Lecture: 02
Paths, Cycles, and Trails

Dr. Rajiv Misra
Associate Professor
Dept. of Computer Science & Engg.
Indian Institute of Technology Patna
rajivm@iitp.ac.in

Preface

Recap of previous Lecture:

- In the previous lecture, we have discussed a brief introduction to the fundamentals of graph theory and how graphs can be used to model the real world problems.

Content of this Lecture:

- In this lecture, we will discuss Peterson graph, Connection in graphs and Bipartite graphs.

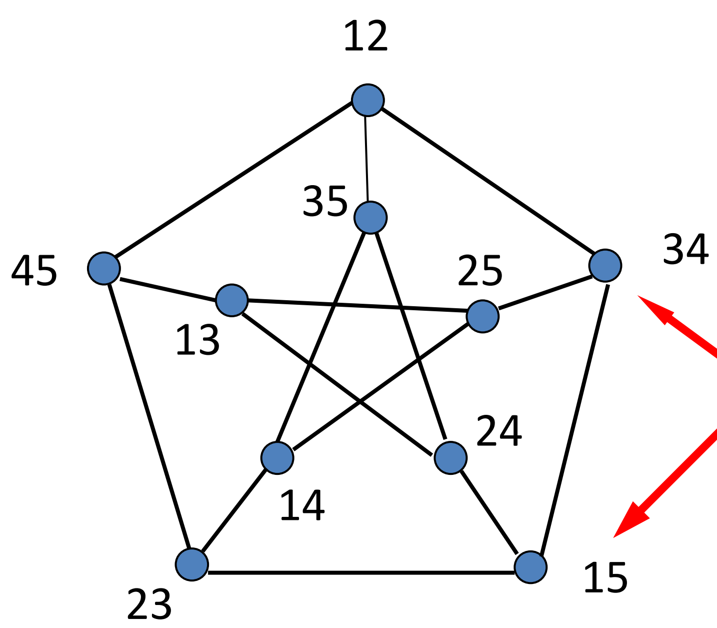
Petersen Graph

- The ***petersen graph*** is the simple graph whose vertices are the 2-element subsets of a 5-element set and whose edges are pairs of disjoint 2-element subsets

Example

- Assume: the set of 5-element be (1, 2, 3, 4, 5)
- Then, 2-element subsets:

(1,2) (1,3) (1,4) (1,5) (2,3) (2,4) (2,5) (3,4) (3,5) (4,5)

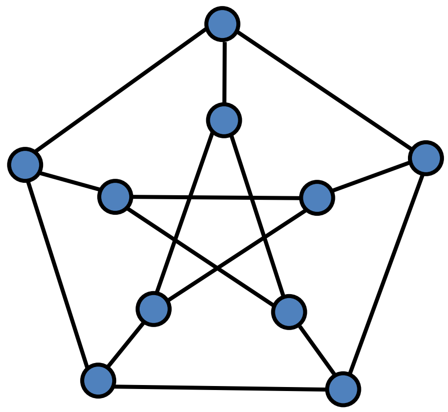


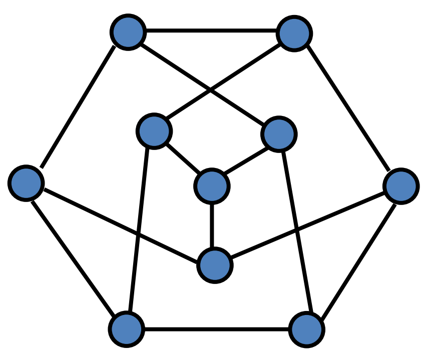
45: (4, 5)

