GRAPH THEORY – LECTURE 2
STRUCTURE AND REPRESENTATION — PART A

Abstract. Chapter 2 focuses on the question of when two graphs are to be regarded as "the same", on symmetries, and on subgraphs. §2.1 discusses the concept of graph isomorphism. §2.2 presents symmetry from the perspective of automorphisms. §2.3 introduces subgraphs.

Outline

2.1 Graph Isomorphism
2.2 Automorphisms and Symmetry
2.3 Subgraphs, part 1

1. Graph Isomorphism

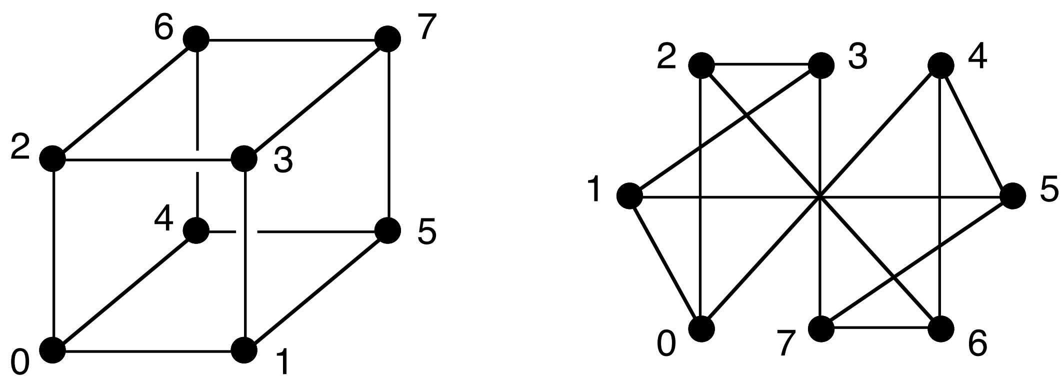


Figure 1.1: Two different drawings of the same graph.

They are (clearly) the "same" because each vertex v has the **exact same set of neighbors** in both graphs.

0.	1	2	4
1.	0	3	5
2.	0	4	6
3.	1	2	7
4.	0	5	6
5.	1	4	6
6.	2	4	7
7.	3	5	6

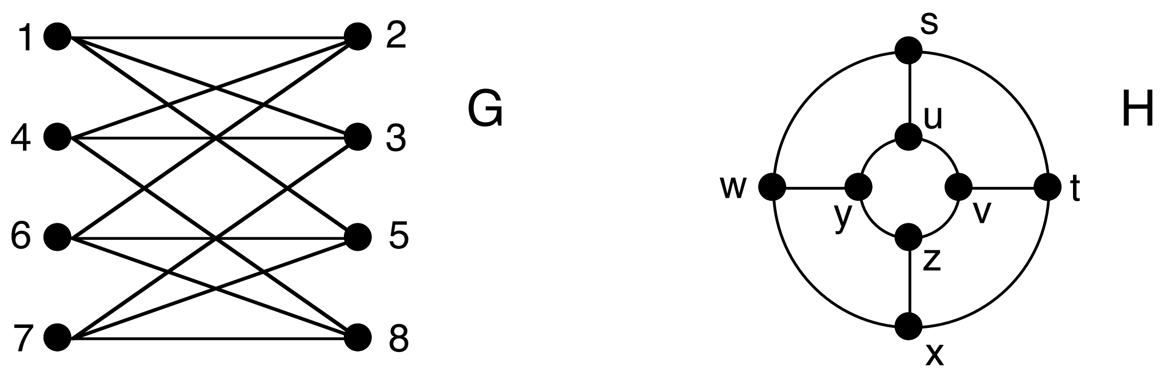


Figure 1.2: Two more drawings of that same graph.

These two graphs are the "same" because, instead of having the same set of vertices, this time we have a bijection $V_G \to V_H$

$$\begin{array}{cccc} 1 \to s & 2 \to t & 3 \to u & 4 \to v \\ 5 \to w & 6 \to x & 7 \to y & 8 \to z \end{array}$$

between the two vertex sets, such that

Nbhds map bijectively to nbhds;
e.g., $N(1) \mapsto N(f(1)) = N(s)$, i.e.

$$N(1) \;=\; \{2,3,5\} \;\mapsto\; \{t,u,w\} \;=\; \{f(2),f(3),f(5)\} \;=\; N(s)$$

Structural Equivalence for Simple Graphs

Def 1.1. Let G and H be two simple graphs. A vertex function $f : V_G \to V_H$ ***preserves adjacency*** if

> for every pair of adjacent vertices u and v in graph G,
> the vertices $f(u)$ and $f(v)$ are adjacent in graph H.

Similarly, f ***preserves non-adjacency*** if

> $f(u)$ and $f(v)$ are non-adj whenever u and v are non-adj.

Def 1.2. A vertex bijection $f : V_G \to V_H$ betw. two simple graphs G and H is ***structure-preserving*** if

> it preserves adjacency and non-adjacency.

That is, for every pair of vertices in G,

$$u \text{ and } v \text{ are adj in } G \iff f(u) \text{ and } f(v) \text{ are adj in } H$$

This leads us to a formal mathematical definition of what we mean by the "same" graph.

Def 1.3. Two simple graphs G and H are ***isomorphic***, denoted $G \cong H$, if

$\exists$ a structure-preserving bijection $f : V_G \to V_H$.

Such a function f is called an ***isomorphism*** from G to H.

