MATH 200
Lectures (Apr 16-18, etc) **Discrete Mathematics**
Wm Kallfelz

GRAPHS

We discussed digraphs (directed graphs) before in terms of a diagram. We can extract a simpler structure:

***Defn.(1) A** <u>simple graph</u>* **G= { V,E } is a collection of vertices $V$and edges E such that**:

a.) $0 < |V| < \infty$

b.) For any pair of distinct vertices $\{x,y\}$ ∃! An edge *e* connecting x,y OR there is no edge. If such a unique edge *e* exists between *x,y*; we say *x* is <u>adjacent</u> to y and write: *x*(adj)*y*, and we also say: edge *e* is incident on vertices *x,y*

This is known as the <u>no paralell edge rule</u> and it enables us to uniquely represent an edge as: {x,y}. Because $x\neq y$ it also automatically follows that there can be no edges of the form: {x,x} (No loops.

Defn.(2)

c.) For any vertex $v\in V(G)$ the <u>degree</u> of v = # of edges incident on v (write: deg(v))

d.) For any graph G, the <u>degree</u> of G = sum total of all degrees per vertex, written: deg(G)

Lemma: The "Handshake" Lemma: deg$G$ = 2$|E|$ (Thm 11.1.1)

Proof: Consider the algorithm for determining $degG = \sum_{v\in V(G)} deg(v)$: Let edge *e* be incident on vertices $v,v' \in V(G)$. **Case 1:** Suppose $v\neq v'$ (*e* is not a loop.) Counting *deg(v)* implies *e* is counted where *e* is incident on *v*. Counting *deg(v')* implies *e* is counted where *e* is incident on *v'*. Thus *e* is counted twice. **Case 2:** Suppose $v= v'$ then *e* is a loop, and is automatically double counted when evaluating *deg(v)* (as the counting procedure involves drawing enclosing *v* in a circle *C* of arbitrarily small radius ε and enumerating points of intersection on *C*.) Since $degG$ involves this procedure at every vertex, every edge in G is double-counted. $\therefore degG = 2|E|$

Exercise: Using the Handshake Lemma, determine: a) $|E(K_n)|$ **b)** $|E(K_{n,m})|$

a.) In the case of K_n, every vertex is adjacent to its (n - 1) neighbors. $\therefore \forall v\in V(K_n)$, $deg(v) = n - 1$. $\therefore deg(K_n) = n(n - 1) = n^2 - n = 2|E(K_n)|$. Therefore: $|E(K_n)|.= {}^n/_2(n - 1) = C(n,2)$

b.) In the case of $K_{n,m}$ $\forall v\in V(K_{n,m})$, $deg(v)= m \vee n$ In fact, there are *m* vertices of degree *n* and vice versa. Therefore:. $deg(K_{n,m}) = mn + nm = 2mn = 2|E(K_{n,m})|.\therefore |E(K_{n,m})|.= mn$

EULER & HAMILTON CIRCUITS

Recall that a **circuit** is a closed walk $W_{u\to u} = u,e_1; v_2,e_1;\ldots; v_{n-1},e_{n-1};v_n (= u)$ containing no repeated edges. An **Euler Circuit (EC)** in a graph G contains all of G's edges. We present the material below in a slightly more condensed and less theoretical manner as that found in text (11.2)

Lemma1 (Nec. Condition for the existence of an EC) **If a graph G contains an EC then $\exists j\in Z^+$:** ***degv=2j*** *$\forall v\in V(G)$*

Proof (by contradiction) Suppose *G* has an EC but there exists a vertex *v* of odd degree 2j + 1. Moving along EC, every time we arrive at *v* we must adjoin a *different* outgoing edge from the (incoming) edge we were just on. Thus *j* incoming edges must be paired with another *j* outgoing edges at *v*. However, since eventually every edge must be traversed in G we're stuck with one more edge *e'* at *v* where we have no choice but to `do a U turn' at *v* lest we visit any of the other *j* ingoing-ougoing edges again. But then we're re-visiting *e'*, violating the conditions of an EC. If we ignore *e'* by not going on it at all, then we haven't used up all the edges in *G*. Either way, there cannot be an EC. Contradiction.

