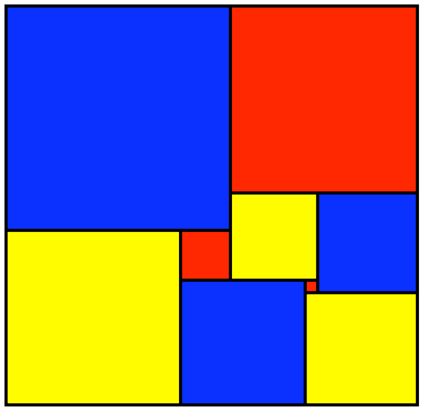
Introduction to Combinatorics

Course Notes for Math 239



Department of Combinatorics and Optimization

University of Waterloo

Part II

Introduction to Graph Theory

Chapter 4

Introduction to Graph Theory

4.1 Definitions

Graph theory is the study of mathematical objects known as “graphs" — a word to which graph theorists have given a rather special meaning. So we must start by defining exactly what a graph is.

Definition 4.1.1. *A* **graph** *G is a finite nonempty set, $V(G)$, of objects, called* **vertices***, together with a set, $E(G)$, of unordered pairs of distinct vertices. The elements of $E(G)$ are called* **edges***.*

For example, we might have

$$V(G) = \{1, 2, 3, 4, 5\}$$

and

$$E(G) = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 5\}, \{3, 4\}, \{3, 5\}, \{4, 5\}\}.$$

For the sorts of results with which we are concerned, it is most convenient to consider the following geometric representation or diagram or **drawing** of a graph. On the page we draw a small circle to correspond to each vertex. For each edge we then draw a line between the corresponding pair of vertices. The only restriction on such a line is that it does not intersect the circle corresponding to any other vertex. For example, the above graph is represented in Figure 4.1(i), (ii) and (iii) in three ways.

If $e = \{u, v\}$ then we say that u and v are **adjacent** vertices, and that edge e is **incident** with vertices u and v. We can also say that the edge e **joins** u and v.

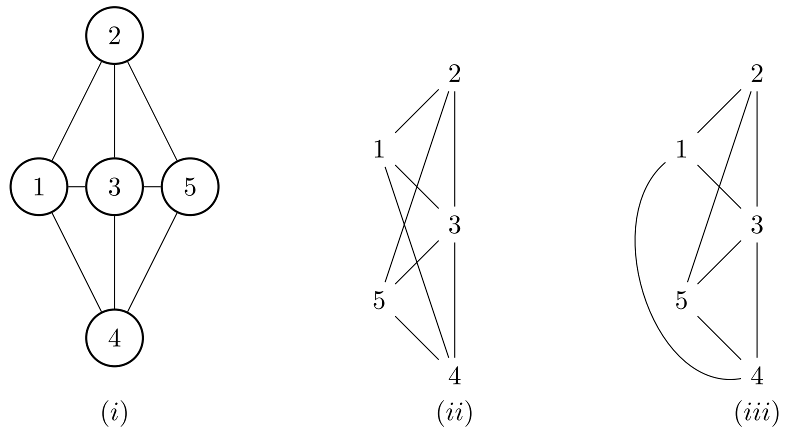


Figure 4.1: Three drawings of the same graph

Vertices adjacent to a vertex u are called **neighbours** of u. The set of neighbours of u is denoted $N(u)$. A graph is completely specified by the pairs of vertices that are adjacent, and the only function of a line in the drawing, representing an edge, is to indicate that two vertices are adjacent. In (i) of Figure 4.1, no edge crosses another; in (ii) of Figure 4.1, the edge $\{1,4\}$ crosses edges $\{2,5\}$ and $\{3,5\}$. A graph which can be represented with no edges crossing is said to be **planar**, so our graph G is planar by Figure 4.1(i).

We give a few examples to illustrate the variety of settings in which graphs arise.

Example 4.1.2. The **word graph** W_n is the graph having $V(W_n)$ equal to the set of all English words having exactly n letters. Two words are adjacent if one can be obtained from the other by replacing exactly one letter by another (in the same position). For example, *seat* is adjacent to *sent* in W_4. In Figure 4.2 we show a drawing of part of W_3.

Example 4.1.3. Given the street map of a city, one can define a **street map graph** as follows. There is a vertex for each street intersection, and an edge for each part of a street joining two intersections and traversing no other intersections. An example is given in Figure 4.3, where intersections are numbered to make the correspondence clear. Such graphs are useful in solving certain kinds of routing and scheduling problems, such as garbage pickup, or delivery of newspapers to carriers.

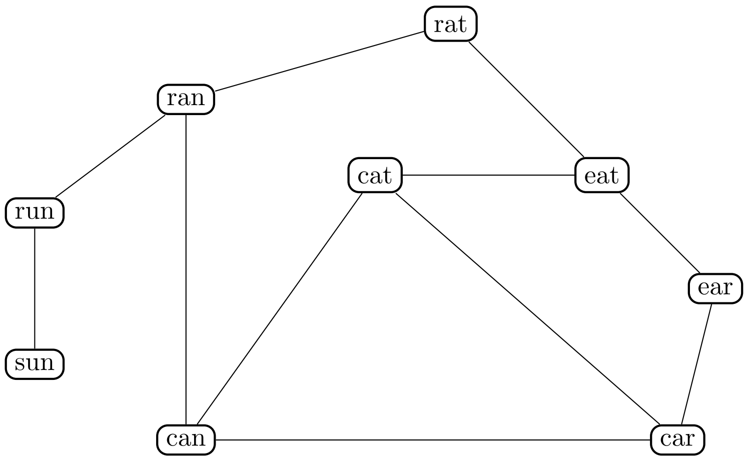


Figure 4.2: Part of W_3

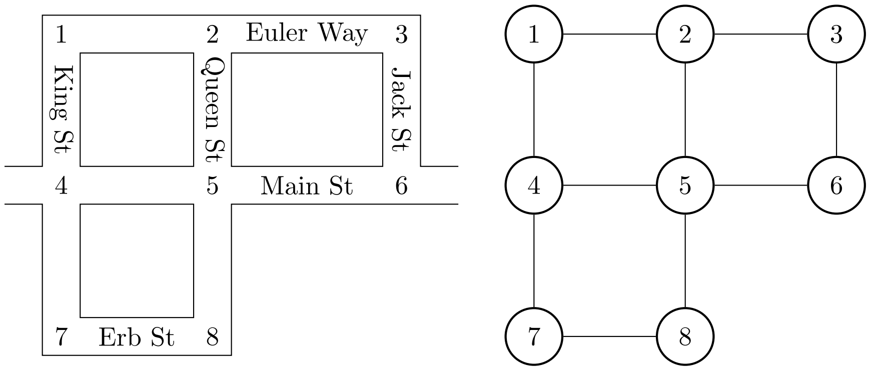


Figure 4.3: Map of Smalltown and its Graph

Example 4.1.4. Another way to obtain a graph from a map, is to begin with a political map, such as the map of the countries of a continent. There is a vertex for each country and two countries are adjacent if they share a boundary. One of the most famous problems in graph theory arose from the question of how many colours are needed to colour such maps so that adjacent countries are not assigned the same colour. Figure 4.4 shows the graph obtained from South America.

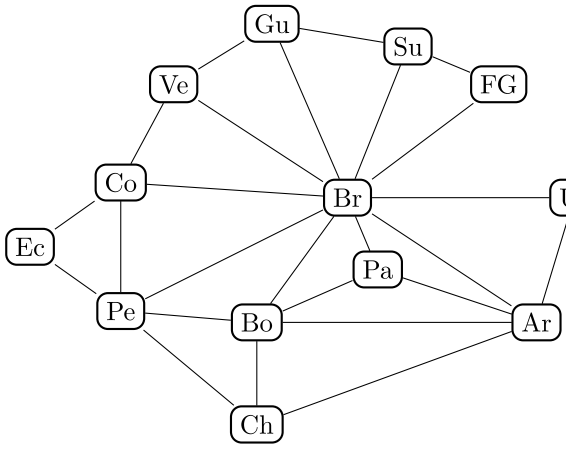


Figure 4.4: Graph of South American Countries

Example 4.1.5. Graphs are often defined from other mathematical objects. For example, we can define the graph $S_{n,k}$ to have $V(S_{n,k})$ equal to the set of k-element subsets of $\{1,2,\ldots,n\}$. Two such subsets are adjacent if they have exactly $k-1$ elements in common. Figure 4.5 shows a drawing of the graph $S_{4,2}$.

Some important points arise from our definition of a graph.

(1) Edges are **unordered** pairs of vertices. Thus the edge $\{v_1, v_2\}$ is not *from* v_1 to v_2 or vice versa; it is simply “between" v_1 and v_2. If we change Definition 4.1.1 to read “ordered pairs" we obtain the definition of a different kind of graph, a **directed graph** or **digraph**.

(2) $E(G)$, being a set, either contains a pair $\{v_1, v_2\}$ or it does not. Thus we do not allow the possibility of “multiple edges” such as exist between the vertices a and b in Figure 4.6.

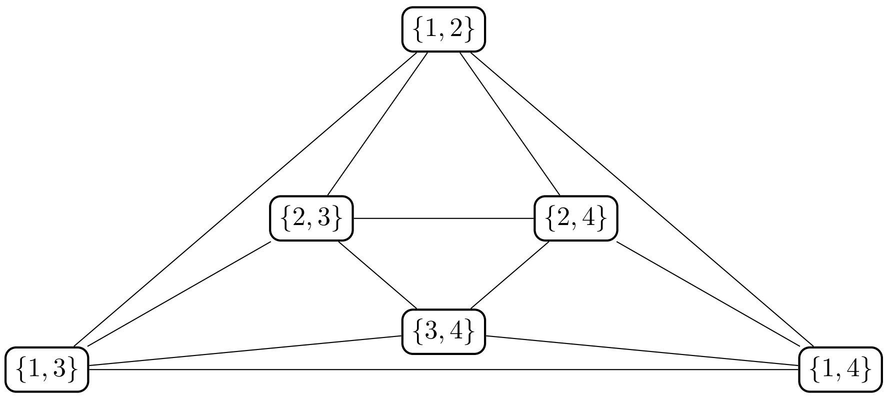


Figure 4.5: A drawing of $S_{4,2}$

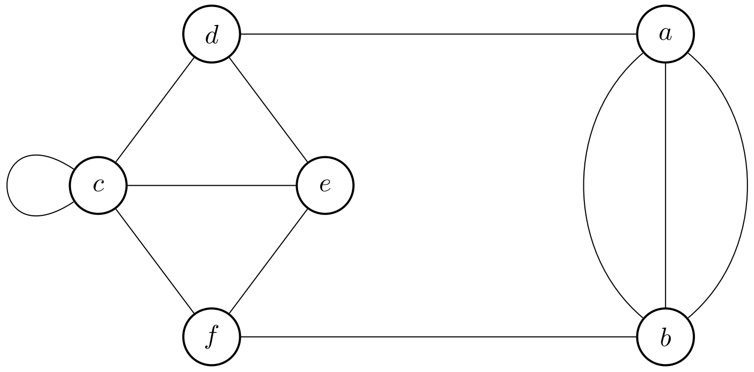


Figure 4.6: A multigraph

(3) The edges are pairs of **distinct** vertices. Hence we cannot have a loop, i.e., an edge joining a vertex to itself as shown in Figure 4.6 at vertex c. Nevertheless in some circumstances it can be convenient to consider loops and/or multiple edges. If we wish to allow loops and multiple edges we will use the term **multigraph** instead of graph. (In some texts "graph" is used to mean "multigraph", and if loops and multiple edges are not allowed the term "simple graph" is used.)

