Left over

Finishing stable matching...

Good for jobs? candidates?

Is the Job-Proposes better for jobs? for candidates?

Definition: A **matching is** x**-optimal** if $x's$ partner is its best partner in any stable pairing.

Definition: A **matching is** x**-pessimal** if $x's$ partner is its worst partner in any stable pairing.

Definition: A **matching is job optimal** if it is x-optimal for **all** jobs x.

..and so on for job pessimal, candidate optimal, candidate pessimal.

Claim: The optimal partner for a job must be first in its preference list.

True? False? False!

Subtlety here: Best partner in any stable matching.
As well as you can be in a globally stable solution!

Question: Is there a job or candidate optimal matching?
Is it possible:
b-optimal pairing different from the b'-optimal matching!
Yes? No?

Understanding Optimality: by example.

A: 1,2 1: A,B
B: 1,2 2: B,A

Consider pairing: $(A,1),(B,2)$.

Stable? Yes.

Optimal for B?
Notice: only one stable pairing.
So this is the best B can do in a stable pairing.
So optimal for B.

Also optimal for A, 1 and 2. Also pessimal for A,B,1 and 2.

A: 1,2 1: B,A
B: 2,1 2: A,B

Pairing S: $(A,1),(B,2)$. Stable? Yes.

Pairing T: $(A,2),(B,1)$. Also Stable.

Which is optimal for A? S Which is optimal for B? S
Which is optimal for 1? T Which is optimal for 2? T

Pessimality?

Job Propose and Candidate Reject is optimal?

For jobs? For candidates?

Theorem: Job Propose and Reject produces a job-optimal pairing.

Proof:
Assume not: there is a job b does not get optimal candidate, g.

There is a stable pairing S where b and g are paired.

Let t be first day job b gets rejected
by its optimal candidate g who it is paired with
in stable pairing S.

b^* - knocks b off of g's string on day t $\implies$ g prefers b^* to b

By choice of t, b^* likes g at least as much as optimal candidate.

$\implies$ b^* prefers g to its partner g^* in S.

Rogue couple for S.
So S is not a stable pairing. Contradiction. □

Notes: S - stable. $(b^*,g^*) \in S$. But (b^*,g) is rogue couple!

Used Well-Ordering principle...Induction.

How about for candidates?

Theorem: Job Propose and Reject produces candidate-pessimal pairing.

T – pairing produced by JPR.

S – worse stable pairing for candidate g.

In T, (g,b) is pair.

In S, (g,b^*) is pair.

g prefers b to b^*.

T is job optimal, so b prefers g to its partner in S.

(g,b) is Rogue couple for S

S is not stable.

Contradiction. □

Notes: Not really induction.
Structural statement: Job optimality $\implies$ Candidate pessimality.

Quick Questions.

How does one make it better for candidates?

Propose and Reject - stable matching algorithm. One side proposes.
Jobs Propose $\implies$ job optimal.
Candidates propose. $\implies$ optimal for candidates.

Residency Matching..

The method was used to match residents to hospitals.

Hospital optimal....

..until 1990's...Resident optimal.

Another variation: couples.

Takeaways.

Analysis of cool algorithm with interesting goal: stability.

"Economic": different utilities.

Definition of optimality: best utility in stable world.

Action gives better results for individuals but gives instability.

Induction over steps of algorithm.

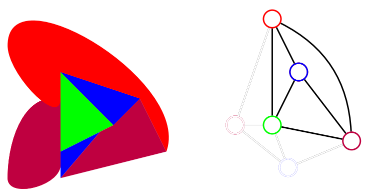
Proofs carefully use definition:
  Stability:
    Improvement Lemma plus every day the job gets to choose.
  Optimality proof:
    Job Optimality:
      contradiction of the existence of a better *stable* pairing.
        that is, no rogue couple by improvement, job choice, and well ordering principle.   Candidate Pessimality:
      contradiction plus cuz job optimality implies better pairing.

Life Lesson: ask, you will do better even if rejection is hard.

Lecture 5: Graphs.

Graphs!
  Definitions: model.
  Fact!

Map Coloring.



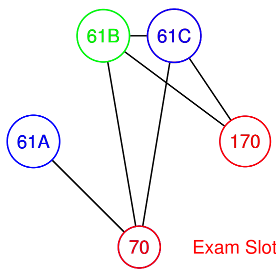
Four colors required!

Theorem: Four colors enough for maps on the plane.

Fewer Colors?

Yes! Three colors.

Scheduling: coloring.



Exam Slot 1.

Exam Slot 2.

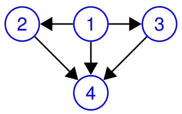
Exam Slot 3.

Graphs: formally.

