CS 105: DIC on Discrete Structures

Graph theory

Graph Isomorphism

Lecture 31
Oct 15 2024

Part 4: Graph theory

Recap of last four lectures:

1. Basics: graphs, paths, cycles, walks, trails; connected graphs.
2. Eulerian graphs and a characterization in terms of degrees of vertices.
3. Bipartite graphs and a characterization in terms of odd length cycles.

Reference: Sections 1.1-1.3 of Chapter 1 from Douglas West.

Part 4: Graph theory

Recap of last four lectures:

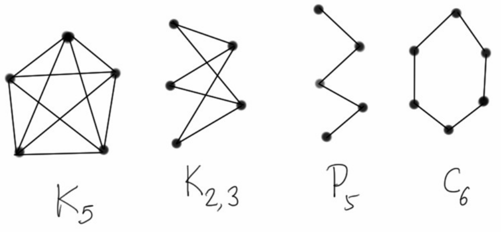
1. Basics: graphs, paths, cycles, walks, trails; connected graphs.
2. Eulerian graphs and a characterization in terms of degrees of vertices.
3. Bipartite graphs and a characterization in terms of odd length cycles.

Reference: Sections 1.1-1.3 of Chapter 1 from Douglas West.

Today

Graph representations and naming

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

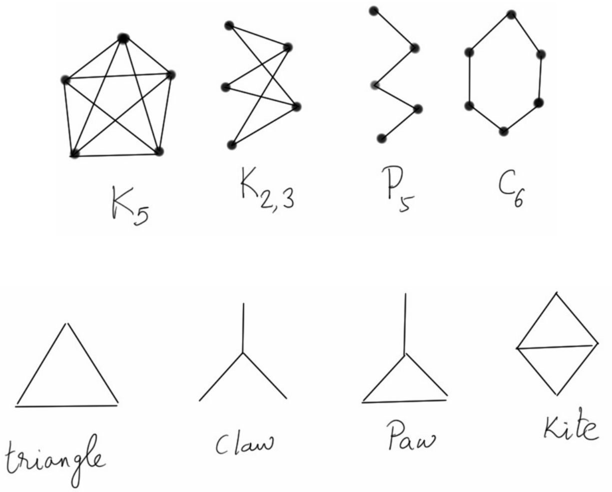


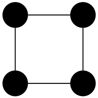
Figure: A whole graph zoo!

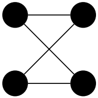
Are these graphs the same?

Are these graphs the same?

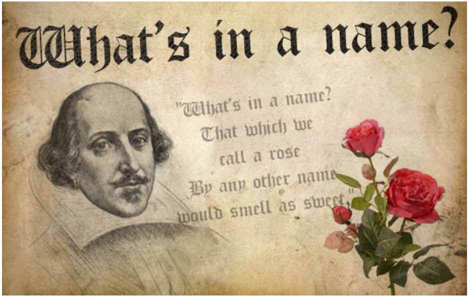
- To compare graphs, we need to name them!

Are these graphs the same?



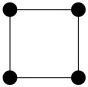


- To compare graphs, we need to name them!



Representing and comparing graphs

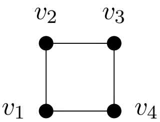
We start with simple graphs...



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

Representing and comparing graphs

We start with simple graphs...



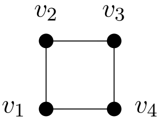
To represent it, we need to name the vertices...

- 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...



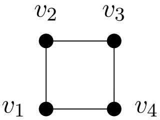
To represent it, we need to name the vertices...

- 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...

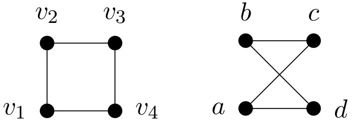
- As an adjacency matrix:

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

- 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...

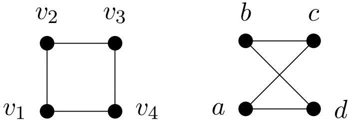
- As an adjacency matrix:

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

- 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:

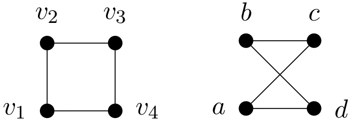
	v_1	v_2	v_3	v_4
v_1	0	1	0	1
v_2	1	0	1	0
v_3	0	1	0	1
v_4	1	0	1	0

	a	b	c	d
a	0	0	1	1
b	0	0	1	1
c	1	1	0	0
d	1	1	0	0

- 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...

