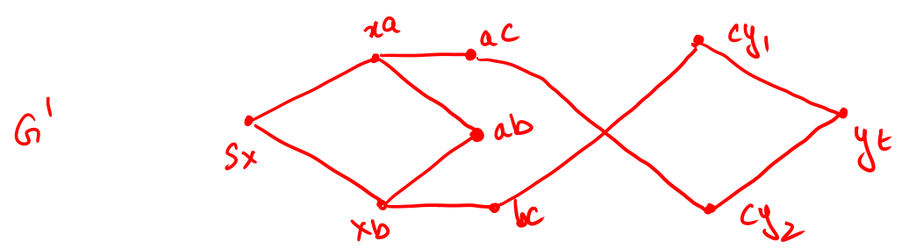
Notation:
$\lambda_e(x, y)$ is the maximum size of a set of pairwise-edge-disjoint x, y-paths
$\kappa_e(x, y)$ the minimum number of edges whose deletion makes y unreachable from x

(Elias-Feinstein-Shannon 1956 and Ford-Fulkerson 1956)
$\lambda_e(x, y) = \kappa_e(x, y)$ (does not matter if multigraph or if $xy \in E(G)$) → THIS is different from menger. Multiple edges are irrelevant there. Needed $xy \notin E(G)$ there.

Theorem: (Edge Menger)
If x, y are distinct vertices of a graph or digraph G, then $\lambda_e(x, y) = \kappa_e(x, y)$



← Add s,t to the graph.



$x \leftrightarrow y$ in G
$\Leftrightarrow$ $sx \leftrightarrow sy$ in $L(G)$

No edge from sx to yt in $L(G')$ since x and y are distinct (needed for vertex Menger)

Proof:

Add new vertices s, t and edges sx, yt (draw)
Does not change $\lambda'(x, y)$ or $\kappa'(x, y)$
Set of edges disconnects x, y in G iff corresponding vertices of $L(G')$ form an sx, yt-cut
Edge disjoint x, y-paths in G iff internally disjoint sx, yt-paths in $L(G')$
$x \neq y$, so sx, yt are not adjacent in $L(G')$
By Menger's Theorem

$$K_{e,G}(x, y) = \kappa_{L(G')}(sx, yt) = \lambda_{L(G')}(sx, yt) = \lambda_{e,G}(x, y)$$

construction

what does this mean? maybe $K(G) = \min_{xy} K(xy)$

Global version of k-connected statement is also often called Menger's theorem. For edges and for digraphs first appeared in Ford-Fulkerson 1956.

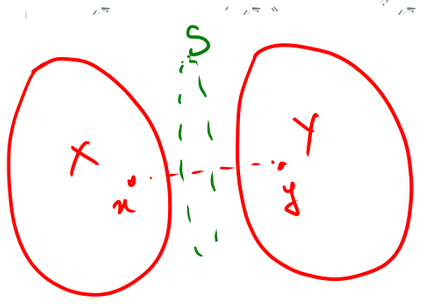
Lemma:

Deletion of an edge reduces vertex connectivity by at most 1

Proof:

Every vertex cut separating set of G separates $G - xy$, so $\kappa(G - xy) \leq \kappa(G)$ (obvious)

If equality does not hold, then $G - xy$ has a separating set S which does not separate G
$G - xy - S$ has some components, call two X, Y with $x \in X$ and $y \in Y$
xy is only edge connecting these components.



Why must $x \in X$? Suppose $x \in S$ (or $y \in S$), then S must be a cut in G as well since then $G - xy - S = G - S$.
$\Rightarrow x, y \notin S$. (Not obvious!)

If $|X| = 1$, $S \cup \{x\}$ leaves nothing behind & so cannot separate G.

If $|X| \geq 2$, $S \cup \{x\}$ separates G, so $\kappa(G) \leq \kappa(G - xy) + 1$. Same if $|Y| \geq 2$. In the remaining case, $|X| = |Y| = 1$

$\Rightarrow$ Otherwise $|S| = n(G) - 2$. Since $|S| < \kappa(G)$, $\kappa(G) = n(G) - 1$ [only happens if G is complete]

$\Rightarrow \kappa(G) \geq n-1$. But we know $\kappa(G) \leq n-1$, so G is complete. Then

$\kappa(G - xy) = n - 2 = \kappa(G) - 1$

Theorem:

Connectivity of G is the max k such that $\lambda(x,y) \geq k$ for all $x, y \in V(G)$

Edge-connectivity of G is the max k such that $\lambda'(x,y) \geq k$ for all $x, y \in V(G)$

Both statements are true for graphs and digraphs.

Proof:

$\kappa_e(G) = \min_{x,y \in V(G)} \kappa(x,y) = \min_{x,y} \lambda_e(x,y)$

So edge connectivity is immediate by previous theorem

For connectivity, we still have $\kappa(G) = \min_{x,y \in V(G)} \kappa(x,y)$

But we only know that $\lambda(x,y) = \kappa(x,y)$ if $xy \notin E(G)$

If $xy \in E(G)$, xy is an x,y-path, so deletion of xy reduces $\lambda(x,y)$ by 1

By previous lemma and Menger's theorem

$$\lambda_G(x,y) = 1 + \lambda_{G-xy}(x,y) = 1 + \kappa_{G-xy}(x,y) \geq 1 + \kappa(G - xy) \geq \kappa(G)$$

Questions similar to that in Menger's Theorem come up in a variety of different contexts. An important example is the Fan Lemma. (Not the same Dirac)

Definition:

Given a vertex x and a set U of vertices an x,U-*fan* is a set of paths from x to U such that any two of them only share the vertex x in common

Theorem: (Fan Lemma, Dirac 1960)

A graph is k-connected iff it has at least $k+1$ vertices and for all $x \in V(G)$ and U with $|U| \geq k$ it has an x,U-fan of size k

Proof:

To see necessary, add a new vertex adjacent to all of U then use Menger's theorem.

To see sufficient, suppose G satisfies fan condition.

Pick $v \in V(G)$ and $U = V(G) - \{v\}$. By fan condition, $\delta(G) \geq k$ (each path hits a neighbor)

Take $w, z \in V(G)$ and $U = N(z)$. $|U| \geq k$, extend w,U-paths with edges to z. This gives k w,z-paths.

A slightly less convoluted extension is below. [This may not be worth doing]

Theorem: (Dirac 1960)

If G is k-connected with $k \geq 2$ and S is a set of k-vertices in G, then G has a cycle including S in its vertex set.

Proof:

Induction on k. The $k = 2$ case is our characterization of 2-connectedness.

For larger $k > 2$, fix G, S

Take $x \in S$. $S - \{x\}$ all lies on a cycle C. If $n(C) = k - 1$ we must have an x, C-fan of size $k - 1$. Two paths in this fan going to adjacent vertices in C can be used to enlarge C

Now assume $n(C) \geq k$.
G has an $x, V(C)$-fan of size k
Claim:

There exist two paths in this fan that form a detour from C that includes x but keeps all of $S - \{x\}$ in C

Let $v_1, \dots, v_{k-1}$ the points of $S - \{x\}$ in the order they appear in C, let V_i be the portion of $V(C)$ starting at v_i up to but not including v_{i+1}
$V_1, \dots, V_{k-1}$ partition C into $k - 1$ sets, so by Pigeonhole two paths in the fan go to the same set. Take these as the detour.

Question:

Why is Menger's theorem helpful? What do all of these clever little arguments actually help us with?

Idea:

If you want to understand some problem, try to view the objects you want to study as paths in some graph of digraph, by cleverly defining a graph
Then use Menger's theorem to get disjoint paths, and translate these back into your desired context.

Example:

Given sets $\boldsymbol{A} = A_1, \dots, A_m$ with union X
A *system of distinct representatives* (SDR) is a set of distinct elements $x_1, \dots, x_m$ such that $x_i \in A_i$

A sufficient and necessary condition is $\left|\bigcup_{i \in I} A_i\right| \geq |I|$ for all $I \subseteq [m]$
(Does this look familiar?)
This problem is just a rephrasing of Hall's matching theorem, and is actually an equivalent formulation of Menger's Theorem

Here's a tougher variant.
Problem:

Let $\boldsymbol{A} = A_1, \dots, A_m$ and $\boldsymbol{B} = B_1 \dots, B_m$ be two families of sets.
A *common system of distinct representatives* (CSDR) is a set of m elements that is an SDR for both $\boldsymbol{A}$ and $\boldsymbol{B}$.