NOTATION: When we regard a vertex function $f : V_G \to V_H$ as a mapping from one graph to another, we may write $f : G \to H$.

ISOMORPHISM CONCEPT

Two graphs related by isomorphism differ only by the names of the vertices and edges. There is a complete structural equivalence between two such graphs.

REPRESENTATION by DRAWINGS

When the drawings of two isomorphic graphs look different, relabeling reveals the equivalence.

One may use functional notation to specify an isomorphism between the two simple graphs shown.

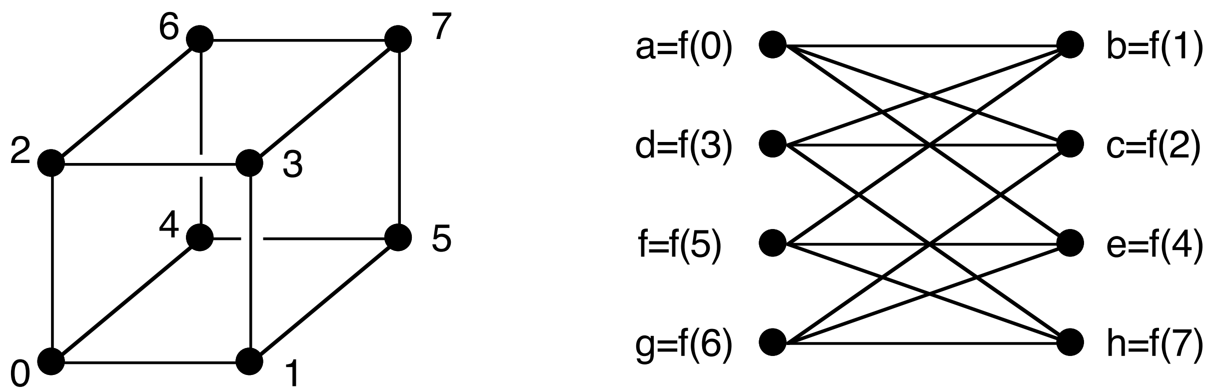


Figure 1.3: Specifying an isom betw two simple graphs.

Alternatively, one may relabel the vertices of the codomain graph with names of vertices in the domain, as in Fig 1.4.

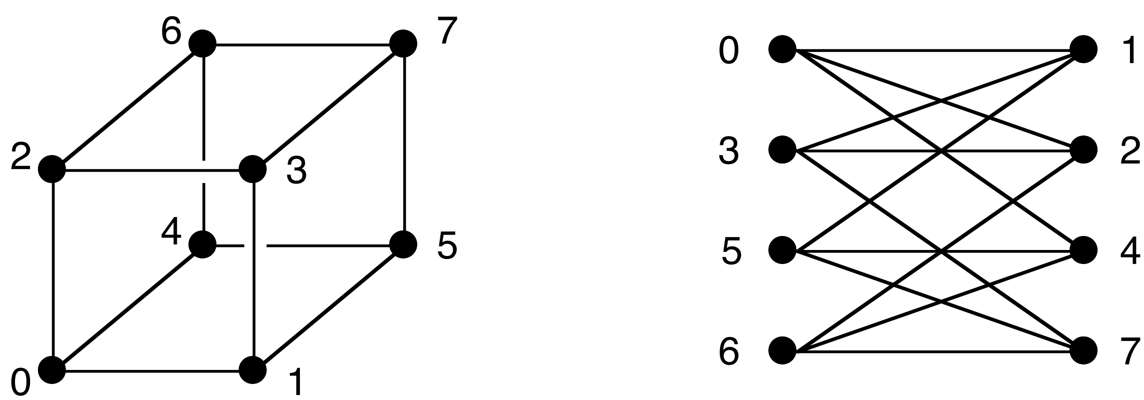


Figure 1.4: Another way of depicting an isomorphism.

ISOMORPHISM (SIMPLE) =
BIJECTIVE on VERTICES
ADJACENCY-PRESERVING
(NON-ADJACENCY)-PRESERVING

Example 1.1. The vertex function $j \mapsto j+4$ depicted in Fig 1.5 is bijective and adjacency-preserving, but it is **not an isomorphism**, since it does not preserve non-adjacency.

In particular, the non-adjacent pair $\{0, 2\}$ maps to the adjacent pair $\{4, 6\}$.

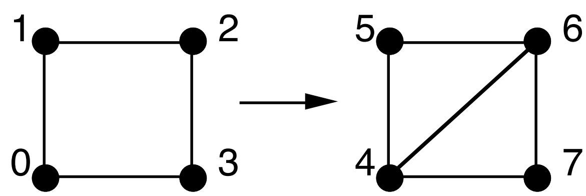


Figure 1.5: Bijective and adj-preserving, but not an isom.

Example 1.2. The vertex function

$$j \mapsto j \ mod \ 2$$

depicted in Figure 1.6 is **structure-preserving**, since it preserves adjacency and non-adjacency, but it is **not an isomorphism** since it is not bijective.

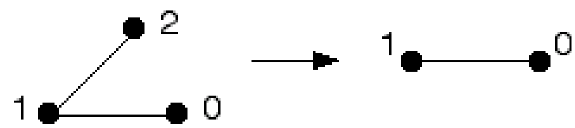


Figure 1.6: Preserves adj and non-adj, but not bijective.

Isomorphism for General Graphs

The mapping $f : V_G \to V_H$ between the vertex-sets of the two graphs shown in Figure 1.7 given by

$$f(i) \ = \ i, \qquad i = 1, 2, 3$$

preserves adjacency and non-adjacency, but the two graphs are clearly not structurally equivalent.

