CS 207: Discrete Structures

Graph theory

Bipartite subgraphs, graph isomorphism

Lecture 26
Sept 22 2015

Topic 3: Graph theory

Topics covered

- Eulerian graphs and a characterization
- Characterization of bipartite graphs using odd cycles.
- Subgraphs and degree sum formula.
- Cliques and independent sets.

Today

- Finding a large bipartite subgraph of a given graph
- Graph representation as a matrix.
- Comparing graphs: isomorphism

Bipartite graphs

Definition

A graph is called bipartite, if the vertices of the graph can be partitioned into $V = X \cup Y$, $X \cap Y = \emptyset$ s.t., $\forall e = (u, v) \in E$,

- either $u \in X$ and $v \in Y$
- or $v \in X$ and $u \in Y$

Example: m jobs and n people, k courses and ℓ students.

- How can we check if a graph is bipartite?
- Can we characterize bipartite graphs?

Some basic stuff that we have already seen

Degree-Sum Formula (also called Handshake Lemma!)

For any graph G with vertex set V and edge set E:

$$\sum_{v \in V} d(v) = 2|E|$$

Some basic stuff that we have already seen

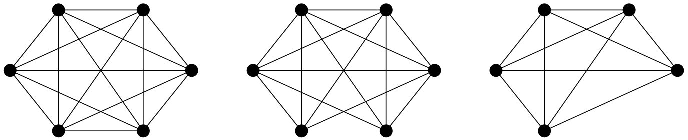
Degree-Sum Formula (also called Handshake Lemma!)

For any graph G with vertex set V and edge set E:

$$\sum_{v \in V} d(v) = 2|E|$$

Subgraphs of a graph G

A subgraph H of a graph G is a graph H such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ (and the assignment of endpoints to edges in H is same as in G).



Bipartite subgraphs of graphs

- Does a graph G always have a bipartite subgraph?
- How large is it?

Bipartite subgraphs of graphs

- Does a graph G always have a bipartite subgraph?
- How large is it?

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

Bipartite subgraphs of graphs

- Does a graph G always have a bipartite subgraph?
- How large is it?

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

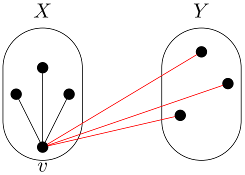
Proof: (by extremality principle)

Bipartite subgraphs of graphs

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

Proof: (by extremality principle) Let H be the bipartite subgraph of G s.t, # edges crossing bipartition is max possible.



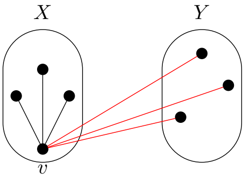
Bipartite subgraphs of graphs

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

Proof: (by extremality principle) Let H be the bipartite subgraph of G s.t, # edges crossing bipartition is max possible.

- Either, $\forall v \in V(G), d_H(v) \geq \frac{d_G(v)}{2}$, then $|E(H)| \geq \frac{|E(G)|}{2}$. $d_H(v)$ denotes degree of v in H and $d_G(v)$ its degree in G.



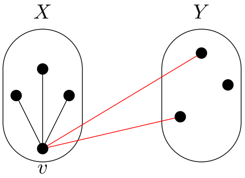
Bipartite subgraphs of graphs

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

Proof: (by extremality principle) Let H be the bipartite subgraph of G s.t, # edges crossing bipartition is max possible.

- Either, $\forall v \in V(G), d_H(v) \geq \frac{d_G(v)}{2}$, then $|E(H)| \geq \frac{|E(G)|}{2}$.
- Or, $\exists v \in V(G)$ such that $d_H(v) \leq \frac{d_G(v)}{2}$.



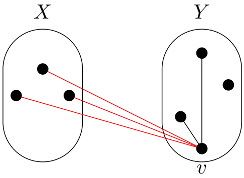
Bipartite subgraphs of graphs

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

Proof: (by extremality principle) Let H be the bipartite subgraph of G s.t, # edges crossing bipartition is max possible.

