Mat 3770
Week 3

Spring 2014

Week 3 — Student Responsibilities

- ▶ Reading: Edge Counting, Planarity

- ▶ Homework from Tucker & Rosen

- ▶ Attendance **Cheekily** Encouraged

Chapter 1 Supplement:
Representing Graphs with Data Structures

- ▶ When working with graphs in a program, we're usually not so much interested in doing some sort of arithmetic operation as we are mainly **searching** – for neighboring nodes, for nodes with the least or maximum degree, etc.

- ▶ Given a graph G=(V, E), we can represent G in a computer program either with an **adjacency matrix** or **adjacency lists**.

- ▶ Either an adjacency matrix or adjacency list can store edge **weights** rather than 0/1 (indicating edge existence); they can also be used to store **directed** edges.

Adjacency Matrix

- ▶ Notice that in an adjacency matrix, in order to find a neighbor, say of F, we have to do essentially a **linear search**. This is true even if we only want to know if F has *any* neighbors.

Linked Lists

- ▶ An adjacency list can be stored in an array (`vector`) or as linked lists.
- ▶ Note that in C++ it is possible to "shrink" and "grow" array/vector size, thus avoiding wasted space. However, there is a trade off in time.

Induction Proofs

The **DREADED** Mathematical Induction Proof!
(And you thought you'd never see it again!)
BWAAAAAHAHAHAHAHAHA!!!!

- ▶ Let P(1), P(2), ..., P(n), P(n+1), ..., be an infinite sequence of statements. And suppose we know:
    1. P(1) is true, and
    2. for some **arbitrary** n ≥ 1,
       if P(n) is true, then P(n+1) is true

   Then, for **every** n ≥ 1, P(n) is true.

Induction Proofs — Steps

- There are three steps to an Induction Proof:

  1. **Base Case** (BC): establish the truth for P(1) (or P(i) for some small or initial i); pick an easy case if possible.

  2. **Induction Hypothesis** (IH): Assume the theorem is true for some **arbitrary** case, say P(n).

     (Note: in **strong induction**, assume all cases **up to** an arbitrary case are true.)

  3. **Inductive Step** (IS): Show (or verify) that **using the IH**, we can prove P(n+1) is true.

7

Prove

Use induction to prove: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ $\forall\, n \geq 1$

BC:

IH:

IS:

8

Another Proof

Use induction to prove: $\sum_{i=1}^{n}(2i-1) = n^2$ $\forall\, n \geq 1$

BC:

IH:

IS:

9

Yet Another Proof

Use induction to prove: $7|(11^n - 4^n)$ $\forall\, n \geq 0$

BC:

IH:

IS:

10

General Position

- A set of lines is in **General Position** if no two are parallel and no three pass through the same point.

General   **Not General**   **Not General**
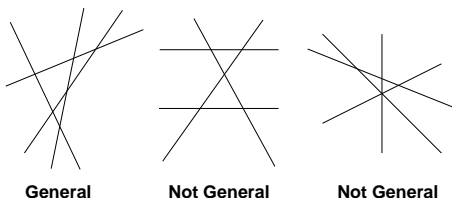
11

Number of Regions

Given n lines in general position in the plane, into how many regions is the plane divided?
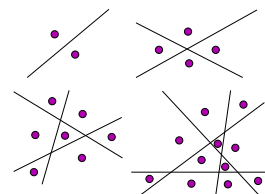
# lines	# regions
0	
1	
2	
3	
4	
5	

12

▸ Do all sets of 4 lines (in general position in the plane) yield the same number of regions? Why?

▸ **Question**: How many new regions seem to get added by the ith line? Why? i of them

▸ Let R(n) be the number of regions formed by n lines in general position in the plane.

▸ Claim: R(n+1) = R(n) + (n+1)

▸ Observation: line n+1 passes through $n+1$ regions of the plane divided by n lines. **Thus**, each of these regions is cut in two, for a net gain of $n+1$ regions.

▸ Where was the definition of **general position** used in that observation? In the phrase, "passes through $n+1$ regions"

▸ So:
$$
\begin{aligned}
R(n) &= n + R(n-1) \\
&= n + (n-1) + R(n-2) \\
&= n + (n-1) + (n-2) + R(n-3) \\
&= n + (n-1) + \ldots + 2 + 1 + R(0) \\
&= \frac{n(n+1)}{2} + 1
\end{aligned}
$$

▸ Prove using induction:
n lines in general position divide the plane into $\frac{n(n+1)}{2} + 1$ or $\frac{(n^2+n)}{2} + 1$ regions

---

Proof

We wish to prove that n lines **in general position** divide the plane into $\frac{n^2+n}{2} + 1$ regions