(4) Note that $V(G)$, and hence $E(G)$, is a **finite** set. If we remove this condition we find ourselves in the realm of infinite graphs—and that is a whole new ballgame!

4.2 Isomorphism

Figure 4.7(i) is the diagram of the graph G, where

$$V(G) = \{p, q, r, s\}, \qquad E(G) = \{\{p, q\}, \{p, r\}, \{q, r\}, \{q, s\}\}.$$

Figure 4.7(ii) is the diagram of the graph H, where

$$V(H) = \{a, b, c, d\}, \qquad E(H) = \{\{a, b\}, \{a, c\}, \{a, d\}, \{c, d\}\}.$$

The graphs G and H are not the same—G has vertices p, q, r, s and H has vertices a, b, c, d—but for almost all purposes they are indistinguishable. We make this idea precise.

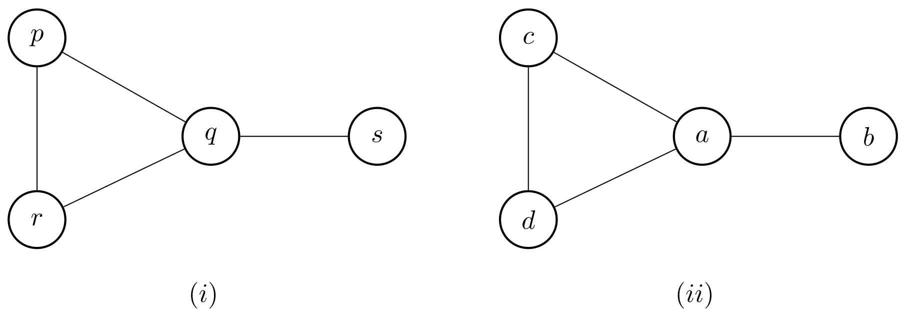


Figure 4.7: Isomorphic graphs

Definition 4.2.1. *Two graphs G_1 and G_2 are* **isomorphic** *if there exists a bijection $f : V(G_1) \to V(G_2)$ such that vertices $f(u)$ and $f(v)$ are adjacent in G_2 if and only if u and v are adjacent in G_1. (We might say that f preserves adjacency.)*

The bijection f in this definition that preserves adjacency is called an **isomorphism**. For example, an isomorphism from G to H is g, defined by

$$g(p) = c,\ g(q) = a, g(r) = d, g(s) = b.$$

Another isomorphism is h, defined by

$$h(p) = d,\ h(q) = a, h(r) = c, h(s) = b.$$

Figure 4.8 shows two other graphs, G and H, that are isomorphic. One isomorphism is the mapping $f : V(G) \to V(H)$ given by

$$\begin{array}{llll} f(1) = a, & f(2) = b, & f(3) = c, & f(4) = h, \\ f(5) = i, & f(6) = j, & f(7) = d, & f(8) = e, \\ f(9) = f, & f(10) = g. & & \end{array}$$

(This graph is called the **Petersen graph**.)

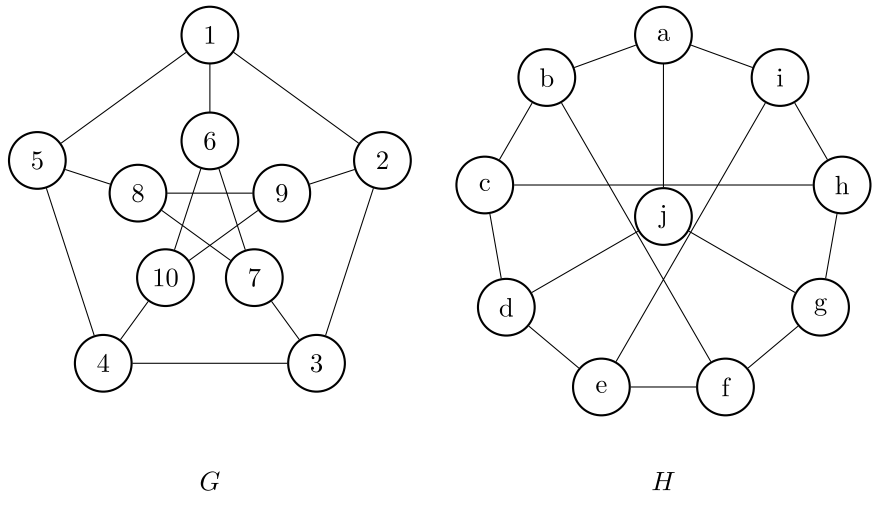


Figure 4.8: Two drawings of the Petersen graph

The collection of graphs that are isomorphic to G forms the **isomorphism class** of G. In almost all cases, a graph has some property if and only if all graphs in its isomorphism class have the property. Thus we generally regard isomorphic graphs as 'the same' even if formally they might not be equal. Even if G has only one vertex, there are infinitely many graphs in its isomorphism class. Fortunately though, the number of isomorphism classes of graphs with a given finite set of vertices is finite. For example, there are exactly 11 isomorphism classes of graphs on 4 vertices, pictured in Figure 4.9. Note that in this figure, the vertices of the graphs are not given explicitly, because however we assign vertices to the drawing, we will still get a graph in the same isomorphism class.

The identity map on $V(G)$ is an isomorphism from the graph G to itself. An isomorphism from G to itself is called an **automorphism** of G.

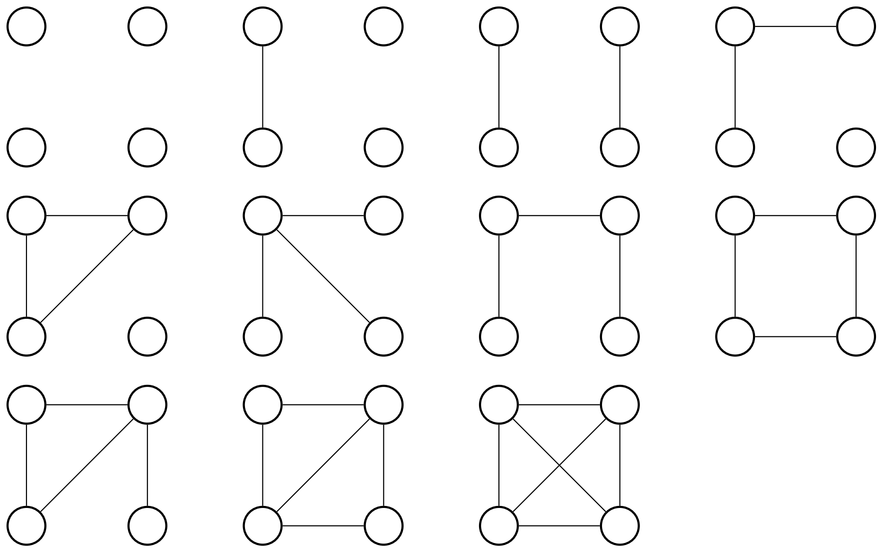


Figure 4.9: The graphs on 4 vertices, up to isomorphism

4.3 Degree

The number of edges incident with a vertex v is called the **degree** of v, and is denoted by $\deg(v)$. For example in G of Figure 4.7 we have $\deg(p) = \deg(r) = 2$, $\deg(q) = 3$, $\deg(s) = 1$; in G of Figure 4.8, all vertices have degree 3. In what follows we generally use 'p' for the number of vertices and 'q' for the number of edges.

Theorem 4.3.1. *For any graph G we have*

$$\sum_{v \in V(G)} \deg(v) = 2|E(G)|.$$

Proof: Each edge has two ends, and when we sum the degrees of the vertices, we are counting the edges twice, once for each end. ∎

This is known as the **Handshaking Lemma** or the **Degree-Sum Formula**.

Corollary 4.3.2. *The number of vertices of odd degree in a graph is even.*

Proof: The sum of all vertex degrees is $2q$, an even number. The sum of the vertices of even degrees is even. Hence the sum of the vertices of odd degrees is also an even number. This implies that there must be an even number of vertices with odd degree. ∎

Corollary 4.3.3. *The average degree of a vertex in the graph G is*

$$\frac{2|E(G)|}{|V(G)|}.$$

A graph in which every vertex has degree k, for some fixed k, is called a k-**regular** graph (or just a **regular** graph). We note one important class of regular graphs.

Definition 4.3.4. *A* **complete graph** *is one in which all pairs of distinct vertices are adjacent. (Thus each vertex is joined to every other vertex). The complete graph with p vertices is denoted by K_p, $p \geq 1$.*

In K_n each vertex is adjacent to the $n-1$ vertices distinct from it, thus K_n is regular with degree $n-1$. The number of edges in K_n is therefore $\binom{n}{2}$, following Corollary 4.3.2.

Figure 4.10 shows K_4, the complete graph on 4 vertices, and therefore 6 edges.

4.4 Bipartite Graphs

A graph in which the vertices can be partitioned into two sets A and B, so that all edges join a vertex in A to a vertex in B, is called a **bipartite** graph, with **bipartition** (A, B). The **complete** bipartite graph $K_{m,n}$ has all vertices in A adjacent to all vertices in B, with $|A| = m$, $|B| = n$. For example, Figure 4.11 is a drawing of $K_{2,3}$.

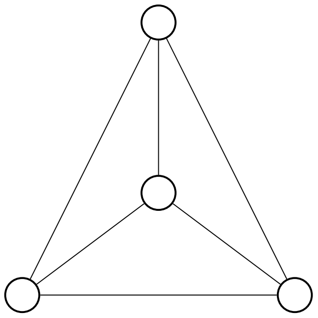


Figure 4.10: K_4

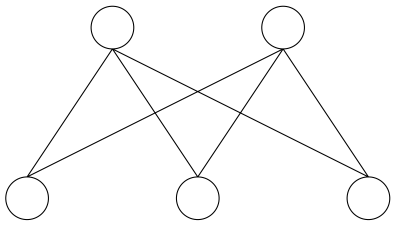


Figure 4.11: The complete bipartite graph $K_{2,3}$

Definition 4.4.1. *For $n \geq 0$, the n-**cube** is the graph whose vertices are the $\{0,1\}$-strings of length n, and two strings are adjacent if and only if they differ in exactly one position.*

For example, Figure 4.12 shows the 3-cube.