- Either, $\forall v \in V(G), d_H(v) \geq \frac{d_G(v)}{2}$, then $|E(H)| \geq \frac{|E(G)|}{2}$.
- Or, $\exists v \in V(G)$ such that $d_H(v) \leq \frac{d_G(v)}{2}$. Then switching v to other partition contradicts choice of H.



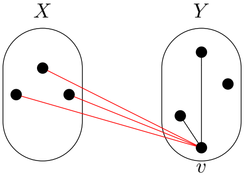
Bipartite subgraphs of graphs

Theorem

Every simple graph G has a bipartite subgraph with at least $|E|/2$ edges.

Proof: (by extremality principle) Let H be the bipartite subgraph of G s.t, # edges crossing bipartition is max possible.

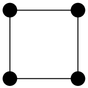
- Either, $\forall v \in V(G), d_H(v) \geq \frac{d_G(v)}{2}$, then $|E(H)| \geq \frac{|E(G)|}{2}$.
- Or, $\exists v \in V(G)$ such that $d_H(v) \leq \frac{d_G(v)}{2}$. Then switching v to other partition contradicts choice of H. $\square$



- Can we make this into an algorithm to produce such H?

Representing and comparing graphs

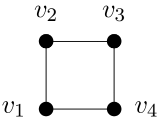
We start with simple graphs...



To represent it, we need to name the vertices...

Representing and comparing graphs

We start with simple graphs...



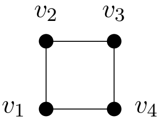
To represent it, we need to name the vertices...

- As an adjacency list:

v_1	v_2, v_4
v_2	v_1, v_3
v_3	v_2, v_4
v_4	v_1, v_3

Representing and comparing graphs

We start with simple graphs...



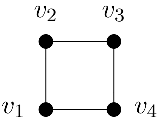
To represent it, we need to name the vertices...

- As an adjacency matrix:

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

Representing and comparing graphs

We start with simple graphs...



To represent it, we need to name the vertices...

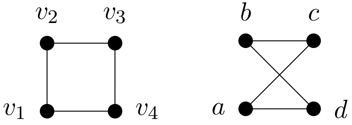
- As an adjacency matrix:

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

- But we want to study properties that are independent of the naming, e.g., connectivity.
- Are two given graphs the "same", wrt these properties?

Representing and comparing graphs

We start with simple graphs...



To represent it, we need to name the vertices...

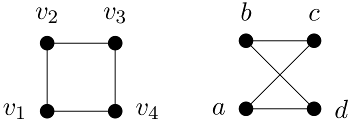
- As an adjacency matrix:

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

- But we want to study properties that are independent of the naming, e.g., connectivity.
- Are two given graphs the "same", wrt these properties?

Representing and comparing graphs

We start with simple graphs...



To represent it, we need to name the vertices...

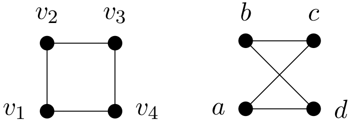
- As an adjacency matrix:

$$
\begin{array}{c}
 \\ v_1 \\ v_2 \\ v_3 \\ v_4
\end{array}
\begin{array}{c}
\begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}
\end{array}
\qquad
\begin{array}{c}
 \\ a \\ b \\ c \\ d
\end{array}
\begin{array}{c}
\begin{array}{cccc} a & b & c & d \end{array} \\
\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}
\end{array}
$$

- But we want to study properties that are independent of the naming, e.g., connectivity.
- Are two given graphs the "same", wrt these properties?

Representing and comparing graphs

We start with simple graphs...



To represent it, we need to name the vertices...

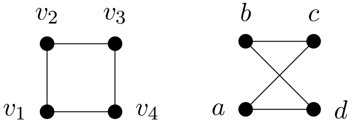
- As an adjacency matrix:

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array}\begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ v_1 \\ v_3 \\ v_2 \\ v_4 \end{array}\begin{array}{c} \begin{array}{cccc} v_1 & v_3 & v_2 & v_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ a \\ b \\ c \\ d \end{array}\begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

- But we want to study properties that are independent of the naming, e.g., connectivity.
- Are two given graphs the "same", wrt these properties?

Representing and comparing graphs

We start with simple graphs...