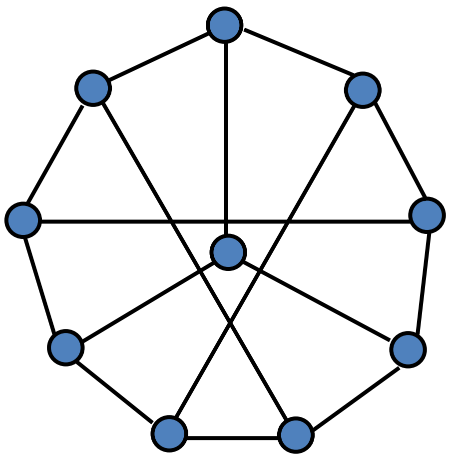
Disjoint, so connected

Example

- Three drawings

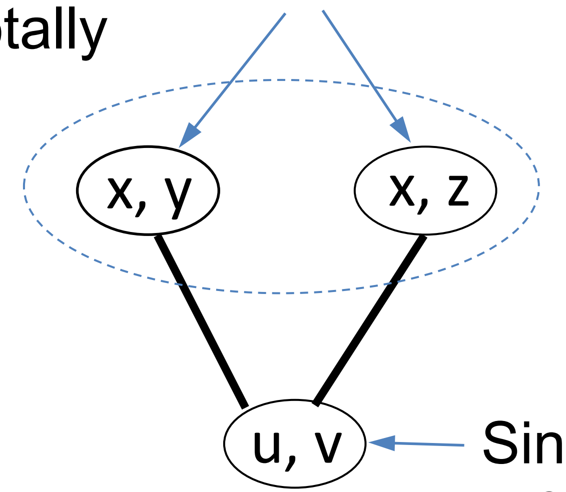






Proof:

No connection,
Joint, One common element.
3 elements in these vertices totally



Since 5 elements totally,
5-3 elements left.
Hence, exactly one of this kind.

Girth

- ***Girth*** : the length of its shortest cycle.

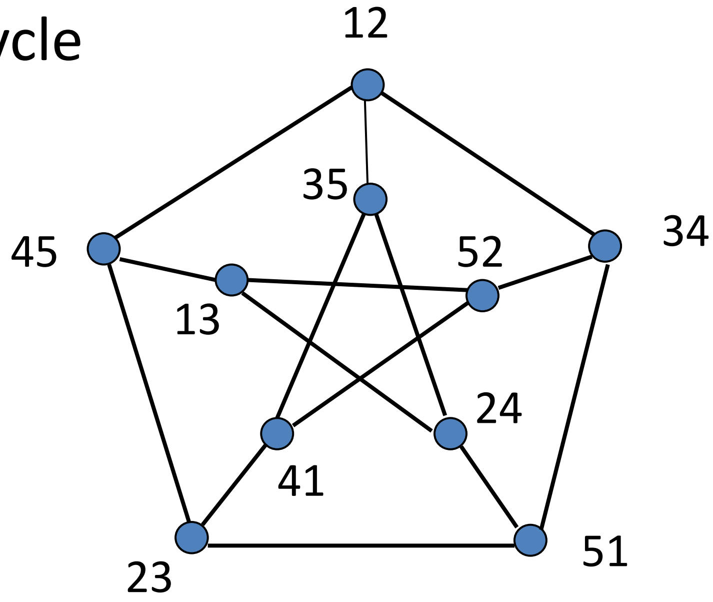
If no cycles, girth is infinite

Girth and Petersen graph

Theorem: The Petersen Graph has girth 5.

Proof:

- Simple → no loop → no 1-cycle (cycle of length 1)
- Simple → no multiple → no 2-cycle
- 5 elements →no three pair-disjoint 2-sets → no 3-cycle
- By previous theorem, two nonadjacent vertices has exactly one common neighbor → no 4-cycle
- 12-34-51-23-45-12 is a 5-cycle.



Walks, Trails

- A ***walk*** : a list of vertices and edges v_0, $e_1, v_1, \ldots., e_k, v_k$ such that, for $1 \leq i \leq k$, the edge e_i has endpoints v_{i-1} and v_i .

- A ***trail*** : a walk with **no repeated edge**.

Paths

- A *u*,*v*-walk or *u*,*v*-trail has first vertex *u* and last vertex *v*; these are its endpoints.
- A ***u*,*v*-path**: a *u*,*v*-trail with no repeated vertex.
- The ***length*** of a walk, trail, path, or cycle is its number of edges.
- A walk or trail is ***closed*** if its endpoints are the same.