Theorem: (Ford-Fulkerson 1958)

Familes $\boldsymbol{A} = A_1, \dots, A_m$ and $\boldsymbol{B} = B_1 \dots, B_m$ have a CSDR if and only if

$$\left|\bigcup_{i \in I} A_i \cap \bigcup_{j \in J} B_j\right| \geq |I| + |J| - m$$

for each pair $I, J \subseteq [m]$

Proof:

Define a digraph G with vertices $a_1, \dots, a_m$ and $b_1, \dots, b_m$ and a vertex for each element in the sets, and two extra vertices s, t
Edges are

$$\{sa_i : A_i \in \boldsymbol{A}\}, \{b_j t : B_j \in \boldsymbol{B}\}$$

$$\{a_i x : x \in A_i \in \boldsymbol{A}\}, \{x b_j : x \in B_j \in \boldsymbol{B}\}$$

An s,t-path selects a member of the intersection of some A_i and B_j (one can only cross over at a common point

Claim:

There exists a CSDR iff there is a set of m pairwise internally disjoint s,t-paths

By Menger's theorem, there is a CSDR iff there is no s,t-cut of size less than m

Take $R \subseteq V(G) - \{s,t\}$ and let

$$I = \{i \in [m]: a_i \notin R\}$$

$$J = \{j \in [m]: b_j \notin R\}$$

Claim:

R is an s,t-cut iff

$$\bigcup_{i \in I} A_i \cap \bigcup_{j \in J} B_j \subseteq R$$

(all points which are both in a set A_i and in a set B_j which are not already covered by R are in R)

Thus,

$$|R| \geq \left| \bigcup_{i \in I} A_i \cap \bigcup_{j \in J} B_j \right| + (m - |I|) + (m - |J|)$$

This lower bound is always at least m (we can essentially choose I and J freely in varying R) iff the main condition holds [so that most terms on the right side cancel]

Section 4.3 - Network Flow Problems

Sunday, April 2, 2023 3:33 PM

Imagine a situation where you have a network of pipes (or roads, or bus lines, or electrical lines, etc.) where valves allow flow in one direction, each pipe having a specified capacity per unit time.
Put a vertex at each junction and model each pipe as an edge, weighted by capacity. Assume there's no buildup at junctions, for simplicity.
Given locations s, t in the network, one might be interested in asking how much flow one can get between s and t

This is an incredibly broad category of problems, one can find more in Ford-Fulkerson 1962 or Ahuja-Magnanti-Orlin 1993.

Definition:

A *network* is a digraph with a nonnegative *capacity* $c(e)$ on each edge e and a distinguished *source vertex* s and *sink vertex* t
Vertices are also called *nodes*
A *flow* f is a function assigning values to each edge e
Given a flow, write $f^+(v)$ for the total flow on edges leaving v and $f^-(v)$ for the total flow on edges entering v
A flow is *feasible* if it satisfies the *capacity constraints*

$$0 \leq f(e) \leq c(e)$$

for each edge and the *conservation constraints*

$$f^+(v) = f^-(v)$$

for each node $v \notin \{s, t\}$

There are a **lot** of questions that can be asked about such things. We'll start by asking about max flows.

Definition:

The *value* $val(f)$ of a flow f is the net flow $f^-(t) - f^+(t)$ into the sink
A *maximum flow* is a feasible flow of maximum value.

(draw an example of a flow on a digraph - pg 176 of the book is a good one to talk through [it has an augmenting path])
(explain why the *zero flow* is always feasible)

Definition:

When f is a feasible flow in a network N, an *f-augmenting path* is a source-to-sink path P in the underlying graph G such that for each $e \in E(P)$
if P follows e in the forward direction, $f(e) < c(e)$
if P follows e in the backward direction, $f(e) > 0$
If we have such a path, let $\epsilon(e) = c(e) - f(e)$ when e is forward on P and let $\epsilon(e) = f(e)$ when e is backward on P.
The *tolerance* of P is $\min_{e \in E(P)} \epsilon(e)$

(explain why this allows increasing the flow using a drawing)

Lemma:

If P is an f-augmenting path with tolerance z, then changing flow by $+z$ on edges followed forward by P and by $-z$ on edges followed backwards by P produces a feasible flow f' with $val(f') = val(f) + z$

Proof:

By definition of tolerance, $0 \leq f'(e) \leq c(e)$ for all edges, satisfying capacity constraints.
For conservation constraints, need only check points on P as only they've changed - but always in a way that cancels out (draw the four cases of directions of P around a node)

Net flow into the sink t increases by z

This means that if we want to find better flows, finding augmenting paths allows us to "redirect fluid" down pipes to improve our situation, iterating until we reach a maximum

Question:

Is there a quick(-ish) way to check that we are at a maximum?

Definition:

A *source/sink cut* $[S,T]$ consists of the edges from a *source set* S to a *sink set* T where S,T partition the set of nodes, with $s \in S$ and $T \in T$
The *capacity* of the cut $[S,T]$ written $cap(S,T)$ is the total of the capacity of the edges in $[S,T]$

(recall for digraphs $[S,T]$ are edges with tail in S and head in T)

Lemma:

If U is a set of nodes in a network, the net flow out of U is the sum of the net flows out of the nodes of U. In particular, if f is feasible and $[S,T]$ is a source/sink cut, then the net flow out of S and net flow into T equal $val(f)$

Proof:

We want to show

$$f^+(U) - f^-(U) = \sum_{v \in U} \left(f^+(v) - f^-(v)\right)$$

This formula is somewhat immediate (edges within U cancel on the right)

Interpret this in the case $U = S$ or $U = T$ for a source/sink cut

Corollary: (Weak duality)

If f is a feasible flow and $[S,T]$ is a source/sink cut, then $val(f) \leq cap(S,T)$

Proof:

By lemma,

$$val(f) = f^+(S) - f^-(S) \leq f^+(S)$$

Capacity constraints require $f^+(S) \leq cap(S,T)$

Note:

Given capacities of a network, the question of finding a source/sink cut with minimum capacity defines the *minimum cut* problem
Again, we have a duality result, and in fact we will ultimately have equality of solutions

Algorithm: (Ford-Fulkerson labeling algorithm)

Input: Feasible flow f in a network
Output: An f-augmenting path or a cut with capacity $val(f)$
Idea:

Find nodes reachable from s by paths with positive tolerance
Reaching t completes an f-augmenting path.
During search, $R =$ nodes labeled as "Reached" and $S \subseteq R$ the "Searched" nodes

Set $R = \{s\}$ and $S = \emptyset$
Loop:
    Choose $v \in R - S$
    For each exiting edge vw with $f(vw) < c(vw)$ and $w \notin R$
        Add w to R
    For each entering edge uv with $f(uv) > 0$ and $u \notin R$
        Add u to R
    Label vertices in R as "reached", record v as the vertex reaching it
    Add v to S and mark it "searched"

    If sink t has been reached (put in R)
        Trace the path reaching t to report an f-augmenting path
        Terminate