▸ **BC**: Let $n = 0$.
lhs: Zero lines $\Rightarrow$ a single region, the plane
rhs: $\frac{n^2+n}{2} + 1 = \frac{0^2+0}{2} + 1 = 1$
and $1 = 1$ $\checkmark$

▸ **IH**: Assume $R(n) = \frac{n^2+n}{2} + 1$ for some arbitrary $n \geq 0$

Proof

▸ **IS**: Show $R(n+1) = \frac{(n+1)^2+(n+1)}{2} + 1$

$$
\begin{aligned}
R(n+1) &= R(n) + (n+1) & \text{by definition} \\
&= \frac{n^2+n}{2} + 1 + n + 1 & \text{by IH, subst} \\
&= \frac{n^2+n}{2} + \frac{2(n+1)}{2} + 1 & \text{by alg. man.} \\
&= \frac{n^2+n+2n+2}{2} + 1 & \text{by alg. man.} \\
&= \frac{(n^2+2n+1) + (n+1)}{2} + 1 & \text{by alg. man.} \\
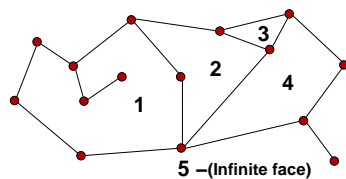&= \frac{(n+1)^2 + (n+1)}{2} + 1 & \text{by alg. man.}
\end{aligned}
$$

Thus, $R(n) = \frac{n^2+n}{2} + 1 \quad \forall \; n \geq 0$

Euler's Formula

▸ Let G = (V, E) be a connected planar graph,
where $v = |V|$ and $e = |E|$.

Euler's Formula: $\quad v - e + r = 2 \quad$ or $\quad r = e - v + 2$

$$
\begin{aligned}
v &= 15 \\
e &= 18 \\
r &= 5
\end{aligned}
$$

▸ **General Formula** (if G is not necessarily connected):
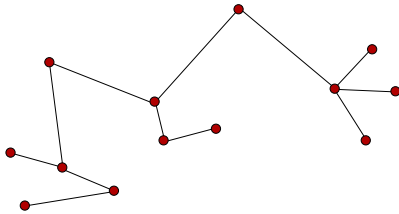$$v - e + r = 1 + c$$
where c is the number of connected components

▸ We shall prove $r = e - v + 2$ by induction on the **number of regions**

▸ **Theorem**. If G = (V, E) is a connected planar graph, where $v = |V|$, $e = |E|$, and r is the number of regions formed by G, then
$$r = e - v + 2$$

Base Case

- Let r = 1. Then we note:
  1. Graph must be a tree (i.e., no cycles)
  2. We need $1 = e - v + 2$, or $e = v - 1$

- We need to show that if G is a tree, then $e = v - 1$ (by induction on v)

If G is a tree, then $e = v - 1$

BC   Let $v = 1$
One vertex → no edges, e = 0
Thus, clearly true $(0 = 1 - 1 = 0)$

IH   Assume a tree of n vertices has $n - 1$ edges for some arbitrary n ≥ 1.

IS   Show a tree G_{n+1} with $n + 1$ vertices has n edges
- Find a vertex of degree 1 (it must be a leaf), and remove it and its edge, forming tree G_n
- By the IH, $e_n = v_n - 1$, so G_{n+1} has 1 more vertex than edges (i.e., $v_{n+1} - 1 = e_{n+1}$)
- Hence, $v_{n+1} - 1 = e_{n+1}$ since we add 1 vertex and 1 edge to G_n to construct G_{n+1}

Thus, if G is a tree, then $e = v - 1$

End of Subproof

Back to Euler

IH   Assume Euler's formula holds for graphs with n faces, for some arbitrary n ≥ 1, i.e., that $n = e - v + 2$

IS   Show the formula holds for any connected planar graph G with n+1 faces, v vertices, and e edges (i.e., n+1=e−v+2, or n=e−v+1)

- Pick a region of G and let a be an edge on that face.

- Then $G - a$ has (n+1)−1 = n regions; $e - 1$ edges, and v vertices

---
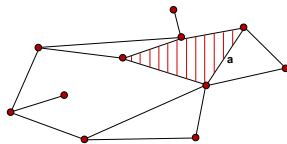
Conclusion

- By the IH:
$$n = (e - 1) - v + 2$$
$$= e - v + 1$$

- Thus, $r = e - v + 2 \quad \forall \, r \geq 1$

Chapter 2: Covering Circuits and Graph Coloring

2.1 Euler Cycles

Cycle: a sequence of consecutively linked edges:

$$((x_1, x_2), (x_2, x_3), \ldots, (x_{n-1}, x_n))$$
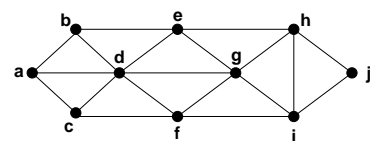
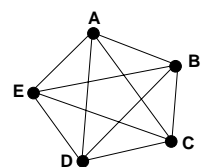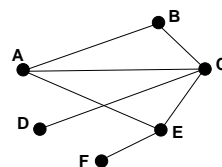whose starting vertex is the ending vertex

$$(x_1 = x_n)$$

and in which no edge can appear more than once.