To represent it, we need to name the vertices...

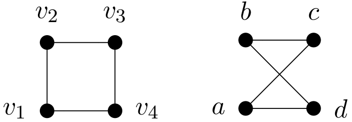
- As an adjacency matrix:

$$\begin{array}{c|c} & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} & \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \qquad \begin{array}{cc} & \begin{array}{cccc} v_1 & v_3 & v_2 & v_4 \end{array} \\ \begin{array}{c} v_1 \\ v_3 \\ v_2 \\ v_4 \end{array} & \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array} \qquad \begin{array}{cc} & \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \end{array} & \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

- Reordering of vertices is same as applying a permutation to rows and colums of $A(G)$.
- So, it seems two graphs are "same" if by reordering and renaming the vertices we get the same graph/matrix.

Representing and comparing graphs

We start with simple graphs...



To represent it, we need to name the vertices...

- As an adjacency matrix:

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \quad \begin{array}{c} \\ v_1 \\ v_3 \\ v_2 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_3 & v_2 & v_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array} \quad \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

- Reordering of vertices is same as applying a permutation to rows and colums of $A(G)$.
- So, it seems two graphs are "same" if by reordering and renaming the vertices we get the same graph/matrix.
- How do we formalize this?

Isomorphism

Definition

An isomorphism from simple graph G to H is a bijection $f : V(G) \to V(H)$ such that $uv \in E(G)$ iff $f(u)f(v) \in E(H)$.

Isomorphism

Definition

An isomorphism from simple graph G to H is a bijection $f : V(G) \to V(H)$ such that $uv \in E(G)$ iff $f(u)f(v) \in E(H)$.

- Thus, it is a bijection that "preserves" the edge relation.
- Can be checked using adjacency matrix by reordering/renaming.
- What are the properties of this function/relation: $R = \{(G, H) \mid \exists \text{ an isomorphism from } G \text{ to } H\}$.

Isomorphism

Definition

An isomorphism from simple graph G to H is a bijection $f : V(G) \to V(H)$ such that $uv \in E(G)$ iff $f(u)f(v) \in E(H)$.

- Thus, it is a bijection that "preserves" the edge relation.
- Can be checked using adjacency matrix by reordering/renaming.
- What are the properties of this function/relation: $R = \{(G, H) \mid \exists$ an isomorphism from G to $H\}$.

Proposition

The isomorphism relation is an equivalence relation.

Isomorphism

Definition

An isomorphism from simple graph G to H is a bijection $f : V(G) \to V(H)$ such that $uv \in E(G)$ iff $f(u)f(v) \in E(H)$.

- Thus, it is a bijection that "preserves" the edge relation.
- Can be checked using adjacency matrix by reordering/renaming.
- What are the properties of this function/relation: $R = \{(G, H) \mid \exists \text{ an isomorphism from } G \text{ to } H\}$.

Proposition

The isomorphism relation is an equivalence relation.

- The equivalence classes are called isomorphism classes.

Isomorphism

Definition

An isomorphism from simple graph G to H is a bijection $f : V(G) \to V(H)$ such that $uv \in E(G)$ iff $f(u)f(v) \in E(H)$.

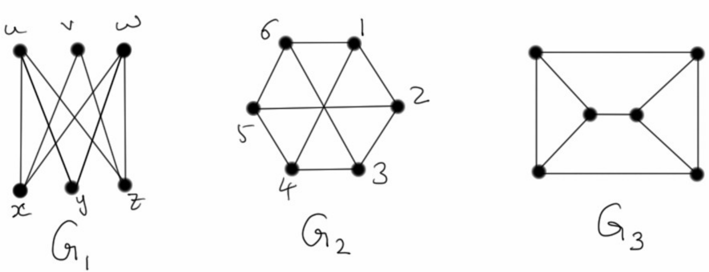
- Thus, it is a bijection that "preserves" the edge relation.
- Can be checked using adjacency matrix by reordering/renaming.
- What are the properties of this function/relation: $R = \{(G, H) \mid \exists \text{ an isomorphism from } G \text{ to } H\}$.