Graph: $G = (V, E)$.
  V - set of vertices.
    $\{A, B, C, D\}$
  $E \subseteq V \times V$ - set of edges.
    $\{\{A,B\},\{A,B\},\{A,C\},\{A,C\},\{B,D\},\{A,D\},\{C,D\}\}$.

For CS 70, usually simple graphs.
  No parallel edges.
Multigraph above.

Directed Graphs



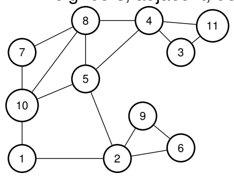
$G = (V, E)$.
V - set of vertices.
$\{1,2,3,4\}$
E ordered pairs of vertices.
$\{(1,2),(1,3),(1,4),(2,4),(3,4)\}$

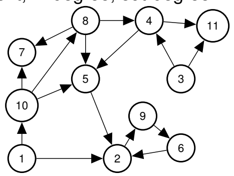
One way streets.
Tournament: 1 beats 2, ...
Precedence: 1 is before 2, ..

Social Network: Directed? Undirected?
Friends. Undirected.
Likes. Directed.

Graph Concepts and Definitions.

Graph: $G = (V, E)$
neighbors, adjacent, degree, incident, in-degree, out-degree



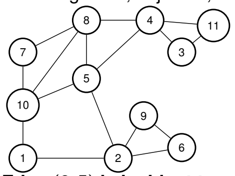


Neighbors of 10? 1,5,7, 8.
u is neighbor of v if $\{u, v\} \in E$.
Edge $\{10, 5\}$ is incident to vertex 10 and vertex 5.
Edge $\{u, v\}$ is incident to u and v.
Degree of vertex 1? 2
Degree of vertex u is number of incident edges.
Equals number of neighbors in simple graph.

Directed graph?
In-degree of 10? 1 Out-degree of 10? 3

Graph Concepts and Definitions.

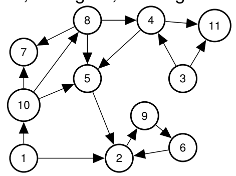
Graph: $G = (V, E)$
neighbors, adjacent, degree, incident, in-degree, out-degree



Edge $(8, 5)$ is incident to:

(A) Vertex 8.
(B) Vertex 5.
(C) Edge $(8, 5)$.
(D) Edge $(8, 4)$.
(E) Vertex 10.

(A) and (B) are true.



The degree of a vertex is:
(A) The number of edges incident to it.
(B) The number of neighbors of v.
(C) Is the number of vertices in its connected component.

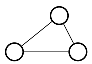
(A) and (B) are true.

Sum of degrees?

The sum of the vertex degrees is equal to

(A) the total number of vertices, $|V|$.
(B) the total number of edges, $|E|$.
(C) What?

(A) and (B) are false. (C) is a fine response to a poll with no correct answers.



Not (A)! Triangle.
Not (B)! Triangle.

What? For triangle number of edges is 3, the sum of degrees is 6.

Could sum always be...

(A) $2|E|$? ..
(B) $2|V|$?
(A) is true.

Quick Proof of an Equality.

The sum of the vertex degrees is equal to ??

Recall:

edge, (u, v), is incident to endpoints, u and v.

degree of u number of edges incident to u

Let's count incidences in two ways.

How many incidences does each edge contribute? 2.

Total Incidences? $|E|$ edges, 2 each. $\rightarrow 2|E|$

What is degree v? Incidences corresponding to v!

Total Incidences? The sum over vertices of degrees!

Thm: Sum of vertex degress is $2|E|$.

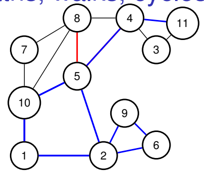
Poll: Proof of “handshake” lemma.

What's true?

(A) The number of edge-vertex incidences for an edge e is 2.
(B) The total number of edge-vertex incidences is $|V|$.
(C) The total number of edge-vertex incidences is $2|E|$.
(D) The number of edge-vertex incidences for a vertex v is its degree.
(E) The sum of degrees is $2|E|$.
(F) The total number of edge-vertex incidences is the sum of the degrees.

(A),(C), (D), (E), and (F).

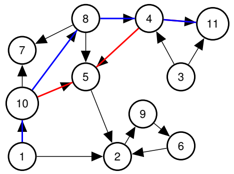
Paths, walks, cycles, tour.



A path in a graph is a sequence of edges.
Path? $\{1,10\}$, $\{8,5\}$, $\{4,5\}$? No!
Path? $\{1,10\}$, $\{10,5\}$, $\{5,4\}$, $\{4,11\}$? Yes!
Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Quick Check! Length of path? k vertices or $k-1$ edges.
Cycle: Path from v_1 to v_{k-1}, + edge (v_{k-1}, v_1) Length of cycle? $k-1$ vertices and edges!
Path is usually simple. No repeated vertex!
Walk is sequence of edges with possible repeated vertex or edge.
Tour is walk that starts and ends at the same node.