Problem 4.4.2. *Determine the numbers of vertices and edges in the n-cube, for $n \geq 0$.*

Solution: The number of $\{0,1\}$-strings of length n is 2^n, for $n \geq 0$, so the n-cube has $p = 2^n$ vertices. Also, every vertex has degree n, since a string of length n is adjacent to the n strings that can be obtained by switching each single element in the string in turn. Thus the n-cube is an n-regular graph, and Theorem 4.3.1 gives

$$\begin{aligned} \sum_{i=1}^{2^n} n &= 2q \\ n2^n &= 2q, \end{aligned}$$

so the n-cube has $q = n2^{n-1}$ edges, for $n \geq 0$. ▌

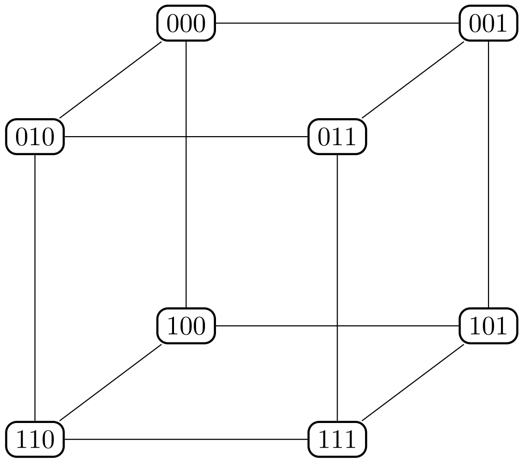


Figure 4.12: The 3-cube

Problem 4.4.3. *Show that the n–cube is bipartite, for $n \geq 0$.*

Solution: Let V be the set of all $\{0,1\}$–strings of length n; V is the vertex set of the n-cube. Partition V into the set A of strings containing an even number of ones and the set B of strings containing an odd number of ones. If vertices x and y are adjacent then the strings x and y differ in exactly one position. Thus, exactly one of x and y contain an even number of ones. Therefore (A,B) is a bipartition and hence the n-cube is bipartite. ∎

Problem Set 4.4

1. For the graphs G_1, G_2, G_3 and H in Figure 4.13, prove that no two of G_1, G_2 or G_3 are isomorphic. Prove that one of them (which?) is isomorphic to H by giving a suitable bijection.

2. A cubic graph is one in which every vertex has degree three. Find all the nonisomorphic cubic graphs with 4, 6 and 8 vertices.

3. For the subset graph $S_{n,k}$ defined in Example 4.1.5, find the number of vertices and the number of edges.

4. The **odd graph** O_n is the graph whose vertices are the n-subsets of a $(2n+1)$-set, two such subsets being adjacent if and only if they are disjoint.

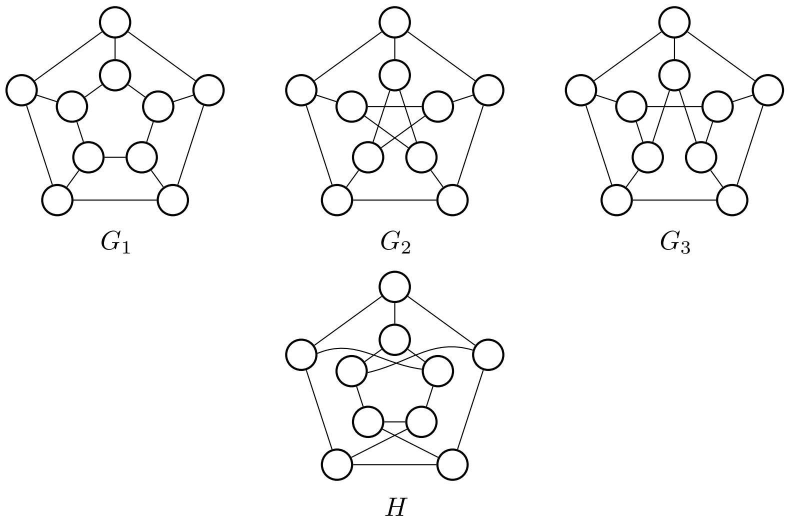


Figure 4.13: Isomorphism exercise

(a) Draw O_1 and O_2.

(b) Prove that O_2 is isomorphic to the Petersen graph (see Figure 4.8).

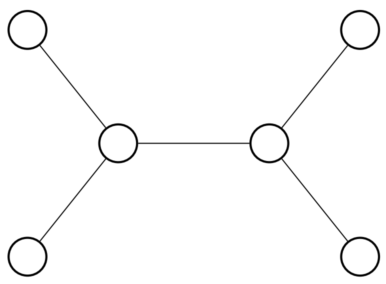
(c) How many vertices and edges does O_n have?

5. The **line-graph** $L(G)$ of a graph G is the graph whose vertex set is $E(G)$ and in which two vertices are adjacent if and only if the corresponding edges of G are incident with a common vertex.

(a) Find a graph G such that $L(G)$ is isomorphic to G.

(b) Find nonisomorphic graphs $G, G^{'}$ such that $L(G)$ is isomorphic to $L(G^{'})$.

(c) If G is the graph



find $L(G), L(L(G))$ and $L(L(L(G)))$.

6. For integer $n \geq 0$, define the graph G_n as follows: $V(G_n)$ is the set of all binary strings of length n having at most one block of 1's. Two vertices are adjacent if they differ in exactly one position.

   (a) Find $|V(G_n)|$

   (b) Make drawings of G_3 and G_4.

   (c) Find $|E(G_n)|$.

7. (a) Draw $K_{m,n}$ for all m, n such that $1 \leq m \leq n \leq 3$.

   (b) How many vertices and edges does $K_{m,n}$ have?

   (c) Let K be a complete bipartite graph on p vertices. Prove that K has at most $\lfloor p^2/4 \rfloor$ edges.

   (d) Let G be a bipartite graph on p vertices. Prove that G has at most $\lfloor p^2/4 \rfloor$ edges.

   (e) Let G be a k-regular bipartite graph with bipartition (X, Y). Prove that $|X| = |Y|$ if $k > 0$. Is this still valid when $k = 0$?

8. The **complement** of the graph G, denoted $\bar{G}$ is the graph with $V(\bar{G}) = V(G)$ and the edge $\{u, v\} \in E(\bar{G})$ if and only if $\{u, v\} \notin E(G)$.

   (a) Let G have vertices 1, 2, 3, 4 and edges $\{1,2\}, \{2,3\}, \{3,4\}, \{1,4\}$. Draw $\bar{G}$.

   (b) Find a 5-vertex graph that is isomorphic to its complement.

   (c) Prove that no 6-vertex graph is isomorphic to its complement.

   (d) Let G_1 and G_2 be two graphs. Prove that G_1 is isomorphic to G_2 if and only if $\bar{G}_1$ is isomorphic to $\bar{G}_2$.

   (e) Find all 2-regular non-isomorphic graphs on 6 vertices (prove that these are the only ones).

   (f) Prove that there are only two 3-regular non-isomorphic graphs on 6 vertices.

9. Make drawings of the 15 nonisomorphic graphs having six vertices and six edges, such that every vertex has degree at least one.

10. Are the graphs in Figure 4.14 isomorphic? Justify your answer.

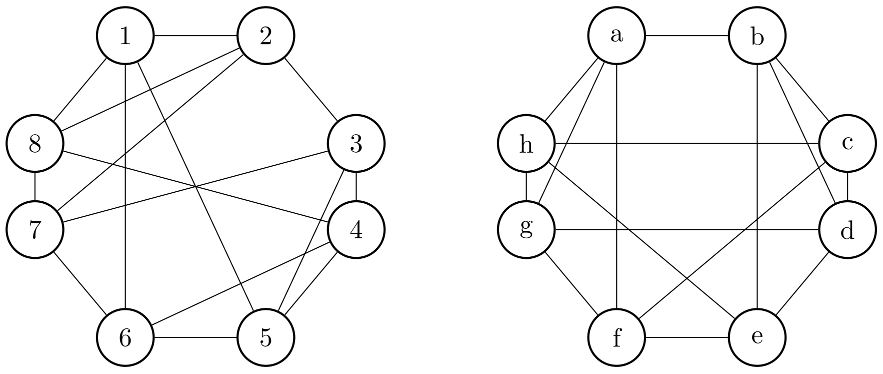


Figure 4.14: Isomorphism exercise

11. For n a positive integer, define the **prime graph** B_n to be the graph with vertex set $\{1,2,\ldots,n\}$, where $\{u,v\}$ is an edge if and only if $u+v$ is a prime number. Prove that B_n is bipartite.

12. (a) Are the two graphs in Figure 4.15 isomorphic? Prove your claim.
    (b) Are the two graphs in Figure 4.16 isomorphic? Prove your claim.

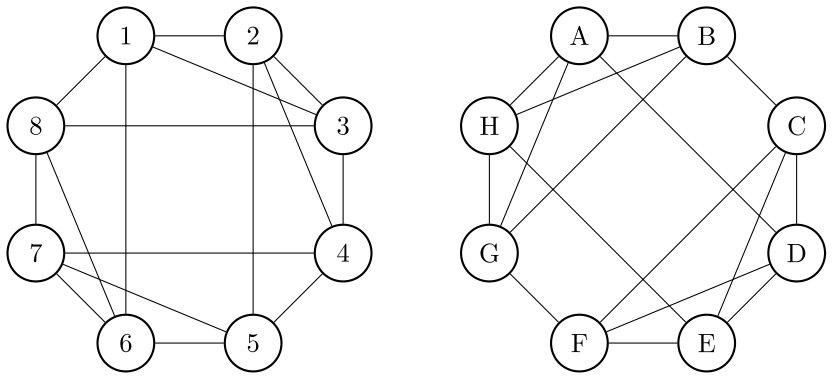


Figure 4.15: Isomorphism exercise

4.5 How to Specify a Graph

One way of specifying a particular graph is to display a drawing of it; but this is not always convenient. Another method is by means of adjacency or incidence matrices.