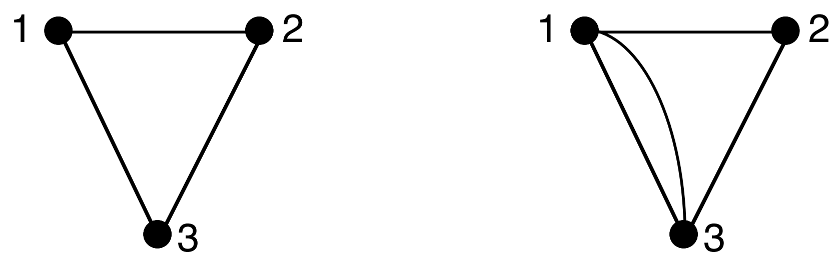
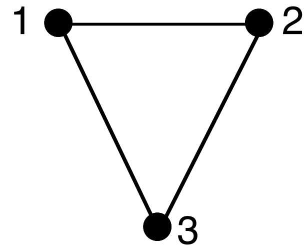


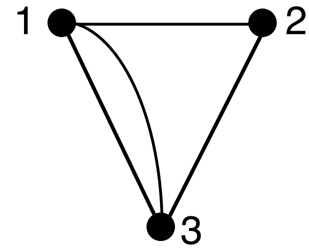
Figure 1.7: These graphs are not structurally equivalent.

Def 1.4. A vertex bijection $f : V_G \to V_H$ between two graphs G and H, simple or general, is ***structure-preserving*** if

(1) the # of edges (even if 0) between every pair of distinct vertices u and v in graph G equals the # of edges between their images $f(u)$ and $f(v)$ in graph H, and

(2) the # of self-loops at each vertex x in G equals the # of self-loops at the vertex $f(x)$ in H.





This general definition of *structure-preserving* reduces, for simple graphs, to our original definition. Moreover, it allows a unified definition of isomorphic graphs for all cases.

Def 1.5. Two graphs G and H (simple or general) are ***isomorphic graphs*** if $\exists$ structure-preserving vertex bijection $f : V_G \to V_H$

This relationship is denoted $G \cong H$.

Isomorphism for Graphs with Multi-Edges

Def 1.6. For isomorphic graphs G and H, a pair of bijections

$$f_V : V_G \to V_H \text{ and } f_E : E_G \to E_H$$

is ***consistent*** if for every edge $e \in E_G$, the function f_V maps the endpoints of e to the endpoints of the edge $f_E(e)$.

Proposition 1.1. *$G \cong H$ iff there is a consistent pair of bijections*

$$f_V : V_G \to V_H \quad \textit{and} \quad f_E : E_G \to E_H$$

Proof. Straightforward from the definitions. □

Remark 1.1. If G and H are isom *simple* graphs, then every structure-preserving vertex bijection $f : V_G \to V_H$ induces a *unique* consistent edge bijection, by the rule: $uv \mapsto f(u)f(v)$.

Def 1.7. If G and H are graphs *with multi-edges*, then an ***isomorphism*** from G to H is specified by giving a consistent pair of bijections $f_V : V_G \to V_H$ and $f_E : E_G \to E_H$.

Example 1.7. Both of the structure-preserving vertex bijections $G \to H$ in Fig 1.8 have six consistent edge-bijections.

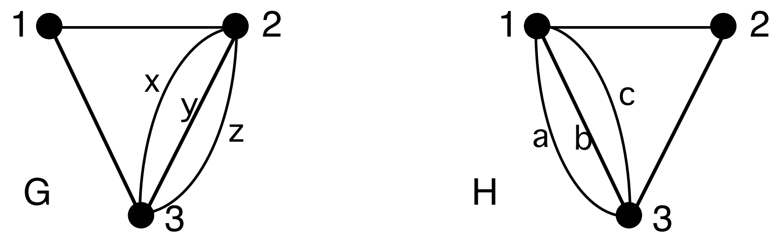


Figure 1.8: There are 12 distinct isoms from G to H.

Necessary Properties of Isom Graph Pairs

Although the examples below involve simple graphs, the properties apply to general graphs as well.

Theorem 1.2. *Let G and H be isomorphic graphs. Then they have the same number of vertices and edges.*

Proof. An isomorphism maps V_G and E_G bijectively. □

Theorem 1.3. *Let $f : G \to H$ be a graph isomorphism and let $v \in V_G$. Then $deg(f(v)) = deg(v)$.*

Proof. Since f is structure-preserving, the # of proper edges and the # of self-loops incident on vertex v equal the corresp #'s for vertex $f(v)$. Thus, $deg(f(v)) = deg(v)$. □

Corollary 1.4. *Let G and H be isomorphic graphs. Then they have the same degree sequence.* □

Corollary 1.5. *Let $f : G \to H$ be a graph isom and $e \in E_G$. Then the endpoints of edge $f(e)$ have the same degrees as the endpoints of e.* □

Example 1.8. In Fig 1.9 below, we observe that Q_3 and CL_4 both have 8 vertices and 12 edges and are 3-regular.

The vertex labelings specify a vertex bijection. A careful examination reveals that this vertex bijection is structure-preserving.
It follows that Q_3 and CL_4 are isomorphic graphs.