    If $R = S$
        Return the cut $[S, \overline{S}]$
        Terminate

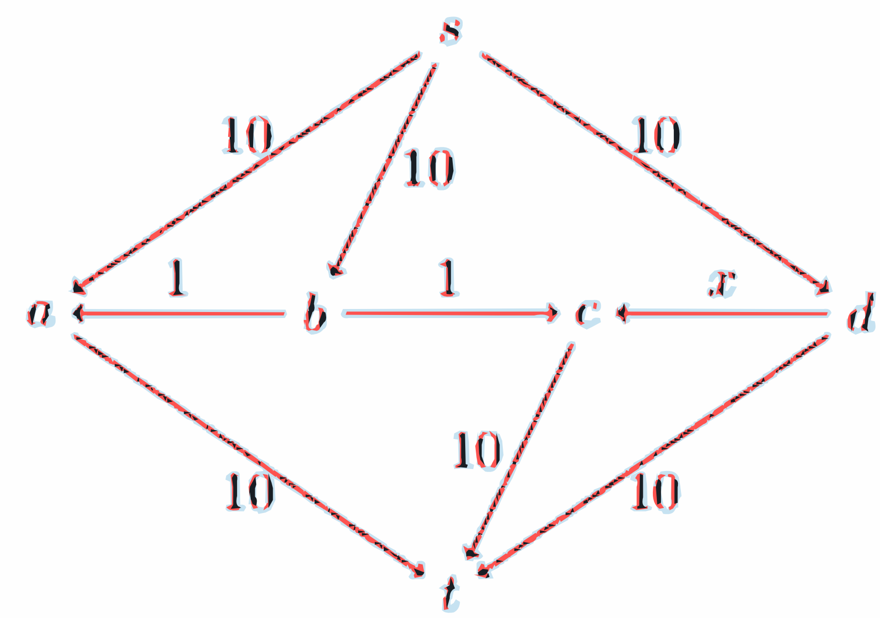
(draw a picture and run an example)

Theorem: (Max-flow Min-cut Theorem - Ford Fulkerson 1956)
In every network, the maximum value of a feasible flow equals the minimum capacity of a source/sink cut

Proof:
Zero flow is always feasible, to take it as start point.
Given a feasible flow, run Ford-Fulkerson algorithm.
If algorithm gives augmenting path, we may use it to increase flow value, then repeat algorithm.
    If capacities are rational, min tolerance is $1/a$ with a the lcm of denominators, so value increases by at least $1/a$ each time and is bounded.
Eventually algorithm must return a cut with equal value
    $[S, \overline{S}]$ is a source/sink cut since $s \in S$ and $t \in \overline{S}$
    Labeling algorithm included no nodes of $\overline{S}$
        so no edge from S to $\overline{S}$ has excess capacity
        and no edge from $\overline{S}$ to S has nonzero flow
    Thus $f^+(S) = cap(S,T)$ and $f^-(S) = 0$
    Thus $val(f) = cap(S,T)$

Question:
Wait, what gives? This proof assumes capacities are rational! What happens if I want a pipe with capacity π or $\sqrt{2}$??

This algorithm actually can break in such examples! It might infinitely loop - here's an example where things will break (take $x = \frac{\sqrt{5}-1}{2}$)



Should probably upload this to the course website, it explains the infinite loop:
https://faculty.math.illinois.edu/~mlavrov/docs/412-spring-2018/infinite-loop.pdf

However, there is a way to modify the algorithm so that it will work for all real capacities
The idea is to find the *shortest* augmenting path (See Ahuja-Magnanti-Orlin 1993)

~~~~~~
~~

Especially for applications in pure mathematics, flows tend to have integer capacities and one ultimately wants solutions for which the flow on each edge is an integer.

Corollary: (Integrality Theorem)
If all capacities in a network are integers, then there is a maximum flow assigning integral flow to each edge
Furthermore, some maximum flow can be partitioned into flows of unit value along paths from source to sink

Proof:
The tolerance in each step of the algorithm is an integer, so it must output an integral flow

For this produced maximum flow:
for each internal node, produce a matching of units of entering flow to units of exiting flow
This produces a collection of s, t-paths and some cycles on the graph.
For each cycle, decrease flow on each edge in the cycle by 1 to remove the cycle without changing the value of the flow.

Note:
This theorem is *really, really* similar to Menger's theorem

Idea 1: (From Max-Flow Min-Cut to Menger)
When x, y are vertices in a digraph D, view D as a network with source x and sink y and capacity 1 on all edges
Units of flow from x to y correspond to pairwise internally disjoint x, y-paths
A flow of value k thus has exactly k such paths

For any source/sink cut $[S, T]$, deleting these disconnects x and y
All capacities are 1, so the size of this set is $cap(S, T)$

Thus
$$\lambda'_D(x, y) \geq \max val(f) = \min cap(S, T) \geq \kappa'_D(x, y)$$
But $\lambda' \leq \kappa'$ always, so equality.

Idea 2: (From Menger to Max-Flow Min-Cut)
~~~~~~

Plan: Take an arbitrary network N with rational capacities, turn it into a digraph to apply Menger's Theorem.

WLOG clear denominators to get integer capacities.
For each edge, if the capacity is j split the edge into j directed edges with the same head and tail as the original edge. This gives a digraph D
By duality on the network, max $val(f) \leq$ min $cap(S, T)$

Take $\lambda'(s, t)$ pairwise edge-disjoint s, t-paths in D, this corresponds to a flow of value $\lambda'(s, t)$ on N, so

max $val(f) \geq \lambda'(s, t)$

Take F a set of $\kappa'(s, t)$ edges disconnecting t from s in D

If $e \in F$, by minimality some s, t-path passes over e but no other edge in F
F must contain all "copies" of the edge e, otherwise we could just take a different one (contains all or none of the copies of each edge)

Thus $\kappa'(s, t)$ is the sum of capacities on a set of edges that disconnects t from s
Let S be vertices reachable from s in $D - F$, this gives a source/sink cut with its complement.

$$cap\left(S, \overline{S}\right) = \kappa'(s, t)$$

Thus min $cap(S, T) \leq \kappa'(s, t)$

These two problems are equivalent! (And in fact are also equivalent to the maximum matching problem!)
The min-cut max-flow algorithm is probably the most computationally convenient algorithm we've discussed, though.

~~~~~~

(There's a section on supply and demand models which is pretty interesting - see what you have time for)
~~~~~~

Section 5.1 - Coloring of Graphs

Monday, April 10, 2023 4:54 PM

Definition:

A k-*coloring* of a graph G is a labeling $f: V(G) \to S$ where $|S| = k$
The labels are *colors*, the vertices of one color form a *color class*
A k-coloring is *proper* if adjacent vertices have different labels
A graph is k-*colorable* if it has a proper k-coloring
The *chromatic number* $\chi(G)$ is the minimum k such that G is k-colorable

Recall: (from the very beginning of the class, almost)

k-colorable and k-partite have the same meaning

Question:

How do I color a graph that has loops?

Definition:

A graph is k-*chromatic* if $\chi(G) = k$
A proper k-coloring of a k-chromatic graph is an *optimal coloring*
If $\chi(H) < \chi(G) = k$ for every proper subgraph H of G, then G is *color-critical* or k-*critical*

Question:

What do 1-critical and 2-critical graphs look like?

Note:

3-critical graphs are odd cycles (by the classification of bipartite graphs)

Definition:

The *clique number* of a graph G is $\omega(G)$ the maximum sizee of a set of pairwise adjacent vertices in G

Proposition:

For every graph G, $\chi(G) \geq \omega(G)$ and $\chi(G) \geq \frac{n(G)}{\alpha(G)}$

Proof:

Points in clique need different colors.
Color classes form independent sets.

Note:

In general, $\chi(G)$ may be strictly larger than $\omega(G)$, a 5-cycle is a perfectly good example

Note:

For graphs G and H

$\chi(G + H) = \max\{\chi(G), \chi(H)\}$
$\chi(G \vee H) = \chi(G) + \chi(H)$

Definition:

The *cartesian product* of G and H is the graph $G \square H$ with vertex set $V(G) \times V(H)$ with (u, v) adjacent to (u', v') iff either

$u = u'$ and $vv' \in E(H)$
$v = v'$ and $uu' \in E(G)$

Definition:

The m by n *grid* is the product $P_m \square P_n$

(draw) (one can remember this box symbol by thinking about $P_2 \square P_2$

Proposition: (Vizing 1963, Aberth 1964)

$\chi(G \square H) = \max\{\chi(G), \chi(H)\}$

Proof:

Since G and H are both contained as subgraphs, one has $\geq$ immediately

Let $k = \max\{\chi(G), \chi(H)\}$
Let f be a proper $\chi(G)$ coloring of G and g a proper $\chi(H)$-coloring of H

Define a coloring h on $G \square H$ by

$$h(u, v) = g(u) + h(v) \; mod \; k$$

Ideally, we'd like a good way to come up with bounds on the chromatic number of a graph
Naively, $\chi(G) \leq n(G)$, but we can do better

Algorithm: (Greedy Coloring)

The *greedy coloring* relative to a vertex ordering $v_1, \ldots, v_n$ of $V(G)$ iterates through the vertices, assigning each the label of the lowest color not equal to the color already assigned to a neighbor

Proposition:

$\chi(G) \leq \Delta(G) + 1$

Proof:

Each point has at most $\Delta(G)$ neighbors, at worst only those many colors will already be claimed.