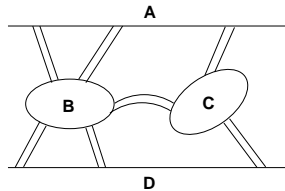A vertex may be visited multiple times, however.

Finding Cycles

Konigsberg Bridge Problem

The old Prussian city of Konigsberg, located on the banks of the Pregel River, included two islands which were joined to the banks and to each other by seven bridges:

Because sex and television hadn't been invented yet, the townspeople strolled about the town and across the bridges, and had entirely too much time to think...

Eventually, someone tried to determine a walk which began at their front door, crossed each bridge exactly once, and allowed them to return to their front door...
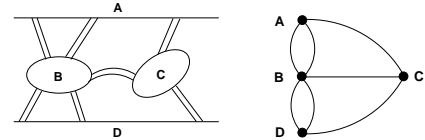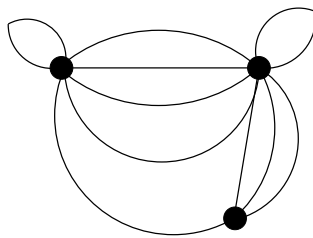

They weren't able to do this, so took the problem to the famous and fabulously well respected mathematician, Leonhard *"Lenny"* Euler!

He was able to solve the problem, and thus spawned
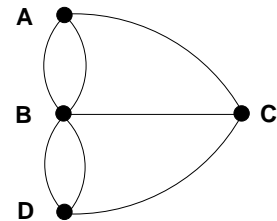**Graph Theory**

Multigraphs

Multigraph: a graph which allows multiple edges between the same vertices, as well as permitting self–loops.

Euler Cycles

► Can we find a tour of the bridges which allows us to return home without crossing any bridge more than one time? Why or why not?
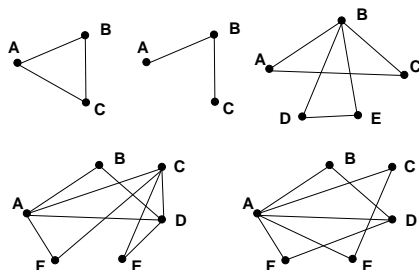

► **Euler Cycle**: a cycle that contains *all* the edges in a graph and *visits* each vertex at least once.
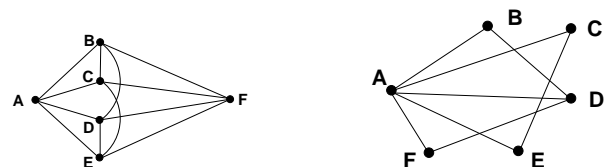

An Investigation

Let G = (V, E) be a graph. Does G contain a path that begins and ends at the same vertex *and* traverses each edge exactly once (i.e., an **Euler Cycle**)?

Question: If an Euler Cycle exists, does it matter *where* we start?

Question: Do we need to *produce* an Euler cycle to know if one exists?

Observation 1

- If G has an Euler Cycle, then every vertex has even degree... Why?

- Suppose degree(v) is odd and
  - **we begin at v**
    Can't end there, no way back

  - **we don't begin at v**
    Must end there, contrary to the definition of Euler Cycle

31

An Algorithm to Find an Euler Cycle

Consider the following algorithm:

Step #1. Starting at an arbitrary vertex, say v, "walk" from edge to edge, *marking* edges as you traverse them, until you can walk no further.

note 1: only choose unmarked edges as you go

note 2: won't necessarily end at the start node

Observation 2: If your walk ended at $u \neq v$, then u has *odd* degree (why?) so G doesn't have an Euler Circuit. Therefore **Stop:Fail**.
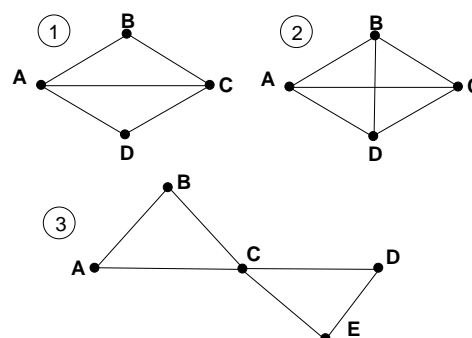
32

Euler Algorithm, Step 2

Step #2. Call the path you made
$$P = v_1, v_2, v_3, \ldots, v_n, v_1$$

Observation 3: Exactly one of the following is true:

1. Some vertex v_i along the path P has an unmarked edge

2. Every edge is in P (therefore **Stop:Success**)

3. G isn't connected (therefore **Stop:Fail**)

33

34

Euler Algorithm, Steps 3—5

Step #3. If the first case holds, (v_i on Path has unmarked edge), then repeat **Step #1** (walk a path) starting at vertex v_i.