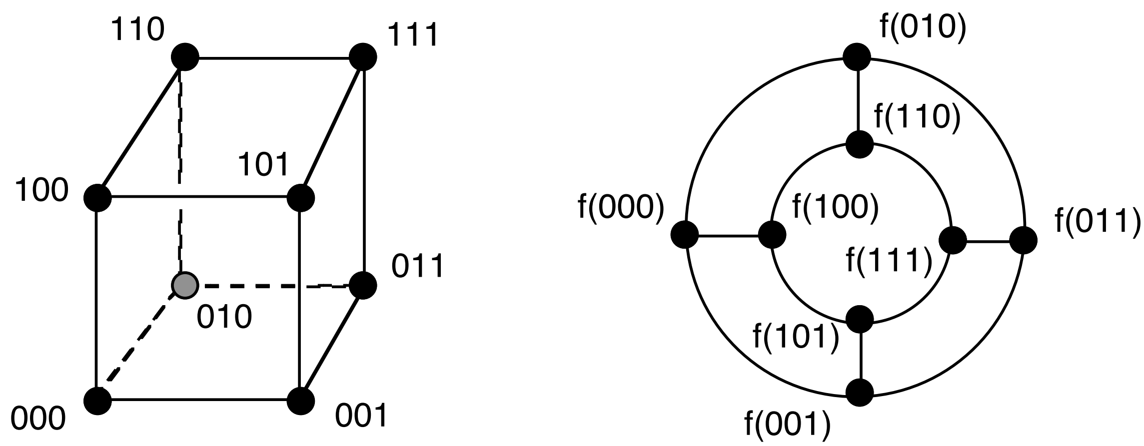


Figure 1.9: Hypercube Q_3 and circ ladder CL_4 are isom.

Def 1.8. The ***Möbius ladder*** ML_n is a graph obtained from the circular ladder CL_n by deleting from the circular ladder two of its parallel curved edges and replacing them with two edges that cross-match their endpoints.

Example 1.9. $K_{3,3}$ and the Möbius ladder ML_3 both have 6 vertices and 9 edges, and both are 3-regular.

The vertex labelings for the two drawings specify an isomorphism.

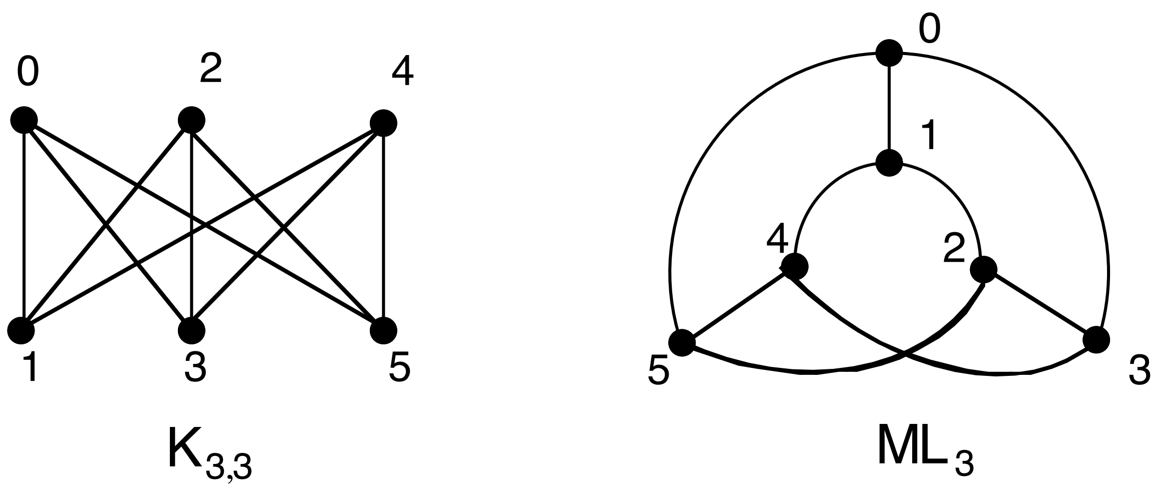


Figure 1.10: $K_{3,3}$ and the Möbius ladder ML_3 are isom.

Isomorphism Type of a Graph

Def 1.9. Each equivalence class under $\cong$ is called an ***isomorphism type***. (Counting isomorphism types of graphs generally involves the algebra of permutation groups — see Chap 14).

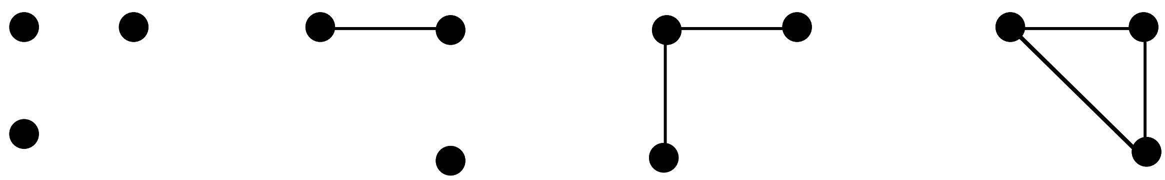


Figure 1.11: The 4 isom types for a simple 3-vertex graph.

Isomorphism of Digraphs

Def 1.10. Two digraphs G and H are ***isomorphic*** if there is an isomorphism f between their underlying graphs that preserves the direction of each edge.

Example 1.10. Notice that non-isomorphic digraphs can have underlying graphs that are isomorphic.