We can improve greedy coloring a bit by picking a good order for vertices.

Proposition: (Welsh-Powell 1967)

If G has degree sequence $d_1 \geq \cdots \geq d_n$ then

$$\chi(G) \leq 1 + \max_i \min\{d_i, i - 1\}$$

Proof:

Order vertices in non-increasing order of degrees. Number of neighbors of v_i already colored is at most $\min\{d_i, i - 1\}$

Picking the right order is the name of the game - every graph has a vertex ordering where the greedy coloring will produce an optimal coloring.
There are some graphs where we can come up with better colorings than this greedy coloring strategy might have us do without being very lucky about the order we've chosen.

Example:

Computers stores variables in memory locations called *registers* in order to do arithmetic
These are quickly accessible locations in memory, but there are relatively few of them. Ideally, we'd like to assign variables that are never used at the same time to the same register, because we'll never need both at once
For any variable, we could in principle record the first and last time we use it, calling the intervening interval *active*

Make a graph whose vertices are the variables. Two vertices are adjacent if they are active at overlapping times.
Number of registers needed = chromatic number of the obtained graph

(draw a picture)

Definition:
An *interval representation* of a graph is a family of intervals assigned to the vertices such that the vertices are adjacent if and only if the corresponding intervals intersect.
A graph with such a representation is an *interval graph*

Proposition:
If G is an interval graph, then $\chi(G) = \omega(G)$

Proof:
Order vertices according to left endpoints of intervals, then greedy color.
Suppose x has label k
Then some $k-1$ preceding points have intervals overlapping the start of the interval for x
These k vertices form a k-clique

Note:
If you computationally implement the greedy coloring algorithm on a "random" graph, it will probably use about twice the minimum needed # of colors.
It can be *very very bad* if you use it on trees.

Lemma:
If H is a k-critical graph, then $\delta(H) \geq k-1$

Proof:
Let $x \in V(H)$
$H - x$ is $k-1$-colorable
If $d_H(x) < k-1$, then $N(x)$ does not use $k-1$ many colors, so we can use one of these remaining on k

Theorem: (Szekeres-Wilf 1968)
If G is a graph, then $\chi(G) \leq 1 + \max_{H \subseteq G} \delta(H)$

Proof:
Let $k = \chi(G)$
If $H' \subseteq G$ is k-critical, then
$$\chi(G) - 1 = \chi(H') - 1 \leq \delta(H') \leq \max_{(H \subseteq G)} \delta(H)$$

We can relate properties of colorings to those of orientations of a graph.

Example:
Every bipartite graph has an orientation with all edges going from partite set 1 to partite set 2
Longest path length is 1

Any orientation of an odd cycle must have two adjacent edges with the same orientation
Longest path length at least 2

Theorem: (Gallai-Roy-Vitaver Theorem (Gallai 1968) (Roy 1967) (Vitaver 1962))

If D is an orientation of G with longest path length $l(G)$, then

$$\chi(G) \leq 1 + l(D)$$

Moreover, there exists an orientation of G where this is an equality.

Proof:

Let D be an orientation, take D' a maximal subdigraph containing no cycle (can always take D' spanning)

Color $V(G)$ by letting $f(v)$ be 1 + (length of longest path in D' ending at v)

Let P a path in D' starting at u. Any path in D' ending at u cannot contain any other points in P - this would form a cycle

Thus f strictly increases along P

f uses colors 1 through $1 + l(D')$ on $V(G)$

For each edge $uv \in E(G)$, either uv or $vu \in E(D')$ or there is a path in one of those directions (maximality)

Thus $f(u) \neq f(v)$

Now, let f be an optimal coloring of G, we want to construct an orientation.

Define an orientation D by $uv \in E(D)$ iff $f(u) < f(v)$ and $uv \in E(G)$

No path in this orientation can be longer than #colors - 1

We had a bunch of bounds for colorability in the previous section. One of the simplest was $\chi(G) \leq 1 + \Delta(G)$. We know this bound is an equality for complete graphs and odd cycles.

Theorem: (Brook 1941)

If G is connected and is not complete or an odd cycle, then $\chi(G) \leq \Delta(G)$

Proof:

Take G connected with $k = \Delta(G)$

We may take $k \geq 3$ (all smaller graphs are trivial or do not satisfy assumptions)

(Idea: Order the vertices in a clever way, then use greedy coloring)

Case 1: G is not k-regular

Take a vertex $v_n \in V(G)$ with degree less than k

Grow a spanning tree of G from v_n, assign indices in decreasing order as they are reached

Every vertex has a higher ordered neighbor, so has at most $k - 1$ lower-indexed neighbors (we know Δ)

Greedy color

Case 2: G is k-regular, with some cut vertex x

$G - x$ has multiple components, let G' be one of them, with edges to x added back in

$d_{G'}(x) < k$, so previous method gives proper k-coloring of G'

Do this for every component, permute colors if necessary so that they agree on the color for x

Case 3: G is k-regular and 2-connected

Suppose v_n has two neighbors v_1, v_2 such that they are not adjacent and $G - \{v_1, v_2\}$ is connected

Index spanning tree of $G - \{v_1, v_2\}$ so that indices increase along paths to v_n, starting at index 3

We now have order $v_1, v_2, \ldots, v_n$

Each element in order has at most $k - 1$ lower-indexed neighbors, except last

But v_1, v_2 are assigned same color, so we get a k-coloring

Now we just need to argue that every 2-connected k-regular graph with $k \geq 3$ has such a choice of v_1, v_2, v_n

Take $x \in V(G)$

If $\kappa(G - x) \geq 2$, take $v_1 = x$, v_2 some vertex at distance 2 from x

(must exist otherwise everything connected to everything)

Take v_n a common neighbor

If $\kappa(G - x) = 1$, let $v_n = x$

$G - x$ can be divided into blocks, say a block is a *leaf block* if it only contains one cut-vertex of $G - x$ (equivalently if it is a leaf in the block-cutpoint graph, which is a tree)

x must have a neighbor in every such block, otherwise G would have a cut-vertex

There are at least two leaf blocks

Must be some neighbors v_1, v_2 of x in different blocks which are nonadjacent

(draw picture, this would probably help)

Since blocks have no cut-vertices $G - \{x, v_1, v_2\}$ is connected

Since $k \geq 3$, x has a third neighbor so $G - \{v_1, v_2\}$ is connected

There are a LOT of interesting variants of coloring problems that have uses in applications. One could color edges, one could examine "generalized colorings", one can allow only certain colorings on each value ["list colorings"], one can discuss colorings of hypergraphs, and on and on

Section 5.2 - Structure of k-Chromatic Graphs

Wednesday, April 12, 2023 2:53 PM

Definition:

We say that a graph G is *perfect* if $\chi(H) = \omega(H)$ for all induced subgraphs H of G

Our goal here is to demonstrate a way to produce a graph with a chromatic number $\chi(G)$ much larger than $\omega(G)$

Definition:

If G is simple, *Mycielski's construction* produces a simple graph G' containing G
Suppose $V(G) = \{v_1, \dots, v_n\}$
Add vertices $U = \{u_1, \dots, u_n\}$ and one additional vertex w
Add edges so that u_i is adjacent to all of $N_G(v_i)$ and let $N(w) = U$

Example:

Apply this construction to K_2 and then again to the resulting graph. Observe what seems to happen to the chromatic number

Theorem: (Mycielski 1955)

From a k-chromatic triangle-free graph G, Mycielski's construction produces a $k+1$-chromatic triangle-free graph G'

Proof:

Let $V(G) = \{v_1, \dots, v_n\}$ with copies $u_1, \dots, u_n$ and additional vertex w
U is an independent set of G'

Thus any triangle containing a point u_i must have *both* other points in $V(G)$
This necessarily gives us a a triangle entirely in G, which doesn't exist

If f is a proper k-coloring of G, set $f(u_i) = f(v_i)$ and $f(w) = k+1$ to get a proper $k+1$-coloring of G'

Now, suppose by way of contradiction f is a proper k-coloring of G'

WLOG $f(w) = k$
Thus $f(U) \subseteq \{1, \dots, k-1\}$
Let A be the color class in $V(G)$ corresponding to color k

For each $v_i \in A$, modify $f(v_i)$ to be equal to $f(u_i)$

Only place this could cause problems is on edges of form $v_i v'$ with $v_i \in A$ and $v' \in V(G) - A$
For any such edge, $u_i v'$ is also an edge, so no color issues arise.