Lemma: Every u,v-walk contains a u,v-path 1.2.5

Proof:

- Use induction on the length 'l' of a u, v-walk W.
  - Basis step: $l = 0$.
  - Having no edge, W consists of a single vertex ($u=v$).
  - This vertex is a u,v-path of length 0.

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

Proof: Continue

Induction step : $l \geq 1$.

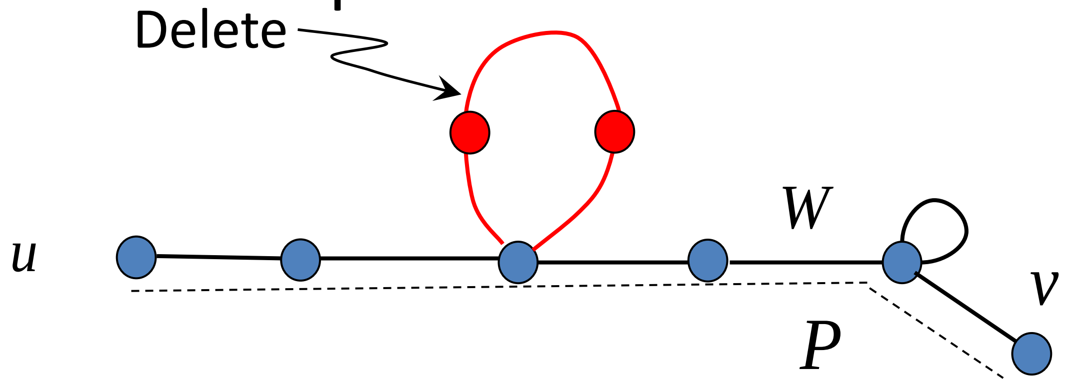
- Suppose that the claim holds for walks of length less than l.
- If W has no repeated vertex, then its vertices and edges form a u,v-path.

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

Proof: Continue

Induction step : $l \geq 1$. **Continue**

- If W has a repeated vertex ***w***, then deleting the edges and vertices between appearances of w (leaving one copy of ***w***) yields a shorter u,v-walk W' contained in W.
- By the induction hypothesis, W' contains a u,v-path P and this path P is contained in W.



Connected and Disconnected

- "**Connected**" is an adjective applies only to **graphs** and to **pairs of vertices**
- *(we never say "v is disconnected" when v is a vertex).*
- Distinction between *connection* and *adjacency*:

***G* has a *u*, *v*-path**	***u v* ϵ *E(G)***
u and *v* are connected	*u* and *v* are adjacent
u is connected to *v*	*u* is joined to *v*
	u is adjacent to *v*

Connection Relation

- By Lemma 1.2.5, we can prove that a graph is connected showing that from each vertex there is a walk to one particular vertex.
- By Lemma 1.2.5, the connected relation is transitive: if G has a u, v-path and a v, w-path, then G has a u, w-path.
- It is also reflexive (paths of length 0) and symmetric (paths are reversible), so *connection* is an equivalence relation.
- A **maximal** connected subgraph of G is a subgraph that is connected and is not contained in any other connected subgraph of G.

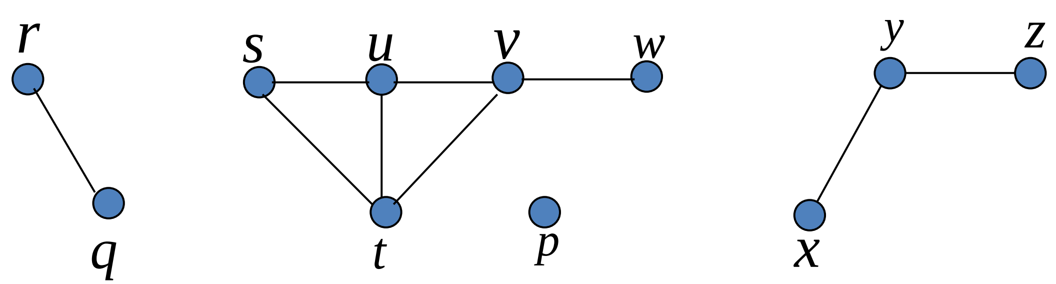
Components

- The ***components*** of a graph G are its **maximal** connected subgraphs. A component (or graph) is ***trivial*** if it has no edges; otherwise it is nontrivial.
- An ***isolated vertex*** is a vertex of degree 0.
- The equivalence classes of the connection relation on $V(G)$ are the vertex sets of the components of G.