Proposition

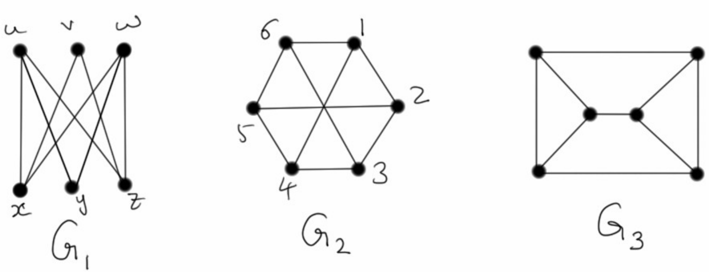
The isomorphism relation is an equivalence relation.

- The equivalence classes are called isomorphism classes.
- When we talked about an "unlabeled" graph till now, we actually meant the isomorphism class of that graph!

Graph isomorphism

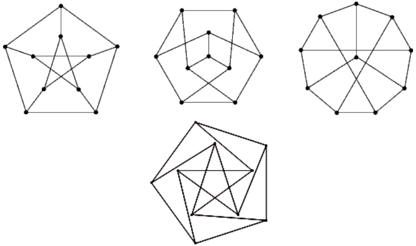


Graph isomorphism



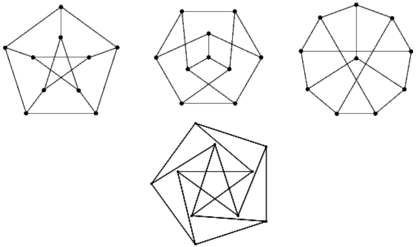
- To show that two graphs are isomorphic, you have to
  1. give names to vertices
  2. specify a bijection
  3. check that it preserves the adjacency relation
- To show that two graphs are non-isomorphic, find a structural property that is different.

Is checking graph isomorphism easy?



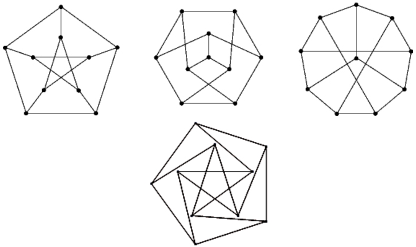
- Qn: Which of these graphs are isomorphic?

Is checking graph isomorphism easy?



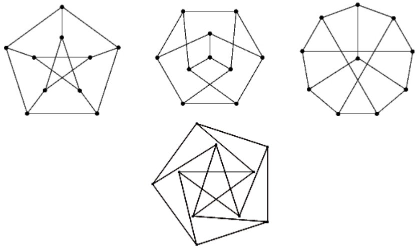
- Qn: Which of these graphs are isomorphic?
- A: All of them!

Is checking graph isomorphism easy?



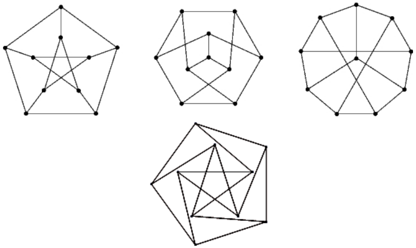
- Qn: Which of these graphs are isomorphic?
- A: All of them!
- This graph is called the Petersen graph and has some very interesting propreties.

Is checking graph isomorphism easy?



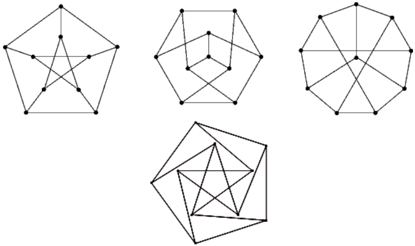
- Qn: Which of these graphs are isomorphic?
- A: All of them!
- This graph is called the Petersen graph and has some very interesting propreties.
  - vertices are 2-element subsets of 5-element set and edges are pairs of disjoint 2-element subsets.

Is checking graph isomorphism easy?



- Qn: Which of these graphs are isomorphic?
- A: All of them!
- This graph is called the Petersen graph and has some very interesting propreties.
  - vertices are 2-element subsets of 5-element set and edges are pairs of disjoint 2-element subsets.
  - 2 vertices that do not share an edge, have exactly 1 common nbr.