Thus, f gives rise to a proper $k-1$-coloring of G, which does not exist by assumption.

Note:

If G is color-critical, Mycielski's construction gives a new color-critical graph.

Note:

The graphs you get by repeatedly applying this construction are not, in general, the smallest possible examples. The number of vertices here grows exponentially. It's enough for it to grow a bit faster than quadratically.

We now know that clique number and chromatic number can differ greatly.
Is there any other way for us to infer information about the structure of a k-chromatic graph?

Proposition:

Every k-chromatic graph with n vertices has at least k choose 2 edges.
Equality holds for a complete graph plus isolated vertices.

Proof:

For every pair of colors i, j there must be an edge between vertices of color i and those of color j - otherwise we could amalgamate the colors.

Definition:

A *complete multipartite graph* is a simple graph G whose vertices can be partitioned into sets so that u adjacent to v if and only if u, v belong to different partite sets.
Equivalently, iff every component of $\overline{G}$ is a complete graph.

When $k \geq 2$, write $K_{n_1,\ldots,n_k}$ as the complete k-partite graph with partite set sizes specified

Definition:

The *Turan graph* $T_{n,r}$ is the complete r-partite graph with n vertices where partite sets differ in size by at most 1
(describe based on pigeonhole principle)

Lemma:

Among simple r-partite graphs with n vertices, the Turan graph is the unique graph with the most edges.

Proof:

Need only consider complete r-partite graphs. The idea is simply to move a vertex from the largest partite set (size i) to the smallest partite set (size j), gaining $i - 1$ edges but losing j
This is positive iff all partite sets are within size 1 of one another

Theorem: (Turan 1941)

Among n-vertex simple graphs with no $r + 1$-clique, $T_{n,r}$ has the maximum number of edges.

Proof:

Certainly there cannot be an $r + 1$-clique in $T_{n,r}$ by r-colorability

Our lemma will give us the result if we can prove the maximum is achieved by an r-partite graph.
We claim:

if G has no $r + 1$-clique, then there is an r-partite graph H with the same vertex set as G and at least as many edges

We work by induction on r

$r = 1$, there aren't any edges.

Take $r > 1$

Suppose G has no $(r + 1)$-clique and has n vertices and $x \in V(G)$ has degree $k = \Delta(G)$

Take G' the induced subgraph on neighbors of x.

In G, x is adjacent to all points in G', so there can't be any r-cliques in G'

By induction hypothesis, there exists $r - 1$-partite H' with vertex set $N(x)$ with $e(H') \geq e(G')$

Let $S = V(G) - N(x)$ and H be the graph formed from H' by adding all possile edges between $N(x)$ and S

S is an independent set, so H is r-partite

Bounds on number of edges:

$$e(H) = e(H') + k(n-k)$$

$$e(G) \leq e(G') + \sum_{v \in S} d_G(v)$$

Corresponding terms in the first equation are larger (max degree is k)

Note:

The Turan graph is the unique extremal graph

~~~~~~

Characterizing color-critical graphs is another important task

Remark:

If G has no isolated vertices, G is color-critical if and only if for every $e \in E(G), \chi(G-e) < \chi(G)$

Proposition:

Let G be k-critical

For $v \in V(G)$, there is a proper k-coloring in which the color on v appears nowhere else, and the other $k-1$ colors all appear on $N(v)$

For $e \in E(G)$, every proper $k-1$-coloring of $G-e$ gives the same color to the two endpoints of e

Proof:

Remove the corresponding object and both proofs are trivial.

Recall that $\delta(G) \geq k-1$ if G is k-critical. We can actually use Koenig-Egervary to get $\kappa'(G) \geq k-1$ as well

Lemma: (Dirac 1953)

Let G have $\chi(G) > k$ and let X, Y be a partition of $V(G)$

If $G[X]$ and $G[Y]$ are k-colorable, then the edge cut $[X, Y]$ has at least k edges

Proof: (Dirac-Sorensen-Toft 1974, Kainen)

Let $X_1, \ldots, X_k$ and $Y_1, \ldots, Y_k$ be the color classes

If there's no edge between X_i and Y_j, then $X_i \cup Y_j$ is an independent set in G

(need to examine possible pairings)

Make a bipartite graph H with vertices $X_1, \ldots, X_k$ and $Y_1, \ldots, Y_k$ and edges connecting X_i and Y_j if there is *no edge in G between them*

If $|[X, Y]| < k$, H has at least $k(k-1)$ edges

m vertices can cover at most km edges in a subgraph of $K_{k,k}$, so $E(H)$ cannot be covered by $k-1$ vertices

Apply Koenig-Egervary to get a perfect matching

Assign a color to each independent set resulting from the matching

Theorem: (Dirac 1953)

Every k-critical graph is $k-1$-connected

Proof:
~~~~~~

G be k-critical, $[X, Y]$ a minimum edge cut, both are $k - 1$-colorable, so $|[X, Y]| \geq k - 1$

(lecture ended here)
There's a useful way to split up graphs that describes more of the relationship between chromatic number and size of vertex cuts.

Definition:

Let $S \subseteq V(G)$. An S-*lobe* of G is an induced subgraph of G whose vertex set consists of S and a component of $G - S$
(draw)

The union of all S-lobes of G is clearly G itself.

Proposition:

If G is k-critical, then G has no cutset consisting of pairwise adjacent vertices.
Notably, if $S = \{x, y\}$ is a cutset of G, then x is not adjacent to y and G has an S-lobe H with $\chi(H + xy) = k$

Proof:

Take S a cutset with lobes $H_1, \ldots, H_t$
Each H_i is $k - 1$-colorable
If all points of S are adjacent, for each H_i, the colors of all points in S must be distinct from one another
By permuting colors, can make the colors on each $v_j \in V(S)$ agree for all i
This gives a $k - 1$-coloring of G

~~~~~~

A topic that will soon be important - *forced subdivisions*

Definition:

An H-*subdivision* is a graph obtained from H by successive edge subdivisions.
(Equivalently, a graph obtained from H by replacing edges with pairwise internally-disjoint paths.

(draw)

Theorem: (Dirac 1952a)

Every graph with chromatic number at least 4 contains a K_4 subdivision.

Proof:

Induct on $n(4)$
$n = 4$ means our graph is K_4
$n > 4$
Let H be a 4-critical subgraph of G
H cannot have a cut-vertex (by prev proposition)
If $\kappa(H) = 2$ and we have a cutset $S = \{x, y\}$, then these points are not adjacent
There must be some S-lobe H' such that $\chi(H' + xy) \geq 4$
$n(H' + xy) = n(H') < n(H) \leq n(G)$, so apply IH
We have a K_4 subdivision F in $H' + xy$

If F doesn't contain xy, no problems
If it does, replace xy in F with an x, y-path through a different S-lobe of H
(One must exist because x, y both must have a neighbor in all components of
~~~~~~

$H - S$)

May now assume H is 3-connected.
Pick $x \in V(G)$, $H - x$ is 2-connected so there exists a cycle C having length at least 3.
By Fan-Lemma, there exists an x, C-fan of size 3
This is a subdivision of K_4

Section 5.3 - Enumerative Aspects

Monday, April 17, 2023 3:20 PM

There's another interesting topic about colorings which we haven't touched yet - counting them.
Namely, how many proper k-colorings of a graph are there?

Definition:

Given $k \in N$ and a graph G, the value $\chi(G;k)$ is the number of proper colorings $f: V(G) \to [k]$
The set of available colors is $[k] = \{1, \dots, k\}$, but we do *not* insist that the k colors all be used in a given coloring.
Permuting the colors of a given coloring is regarded as producing a different coloring.