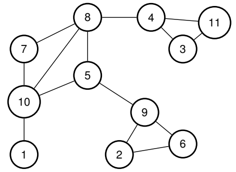
Quick Check!
Path is to Walk as Cycle is to ?? Tour!

Directed Paths.



Path: $(v_1, v_2), (v_2, v_3), \ldots (v_{k-1}, v_k)$.
Paths, walks, cycles, tours ... are analagous to undirected now.

Connectivity: undirected graph.



u and v are connected if there is a path between u and v.

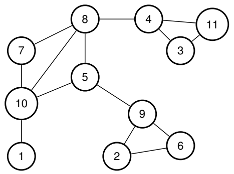
A connected graph is a graph where all pairs of vertices are connected.

If one vertex x is connected to every other vertex.
Is graph connected? Yes? No?

Proof: Use path from u to x and then from x to v. □

May not be simple!
Either modify definition to walk.
Or cut out cycles. .

Connected Components: Quiz.



Is graph above connected? Yes!

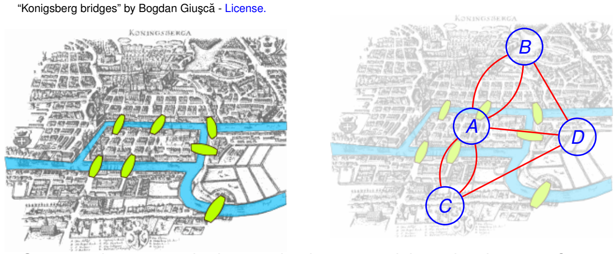
How about now? No!

Connected Components? $\{1\}, \{10,7,5,8,4,3,11\}, \{2,9,6\}$.
Connected component - maximal set of connected vertices.
Quick Check: Is $\{10,7,5\}$ a connected component? No.

Konigsberg bridges problem.

Can you make a tour visiting each bridge exactly once?

"Konigsberg bridges" by Bogdan Giuşcă - License.



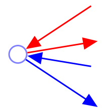
Can you draw a tour in the graph where you visit each edge once?
Yes? No?
We will see!

Eulerian Tour

An Eulerian Tour is a tour that visits each edge exactly once.

Theorem: Any undirected graph has an Eulerian tour if and only if all vertices have even degree and is connected.
Proof of only if: Eulerian $\implies$ connected and all even degree.

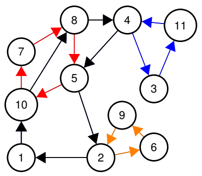
Eulerian Tour is connected so graph is connected.
Tour enters and leaves vertex v on each visit.
Uses two incident edges per visit. Tour uses all incident edges.
Therefore v has even degree. □



When you enter, you can leave.
For starting node, tour leaves first ....then enters at end.
Not The Hotel California.

Finding a tour!

Proof of if: Even + connected $\implies$ Eulerian Tour.
We will give an algorithm. First by picture.



1. Take a walk starting from v (1) on "unused" edges
... till you get back to v.
2. Remove tour, C.
3. Let $G_1, \ldots, G_k$ be connected components.
   Each is touched by C.
      Why? G was connected.
   Let v_i be (first) node in G_i touched by C.
    Example: $v_1 = 1$, $v_2 = 10$, $v_3 = 4$, $v_4 = 2$.
4. Recurse on $G_1, \ldots, G_k$ starting from v_i
5. Splice together.
   1,10,7,8,5,10 ,8,4,3,11,4 5,2,6,9,2 and to 1!

Recursive/Inductive Algorithm.

1. Take a walk from arbitrary node v, until you get back to v.

Claim: Do get back to v!
Proof of Claim: Even degree. If enter, can leave except for v. □

2. Remove cycle, C, from G.
Resulting graph may be disconnected. (Removed edges!)
Let components be $G_1, \ldots, G_k$.
Let v_i be first vertex of C that is in G_i.
  Why is there a v_i in C?
    G was connected $\implies$
      a vertex in G_i must be incident to a removed edge in C.

Claim: Each vertex in each G_i has even degree and is connected.
  Prf: Tour C has even incidences to any vertex v. □

3. Find tour T_i of G_i starting/ending at v_i. Induction.
4. Splice T_i into C where v_i first appears in C.

Visits every edge once:
  Visits edges in C exactly once.
  By induction for all edges in each G_i. □

Poll: Euler concepts.