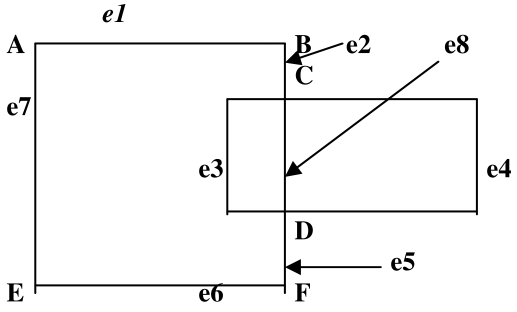
Suppose G is connected and every vertex is even degreed. The ***Euler Circuit Algorithm*** automatically finds an EC regardless of the complexity of G:

Defn (3) The *ECA* is precisely denoted as:

```
1.) Have all edges been used? If yes STOP
2.) Pick a vertex X and an edge e  incident on X
    3.) Choose a vertex Y incident on e.
    4.) Is Y=X? If yes, goto 7.)
        5.) Pick an unused edge e' on Y and a vertex Z incident on e'
        6.) Let Z = Y. Goto 4.)
    7.) Are there any vertices with unused edges? If no STOP
    8.) Choose a vertex V with unused edges
    9.) Pick an unused edge e" on V and a vertex W incident on e"
    10.) Is W=V? If yes, adjoin this list with the previous edge list, go to 7.)
    11.) Pick an unused edge e"' on W and a vertex T incident on e"'. Let T=W,goto 10.)
```

- *Note: Keep in mind the convention is obeyed where the next line is automatically read whenever the the "IF…THEN…" conditional fails. For example, the 'No' case in line 4) means automatically that line 5.) will be the next execution. Keep in mind also that 'Let X = Y' means* ***assign*** *the value (so '=' is not a comparison in this case)*
- *We recongize lines 1.) - 6.) , and lines 7.) - 11.) as finding circuits in the graph G. The most important command is really 10.) as the Euler circuit is explicitly constructed.*

Example:



We run the ECA in 'real time' here:

1.) NO→2.) X=A,e7→3.) Y=E →4.) NO→5.) e6,Z=F→6.) Y=F→4.)NO →5.) e5,Z=D→ 6.) Y=D→4.)NO→5.) e8,Z=C→6.) Y=C→4.) NO→5.) e2,Z=B→6.) Y=B→4.)NO →5.) e1,Z=A→ 6.) Y=A→4.)YES→7.)YES→8.)V=D→9.) e3,W=C→10.) NO→11.) e4,T=W=D→10.) YES *ADJOIN* →7.) NO, STOP.

Now the *actual output* of the ECA is generated in 'sample time,' i.e., via the following list:

A,e7;E,e6;F,e5;D,e8;C,e2;B,e1
D,e3;C,e4
***ADJOIN* (insert second list to top list immediately left of vertex D)**
→ A,e7;E,e6;F,e5;{ D,e3;C,e4};D,e8;C,e2;B,e1
STOP

So the final output generated is:

A,e7;E,e6;F,e5; D,e3;C,e4;D,e8;C,e2;B,e1

(Which is an Euler Circuit)

- **How does the ECA work?** By randomnly selecting unused edges until a circuit is formed (Steps 1.) - 4.)) and once a circuit is found, the process is iterated until all edges are used up (steps 6.) - 11.)). Once the edges are used up, the circuits are adjoined to form the final Euler circuit. Suppose the ECA halted prematurely. Then (**case 1**) the graph could be disconnected OR (**case 2**) there exist vertices of odd degree. For the ECA keeps looking for cycles at a given vertex which implies that the initial edge at a particular vertex comprising a cycle is different from the final edge. (Example: In the graph above, the ECA found one cycle at A by initially departing along **e7** and arriving at **e1**.) So if a vertex were odd degreed, then the ECA would always overlook one edge at that vertex and thus not use up all of G's edges. Expressing the ***contrapositive*** of the above we arrive at the following :

Lemma2 (Sufficiency conditions for the existence of an EC) If a graph is connected and every vertex even-degreed, then it has an EC.

Proof: *As shown above, If G is connected and every vertex even-degreed, then the ECA will use up all of* G's edges.

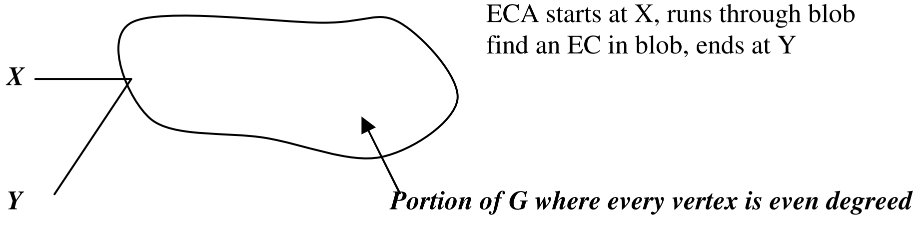
Combining both Lemmas:

Theorem (I) (11.2.4, text)***A graph G has an EC $\Leftrightarrow G$ is connected and every vertex is even degreed***

Proof: Lemmas (1) and (2).