Example

- The graph below has four components, one being an ***isolated vertex***.

- The vertex sets of the components are {p}, {q, r}, {s, t, u, v, w}, and {x, y, z}; these are the equivalence classes of the connection relation.



Adding/Removing an edge

- Components are pairwise disjoint; no two share a vertex. Adding an edge with endpoints in distinct components combines them into one component.

- Thus adding an edge decreases the number of components by 0 or 1, and deleting an edge increases the number of components by 0 or 1.

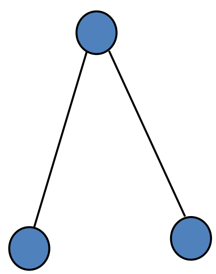
Theorem: Every graph with *n* vertices and *k* edges has at least *n-k* components 1.2.11

Proof:

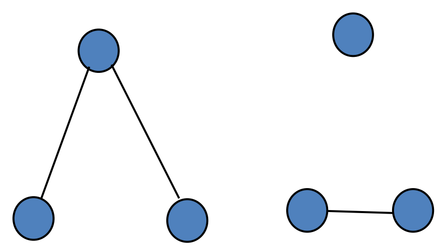
- An *n*-vertex graph with no edges has *n* components
- Each edge added reduces this by at most 1
- If *k* edges are added, then the number of components is at least *n* - *k*

Examples

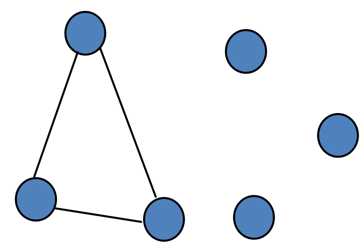
***n* =2, *k* =1**,
1 component



***n* =3, *k* =2**,
1 component



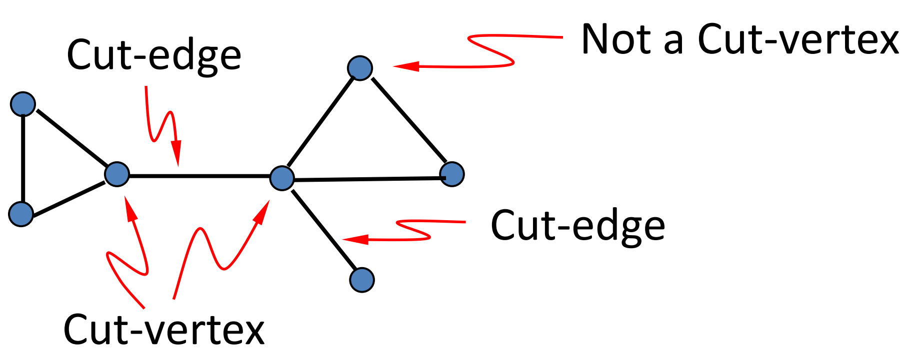
***n* =6, *k* =3**,
3 components



***n* =6, *k* =3**,
4 components

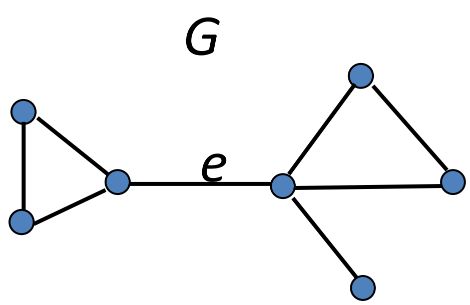
Cut-edge, Cut-vertex

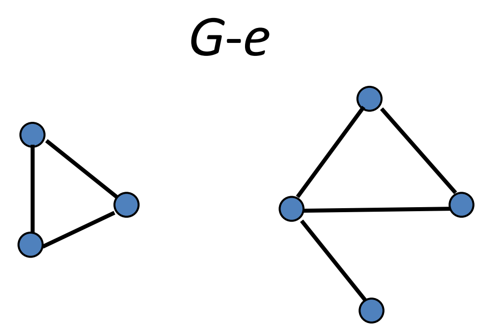
- A ***cut-edge*** or ***cut-vertex*** of a graph is an edge or vertex whose deletion increases the number of components.