Question:

What is $\chi(K_n;k)$?

What is $\chi\left(\overline{K}_n;k\right)$?

Proposition:

If T is a tree with n vertices, then $\chi(T;k) = k(k-1)^{n-1}$

Proof:

Fix a vertex $v \in T$ as the root. It can be colored arbitrarily.
Extending the proper coloring on from there in order of the tree, at each step we can assign $(k-1)$ colors to each newly reached vertex

We are starting to notice a pattern here - we keep getting polynomials of degree n

Proposition:

Let $x_{(r)} = x(x-1)\cdots(x-r+1)$. If $p_r(G)$ is the number of partitions (order does not matter!) of $V(G)$ into r nonempty independent sets, then

$$\chi(G;k) = \sum_{r=1}^{n(G)} p_r(G) k_{(r)}$$

which is a polynomial in k of degree $n(G)$

Proof:

If a given coloring uses exactly r colors, it partitions $V(G)$ into r nonempty independent sets.
This can happen in $p_r(G)$ ways, by definition
If k colors are available but I'm only using r of them, there are $k_{(r)}$ ways to pick which colors are used and in which order.

Corollary:

$\chi(G;k)$ is monic

Proof:

$p_n(G) = 1$

Example:

Consider $G = C_4$ (good luck with this computation...)

Note:

Unless the graph has no edges, $p_1(G) = 0$

This is usually a horrifically complex way to compute the chromatic polynomial.

[remember that $G \cdot e$ is the graph contracted along the edge e - also note that multiple edges don't affect colorings in any way, so WLOG we may think only of simple graphs]

Theorem: (Chromatic Recurrence)

If G is a simple graph and $e \in E(G)$, then $\chi(G;k) = \chi(G-e;k) - \chi(G \cdot e;k)$

Proof:

Every proper k-coloring of G is a proper k-coloring of $G - e$

Proper k-colorings of $G - e$ are proper k-colorings of G if and only if they assign different colors to endpoints of e

If it assigns the same color to endpoints of e, it will correspond to a proper k-coloring of $G \cdot e$

Example:

Compute $\chi(G;k)$ for C_4 again, using previous propositions about trees and complete graphs.

This past proposition is a lot like the recursive one we had for computing the number of spanning trees. It is about as useful. Characterizing the chromatic polynomial more explicitly is an important (and fairly tricky!) topic.

Theorem: (Whitney 1933)

The chromatic polynomial $\chi(G;k)$ of a simple graph has degree $n(G)$, with integer coefficients alternating in sign and is of the form $k^n - e(G)k^{n-1} + \cdots$

Proof:

We induct on $e(G)$. This follows from the empty graph case if $e(G) = 0$

Suppose G has n vertices.

$G - e$ and $G \cdot e$ have fewer edges, so the inductive case applies to each

$$\chi(G-e;k) - \chi(G \cdot e;k) = \chi(G;k)$$

$$\begin{aligned} &k^n - [e(G)-1]k^{n-1} + a_2 k^{n-2} - \cdots \\ &-\left(k^{n-1} - b_1 k^{n-2} + b_2 k^{n-3} - \cdots\right) \\ &= k^n - e(G)k^{n-1} + \cdots \end{aligned}$$

If you want a technically complete formula, this one is the result of repeatedly using the recursive definition ad nauseum. It is *totally impractical* to actually use, since it's a summation with exponentially many terms.

Theorem: (Whitney 1932)

Let $c(G)$ denote the number of components of a graph G

Given a set $S \subseteq E(G)$, let $G(S)$ denote the spanning subgraph with edge set S

Then

$$\chi(G;k) = \sum_{S \subseteq E(G)} (-1)^{|S|} k^{c(G(S))}$$

I won't even do the proof of this, because it's essentially a useless formula.

~~~~~~

I think I will skip the rest of this section, so as to spend more time on subsequent material.
~~~~~~

Section 6.1 - Embeddings and Euler's Formula

Wednesday, April 19, 2023 2:50 PM

Note:

During this whole semester, we've drawn graphs on the board or on paper. Doing so is necessarily (and ideally inconsequentially!) embedding our graphs in the plane. However, sometimes when we do so edges of our graphs cross each other. This makes the graph much more difficult to read, and also in some intuitive sense makes the drawing of the graph seem less natural.

Proposition:

K_5 and $K_{3,3}$ cannot be drawn without crossings.

(**Definition:** A *chord* of a path or cycle C is an edge whose endpoints both lie in C

Proof:

Draw either graph in the plane and let C be a spanning cycle. If no edges cross, other edges are either inside or outside this cycle.

Say that two chords *conflict* if their endpoints on C occur in alternating order - two such chords cannot both be internal or external

Both K_5 and $K_{3,3}$ have 3 pairwise conflicting chords for a spanning cycle

Ok, so this is a start point for the idea of drawing graphs in the plane - it isn't always possible. But to get at this more systematically, we need a clearer definition of what 'drawing' is

Definition:

A *curve* is the image of a continuous function $f: [0,1] \to R^2$

A *polygonal curve* is a curve consisting of finitely many line segments

A u, v-curve starts and ends at u and v, respectively

A curve is *closed* if its first and last points coincide, and *simple* if it has no repeated points save possibly the first and last

A *drawing* of a graph G is a function f with domain $V(G) \cup E(G)$ which assigns each vertex to a point

$$f(v) \in R^2$$

and assigns each edge with endpoints u, v to a polygonal $f(u), f(v)$-curve

The images of vertices must be distinct.

A point of $f(e) \cap f(e')$ that is not a common endpoint for distinct edges e, e' is a *crossing*

(I won't prove this, but we can assume WLOG that three way crossings never happen, by slightly perturbing drawings)

These definitions allow us to specify graphs which can be drawn nicely.

Definition:

A graph is *planar* if it has a drawing without crossings

Such a drawing is a *planar embedding* of G

A *plane graph* is a particular planar embedding of a planar graph

Key to this theory will be thinking about the regions enclosed by drawings of graphs.

Definition:

An *open set* in the plane is a set $U \subseteq R^2$ such that $\forall p \in U$, there is a small ball around p contained wholly in U

A *region* is an open set U containing a polygonal u, v-curve for every $u, v \in U$

The *faces* of a plane graph are the maximal regions of the plane that contain no point used in the embedding.

Note that every graph we draw has a bunch of internal faces and one enormous unbounded face

Theorem: (Jordan Curve Theorem - special case)
A simple closed polygonal curve C consisting of finitely many segments partitions the plane into exactly two faces, each having C as boundary.

Proof: (skip)

Definition:
Given a plane graph G, the *dual graph* G^* is a plane graph whose vertices correspond to the faces of G
(Edges of G^* connect faces of G which are separated by an edge of G.)
In particular, for each edge e of G which borders on faces U, V, we obtain a *dual edge* $e^* \in E(G^*)$ connecting U, V

(draw)

Example:
Draw the dual graph of an embedding of the cube (this should be another Euler solid)

Note:
For any planar graph $(G^*)^*$ is (naturally) isomorphic to G
I won't prove this because I would have to be more careful about how I'm defining the embedding of the dual graph.

Note:
Dual graphs may have multiple edges
(draw)

Note:
Two embeddings of a planar graph may have nonisomorphic duals (example in the book)

This doesn't happen if the graph is 3-connected

Definition:
The *length* of a face in a plane graph G is the total length of the closed walk(s) in G bounding the face

(there's only one walk if the graph is connected)

Proposition:

If $l(F_i)$ is the length of face F_i in a plane graph G, then $2e(G) = \sum l(F_i)$

Proof:

$e(G) = e(G^*)$ and the length of a face is the degree in G^* of that face

Sum up degrees in dual graph.

Thinking about planar graphs and their duals presents a nice little variation of coloring problems. Coloring nonadjacent vertices of G^* amounts to coloring nonadjacent faces of G. As such, we can talk about coloring maps in terms of colorings of duals of planar graphs!

And coloring the duals of planar graphs is an identical problem to coloring planar graphs (since every such graph is the dual of its dual)

Theorem: (Four Color Theorem)

Every planar graph admits a proper 4-coloring.