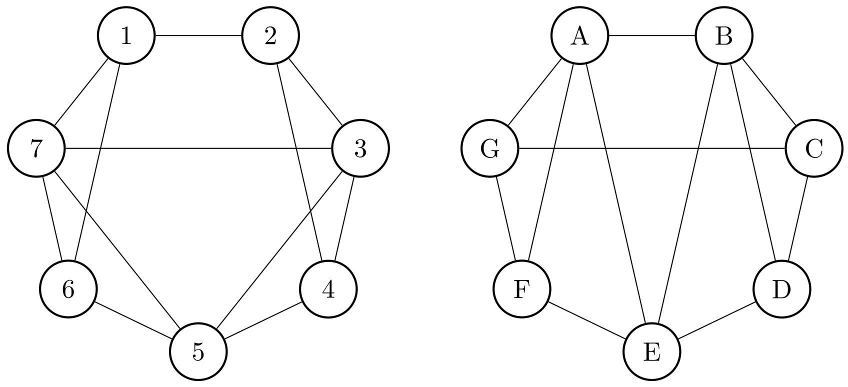


Figure 4.16: Isomorphism exercise

Definition 4.5.1. *The* **adjacency matrix** *of a graph G having vertices $v_1, v_2, \dots, v_p$ is the $p \times p$ matrix $A = [a_{ij}]$ where*

$$a_{ij} = \begin{cases} 1, & \text{if } v_i \text{ and } v_j \text{ are adjacent;} \\ 0, & \text{otherwise.} \end{cases}$$

Clearly A is a symmetric matrix and, since we do not allow loops, its diagonal elements are all zero.

To define an incidence matrix we must name the edges of G; we shall call them $e_1, e_2, \dots, e_q$.

Definition 4.5.2. *The* **incidence matrix** *of a graph G with vertices $v_1 \dots v_p$ and edges $e_1 \dots e_q$ is the $p \times q$ matrix $B = [b_{ij}]$ where*

$$b_{ij} = \begin{cases} 1, & \text{if } v_i \text{ is incident with } e_j; \\ 0, & \text{otherwise.} \end{cases}$$

Clearly, each column of B contains exactly two 1's.

Consider the product BB^t. Its (i, j)-element is

$$\sum_{k=1}^{q} b_{ik} b_{jk}.$$

For $i \neq j$ this sum is the number of edges incident with both v_i and v_j; for $i = j$ it is the number of edges incident with v_i, which is $\deg(v_i)$. Thus

$$BB^t = A + \operatorname{diag}(\deg(v_i), \dots, \deg(v_p)).$$

Another way of specifying a graph is to give, for each vertex, a list of the vertices adjacent to it. For example:

vertex	adjacent vertices
1	3 4 5 7
2	3 5 7
3	1 2 4 5 6
4	1 3 5 6
5	1 2 3 4
6	3 4 7
7	1 2 6

This is called an **adjacency list**. In practice this would be stored as a dictionary or hash.

Finally we could also specify a graph by giving its vertex set together with the list of its edges. This is likely to be most useful when the graph is sparse, i.e., does not have many edges.

It is important to note that in many cases the vertices of the graph may not arrive as non-negative integers. There were examples of this in Problem Set 4.4.

Problem Set 4.5

1. (a) Find the adjacency matrix A and the incidence matrix B for the graph in Figure 4.17.

   (b) Give an interpretation of

   (i) the diagonal elements of the matrix A^2; and

   (ii) the off-diagonal elements of the matrix A^2.

2. (a) Order the vertices of the graph in Figure 4.18 so that the adjacency matrix has the form $\begin{bmatrix} 0 & \mathbf{M} \\ \mathbf{M}^t & 0 \end{bmatrix}$ for some matrix $\mathbf{M}$.

   (b) Prove that any bipartite graph has a vertex ordering which gives an adjacency matrix of the above form for some $\mathbf{M}$.

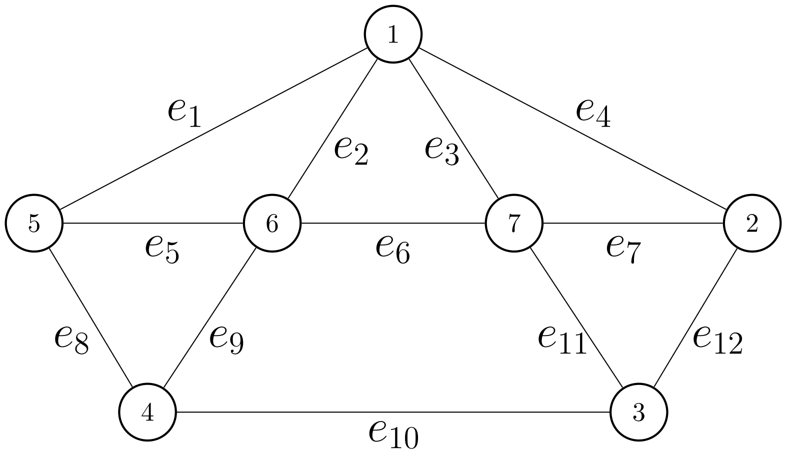


Figure 4.17: Graph for problem 1

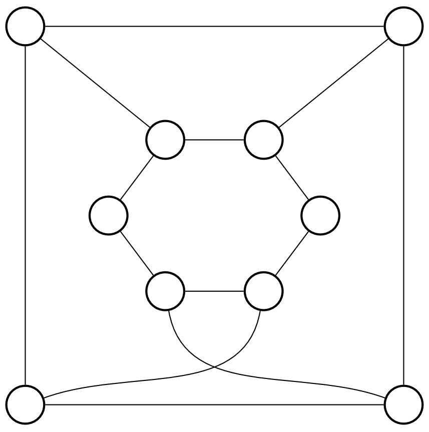


Figure 4.18: Graph for problem 2

4.6 Paths and Cycles

A subgraph of a graph G is a part of G (or possibly the whole of G). More rigorously we have

Definition 4.6.1. *A* **subgraph** *of a graph G is a graph whose vertex set is a subset U of $V(G)$ and whose edge set is a subset of those edges of G that have both vertices in U.*

Thus if H is a subgraph of G, then H has some (perhaps all) the vertices of G and some (perhaps all) the edges that, in G, join vertices of H. (Clearly we cannot have any edges in H that involve vertices not in H). If $V(H) = V(G)$, that is H has all vertices of G, we say H is a **spanning subgraph** of G.

If H is a subgraph of G and H is not equal to G, we say it is a **proper subgraph** of G. If D is a subgraph of G and C is a subgraph of D, then C is a subgraph of G.

A **walk** in a graph G from v_0 to v_n, $n \geq 0$, is an alternating sequence of vertices and edges of G

$$v_0 e_1 v_1 e_2 v_2 \dots v_{n-1} e_n v_n$$

which begins with vertex v_0, ends with vertex v_n and, for $1 \leq i \leq n$, edge $e_i = \{v_{i-1}, v_i\}$. Such a walk can also be called a v_0, v_n**-walk**. Note that the **length** of a walk is the number of edges in it (in this case, n). Also, because we can reverse a walk to get a walk from v_n to v_0, we refer, where convenient, to a walk **between** a pair of vertices, instead of from one to the other. A walk is said to be **closed** if $v_0 = v_n$.

A **path** is a walk in which all the vertices are distinct. A path that starts at v_0 and ends at v_n is called a v_0, v_n**-path**. Observe that since all the vertices in a path are distinct, so are all the edges.

Figure 4.19 shows (by heavy lines) a path in a graph, from vertex 1 to vertex 2.

Since graphs have no multiple edges, consecutive vertices v_{i-1} and v_i determine the edge e_i of a walk. Hence, in describing a walk we often omit the edges.

Theorem 4.6.2. *If there is a walk from vertex x to vertex y in G, then there is a path from x to y in G.*

Proof: We use a form of proof by contradiction.

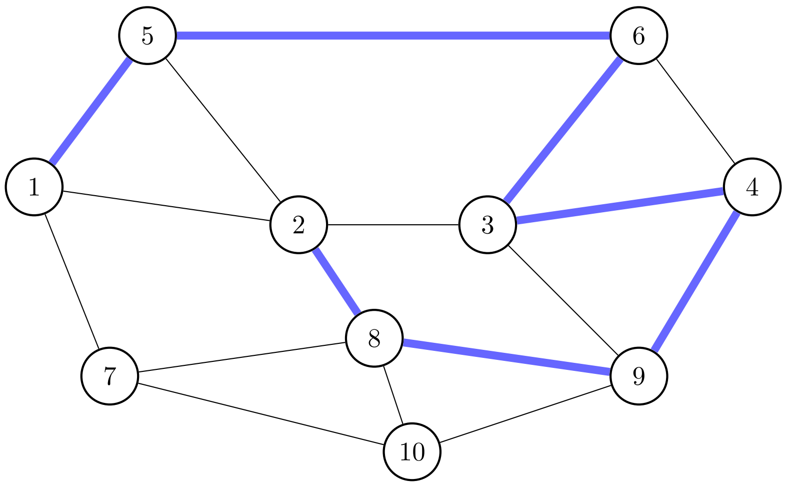


Figure 4.19: A path

Suppose that $v_0, \dots, v_n$ is a walk in G from $v_0 = x$ to $v_n = y$ with minimal length (among all walks from x to y). If the walk is not a path, then $v_i = v_j$ for some $i < j$, and then $v_0 \dots v_i v_{j+1} \dots v_n$ would be a walk from x to y whose length is less than the minimum possible. But this contradicts our initial choice, and so we conclude that the vertices in our original walk are all distinct, and therefore it is a path. ▌

Corollary 4.6.3. *Let x, y, z be vertices of G. If there is a path from x to y in G and a path from y to z in G, then there is a path from x to z in G.*

Proof: The path from x to y followed by the path from y to z is a walk from x to z in G, so by Theorem 4.6.2 there is a path from x to z in G. ▌

A **cycle** in a graph G is a subgraph with n distinct vertices $v_0, v_1 \dots v_{n-1}$, $n \geq 1$, and n distinct edges $\{v_0, v_1\}$, $\{v_1, v_2\}$, $\dots \{v_{n-2}, v_{n-1}\}$, $\{v_{n-1}, v_0\}$. Equivalently, a cycle is a connected graph that is regular of degree two. (The definition of connectedness is given in Section 4.8.)

The subgraph we get from a cycle by deleting one edge is called a **path**. This is inconsistent with the original definition of path, because we defined a path to be a type of walk, and a walk is a sequence of vertices and edges and so is not a graph. But it is very convenient to use the word path in both senses, and we will.

A cycle with n edges is called an n-cycle or a cycle of length n. A cycle of length 1 has one vertex, v_0, and one edge $\{v_0, v_0\}$ which is a loop. Therefore, since graphs have no loops, a graph has no cycle of length one. A cycle of length 2 has two vertices, v_0 and v_1, and two distinct edges, called multiple edges, joining v_0 and v_1. Therefore, a graph has no cycle of length two. Thus the shortest possible cycle in a graph is a 3-cycle, often called a triangle.

Note that for a cycle with $n \geq 3$ vertices, $v_0, \dots, v_{n-1}$, then $v_i v_{i+1} \dots v_{n-1} v_0 \dots v_i$ and $v_i v_{i-1} \dots v_0 v_{n-1} \dots v_i$ are both closed walks for each $i = 0, \dots, n-1$, and in this way there are $2n$ closed walks of length n associated with a given n-cycle.