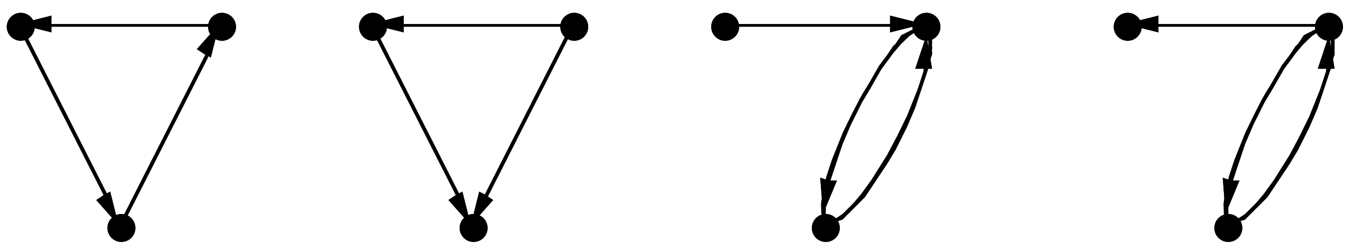


Figure 1.12: Four non-isomorphic digraphs.

Def 1.11. The ***graph-isomorphism problem*** is to devise a practical general algorithm to decide graph isomorphism, or, alternatively, to prove that no such algorithm exists.

2. Automorphisms & Symmetry

Def 2.1. An isomorphism from a graph G to itself is called an ***automorphism***.

Thus, an automorphism π of graph G is a structure-preserving *permutation*

$$\pi_V \text{ on } V_G$$

along with a (consistent) permutation

$$\pi_E \text{ on } E_G$$

We may write $\pi = (\pi_V, \pi_E)$.

Remark 2.1. The proportion of vertex-permutations of V_G that are structure-preserving is a measure of the *symmetry* of G.

Permutations and Cycle Notation

The most convenient representation of a permutation, for our present purposes, is as a *product of disjoint cycles*.

Remark 2.2. As explained in Appendix A4, every permutation can be written as a composition of disjoint cycles.

Example 2.1. The permutation

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 7 & 4 & 1 & 8 & 5 & 2 & 9 & 6 & 3 \end{pmatrix}$$

which maps 1 to 7, 2 to 4, and so on, has the ***disjoint cycle form***

$$\pi = \begin{pmatrix} 1 & 7 & 9 & 3 \end{pmatrix} \begin{pmatrix} 2 & 4 & 8 & 6 \end{pmatrix} \begin{pmatrix} 5 \end{pmatrix}$$

Geometric Symmetry

A geometric symmetry on a graph drawing can be used to represent an automorphism on the graph.

Example 2.2. $K_{1,3}$ has six automorphisms. Each of them is realizable by a rotation or reflection of Fig 2.2.

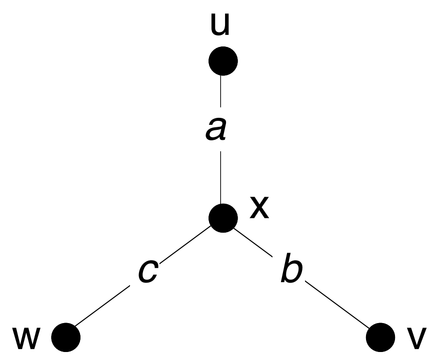
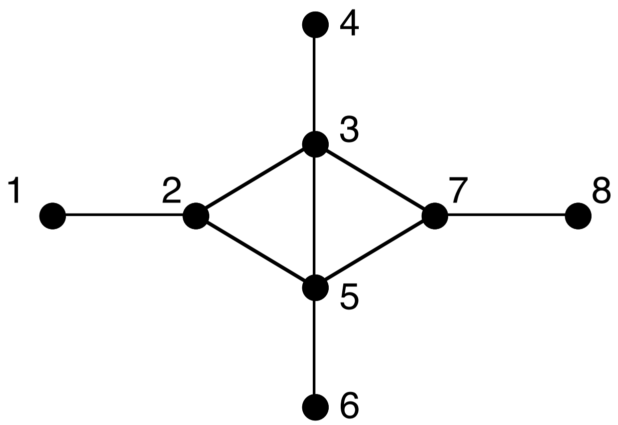


Figure 2.1: The graph $K_{1,3}$.

Symmetry	Vertex permutation	Edge permutation
identity	$(u)\ (v)\ (w)\ (x)$	$(a)\ (b)\ (c)$
120° rotation	$(x)\ (u\ v\ w)$	$(a\ b\ c)$
240° rotation	$(x)\ (u\ w\ v)$	$(a\ c\ b)$
refl. thru a	$(x)\ (u)\ (v\ w)$	$(a)\ (b\ c)$
refl. thru b	$(x)\ (v)\ (u\ w)$	$(b)\ (a\ c)$
refl. thru c	$(x)\ (w)\ (u\ v)$	$(c)\ (a\ b)$

There are no other automorphisms of $K_{1,3}$.

Example 2.3. It is easy to verify that these vertex-perms are structure-preserving, so they are all graph automorphisms.



Automorphisms

$\lambda_0 = (1)(2)(3)(4)(5)(6)(7)(8)$

$\lambda_1 = (1\ 8)(2\ 7)(3)(4)(5)(6)$

$\lambda_2 = (1)(2)(3\ 5)(4\ 6)(7)(8)$

$\lambda_3 = (1\ 8)(2\ 7)(3\ 5)(4\ 6)$

Figure 2.2: A graph with four automorphisms.