This proof is a *little bit tricky*

We can think a bit more concretely about the duality relationship here, though

Many of the concepts which we've talked about in class have a really nice duality relationship for planar graphs (this does not extend to non-planar graphs!)

Theorem:

Edges in a plane graph G form a cycle if and only if the corresponding dual edges form a bond (minimal edge cut) in G^*

Proof:

Take $D \subseteq E(G)$

If D does not contain a cycle, it does not enclose a region

Then $G^* - D^*$ is connected (there's a way to get from any face to any other)

Thus D^* does not contain an edge cut

If D makes up the edges of a cycle, by Jordan curve theorem this encloses a region

Corresponding edge set D^* contains all dual edges between this region and outside (again, by JCT)

Thus D^* contains an edge cut

If D contains a cycle, then D^* contains an edge cut

Hence cycles in $E(G)$ are minimal edge cuts in $E(G^*)$

We can say even more!

Theorem:

For a plane graph G, the following are equivalent

G is bipartite

Every face of G has even length

The dual graph G^* is Eulerian

Proof:

B and C are equivalent by our standard characterization theorem for Eulerian graphs, just applied to G^*

A → B

A face boundary consists of closed walks. Odd closed walks contain odd cycles, none in bipartite

B → A

Let C a cycle in G, it forms a simple closed curve enclosing a region F
Every face of G is entirely in F or entirely out of F
Sum face lengths for all faces inside F, get an even number

This sum counts each edge in C once, and each edge entirely contained in F twice
Hence all cycles have even length.

For planar graphs, we can establish a relation between the number of vertices, edges, and faces

Theorem: (Euler 1758)
If a connected plane graph G has exactly n vertices, e edges, and f faces, then
$n - e + f = 2$

Proof:
We induct on n.
$n = 1$, then G consists only of loops. Each is a non-intersecting closed curve, and there's an outer region plus 1 for each edge.
$n > 1$
G is connected, so it has an edge which isn't a loop. Select such an edge, e
Contract G along e to get a new plane graph with n', e', f'
$n' = n - 1$
$e' = e - 1$
$f' = f$
Apply inductive hypothesis

Note:
If G has k components, $n - e + f = k + 1$ instead (this is effectively just a corollary of the main theorem)

Theorem:
If G is a simple planar graph with at least three vertices, then $e(G) \leq 3n(G) - 6$
If G is triangle-free, then $e(G) \leq 2n(G) - 4$

Proof:
WLOG by adding edges, may assume G is connected

Since G is simple, there are no loops and no multiple edges. With at least three vertices, so every face is bordered by at least 3 edges.
$2e = \sum l(F_i) \geq 3f$
Put into Euler's formula

If triangle-free, all faces have length at least 4, do the same thing

Corollary:
K_5 and $K_{3,3}$ are nonplanar

Proof:
For K_5, $n = 5, e = 10$, violates previous

For $K_{3,3}$, no triangles and $n = 6, e = 9$

Definition:
A *maximal planar graph* is a simple planar graph that is not a spaning subgraph of another planar graph.

A *triangulation* is a simple plane graph where every face boundary is a 3-cycle

Proposition:

For a simple n-vertex plane graph G, TFAE

G has $3n - 6$ edges

G is a triangulation

G is a maximal plane graph

Proof:

First two are equivalent by noting that G is a triangulation iff $l(F_i) = 3$ for all faces, so $2e = 3f$, thus equality

Second two are equivalent by noting that we can add an edge to split up a face iff the face is not bounded by a triangle (any way to split it is a chord of the cycle)

Section 6.2 - Characterization of Planar Graphs

Wednesday, April 19, 2023 4:56 PM

Motivating Question:

Which graphs are and are not planar?

Proposition:

If G has a subgraph that is a subdivision of K_5 or $K_{3,3}$, then G is nonplanar

Proof:

Every subgraph of a planar subgraph is planar, so if G has any nonplanar subsets it is not planar
WLOG may take G a subdivision of K_5 or $K_{3,3}$

If G was planar, we could contract its edges to obtain a planar embedding of K_5 or $K_{3,3}$

Theorem: (Kuratowski 1930)

A graph is planar if and only if it does not contain a subdivision of K_5 or $K_{3,3}$

This proof is a bit longer, so we'll break the proof up into lemmas in and of itself.

Definition:

A *Kuratowski subgraph* of G is a subgraph of G that is a subdivision of K_5 or $K_{3,3}$
A *minimal nonplanar graph* is a nonplanar graph such that every proper subgraph is planar

Goal 1: Show that a smallest possible nonplanar graph with no Kuratowski subgraph is 3-connected

Lemma:

If F is the edge set of the face of a planar embedding of G, then G has an embedding with F the edge set of the unbounded face.

Proof:

Embed on the sphere instead, then rotate desired face

Lemma:

Every minimal nonplanar graph is 2-connected

Proof:

Let G be a minimal nonplanar graph.
If G disconnected, embed one component inside one face of the rest (contradiction)

If G has a cut vertex v, let $G_1, \dots, G_k$ be $\{v\}$-lobes of G
Each is planar
Each can be embedded with v on the outside face
Squeeze embeddings of each into sectors of the plane centered on v to embed whole graph (contradiction)

Lemma:

Let $S = \{x, y\}$ be a separating 2-set of G. If G is nonplanar, adding xy some S-lobe of G yields a nonplanar graph

Proof:

Let $G_1, \ldots, G_k$ be S-lobes of G, let $H_i = G_i \cup xy$
If all H_i are planar, can embed all with xy on outside face
Then, starting with H_1, may embed H_i inside a single face with boundary containing xy
Then, delete xy if G doesn't contain it.

Lemma:
If G is a graph with fewest edges among all nonplanar graphs without Kuratowski subgraphs, then G is 3-connected

Proof:
G must be a minimal nonplanar graph, so G is 2-connected. Let $S = \{x, y\}$ a separating set
Union of xy with some S-lobe H is nonplanar.
By minimality, $H \cup xy$ must have a Kuratowski subgraph F.
All of F is in G, except possibly xy

Take an x, y-path through some other S-lobe
We obtain a Kuratowski subgraph of G

Hence G is 3-connected

Goal 2: Show that every 3-connected graph with no Kuratowski subgraph is planar.

Our strategy is going to be inductive, but we need a bit of groundwork to establish the technique we're going to use to reduce our graphs in size. The operation will be a contraction of a well chosen edge.

Fact: (don't bother to prove this one, or leave it as an exercise)
Every 3-connected graph G with at least 5 vertices has an edge e such that $G \cdot e$ is 3-connected

We need to argue that this operation doesn't create any Kuratowski subgraphs.

Lemma:
If G has no Kuratowski subgraph, then neither does $G \cdot e$

Useful term for this proof:
A *branch vertex* of a subdivision H' of H is a vertex of degree at least 3 in H'
(they don't correspond to the interior of created paths in the subdivision)

Proof:
We wish to show that if $G \cdot e$ has a Kuratowski subgraph, then so does G

Let $e = xy$ and let $z \in G \cdot e$ be the contracted vertex.
If Kuratowski subgraph H doesn't contain z, we're done.
If $z \in V(H)$ but isn't a branch vertex:
we get a Kuratowski subgraph of G by taking H and replacing z with x or y or e

If $z \in V(H)$ is a branch vertex but at most one edge incident on z in H is incident to x in G:
Expand z into xy

If $z \in V(H)$ is a branch vertex with exactly 4 edges in H incident on z, 2 of which are incident on x in G
H is a subdivision of K_5 (only way to get 4 incident)
Let u_1, u_2 be the next branch vertices in H reached by following the paths from the edges incident on x
Let v_1, v_2 be similarly defined for y
There are only 5 branch vertices, so these are all of them. (draw)

Draw corresponding subgraph with z replaced by xy, delete u_1v_1-path and u_2v_2-path
This is a $K_{3,3}$ subdivision

We're almost ready. We'll actually prove something better in the process.

Definition:

A *convex embedding* of a graph is a planar embedding in which each face is a convex region.

Theorem: (Tutte 1960)

If G is a 3-connected graph with no subdivision of K_5 or $K_{3,3}$, then G has a convex embedding in the plane with no three vertices on a line.