Observation #4: If your walk doesn't end at v_i then v_i has odd degree. Therefore **Stop:Fail**.

Step #4. Augment the path P by adding the new path (insert it in P at v_i)

Step #5. Repeat from **Step #3**.

35

Results

Claim: If the algorithm succeeds, then G clearly has an Euler Cycle.

Just as important: If the algorithm *fails* then G **doesn't** have an Euler Cycle (since G either has a vertex of odd degree or G isn't connected).

Thus the algorithm will definitely *find* an Euler Cycle if one exists and will produce *proof* that G doesn't have one otherwise.

The algorithm also shows:
Theorem. If G is connected and each vertex has even degree, then G has an Euler Cycle. (Why?)
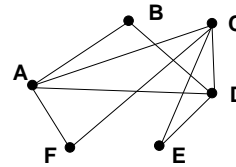
36

Algorithm Efficiency

Question: How efficient is this algorithm?

- Well, what does it do?
  - walks along the graph
  - glues little cycles together
  - builds new little cycles
- Until success or failure.

Adjacency Matrix

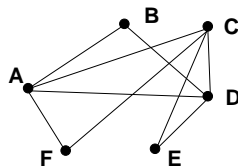How efficient may depend on what data structure is used to store the edges.

	A	B	C	D	E	F
A	0	1	1	1	0	1
B	1	0	0	1	0	0
C	0	0	0	1	1	1
D	1	1	1	0	1	0
E	0	0	1	1	0	0
F	1	0	1	0	0	0

Finding "next edge" in the path could take $n = |V|$ steps, and this happens once for each edge; thus, $|V| * |E|$ steps are required.

Linked Lists

Pseudo–Code

```
// S - Set of vertices with unmarked edges
// P - Euler Path we're building
S = V
P = {}
fail = false
while vertex remains with unmarked edge in S
        and haven't failed
 1.choose vertex with unmarked edge, u
 2."walk" a path P' to vertex v, marking edges
 3.if u doesn't equal v
        then fail = true
    else augment P with P'
endwhile

if not all edges marked, or G not connected,
  then fail = true

if not fail, then Euler Cycle exists
```

Theorem. An undirected multigraph has an Euler Cycle IFF it is connected and all vertices have even degree.

The proof follows the same construction as we gave for the Euler Cycle on a regular graph:

Take a walk, leaving breadcrumbs as we go.

If we do not return to where we started, yet all the edges we may take are marked, there is no cycle.

Otherwise, if there is a vertex on our path with an unmarked edge, repeat the walk with the same results.

Associated Problems

In graphs that do not contain Euler Cycles, we must traverse some edges more than one time (called **Deadheading edges**) to obtain a **tour**. (Example in text: street cleaning)

- **New Problem**: minimize the number of deadheading edges (i.e., find the minimum set of edges needed to make all vertices of even degree and the graph connected.

- **Another Problem**: minimize the number of U–turns and turns the street cleaner must make (takes time and wastes fuel). This same concept is used in VLSI chip design — minimize layers and vias.

Trails

Trail: a sequence of consecutively linked edges in which no edge appears more than once.

Trail is to **Path**
as
Cycle is to **Circuit**

Both trails and cycles allow repeated vertices

Euler Trail: a trail which contains all the edges in a graph.

(**Note**: it will visit each vertex at least once assuming the graph is connected.)

A Corollary

Corollary (to Euler's Theorem): A multigraph has an Euler Trail, but not an Euler Cycle, IFF it is connected and has exactly two vertices of odd degree.

Proof

($\Rightarrow$) Assume multigraph G has an Euler trail T but no Euler cycle and show it has exactly two vertices of odd degree.

We note the starting and ending vertices of T, the Euler trail, must have odd degree while all other vertices must have even degree (by the same reasoning used to show all vertices in a graph with an Euler Cycle must have even degree). Further, G must be connected.

($\Leftarrow$) Assume multigraph G has exactly two vertices of odd degree (say p and q), and show it has an Euler Trail but not an Euler Cycle.

Let us add a supplementary edge $< p, q >$ to G, obtaining the graph H. H is connected and has all vertices of even degree. Hence by the Euler Cycle theorem, H has an Euler Cycle.

Let us call this Cycle C. Now, remove the edge $< p, q >$ from C. This reduces the Euler Cycle to an Euler Trail and includes all edges of G.