Limitations of Geometric Symmetry

Example 2.4. The leftmost drawing has 5-fold rotational symmetry that corresponds to the automorphism (0 1 2 3 4) (5 6 7 8 9), but this automorphism does not correspond to any geometric symmetry of either of the other two drawings.

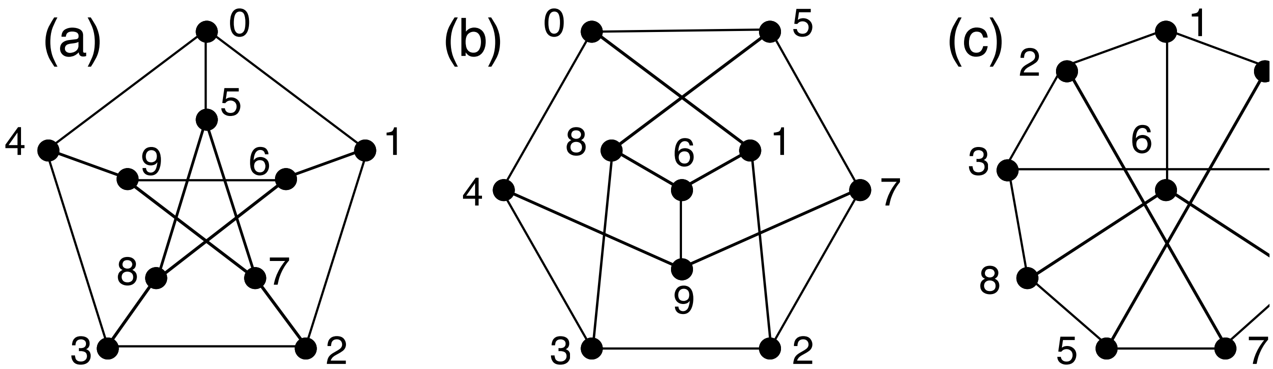


Figure 2.3: Three drawings of the Petersen graph.

Vertex- and Edge-Transitive Graphs

Def 2.2. A graph G is ***vertex-transitive*** if for every vertex pair $u, v \in V_G$, there is an automorphism that maps u to v.

Def 2.3. A graph G is ***edge-transitive*** if for every edge pair $d, e \in E_G$, there is an automorphism that maps d to e.

Example 2.5. $K_{1,3}$ is edge-trans, but not vertex-trans, since every autom must map the 3-valent vertex to itself.

Example 2.7. The hypercube graph Q_n is vertex-trans and edge-trans for every n. (See Exercises.)

Example 2.8. Every circulant graph $circ(n; S)$ is vertex-transitive. In particular, the vertex function $i \mapsto i + k \mod n$ is an automorphism. Although $circ(13 : 1, 5)$ is edge-transitive, some circulant graphs are not. (See Exercises.)

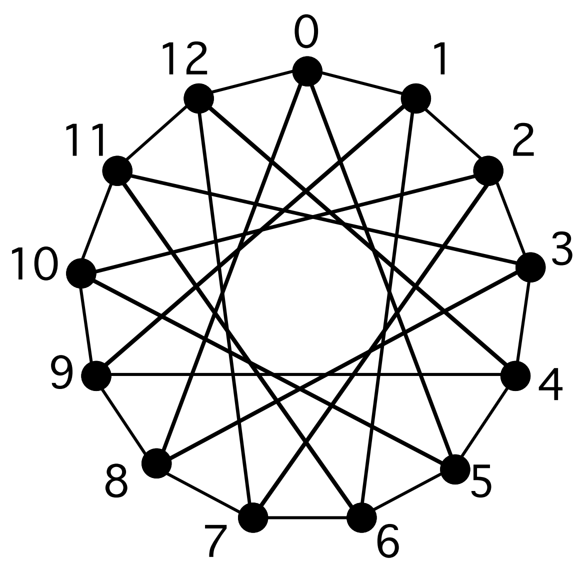


Figure 2.4: The circulant graph $circ(13 : 1, 5)$.

Vertex Orbits and Edge Orbits

Def 2.4. The equivalence classes of the vertices of a graph G under the action of the automorphisms are called ***vertex orbits***. The equivalence classes of the edges are called ***edge orbits***.

Example 2.10.

$$\begin{array}{rllll} \text{vertex orbits:} & \{1,8\}, & \{4,6\}, & \{2,7\}, & \{3,5\} \\ \text{edge orbits:} & \{12,78\}, & \{34,56\}, & \{23,25,37,57\}, & \{35\} \end{array}$$