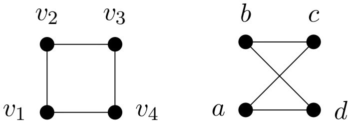
- As an adjacency matrix:

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

- 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:

	v_1	v_2	v_3	v_4
v_1	0	1	0	1
v_2	1	0	1	0
v_3	0	1	0	1
v_4	1	0	1	0

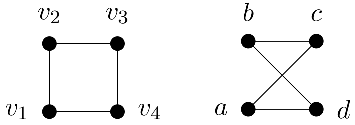
	v_1	v_3	v_2	v_4
v_1	0	0	1	1
v_3	0	0	1	1
v_2	1	1	0	0
v_4	1	1	0	0

	a	b	c	d
a	0	0	1	1
b	0	0	1	1
c	1	1	0	0
d	1	1	0	0

- 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:

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

- 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.

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: $R = \{(G, H) \mid \exists \text{ an isomorphism from } G \text{ to } H\}$.

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: $R = \{(G, H) \mid \exists$ an isomorphism from G 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!

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.

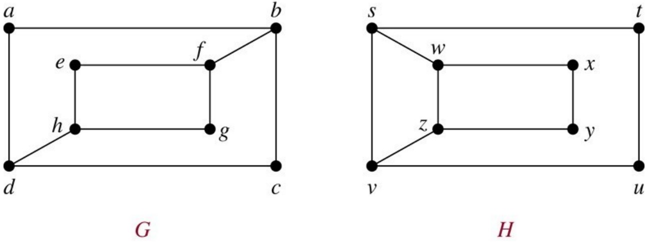
Theorem

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

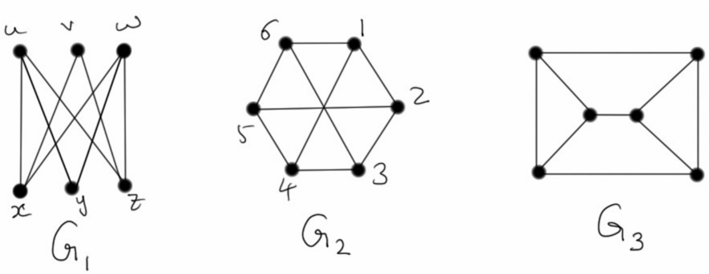
Intuitively, if two graphs are isomorphic then all structural properties, i.e., properties that do not depend on the naming of vertices are preserved.

Theorem

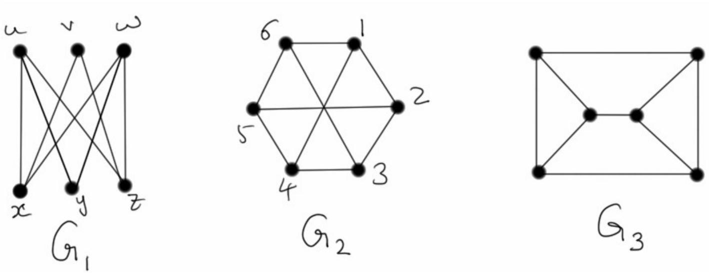
If G is isomorphic to H, then the following properties are preserved:

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. $\ldots$

Graph isomorphism

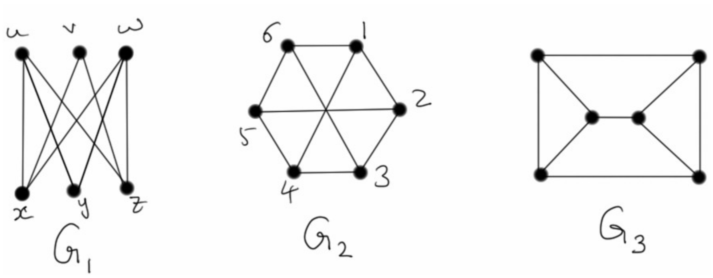


Graph isomorphism



Exercise 1: Which of these graphs are isomorphic? Justify!

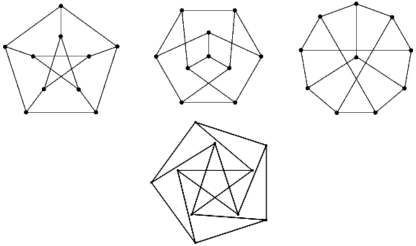
Graph isomorphism



Exercise 1: Which of these graphs are isomorphic? Justify!