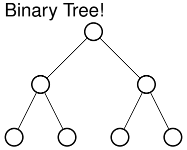
Mark correct statements for a connected graph where all vertices have even degree. (Below, tours uses edges at most once, but may involve a vertex several times.

(A) Removing a tour leaves a graph of even degree.
(B) A tour connecting a set of connected components, each with a Eulerian tour is really cool! Eulerian even.
(C) There is no hotel california in this graph.
(D) After removing a set of edges E' in a connected graph, every connected component is incident to an edge in E'
(E) If one walks on new edges, starting at v, one must eventually get back to v.
(F) Removing a tour leaves a connected graph.

Only (F) is false.

A Tree, a tree.

Graph $G = (V, E)$.
Binary Tree!



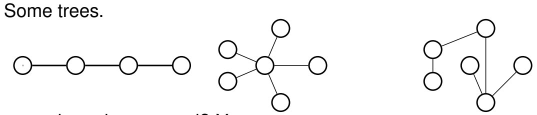
More generally.

Trees.

Definitions:

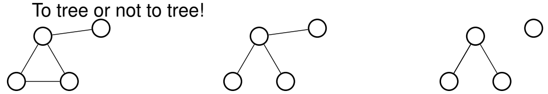
  A connected graph without a cycle.
  A connected graph with $|V| - 1$ edges.
  A connected graph where any edge removal disconnects it.
  A connected graph where any edge addition creates a cycle.

Some trees.



no cycle and connected? Yes.
$|V| - 1$ edges and connected? Yes.
removing any edge disconnects it. Harder to check. but yes.
Adding any edge creates cycle. Harder to check. but yes.

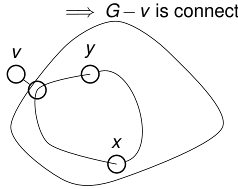
To tree or not to tree!



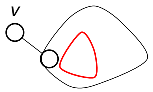
Equivalence of Definitions.

Theorem:
"G connected and has $|V| - 1$ edges" $\equiv$
    "G is connected and has no cycles."

Lemma: If v is degree 1 in connected graph G, $G - v$ is connected.
Proof:
  For $x \neq v, y \neq v \in V$,
   there is path between x and y in G since connected.
   and does not use v (degree 1)
   $\implies$ $G - v$ is connected. □



Proof of only if.



Thm:
"G connected and has $|V|-1$ edges" $\implies$
"G is connected and has no cycles."

Proof of $\implies$: By induction on $|V|$.
Base Case: $|V|=1$. $0=|V|-1$ edges and has no cycles.

Induction Step:
Claim: There is a degree 1 node.
Proof: First, connected $\implies$ every vertex degree ≥ 1.
Sum of degrees is $2|E|=2(|V|-1)=2|V|-2$
Average degree $(2|V|-2)/|V|=2-(2/|V|)$. Must be a degree 1 vertex.
Cuz not everyone is bigger than average! □
By degree 1 removal lemma, $G-v$ is connected.
$G-v$ has $|V|-1$ vertices and $|V|-2$ edges so by induction
$\implies$ no cycle in $G-v$.
And no cycle in G since degree 1 cannot participate in cycle. □

Proof of if

Thm:
"G is connected and has no cycles"
$\implies$ "G connected and has $|V|-1$ edges"
Proof:
Walk from a vertex using untraversed edges.
Until get stuck.
Claim: Degree 1 vertex.
Proof of Claim:
Can't visit more than once since no cycle.
Entered. Didn't leave. Only one incident edge. □
Removing node doesn't create cycle.
New graph is connected.
Removing degree 1 node doesn't disconnect from Degree 1 lemma.
By induction $G-v$ has $|V|-2$ edges.
G has one more or $|V|-1$ edges. □

Poll: Oh tree, beautiful tree.

Let G be a connected graph with $|V|-1$ edges.

(A) Removing a degree 1 vertex can disconnect the graph.
(B) One can use induction on smaller objects.
(C) The average degree is $2-2/|V|$.
(D) There is a hotel california: a degree 1 vertex.
(E) Everyone can be bigger than average.

(B), (C), (D) are true

Lecture Summary.

Graphs.
Basics.
Degree, Incidence, Sum of degrees is $2|E|$. Connectivity.
Connected Component.
maximal set of vertices that are connected.
Algorithm for Eulerian Tour.
Take a walk until stuck to form tour.
Remove tour.
Recurse on connected components.

Trees: degree 1 lemma $\implies$ equivalence of several definitions.
G: n vertices and $n-1$ edges and connected.
remove degree 1 vertex.
$n-1$ vertices, $n-2$ edges and connected $\implies$ acyclic.
(Ind. Hyp.)
degree 1 vertex is not in a cycle.
G is acyclic.