Figure 4.20 shows (by heavy lines) an 8-cycle in a graph.

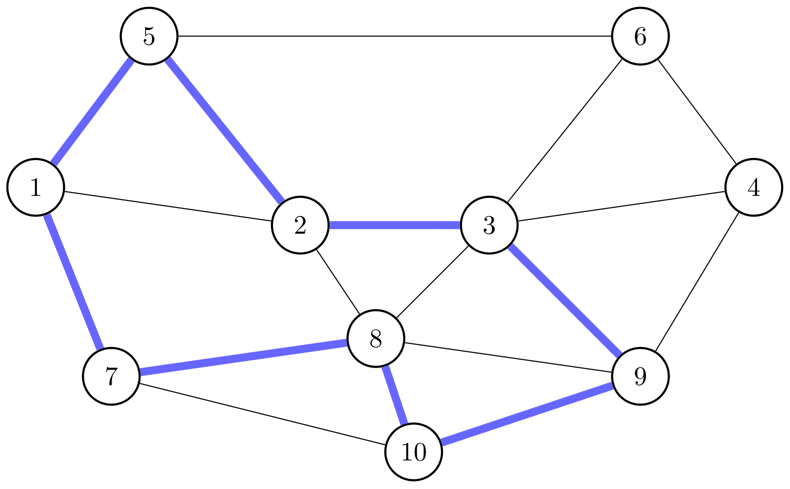


Figure 4.20: A cycle

In a cycle, every vertex has degree exactly 2. There's one condition that guarantees that a graph contains a cycle.

Theorem 4.6.4. *If every vertex in G has degree at least 2, then G contains a cycle.*

Proof: Let $v_0, v_1, \dots, v_k$ be a longest path in G. The vertex v_0 has v_1 as one neighbour. Since v_0 has degree at least 2, v_0 has another neighbour x. If x is not on the path, then $x, v_0, v_1, \dots, v_k$ is a path longer than our longest path, which is a contradiction. Hence x must be on the path, and $x = v_i$ for some $i \geq 2$. Then $v_0, v_1, \dots, v_i, v_0$ is a cycle in G. ▌

The **girth** of a graph G is the length of the shortest cycle in G, and is denoted by $g(G)$. If G has no cycles, then $g(G)$ is infinite (but you may choose to ignore this fact).

A spanning cycle in a graph is known as a **Hamilton cycle** (so the cycle in Figure 4.20 is not a Hamilton cycle). Although it is easy to decide if a given cycle is a Hamilton cycle, it can be surprisingly difficult to find one in a graph or to certify that it has no Hamilton cycle.

Problem Set 4.6

1. Let G be a graph with minimum degree k, where $k \geq 2$. Prove that
   (a) G contains a path of length at least k;
   (b) G contains a cycle of length at least $k+1$.

2. Let A be the adjacency matrix of a graph G.
   (a) Show that the $(i,j)^{th}$ entry of A^k is the number of walks of length k from i to j.
   (b) Assume A satisfies the matrix equation $A^2 + A = (k-1)I + J$ where I is the identity matrix and J is the matrix of all 1's. Explain in graph theory terms the properties G possesses given by the matrix equation.

3. (a) If A is the adjacency matrix of a graph G, show that for $i \neq j$, the $(i,j)^{th}$ element of A^2 is the number of paths of length 2 in G between vertices v_i and v_j.
   (b) What are the diagonal elements of A^2?

4. Let G be the graph whose set of vertices is the set of all "lower 48" states of the United States, plus Washington, DC, with two vertices being adjacent if they share a boundary. (For example, *California* is adjacent to *Arizona.*) Let H be the subgraph of G whose vertices are those of G whose first letter is one of W, O, M, A, N, and whose edges are the edges of G whose ends have this property. (For example, *California* is not a vertex of H, but *Arizona* and *New Mexico* are, and they are adjacent in H.) Find a path in H from *Washington* to *Washington, DC.*

5. Consider the word graph W_n defined in Example 4.1.2.
   (a) Find a cycle through *math* in W_4.
   (b) Find a path from *pink* to *blue* in W_4.

6. For $n \geq 2$, prove that the n-cube contains a Hamilton cycle.

7. Prove that the complete bipartite graph $K_{m,n}$ has a Hamilton cycle if and only if $m = n$ and $m > 1$.

8. Show that if there is a closed walk of odd length in the graph G, then G contains an odd cycle (that is, G has a subgraph which is a cycle on an odd number of vertices).

9. A **diagonal** of a cycle in a graph is an edge that joins vertices that are not consecutive in the cycle.

   (a) Prove that a shortest cycle (if one exists) has no diagonal.

   (b) Prove that a shortest odd cycle (if one exists) has no diagonal.

   (c) Give an example of a graph in which a shortest even cycle has a diagonal.

10. (a) Prove that a k-regular graph of girth 4 has at least $2k$ vertices ($k \geq 2$).

    (b) For $k = 2, 3$, find a k-regular graph of girth 4 with precisely $2k$ vertices. Generalize these examples, i.e. find one for each $k \geq 2$.

    (c) Prove that a k-regular graph of girth 5 has at least $k^2 + 1$ vertices ($k \geq 2$).
    Remark: The only values of k for which such a graph with exactly $k^2 + 1$ vertices can exist are $k = 2, 3, 7, 57$. This surprising result can be proved using elementary matrix theory (i.e., what you study in MATH 235). Examples are known for $k = 2, 3, 7$, but no example has yet been found for $k = 57$. Such a graph would have $57^2 + 1 = 3250$ vertices.

    (d) Prove that a k-regular graph of girth $2t$, where $t \geq 2$, has at least $\frac{2(k-1)^t - 2}{k-2}$ vertices.

    (e) Prove that a k-regular graph of girth $2t + 1$, where $t \geq 2$, has at least $\frac{k(k-1)^t - 2}{k-2}$ vertices.

    (f) For $k = 2, 3$, give an example of a k-regular graph of girth five with exactly $k^2 + 1$ vertices.

4.7 Equivalence Relations

You have met equivalence relations in your first algebra course, but these are important in nearly all areas of mathematics, including graph theory.

Formally, if S and T are sets, then a relation $\mathscr{R}$ between S and T is a subset of $S \times T$. The idea is that if $a \in S$ and $b \in T$, then a and b are related if and only if (a, b) belongs to the subset. If a and b are related we may say that they are **incident**. Thus if G is a graph, then "is contained in" is a relation on $V(G) \times E(G)$.

We will be most concerned with the case where $S = T$. In this case we usually refer to a relation on S, and do not mention $S \times S$. By way of example, "is adjacent to" is a relation on the vertices of a graph. A relation on S is **reflexive** if each element of S is related to itself. So "is adjacent to" is not a reflexive relation on the vertex set $V(G)$ of a graph G. However this relation is **symmetric**, that is, if a is related to b then b is related to a. On the integers, the relation "divides" is reflexive but not symmetric.

There is a third important property a relation may have. Suppose that we are given a relation on a set V and, if $a, b \in V$, we write $a \approx b$ to denote that a and b are related. We say the relation is **transitive** if whenever $a \approx b$ and $b \approx c$, then $a \approx c$. The relation "divides" on the integers is transitive, as is the relation $\leq$ on $\mathbb{R}$. The relation "is a subgraph of" on the subgraphs of G is reflexive and transitive.

We say a relation is an **equivalence relation** if it is reflexive, symmetric and transitive.

As an example, the relation "is joined by a walk to" on the vertices of a graph G is an equivalence relation—each of the three properties is very easy to verify.

The canonical example is equality. Another example you have met is "congruent modulo m" on the integers. We generally use equivalence relations to partition things. Thus the equivalence relation "congruent mod 5" splits the integers into five classes. The key is that if $\approx$ is an equivalence relation on a set V and $C(a)$ is the set

$$\{v \in V : v \approx a\}$$

then any two elements of $C(a)$ are equivalent, and any element of v that is equivalent to something in $C(a)$ is itself an element of $C(a)$. Hence if $b \in V$ then either $C(a) = C(b)$ or $C(a) \cap C(b) = \emptyset$.

4.8 Connectedness

Definition 4.8.1. *A graph G is* **connected** *if, for each two vertices x and y, there is a path from x to y.*

Theorem 4.8.2. *Let G be a graph and let v be a vertex in G. If for each vertex w in G there is a path from v to w in G, then G is connected.*

Proof: For any vertices x and y in G, there is a path from v to x and a path from v to y. If we reverse the path from v to x we obtain a path from x to v, and now Corollary 4.6.3 implies that there is a path from x to y in G, so G is connected by Definition 4.8.1. ▮

If G is a graph on n vertices, then to certify that G is connected according to the definition of connected, we must provide a total of $\binom{n}{2}$ paths. The above theorem reduces the workload: only n paths are needed.

Problem 4.8.3. *Prove that the n-cube is connected for each $n \geq 0$.*

Solution: We use Theorem 4.8.2, and prove that there is a path from vertex $v_0 = 0\ldots0$ (with n 0's) to x for all other vertices x in the n-cube. Now x is a $\{0,1\}$-string of length n, and suppose that x has k 1's in positions $i_1, \ldots, i_k$, where $1 \leq i_1 < \ldots < i_k \leq n$, with $1 \leq k \leq n$. Now let v_j be the $\{0,1\}$-string with 1's in positions $i_1, \ldots, i_j$, and 0's elsewhere for $j = 1, \ldots, k$. Then $v_0 v_1 \ldots v_k$ is a path from $0\ldots0$ to x, so the n-cube is connected, by Theorem 4.8.2. ▮

Definition 4.8.4. *A* **component** *of G is subgraph C of G such that*

(a) C is connected.

(b) No subgraph of G that properly contains C is connected.

These two conditions may sometimes be stated in the form "a component of G is a subgraph which is maximal, subject to being connected". Here when we say a subgraph C of G is maximal, subject to having some property, then this means that any subgraph that properly contains C does not have this property.

Note that we could have defined a component of G to be the equivalence class of a vertex, relative to the relation "is joined by a walk to". Since this is indeed an equivalence relation, it follows immediately that the components of G partition its vertex set.

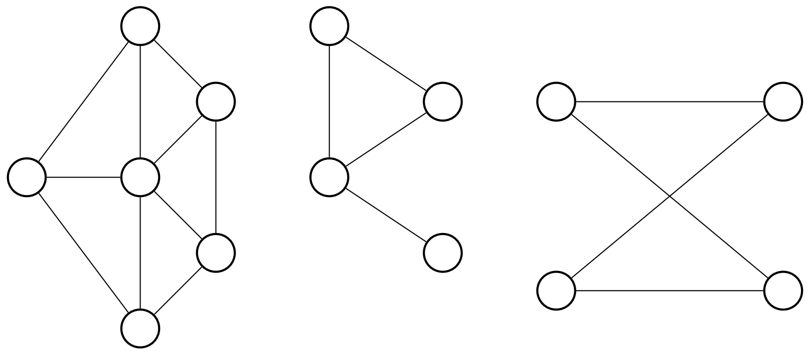


Figure 4.21: A disconnected graph

Figure 4.21 shows a graph having three components. Note that there are paths between every pair of vertices in the same component, but not between pairs of vertices in different components.