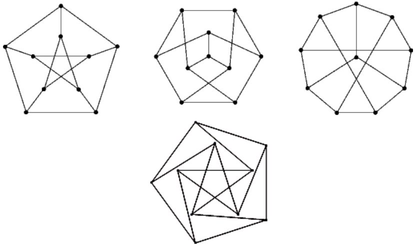
- To show that two graphs are isomorphic, you have to
  1. give names to vertices
  2. specify a bijection
  3. check that it preserves the adjacency relation
- To show that two graphs are non-isomorphic, find a structural property that is different.

Is checking graph isomorphism easy?



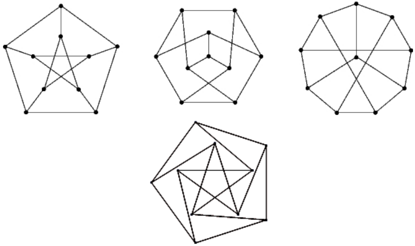
- Exercise 2: Which of these graphs are isomorphic?

Is checking graph isomorphism easy?



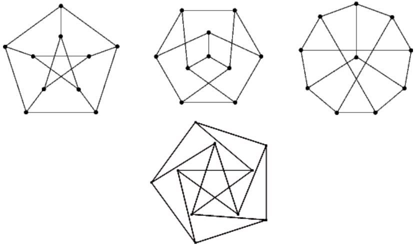
- Exercise 2: Which of these graphs are isomorphic?
- A: All of them!

Is checking graph isomorphism easy?



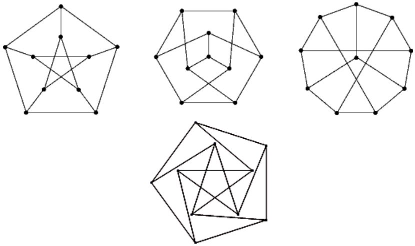
- Exercise 2: Which of these graphs are isomorphic?
- A: All of them!
- This graph is called the Petersen graph and has some very interesting propreties.

Is checking graph isomorphism easy?



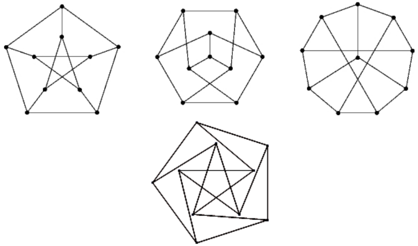
- Exercise 2: Which of these graphs are isomorphic?
- A: All of them!
- This graph is called the Petersen graph and has some very interesting propreties.
  - vertices are 2-element subsets of 5-element set and edges are pairs of disjoint 2-element subsets.

Is checking graph isomorphism easy?



- Exercise 2: 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?



- Exercise 2: 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.

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

What if $G = H$?

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

What if $G = H$?

An automorphism of G is an isomorphism from G to itself, i.e. a bijection $f : V(G) \to V(G)$ s.t. $uv \in E(G)$ iff $f(u)f(v) \in E(G)$.

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

What if $G = H$?

An automorphism of G is an isomorphism from G to itself, i.e. a bijection $f : V(G) \to V(G)$ s.t. $uv \in E(G)$ iff $f(u)f(v) \in E(G)$.

- What are the automorphisms of P_4?

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

What if $G = H$?

An automorphism of G is an isomorphism from G to itself, i.e. a bijection $f : V(G) \to V(G)$ s.t. $uv \in E(G)$ iff $f(u)f(v) \in E(G)$.

- What are the automorphisms of P_4?
- How many automorphisms does K_n have?

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

What if $G = H$?

An automorphism of G is an isomorphism from G to itself, i.e. a bijection $f : V(G) \to V(G)$ s.t. $uv \in E(G)$ iff $f(u)f(v) \in E(G)$.

- What are the automorphisms of P_4?
- How many automorphisms does K_n have?
- How many automorphisms does $K_{r,s}$ have?

Graph Automorphisms

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 $f(u)f(v) \in E(H)$.

An automorphism of G is an isomorphism from G to itself, i.e. a bijection $f : V(G) \to V(G)$ s.t. $uv \in E(G)$ iff $f(u)f(v) \in E(G)$.

Automorphisms are a measure of symmetry.

Practical applications in graph drawing, visualization, molecular symmetry, structured boolean satisfiability, formal verification