Cut-edge, Cut-vertex

- *G-e* or *G-M* : The subgraph obtained by deleting an edge *e* or set of edges *M*
- *G-v* or *G-S* : The subgraph obtained by deleting a vertex *v* or set of vertices *S*



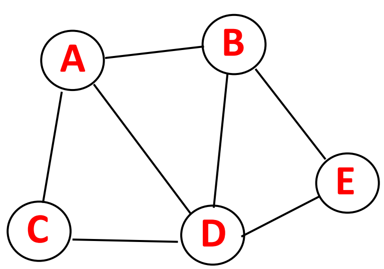


Induced Subgraph

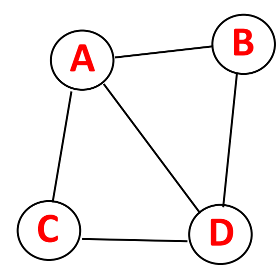
- An ***induced subgraph*** :
- A subgraph obtained by deleting a set of vertices
- We write $\boldsymbol{G}[T]$ for $\boldsymbol{G}-\overline{\boldsymbol{T}}$, where $\overline{\boldsymbol{T}} = \boldsymbol{V}(\boldsymbol{G})-\boldsymbol{T}$

$\boldsymbol{G}[T]$ is the subgraph of $\boldsymbol{G}$ induced by $\boldsymbol{T}$

Example: Assume T: {A, B, C, D}



G



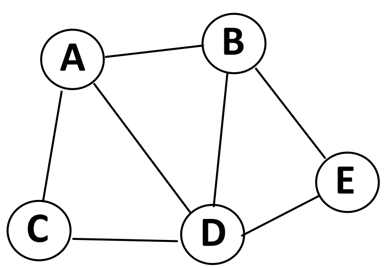
G[*T*]

More Examples:

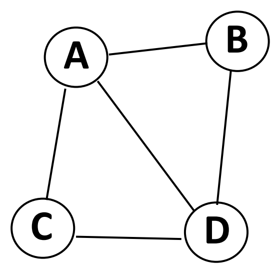
G_2 is the subgraph of G_1 induced by (A, B, C, D)

G_3 is the subgraph of G_1 induced by (B, C)

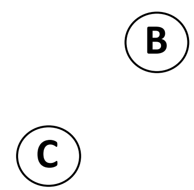
G_4 is **not** the subgraph induced by (A, B, C, D)



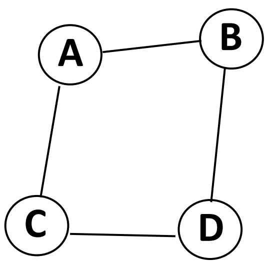
G_1



G_2



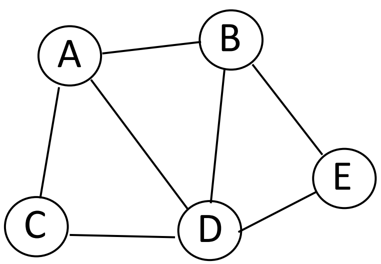
G3



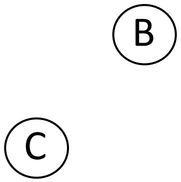
G_4

Induced Subgraph

- A set ***S*** of vertices is an independent set if and only if the subgraph induced by it has no edges.
- Example:
- G_3 is an example.