While it is easy to convince someone of the existence of a path between two vertices (show them the path), it is less clear how you might convince them that a path does not exist. We introduce a convenient way of doing this.

If we are given a partition (X, Y) of $V(G)$ such that there are no edges having an end in X and an end in Y, then there is no path from any vertex in X to any vertex in Y. So, if X and Y are both nonempty, G is not connected. Given a subset X of the vertices of G, the **cut** induced by X is the set of edges that have exactly one end in X.

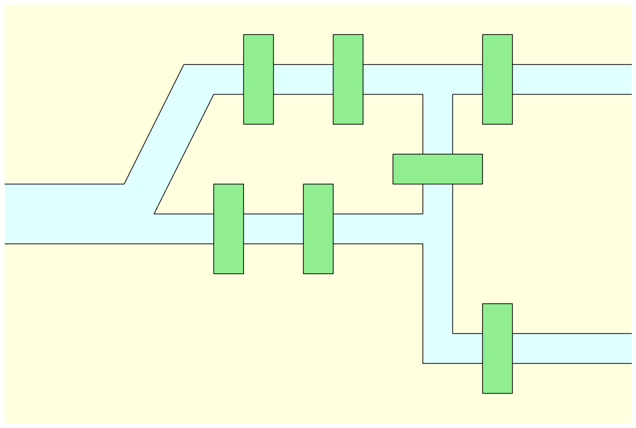
Theorem 4.8.5. *A graph G is not connected if and only if there exists a proper nonempty subset X of $V(G)$ such that the cut induced by X is empty.*

Proof: Let G be a connected graph, and let X be a proper nonempty subset of $V(G)$. Choose vertices $u \in X$ and $v \in V(G) \setminus X$. Since G is connected, there exists a path $x_0 x_1 \ldots x_n$ from u to v. Choose k as large as possible such that $x_k \in X$. Since $x_n = v \notin X, k < n$, the edge $x_k x_{k+1}$ is in the cut induced by X and hence this cut is not empty.

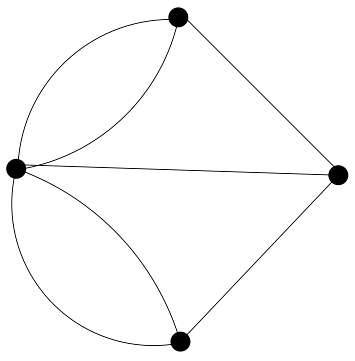
Conversely, suppose that G is not connected and let C be a component of G. Consider the partition (X, Y) of $V(G)$ where $X = V(C)$ and $Y = V(G) \setminus V(C)$. Since C is connected and G is not, X is a proper non-empty subset of $V(G)$. Since C is a component, if the vertex y is adjacent to a vertex in C, then $y \in V(C)$. Hence the cut induced by X is empty. ▌

4.9 Eulerian Circuits

One of the earliest problems in graph theory is the **Seven Bridges of Königsberg** problem. In the 18th century in the town of Königsberg (now Kaliningrad, Russia), there were 7 bridges that crossed the river Pregel, which cuts through the city and there were 2 islands in the middle of the river. The layout can be roughly represented in the following diagram:



The question is that can a resident of the city leave home, cross every bridge exactly once, and then return home? We can formulate the layout of the city as a graph: Create 4 vertices representing the land areas (two shores and two islands), and create an edge for each bridge, joining the vertices representing the two land areas on either side of the bridge. In this case, we obtain the following graph (with multiple edges):



In terms of graph theory, the question becomes "is there a closed walk that uses every edge exactly once?" We use a definition for this type of walk.

Definition 4.9.1. *An* Eulerian circuit *of a graph G is a closed walk that contains every edge of G exactly once.*

In 1736, Swiss mathematician Leonhard Euler noted that there cannot be an Eulerian circuit for the 7 bridges of Königsberg problem. The reason being that if an Eulerian circuit exists, each time we visit a vertex, we must use 2 distinct edges incident with that vertex: use one edge to go to the vertex, use one edge to leave the vertex. So every vertex must have even degree. However, in this graph, every vertex has odd degree, so such a walk is not possible.

On the other hand, suppose we have a connected graph and every vertex has even degree. Does it have an Eulerian circuit? Euler noted that this is indeed true, and we give a proof here.

Theorem 4.9.2. *Let G be a connected graph. Then G has an Eulerian circuit if and only if every vertex has even degree.*

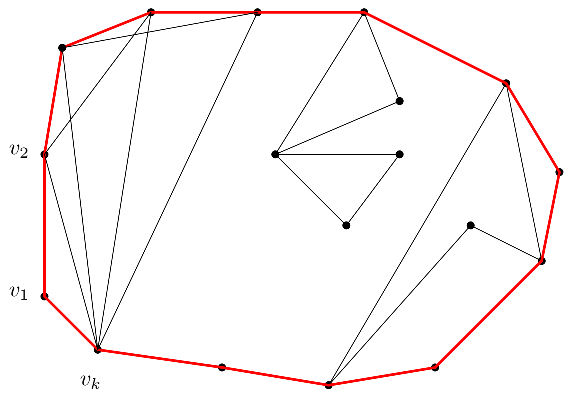
Proof: ($\Rightarrow$) A closed walk contributes 2 to the degree of a vertex for each visit. Since an Eulerian circuit uses each edge of the graph exactly once, each vertex in the graph must have even degree.

($\Leftarrow$) We will prove by strong induction on the number of edges m in G.

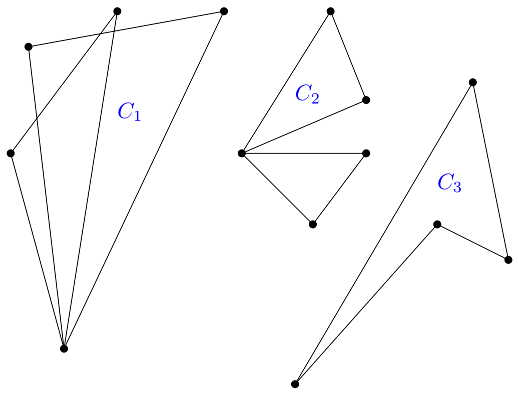
Base case: If G has 0 edges, then G consists of exactly one isolated vertex. It has a trivial closed walk as an Eulerian circuit.

Induction hypothesis: Assume that for some $m \geq 1$, any connected graph with less than m edges where every vertex has even degree has an Eulerian circuit.

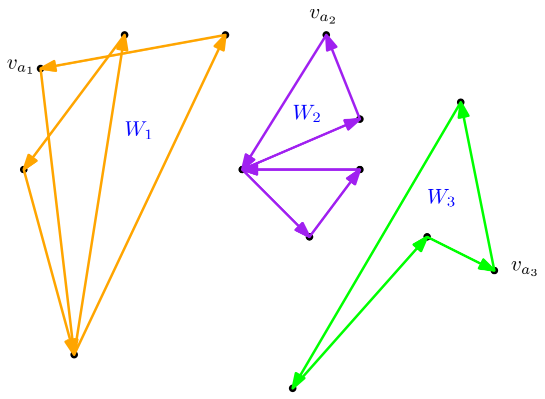
Induction step: Let G be a connected graph with m edges where every vertex has even degree. This implies that every vertex has degree at least 2. Therefore, by Theorem 4.6.4, there exists a cycle C in G, say the vertices on the cycle in order are $v_1, v_2, \ldots, v_k, v_1$.



Remove edges of C from G and remove isolated vertices to obtain G'. Since every vertex is incident with either 0 or 2 edges in C, every vertex in G' still has even degree. Now G' consists of components $C_1, \dots, C_l$, each containing less than m edges.

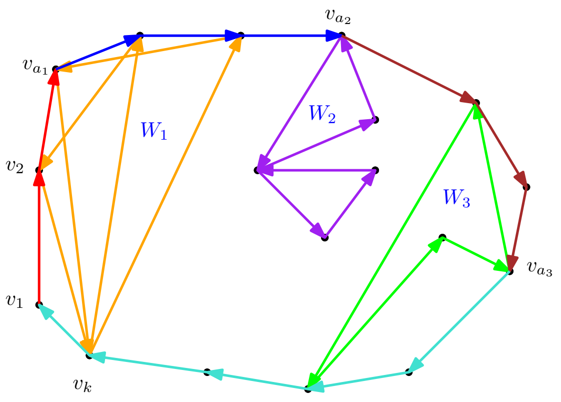


By induction hypothesis, each component C_i has an Eulerian circuit W_i. Moreover, each component must have a common vertex with C, for otherwise G is disconnected. Let v_{a_i} be one vertex of C_i. Rearrange the components so that $a_1 < a_2 < \cdots < a_l$, and let W_i start and end at v_{a_i}.



Then we can construct an Eulerian circuit for G by walking along C and making detours W_i as we hit v_{a_i}:

$$v_1, \dots, v_{a_1-1}, W_1, v_{a_1+1}, \dots, v_{a_2-1}, W_2, v_{a_2+1}, \dots\dots\dots, v_{a_l-1}, W_l, v_{a_l+1}, \dots, v_k, v_1.$$



∎

4.10 Bridges

If $e \in E(G)$, we denote by $G - e$ (or by $G \backslash e$) the graph whose vertex set is $V(G)$ and whose edge set is $E(G) \backslash \{e\}$. (So $G - e$ is the graph obtained from G by deleting the edge e.)

Definition 4.10.1. *An edge e of G is a* **bridge** *if $G - e$ has more components than G.*

Thus if G is connected, a bridge is an edge such that $G - e$ is not connected. Figure 4.22 shows a connected graph with a bridge e. Some texts, and some instructors, use **cut-edge** as a synonym for bridge.