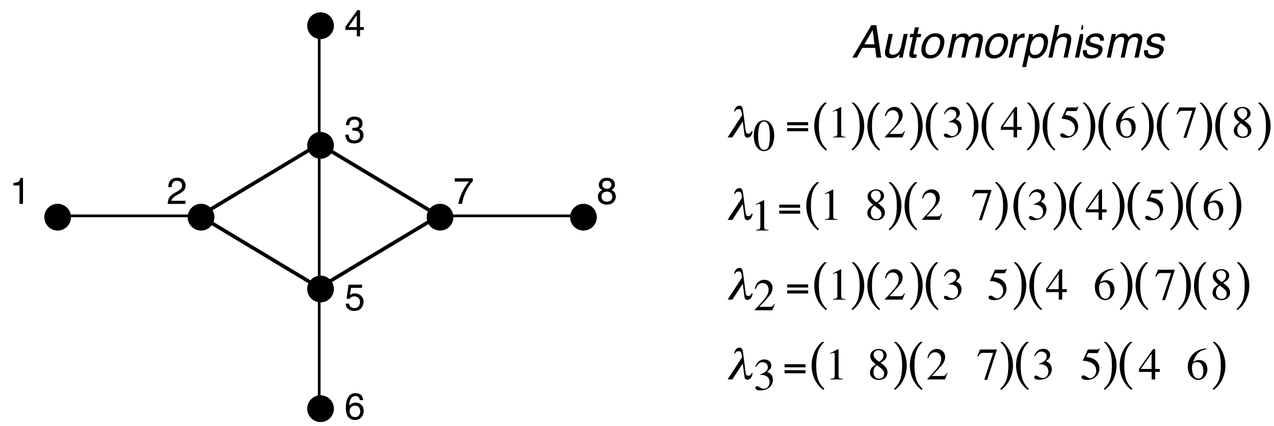


Figure 2.5: Graph of Example 2.3.

Theorem 2.1. *All vertices in the same orbit have the exact same degree.*

Proof. This follows immediately from Theorem 1.3. □

Theorem 2.2. *All edges in the same orbit have the same pair of degrees at their endpoints.*

Proof. This follows immediately from Corollary 1.5. □

Example 2.11. Each of the two partite sets of $K_{m,n}$ is a vertex orbit. The graph is vertex-transitive if and only if $m = n$; otherwise it has two vertex orbits. However, $K_{m,n}$ is always edge-transitive (see Exercises).

How to Find the Orbits

We illustrate how to find orbits by consideration of two examples. (It is not known whether there exists a polynomial-time algorithm for finding orbits. Testing all $n!$ vertex-perms for the adjacency preservation property is too tedious an approach.) In addition to using Theorems 2.1 and 2.2, we observe that if an automorphism maps vertex u to vertex v, then it maps the neighbors of u to the neighbors of v.

Example 2.12. In Fig 2.6, the vertex orbits are

$$\{0\}, \ \{1, 4\}, \ \text{ and } \ \{2, 3\}$$

The edge orbits are

$$\{23\}, \ \{01, 04\}, \ \text{ and } \ \{12, 13, 24, 34\}$$

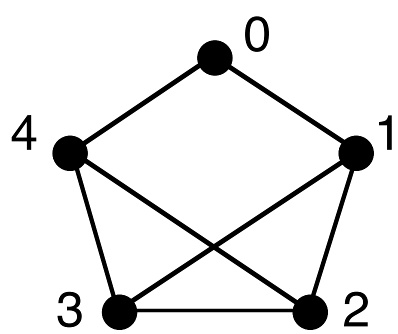


Figure 2.6: Find the vertex orbits and the edge orbits.

Example 2.13. We could approach the 4-regular graph of Figure 2.7 by recognizing the symmetry (0 5)(1 4)(2)(3)(6) and seeking to find others. However, it is possible to expedite the determination of orbits.

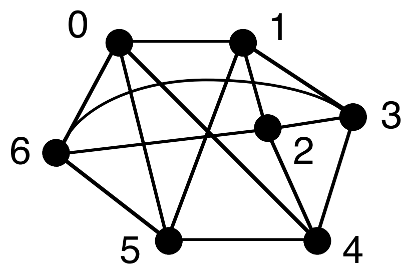


Figure 2.7: Find the vertex orbits and the edge orbits.

When we look at vertices 0, 2, 3, and 5, we discover that each of them has a set of 3 neighbors that are independent, while vertices 1, 4, and 6 each have two pairs of adjacent vertices. This motivates us to redraw the graph as in Figure 2.8.

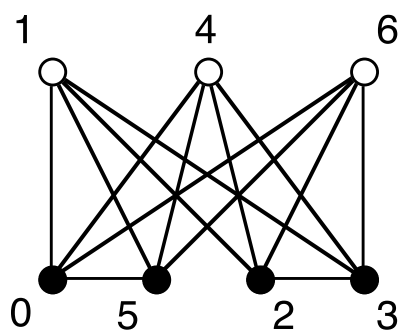


Figure 2.8: Find the vertex orbits and the edge orbits.

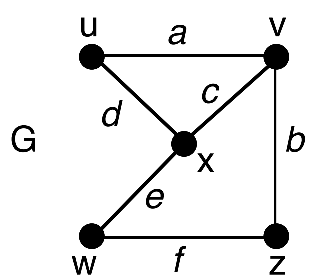
In that form, we see immediately that there are two vertex orbits, namely $\{0, 2, 3, 5\}$ and $\{1, 4, 6\}$. One of the two edge orbits is $\{05, 23\}$, and the other contains all the other edges.

3. Subgraphs

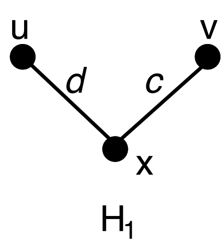
Def 3.1. A ***subgraph*** of a graph G is a graph H whose vertices and edges are all in G. If H is subgraph of G, we may also say that G is a ***supergraph*** of H.

Def 3.2. A ***proper*** subgraph H of G is a subgraph such that V_H is a proper subset of V_G or E_H is a proper subset of E_G.

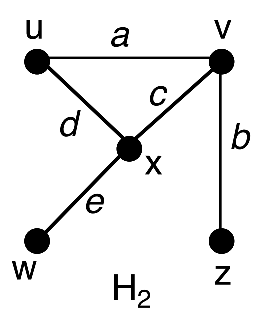
Example 3.1. Fig 3.1 shows the line drawings and corresponding incidence tables for two proper subgraphs of a graph.



edge	a	b	c	d	e	f
endpts	u	v	v	u	x	w
	v	z	x	x	w	z



edge	c	d
endpts	v	u
	x	x



edge	a	b	c	d	e
endpts	u	v	v	u	x
	v	z	x	x	w

Figure 3.1: A graph G and two (proper) subgraphs H_1 and H_2.