Corollary: ***A graph G has an Euler path (a walk using all edges and vertices without repeating any edges) if G is connected and all but 2 vertices are even degreed.***

Proof: Running the ECA we begin at the first odd degreed vertex and we'll end up at the second odd degreed vertex. Schematically WLOG this procedure can be modelled in the simplest case where two vertices have degree 1. Then, as suggested in the diagram below:



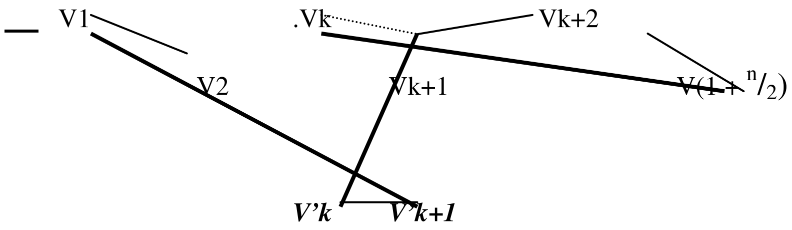
Theorem (I) guaranteed necessity and sufficiency conditions for devising an automatic procedure for finding an EC in a graph G. This solves the problem of finding all ECs in a graph, if they exist, asn an effective (computable) procedure was devised and showed to work under all conditions even degree and connectivity without exception. Unfortunately, the same does not hold for asking a more restrictive question: 'What are all the **Hamilton circuits** in a graph G?' (Recall that an HC is an EC with no repeated vertices.) We do, however, have a *sufficiency condition:*

Lemma 3: Let G be a graph, with $n=|V(G)|$. If $\forall v\in V(G)$., $deg(v) \geq {}^{n}/_{2}$ then G has a HC

Proof: Either 2 vertices u,v are connected by an edge, or $deg(u)+deg(v) \geq n$. In the latter case, u,v must have a common neighbor (WHY?). **Therefore G is connected.** Consider a simple circuit C in G whose endpoint is only adjacent to other vertices in C (since G connected, this is always possible.) **Case 1:** $|V(C)|$.= $|V(G)|$. Then C is a HC. .**Case 2:** $|V(C)|$.< $|V(G)|$. Form a path P in G with at $(1 + {}^{n}/_{2})$ vertices (this is always possible, since $deg(v) \geq {}^{n}/_{2}$). Then an HC can always be naturally constructed from P using the following procedure:

a.) Let v_k, $v_{k+1} \in V(P)$ (where $1 \leq k \leq \lfloor {}^{n}/_{2} \rfloor$) where Let $v_k adj\ v_{k+1}$

b.) Remove the edge connecting v_k, v_{k+1} and adjoin v_k with the last vertex found in P. Adjoin v'_k *with* v'_{k+1} and come back down to v_1 (See figure below)



c.) We have constructed a simple circuit C in G. To extend C into an HC is done by assigning v'_{k+1} to be the first vertex of the complementary path P' composed of ***all*** the ${}^{n}/_{2}$ -1 vertices not found in P. (The fact that such a path P' can be constructed is due to the same reason why P can be formed.) Assign v'_k to be the ***first*** vertex of the complementary path P'. Then C is a HC

Exercise: When is $K_{m,n}$ guaranteed to have a HC?

Answer: The total number of vertices = m+n. And for any vertex in this graph, its degree is either m or n. According to the above Lemma, then:

$$m \geq (m + n)$$
$$n \geq (m + n)$$

Solution: whenever m=n

Though there are many algorithms that can find a HC, there exist no algortihms which can exist all HCs for any given graph G because as of now, no necessary conditions have been formulated for the existence of an HC. This asymmmetry can at least be appreciated when one views trying to index vertices in terms of edges. According to the Handshake Theorem, $|E|$ is a polynomial of $|V|$ (which is at most 2nd degree in the case of a simple graph, as $|K_n| = C(n,2)$.) This implies $|V|$ is a complicated root of $|E|$. For example, in the case of K_n, if $|E|$=m, then:

$$|V| = \left\lfloor {}^{1}/_{2}\{1 + [1+8m]^{1/2}\}\right\rfloor$$

So, just because edges can be naturally indexed in terms of the numbers of vertices, the converse certainly does not hold. Trying to establish necessity conditions for the existence of a HC means trying to determine some general property the topology of the vertices in terms of the edges. Going in this direction is much more difficult than in the other direction (trying to establish topological properties of edges given vertices)