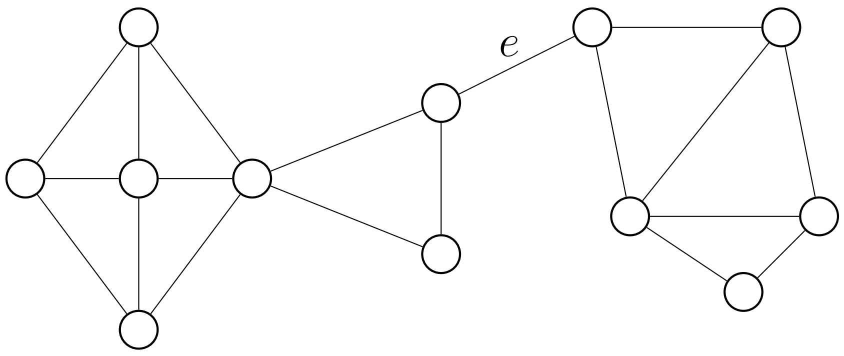


Figure 4.22: A bridge

Lemma 4.10.2. *If $e = \{x, y\}$ is a bridge of a connected graph G, then $G - e$ has precisely two components; furthermore, x and y are in different components.*

Proof: Let $e = \{x, y\}$. If e is a bridge, then $G - e$ has at least two components. Let V_x be the set of vertices in the same component of $G - e$ as x. Let z be any vertex of $G - e$ not in V_x. Because there exists a path from x to z in G but not in $G - e$, every path from x to z in G contains edge e and so must be of the form

$$xeye_2v_2e_3v_3 \cdots v_{n-1}e_nz.$$

But $ye_2v_2 \cdots e_nz$ is a path from y to z in $G - e$. Thus z is in the same component of $G - e$ as y, for every vertex z not in V_x. Hence, $G - e$ has 2 components, one containing vertex x and the other containing vertex y. ▮

Theorem 4.10.3. *An edge e is a bridge of a graph G if and only if it is not contained in any cycle of G.*

Proof: We begin by proving the following implication: if edge $e = \{x, y\}$ is an edge of some cycle of G, then e is not a bridge of G.

By hypothesis, there is a cycle

$$x, e_1, v_1, e_2, v_2, \ldots, v_{n-1}, e_n, y, e, x$$

in G. Then

$$x, e_1, v_1, e_2, v_2, \ldots, v_{n-1}, e_n, y$$

is a path from x to y in $G - e$. Hence e is not a bridge of G by Lemma 4.10.2. This establishes the implication.

To complete the proof, we must establish the converse – if edge e is not a bridge of graph G, then e is an edge of some cycle. Suppose $e = uv$ is not a bridge. Then u and v must lie in the same component of $G - e$, and so there is a path P that joins them. Together with e this path forms a cycle that contains e. ▮

Corollary 4.10.4. *If there are two distinct paths from vertex u to vertex v in G, then G contains a cycle.*

Proof: Let P_1 and P_2 be distinct paths from u to v. Suppose that P_1 is given by $x_0x_1 \ldots x_n$ and P_2 is given by $y_0y_1 \ldots y_m$. Thus $u = x_0 = y_0$ and $v = x_n = y_m$. Let i be the first index such that $x_{i+1} \neq y_{i+1}$. Then $e = \{x_i, x_{i+1}\}$ is an edge in P_1 but not in P_2.

Consider the walk

$$x_i, x_{i-1}, \dots, x_0 = u = y_0, y_1, \dots, y_m = v = x_n, x_{n-1}, \dots, x_{i+1}.$$

It is a walk from x_i to x_{i+1} that does not use the edge e. So it is also a walk in $G-e$, hence x_i, x_{i+1} are in the same component in $G-e$. By Lemma 4.10.2, e is not a bridge. By Theorem 4.10.3, e must be part of a cycle. Hence G contains a cycle. ▌

We will often use the contrapositive form of this result: If graph G has no cycles, then each pair of vertices is joined by at most one path.

Problem Set 4.10

1. Prove that the prime graph B_n defined in Problem Set 4.4 is connected for every n. You may use without proof the following fact: For every integer $k \geq 2$ there is a prime number r such that $k < r < 2k$.
2. Prove that, if G is connected, any two longest paths have a vertex in common.
3. Which graphs, with at least one edge, have the property that every edge is a bridge?
4. If every vertex of a graph H with p vertices has degree at least $p/5$, prove that H cannot have more than 4 components.
5. If edge e is not a bridge of a connected graph G, prove that e is an edge of some cycle.
6. Prove that a 4-regular graph has no bridge.
7. Let A_n be the graph whose vertices are the $\{0,1\}$-strings of length n, and edges are between strings that differ in exactly two positions, $n \geq 2$.
   (a) How many edges does A_n have?
   (b) Is A_n bipartite for any $n \geq 2$?
   (c) How many components does A_n have?
8. Let B_n be the graph whose vertices are the $\{0,1\}$-strings of length n, and edges are between strings that differ in exactly two consecutive positions, $n \geq 2$.

   (a) How many edges does B_n have?
   (b) How many components does B_n have?

9. Let G be a graph in which exactly two of the vertices u, v have odd degree. Prove that G contains a path from u to v.

4.11 Certifying Properties

In graph theory we usually find that each time we meet a new property, two questions arise. First, how do we certify that a graph has the property. Second, how do we certify that a graph does not have that property.

So we can certify that a graph is connected by providing, for each pair of distinct vertices, a path in the graph the joins the two vertices. We can certify that a graph G is not connected by producing a cut—two non-empty sets of vertices A and B that partition $V(G)$, such that no edge of G joins a vertex in A to a vertex in B.

As another example, you can certify that an edge $e = uv$ in a connected graph G is a bridge by producing a cut (A, B) for $G - e$ where $u \in A$ and $v \in B$. You can certify that e is not a bridge by producing a cycle that contains e. (You could also certify this by showing that $G - e$ is connected, but this would be more work as a rule.)

Certificates are required to be easy to verify; they need not be easy to find. So you might have to work hard to find a certificate, but it must still be easy for the marker to verify. (The precise rule is that it must be possible to check the correctness of your certificate in *polynomial time*.)

It is an experimental fact that if there is a good certificate for verifying that a graph has some property, and a good certificate for verifying that it does not, then there is an efficient algorithm for testing if a graph has the property.

Chapter 5

Trees

5.1 Trees

A very special and important kind of graph is a **tree**.

Definition 5.1.1. *A* **tree** *is a connected graph with no cycles.*

Figure 5.1 shows a typical tree.

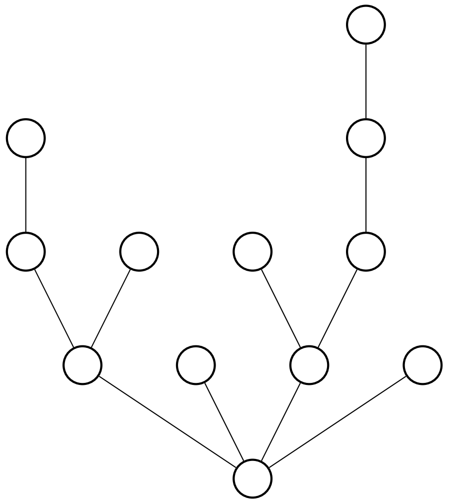


Figure 5.1: A tree

If connectedness is not required, then the graph is a **forest**.

Definition 5.1.2. *A* **forest** *is a graph with no cycles.*

We now prove some properties of trees and forests.

Lemma 5.1.3. *If u and v are vertices in a tree T, then there is a unique u,v-path in T*

Proof: For any 2 vertices u and v in T, there is at least 1 path joining them since T is connected. Since T has no cycles, there is at most one path by Corollary 4.10.4. This establishes the result. ▌

Lemma 5.1.4. *Every edge of a tree T is a bridge.*

Proof: An edge e of T is not in a cycle, so, by Theorem 4.10.3, e is a bridge. ▌

One main property of trees is that any tree with p vertices have the same number of edges: $p-1$.

Theorem 5.1.5. *If T is a tree, then $|E(T)| = |V(T)| - 1$.*

Proof: The proof is by strong induction on q, the number of edges. When $q = 0$ there is just one tree. It has one vertex and no edges, and the theorem holds for it.

Suppose that the theorem is true for trees on fewer than q edges, and let T be any tree with q edges, for $q \geq 1$. Let $e = \{u, v\}$ be any edge. By Lemma 5.1.4, e is a bridge. Therefore, $T - e$ is not connected and, by Lemma 4.10.2, it has exactly two components, say T_1 and T_2. Both are connected (as they are components), and do not contain cycles (as a cycle in $T - e$ is a cycle in T). Hence both T_1 and T_2 are trees. Since both of them have fewer than q edges, by induction, $|E(T_1)| = |V(T_1)| - 1$ and $|E(T_2)| = |V(T_2)| - 1$. Then

$$|E(T)| = |E(T_1)| + |E(T_2)| + 1 = (|V(T_1)| + |V(T_2)|) - 1 = |V(T)| - 1.$$

▌

Corollary 5.1.6. *If G is a forest with k components, then $|E(G)| = |V(G)| - k$.*

Proof: Let $T_1, \dots, T_k$ be the k components of G. Since each component is connected and G does not contain any cycle, each T_i is a tree. By Theorem 5.1.5, $|E(T_i)| = |V(T_i)| - 1$ for each i. Adding the k equations, we get

$$\sum_{i=1}^{k} |E(T_i)| = \sum_{i=1}^{k} (|V(T_i)| - 1) = \left(\sum_{i=1}^{k} |V(T_i)| \right) - k.$$

Since $|E(G)| = \sum_{i=1}^{k} |E(T_i)|$ and $|V(G)| = \sum_{i=1}^{k} |V(T_i)|$, we get $|E(G)| = |V(G)| - k$. ▌

Definition 5.1.7. *A* **leaf** *in a tree is a vertex of degree 1.*

Theorem 5.1.8. *A tree with at least two vertices has at least two leaves.*

Proof: Let P be a longest path in the tree T with end vertices u and v. Since any edge gives a path of length 1, P must have length at least 1, so $u \neq v$.

Now one vertex adjacent to v is in P. If $\deg(v) > 1$, then there must be another vertex, w, adjacent to v. Vertex w cannot be in P, since this would imply a cycle in T, whereas T has no cycles. Since w is not in P, we can extend P by adding the edge $\{v, w\}$ to it to get a longer path. This is a contradiction. Hence $\deg(v) = 1$. Similarly $\deg(u) = 1$, which proves the theorem. ▌

This proof works if, instead of choosing a longest path, we choose a path which is not a subgraph of a path in T with more edges. (We might say that our path is "maximal under inclusion".) The advantage of this choice is that it is easier to decide if a path is maximal under inclusion than to decide if it is a longest path—for to do the latter we must consider all paths in the tree.

The following alternate proof gives more detailed information about how many vertices of degree one a tree can have given the degrees of other vertices.