G_1



G3

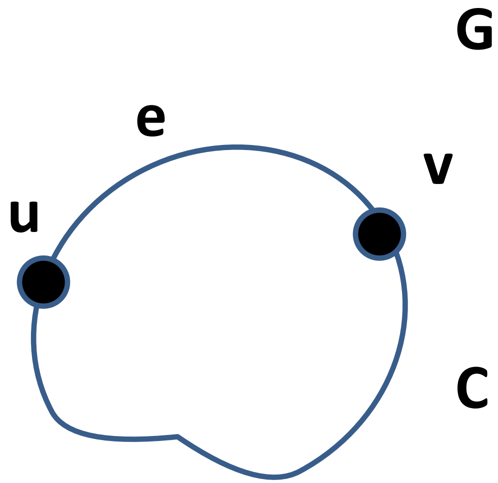
Theorem: An edge *e* is a cut-edge if and only if *e* belongs to no cycles. 1.2.14

Proof : ⇨ (Necessity) e is cut-edge if e is not on cycle

Contrapositive: e is on cycle if e is not cut-edge

- Say e is on a cycle C
- In G-e, u and v are in same connected component
- Therefore, e is not a cut-edge

■

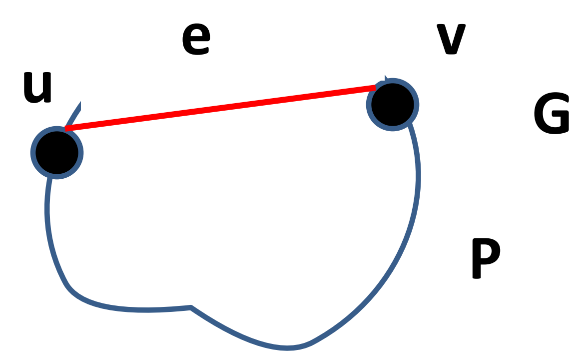


Theorem: An edge *e* is a cut-edge if and only if *e* belongs to no cycles. 1.2.14

Proof : $\Leftarrow$ **e is cut-edge only if e belongs to no cycle**

Contrapositive: e is not a cut-edge $\Longrightarrow$ e is on a cycle

- e = uv is not a cut-edge
- Then G \ e, there is a path from u to v
- (u & v are in the same component)
- P: $u \xrightarrow{*} v$
- In G, eP: $u \xrightarrow[e]{} v \xrightarrow[P]{*} u$
- i.e. there is a cycle that contains e



■

Bipartite Graphs

- Our next goal is to characterize bipartite graphs using cycles. Characterizations are equivalence statements, like Theorem[cut-edge]. When two conditions are equivalent, checking one also yields the other for free.
- Characterizing a class **G** by a condition P means proving the equivalence "$G \in \mathbf{G}$ if and only if G satisfies P".
- In other words, P is both a **necessary** and a **sufficient** condition for membership in **G**.

Necessity	**Sufficiency**
$G \in \mathbf{G}$ *only if* G satisfies P	$G \in \mathbf{G}$ *if* G satisfies P
$G \in \mathbf{G} \Longrightarrow$ G satisfies P	G satisfies P $\Longrightarrow G \in \mathbf{G}$

Bipartite Graphs

- Recall that a loop is a cycle of length 1; also two distinct edges with the same endpoints form a cycle of length 2.
- A walk is **odd** or **even** as its length is odd or even.
- As in Lemma 1.2.5, a closed walk **contains** a cycle C if the vertices and edges of C occur as a sublist of W, in cyclic order but not necessarily consecutive.
- We can think of a closed walk or a cycle as starting at any vertex; the next lemma requires this view point.

Lemma: Every closed odd walk contains an odd cycle
1.2.15

Proof:1/3

- Use induction on the length l of a closed odd walk W.
- l=1. A closed walk of length 1 traverses a cycle of length 1.
- We need to prove the claim holds if it holds for closed odd walks shorter than W.