The usual meaning of the phrase "H is a subgraph of G" is that H is merely isomorphic to a subgraph of G.

Spanning Subgraphs

Def 3.3. A subgraph H is said to ***span*** a graph G if $V_H = V_G$.

Def 3.4. A ***spanning tree*** is a spanning subgraph that is a tree.

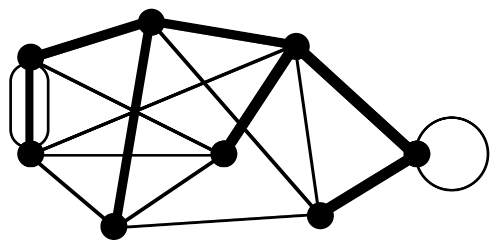


Figure 3.3: A spanning tree.

Def 3.5. An acyclic graph is called a ***forest***.

G

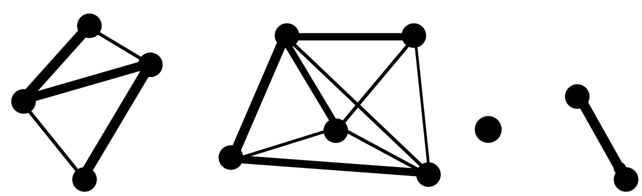


Figure 3.4: A spanning forest H of graph G.

Cliques and Independent Sets

Def 3.6. A subset S of V_G is called a ***clique*** if every pair of vertices in S is joined by at least one edge, and no proper superset of S has this property.

Def 3.7. The ***clique number*** of a graph G is the number $\omega(G)$ of vertices in a largest clique in G.

Example 3.2. In Fig 3.5, the vertex subsets, $\{u, v, y\}$, $\{u, x, y\}$, and $\{y, z\}$ induce complete subgraphs, and $\omega(G) = 3$.

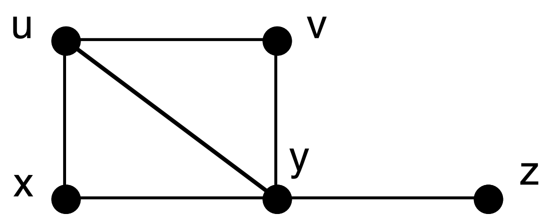


Figure 3.5: A graph with three cliques.

Def 3.8. A subset S of V_G is said to be an ***independent set*** if no pair of vertices in S is joined by an edge.

Def 3.9. The ***independence number*** of a graph G is the number $\alpha(G)$ of vertices in a largest independent set in G.

Remark 3.1. Thus, the clique # $\omega(G)$ and the indep # $\alpha(G)$ are *complementary* concepts (in the sense described in §2.4).

Induced Subgraphs

Def 3.10. ***Subgraph induced on subset*** U of V_G, denoted $G(U)$.

$$V_{G(U)} = U \quad \text{and} \quad E_{G(U)} = \{e \in E_G : endpts(e) \subseteq U\}$$

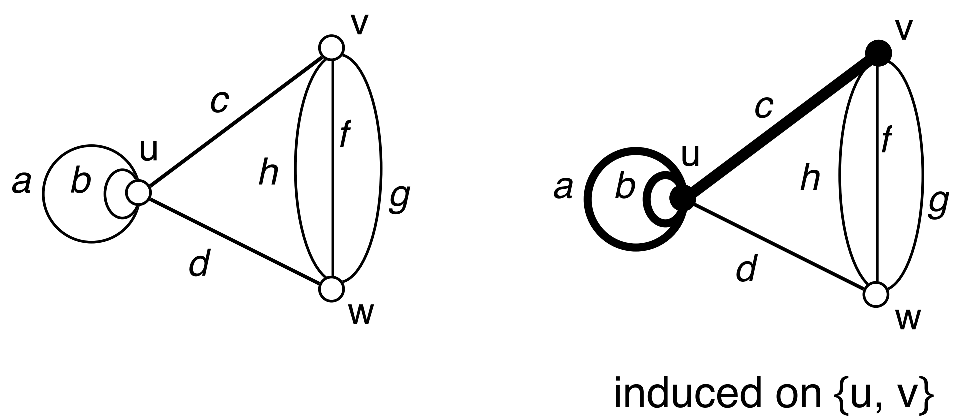


Figure 3.6: A subgraph induced on a subset of vertices.

7. Supplementary Exercises

Exercise 1 Draw all isomorphism types of general graphs with 2 edges and no isolated vertices.

Exercise 14 List the vertex orbits and the edge orbits of the graph of Fig 7.1.

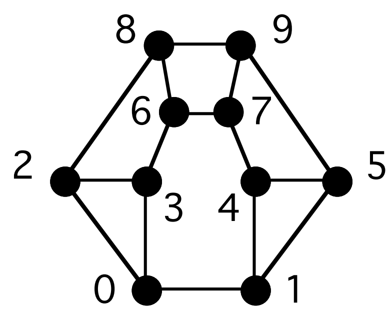


Figure 7.1:

Exercise 17 Some of the 4-vertex, simple graphs have exactly two vertex orbits. Draw an illustration of each such isomorphism type.