Proof: (Thomassen 1980)

Induction on $n(G)$

$n = 4$, the only possible graph is K_4, which has a convex embedding. (a triangle with a point in the middle)

$n(G) \geq 5$

There is an edge e such that $G \cdot e$ is 3-connected, contracting it gives a vertex z
$H = G \cdot e$ has no Kuratowski subgraph, so H has a convex embedding with no three vertices on a line by induction.

If one deletes all edges incident on z (but not z) in this embedding, some face contains z (might be unbounded face)
$H - z$ is 2-connected, so boundary of this face is a cycle C, which contains all neighbors of z

each is a neighbor of x or y in the original graph [replace z by xy

In cyclic order in C

Let $x_1, \ldots, x_k$ be neighbors of x

If all neighbors of y lie between x_i and x_{i+1} (inclusive), we're done [draw picture with y placed in the face]

If this doesn't happen, two possibilities:

1) y has three neighbors u, v, w which are all neighbors of x
2) y has neighbors u, v such that x_i, u, x_{i+1}, v are in cyclic order on C

Case 1 - C along with edges from x, y to u, v, w forms a K_5 subdivision
Case 2 - C along with paths uyv, x_ixx_{i+1}, xy forms a $K_{3,3}$ subdivision
(draw both of these)

We're done.

Final Projects

Wednesday, May 3, 2023 5:34 PM

5% from topic proposals
50% of grade from presentations
45% of grade from paper

Presentation Rubric

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent, with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions, little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

Final Paper Rubric

	5 points	4 points	3 points	2 points	1 point	0 points
Paper Length	10-20 pages		Length has been padded by writing up trivialities or excessive examples	Length has been padded with pictures and formatting	Not long enough	No paper
Bibliography	Sufficient sources, reasonable formatting	Sufficient sources, no formatting of any reasonable kind	Sufficient sources, but too heavy a reliance on a single source	Bad sources used without any formatting	Insufficient sources	None
Grammar and Spelling	Fine	Mostly insignificant errors		Grammatical errors mildly impair understanding	Severe grammatical errors impact comprehensibility	Work is not written in full sentences.
Clarity of Exposition (counts twice)	Writing is exceptionally clear and understandable, at a level readable by a student in class	At most one significant exception to the previous	Writing is in some places is too dense or assumes too much of the reader.	Writing is in many places too dense or assumes too much of the reader. OR assumes too little of the reader		Incomprehensible
Explanations in Own Words	Yes					No
Mathematical Content (counts twice)	Paper develops in depth ideas of mathematical or applied interest, related to graph theory	Reasonable depth of discussion, but weak connection to graph theory	Less depth of discussion, well-explored connection to graph theory	Less depth of discussion, weak connection to graph theory	Discussion is surface level, but connected to graph theory	Discussion is surface level, weak connection to graph theory
Mathematical Validity	Statements in paper are mathematically valid, free from even minor typos, and formatted well	Minor typos or minor bad formatting decisions with respect to mathematics have occurred	Some statements are not mathematically valid or are not stated sufficiently precisely	Key portions of the paper are not mathematically valid or not stated sufficiently precisely OR many significant typos or bad formatting decisions	The previous, but with AND	Paper is so vague that virtually no statements are mathematically meaningful

List of Students:

QI QI (+1 point for early practice)
Questions to ask:

1. You've given us an interesting characterization of bipartite graphs. Can you say anything about k -partite graphs?
2. Does the largest eigenvalue of a graph tell us anything about a graph? How large can it be?
3. Can you tell us more about the relationship between this Green's function and the graph Laplacian?

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent, with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions, little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

Yutian Wang
Questions to Ask

1. You've defined ergodicity as the existence of this limiting state. Can you say anything about the rate of convergence to that state?
2. Suppose I take a Markov chain with a single recurrence stage, but a period greater than 1. Is there anything you can do to it to understand its behavior in a way similar to what you've done here?
3. Suppose the Markov matrix is symmetric. Can I say anything more about my Markov chain?

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent

Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent, with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions, little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

Aaron Weiner
Questions to ask

1. Your algorithm finds the shortest path among those with the fewest number of transfers. Suppose I wanted to strike a balance between minimizing travelled distance and minimizing number of transfers, is there any way to take these structures you've defined and use them to handle this optimization?
2. You've discussed my way of finding an efficient route through a network as a traveler. As a designer of a network, is there a ny way for us to use this paradigm to talk about how efficient the network is as a whole?

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent, with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions, little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

William Hagerstrom
Questions to ask

1. Suppose I have a game in which it is possible for a move to take me back to an earlier state of the game (the game tree is no longer a tree). Can I modify this algorithm to still work in such a setting?
2. How do we evaluate the quality of leaves when we've reached max iteration depth?
3. How can I guarantee that alpha-beta pruning will provide a speed-up over minimax? (If I evaluate "bad strategies" in the game tree first, alpha-beta pruning won't help much.)

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent, with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions, little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

			successful		talk	knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

Vuong Ho
Questions to Ask
1. Suppose I consider a complete graph on 4 vertices with the diagonals having large weights. Can you talk us through specifically how an ant colony algorithm would work and how it would decide not to pick the diagonals?
2. The ants build solutions out of "solution components". How are these "components" related to the structure of a graph?
3. Many of these algorithms seem to require careful tuning of parameters for effective performance. Before running such a computationally intensive algorithm, how can we be sure we've chosen these randomly?

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers

Zeyu Nie
Questions to ask
1. Are there any improvements that can be made to the Christafides algorithm to get better than a 3/2 -approximation?
2. Can you talk us through the Held-Karp algorithm applied to a complete graph on four vertices with diagonals having large weights?
3. The initial step of your discussion takes us from a general graph to a complete graph by adding edges with large weights. However, this removes some of the structure of the graph that might be useful - some graphs might have bottlenecks which limit the possibilities we have to search through. Are there any common algorithms for TSP that make use of detailed information about the structure of the underlying graph?

	5 points	4 points	3 points	2 points	1 point	0 points
Presentation Fits Parameters of Assignment	12-18 minutes		Too long, cut off		Too short	No presentation
Practice Presentation	Fully prepared	mostly prepared	somewhat prepared	unprepared		no practice presentation
Quality of Presentation	Engaging, well spoken, clear familiarity with material	Mostly well spoken, well demonstrated understanding	Mostly familiar with material, not engaging with audience	Seems unfamiliar with material or reading from a script	Seems unfamiliar with material and reading from a script	Seems unfamiliar with material and unprepared
Use of Slides/Chalkboard	Effective use of visual aids to reinforce material in presentation	Use of visual aids, but limited communication of what they convey to the audience	Visual aids used, but not adequately explained	Some purely verbal explanations making no or limited use of visual aids	Many purely verbal explanations making no or limited use of visual aids	No or completely ineffective use of slides or chalkboard
Mathematical Content (this category counts twice in score)	Good mathematical content, relevant to graph theory, clearly explained in some depth	Good mathematical content, mostly relevant to graph theory, mostly explained in some depth	Some mathematical content, somewhat related to graph theory, mostly explained	Some mathematical content, but either not related to graph theory or not explained in much detail or only shallowly explored	Some mathematical content, but very shallow and vague discussions, seldom touching upon ideas from graph theory	Mathematical content either missing or so vague so as to be absent
Clarity of Explanation (this category counts twice in score)	Presentation stands on own right as effective explanation of ideas	Supplemented with questions, presentation is effective explanation of ideas Understandable to peers	Presentation relies on some unexplained background, but mostly stands on own Mostly understandable to peers	Heavy reliance on background material in order to make presentation understandable Not understandable to peers		Presentation is incoherent with even background knowledge being insufficient to understand material
Response to Questions	Responds to questions with insight, demonstrates understanding	Response to questions demonstrates some understanding	Some unclear or muddled explanations fail to demonstrate understanding, others more successful		Student actively confused by questions little thought seems to have gone into the topic aside from the narrow confines of the talk	Student did not provide answers to any questions and fails to demonstrate any external knowledge of their material
Asked Questions of Others	Student asked at least one relevant question to one of their peers		Student asked a somewhat relevant question to one of their peers			No or completely irrelevant questions asked of peers