Lemma: Every closed odd walk contains an odd cycle

Proof: 2/3

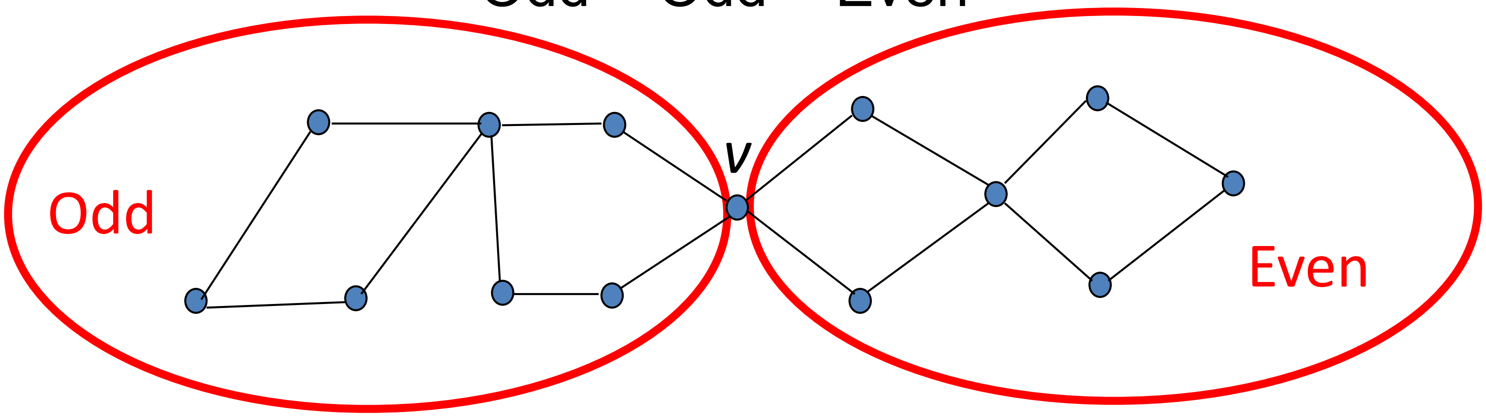
- Suppose that the claim holds for closed odd walks shorter than *W*.
- If *W* has no repeated vertex (other than first = last), then *W* itself forms a cycle of odd length.
- Otherwise, **(*W* has repeated vertex)**
  - Need to prove: If repeated, *W* includes a shorter closed odd walk. By induction, the theorem hold

Lemma: Every closed odd walk contains an odd cycle

Proof: 3/3

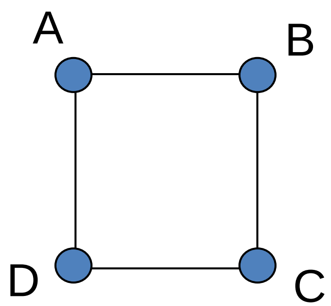
- If W has a repeated vertex v, then we view W as starting at v and break W into two v,v-walks
  - Since W has odd length, one of these is odd and the other is even. (see the next page)
  - The odd one is shorter than W, by induction hypothesis, it contains an odd cycle, and this cycle appears in order in W

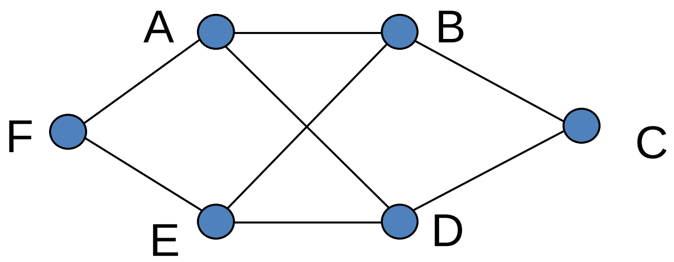
Odd = Odd + Even

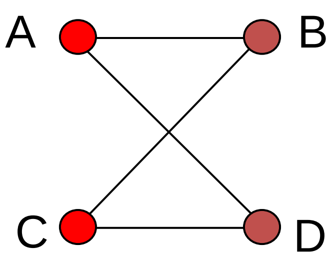


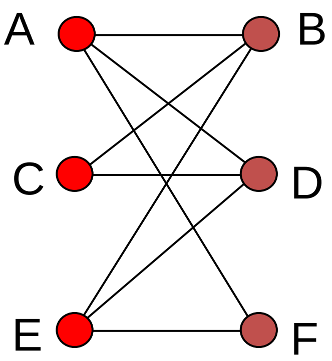
Theorem: A graph is bipartite if and only if it has no odd cycle 1.2.18

Examples:





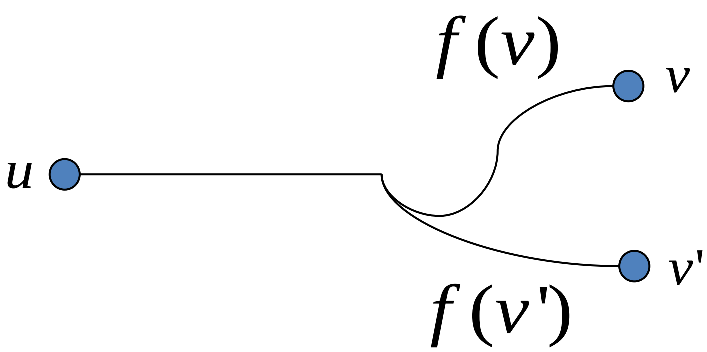




Theorem: A graph is bipartite if it has no odd cycle.

Proof: (sufficiency1/3)

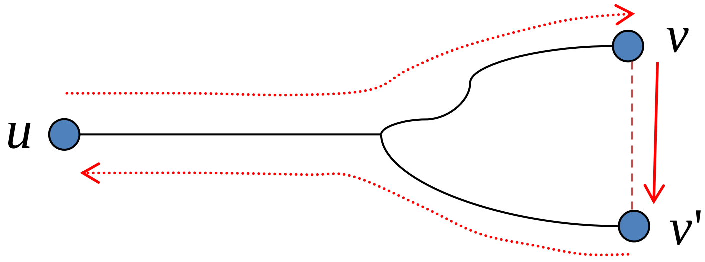
- Let G be a graph with no odd cycle.
- We prove that G is bipartite by constructing a bipartition of each nontrivial component H.
- For each $v \in V(H)$, let $f(v)$ be the minimum length of a u, v-path. Since H is connected, $f(v)$ is defined for each $v \in V(H)$.



Theorem: A graph is bipartite if it has no odd cycle.

Proof: (sufficiency2/3)

- Let X={$v \in V(H)$: $f(v)$ is even} and Y={$v \in V(H)$: $f(v)$ is odd}
- An edge *v, v′* within *X* (or *Y*) would create a closed odd walk using a shortest *u, v*-path, the edge *v, v′* within *X* (or *Y*) and the reverse of a shortest *u, v′*-path.



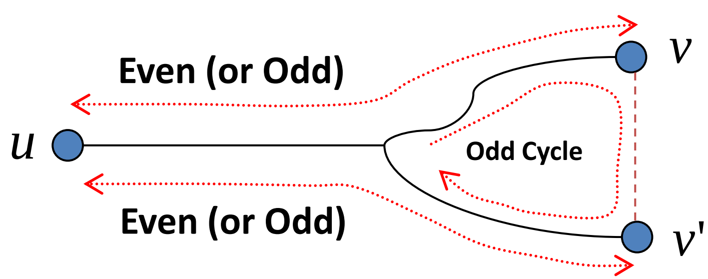
A closed odd walk using

1) **a shortest *u, v*-path,**
2) **the edge *v, v′* within *X* (or *Y*) , and**
3) **the reverse of a shortest *u, v′*-path.**

Theorem: A graph is bipartite if it has no odd cycle.

Proof: (sufficiency 3/3)

- By Lemma 1.2.15, such a walk must contain an odd cycle, which contradicts our hypothesis
- Hence ***X*** and ***Y*** are independent sets. Also ***X***∪***Y*** = ***V***(***H***), so ***H*** is an ***X***, ***Y***-bipartite graph



Because:

even **(or *odd*) +** ***even*** **(or *odd*) =** ***even***

even **+ 1 =** ***odd***

Since no odd cycles, *vv'* doesn't exist.

We have:

X **and** ***Y*** **are independent sets**

Theorem: A graph is bipartite only if it has no odd cycle.

Proof: (necessity)

- Let ***G*** be a bipartite graph.
- Every walk alternates between the two sets of a bipartition
- So every return to the original partite set happens after an even number of steps
- Hence ***G*** has no odd cycle

Conclusion

- In this lecture, we have discussed the useful properties of connection, paths, and cycles and how the statements in graph theory can be proved using the principle of induction.

- In upcoming lecture, we will discuss eulerian circuits, the fundamental parameters of a graph i.e. the degree of the vertices, counting and extremal problems.