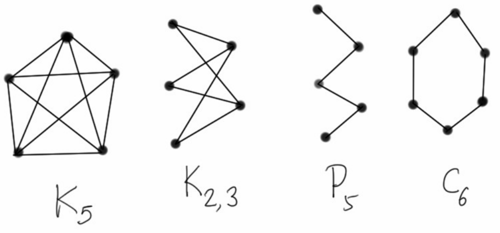
Is checking graph isomorphism easy?



- Qn: Which of these graphs are isomorphic?
- A: All of them!
- This graph is called the Petersen graph and has some very interesting propreties.
  - vertices are 2-element subsets of 5-element set and edges are pairs of disjoint 2-element subsets.
  - 2 vertices that do not share an edge, have exactly 1 common nbr.

Further reading: Graph and sub-graph isomorphism problems.

Some special graphs and notations



- Complete graphs K_n
- Complete bipartite graphs $K_{i,j}$
- Paths P_n
- Cycles C_n

Some special graphs and notations

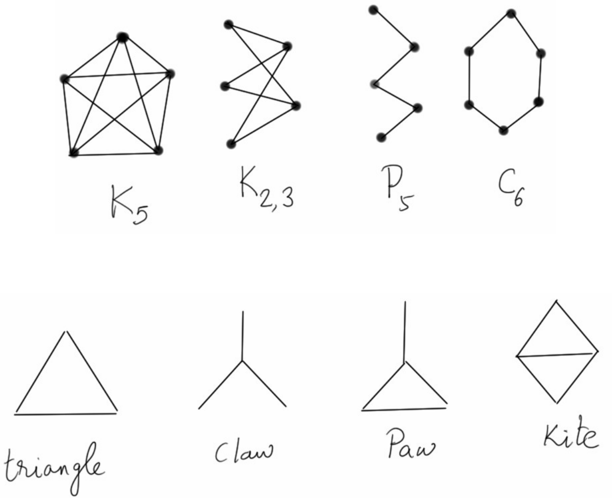


Figure: A whole graph zoo!

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

Theorem

If G is isomorphic to H, then the following properties are preserved:

1. G, H have same # vertices.
2. G, H have same # edges.

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

- Are C_5 and $P_5 \cup \{e\}$ isomorphic?

Theorem

If G is isomorphic to H, then the following properties are preserved:

1. G, H have same # vertices.
2. G, H have same # edges.

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

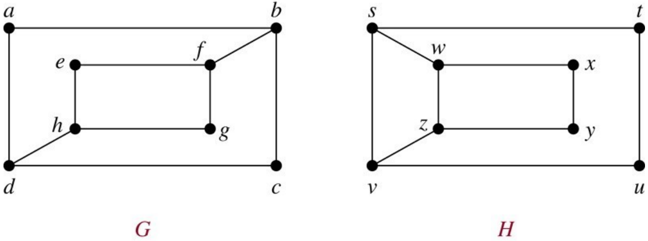
Theorem

If G is isomorphic to H, then the following properties are preserved:

1. G, H have same # vertices.
2. G, H have same # edges.
3. G, H have the same # vertices of degree k, $\forall k \in \mathbb{N}$.

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.



Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

Theorem

If G is isomorphic to H, then the following properties are preserved:

1. G, H have same # vertices.
2. G, H have same # edges.
3. G, H have the same # vertices of degree k, $\forall k \in \mathbb{N}$.

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

Theorem

If G is isomorphic to H, then the following properties are preserved:

1. G, H have same # vertices.
2. G, H have same # edges.
3. G, H have the same # vertices of degree k, $\forall k \in \mathbb{N}$.
4. G has k paths/cycles of length r iff H has k paths/cycles of length r.

Properties of isomorphic graphs

Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

Theorem

If G is isomorphic to H, then the following properties are preserved:

1. G, H have same # vertices.
2. G, H have same # edges.
3. G, H have the same # vertices of degree k, $\forall k \in \mathbb{N}$.
4. G has k paths/cycles of length r iff H has k paths/cycles of length r.
5. G is bipartite iff H is bipartite.
6. G contains K_n as a subgraph iff H does.
7. …