Alternate proof of Theorem 5.1.8: Let T be a tree and let n_r denote the number of vertices of degree r in T. Set $p = |V(T)|$ and assume $p \geq 2$. By Theorem 4.3.1 we have

$$2p - 2 = \sum_{v \in V(T)} \deg(v)$$

and therefore

$$-2 = \sum_{v} (\deg(v) - 2) = \sum_{r=0}^{p-1} n_r (r-2).$$

In the last sum, $n_0 = 0$ (because in a connected graph with at least two vertices, every vertex has degree at least 1) and so we find that

$$-2 = -n_1 + \sum_{r \geq 3} (r-2) n_r.$$

Therefore

$$n_1 = 2 + \sum_{r \geq 3} (r-2) n_r.$$

Since $(r-2)n_r \geq 0$ when $r \geq 3$, it follows that $n_1 \geq 2$. ▌

The above proof implies that if T contains a vertex of degree r, where $r \geq 3$, then $n_1 \geq 2 + (r-2) = r$. As an exercise, use a version of the first proof to show that a tree that contains a vertex of degree r has at least r vertices of degree one.

Problem Set 5.1

1. (a) Draw one tree from each isomorphism class of trees on six or fewer vertices.

   (b) For each tree in (a), exhibit a bipartition (X, Y) by coloring the vertices in X with one colour and the vertices in Y with another.

2. Prove that every tree is bipartite.

3. What is the smallest number of vertices of degree 1 in a tree with 3 vertices of degree 4 and 2 vertices of degree 5? Justify your answer by proving that this is the smallest possible number, and by giving a tree which has this many vertices of degree 1.

4. Find the smallest number r of vertices in a tree having two vertices of degree 3, one vertex of degree 4, and two vertices of degree 6. Justify your answer by proving that any such tree has at least r vertices, and by giving an example of such a tree with exactly r vertices.

5. A **cubic** tree is a tree whose vertices have degree either 3 or 1. Prove that a cubic tree with exactly k vertices of degree 1 has $2(k-1)$ vertices.

6. Prove that a forest with p vertices and q edges has $p-q$ components.

7. Let $p \geq 2$. Show that a sequence $(d_1, d_2, \ldots, d_p)$ of positive integers is the degree sequence of a tree on p vertices if and only if $\sum_{i=1}^{p} d_i = 2p-2$. (Hint: use induction on p.)

5.2 Spanning Trees

A spanning subgraph which is also a tree is called a **spanning tree**. The reason that spanning trees are important is that, of all the spanning subgraphs, they have the fewest edges while remaining connected. Figure 5.2 shows a graph with a spanning tree indicated by heavy lines.

Theorem 5.2.1. *A graph G is connected if and only if it has a spanning tree.*

Proof: ("if" part.) We are given that G has a spanning tree T. Then Lemma 5.1.3 implies that there is a path in T between every pair of vertices of T. But each

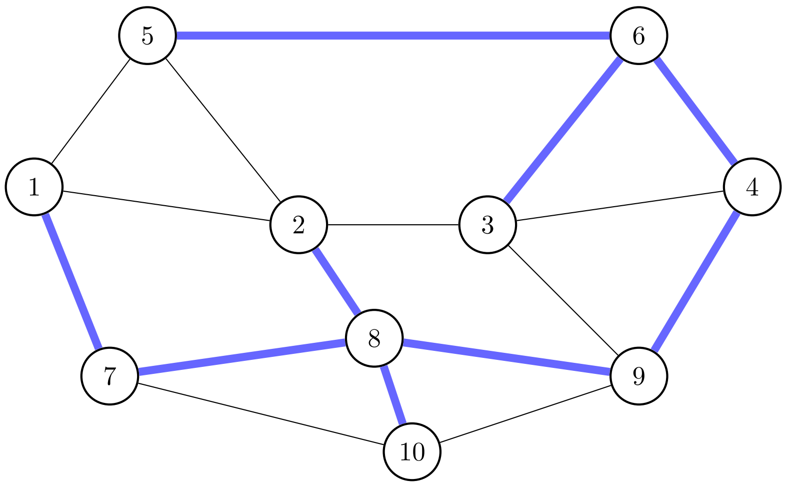


Figure 5.2: A spanning tree

of these paths is also contained in G, and G has the same vertices as T, so from Definition 4.8.1 we conclude that G is connected.

("only if" part.) We are given that G is connected. If G has no cycles, then G itself is a spanning tree of G. Otherwise G has a cycle. Remove any edge e of some cycle. Then $G - e$ is connected, by Theorem 4.10.3, and has fewer cycles than G.

Repeat this process, removing an edge on a cycle at each stage, until we have a connected, spanning subgraph with no cycles. This subgraph is a spanning tree of G. ▌

To show that a graph is connected, using the definition, you need to give a path between any pair of vertices. However, Theorem 5.2.1 provides a much more succinct method: give a spanning tree.

Corollary 5.2.2. *If G is connected, with p vertices and $q = p - 1$ edges, then G is a tree.*

Proof: Let G be a connected graph with p vertices and $q = p - 1$ edges. By Theorem 5.2.1, G has a spanning tree T. Now T is a tree with p vertices, so, by Theorem 5.1.5, T has $p - 1$ edges. However, as G has only $p - 1$ edges, it must be the case that $G = T$. Therefore, G is a tree. ▌

There are a couple of ways to get different spanning trees by exchanging two edges.

Theorem 5.2.3. *If T is a spanning tree of G and e is an edge not in T, then $T+e$ contains exactly one cycle C. Moreover, if e' is any edge on C, then $T+e-e'$ is also a spanning tree of G.*

Proof: Let $e=\{u,v\}$. Any cycle in $T+e$ must use e, since T has no cycles. Such a cycle consists of e along with a u,v-path in T. By Lemma 5.1.3, there is a unique u,v-path in T, hence there is exactly one cycle C in $T+e$.

If e' is any edge in C, then e' is not a bridge (Theorem 4.10.3). So $T+e-e'$ is still connected. Since it has $n-1$ edges, by Corollary 5.2.2, it is a tree. ▌

Theorem 5.2.4. *If T is a spanning tree of G and e is an edge in T, then $T-e$ has 2 components. If e' is in the cut induced by one of the components, then $T-e+e'$ is also a spanning tree of G.*

Proof: The first statement is a direct consequence of Lemma 4.10.2. Let C_1 and C_2 be the two components of $T-e$. Suppose $e'=\{u,v\}$ where $u\in V(C_1)$ and $v\in V(C_2)$.

We wish to show that $T-e+e'$ is connected using Theorem 4.8.2. Let $x\in V(C_1)$. For any $y\in V(C_1)$, there exists an x,y-path since C_1 is connected. Suppose $y\in V(C_2)$. Since C_1 and C_2 are connected, there exist an x,u-path P_1 and a v,y-path P_2. Then P_1,e',P_2 form an x,y-path. Since there exists an x,y-path for any vertex y, $T-e+e'$ is connected.

Since $T-e+e'$ has $n-1$ edges, by Corollary 5.2.2, it is a tree. ▌

5.3 Characterizing Bipartite Graphs

Using Theorem 5.2.1 we will prove an important characterization of bipartite graphs. Note that, we can convince someone that a graph is bipartite by giving them a bipartition. However, it is not so clear how you would convince them that a graph does not have a bipartition. (Checking every partition of $V(G)$ would be tedious.)

One idea is to note that any subgraph of a bipartite graph is bipartite, and so we could try to find a subgraph that is "obviously" not bipartite. An **odd cycle** is a cycle on an odd number of vertices.

Lemma 5.3.1. *An odd cycle is not bipartite.*

Proof: Suppose that C is a cycle with vertex set

$$\{-k, -k+1, \ldots, k\}$$

where $i \sim i+1$ if $-k \leq i < k$ and $k \sim -k$. Suppose C is bipartite with bipartition (A, B). Without loss of generality, $0 \in A$. Then $1, -1 \in B$ and it follows easily that for $j = 1, \ldots, k$, the vertices j and $-j$ must be in the same partition. But k and $-k$ are adjacent and they are in the same partition. This is a contradiction, hence C is not bipartite. ▌

So you could certify a graph is not bipartite by producing a subgraph that is an odd cycle. It is quite surprising, but this is all you need to do.

The converse is also true, as we see in the following result.

Theorem 5.3.2. *A graph is bipartite if and only if it has no odd cycles.*

Proof: Given Lemma 5.3, it suffices to prove that if G is not bipartite, then it contains an odd cycle.

Since G is not bipartite, at least one component H of G is not bipartite. (If all components are bipartite, then we could find a bipartition of G by combining the bipartitions of the individual components.) Since H is connected, by Theorem 5.2.1, there exists a spanning tree T in H.

Trees are bipartite (see Problem Set 5.1, Problem 2), so let (A, B) be a bipartition of T. Since H is not bipartite, (A, B) is not a bipartition of H and therefore there exists an edge $\{u, v\}$ of H such that both u and v are in A, or both are in B. By swapping A and B if needed, we may assume that $u, v \in A$.

Since T is connected, there exists a u, v-path P in T, with vertices $x_0 x_1 \ldots x_n$ where $u = x_0$ and $v = x_n$. Since $x_0 = u \in A$ and T is bipartite, the vertices along P must alternate between A and B. So $x_0, x_2, x_4, \ldots \in A$ and $x_1, x_3, x_5, \ldots \in B$. Since $x_n \in A$, n must be even, hence P has even length. However, $x_0 x_n = uv \in E(H)$, so $P + \{u, v\}$ is an odd cycle in H, which is in G. Hence G contains an odd cycle, as claimed. ▌

Problem Set 5.3

1. Let r be a fixed vertex of a tree T. For each vertex v of T, let $d(v)$ be the length of the path from v to r in T. Prove that

   (a) for each edge uv of T, $d(u) \neq d(v)$, and

   (b) for each vertex x of T other than r, there exists a unique vertex y such that y is adjacent to x and $d(y) < d(x)$.

2. Let r be a fixed vertex in a graph G. Suppose that, for each vertex v of G we have an integer $d(v)$ such that

   (i) for each edge uv of G, $d(u) \neq d(v)$, and

   (ii) for each vertex x of G other than r, there exists a unique vertex y such that y is adjacent to x and $d(y) < d(x)$.

   Prove that G is a tree.

5.4 Breadth-First Search

We now consider an algorithm for finding a spanning tree in a graph G, if one exists. Note that, by Theorem 5.2.1, this is also a very good way of deciding whether a graph is connected. To describe the algorithm properly we must decide how our graph will be presented to us, and also in what form we will present its output. We assume that the graph is given as a list of edges.

We could present the spanning tree (if it exists) as a list of edges but, given a list of edges it is not obvious if it is a tree, and so we take a more sophisticated approach. We will represent a tree by a function.

Suppose T is a tree and u is a vertex in T. For each vertex x in T distinct from u there is a unique path of length at least one from x to u. Define $\text{pr}(x)$ to be the neighbour of x in this path. We define $\text{pr}(u)$ to be $\emptyset$. The pr is a function from $V(T)$ to $V(T)$. We might call $\text{pr}(x)$ the **parent** or **predecessor** of x.

Clearly if we are given a tree it is easy to write down its predecessor function and, conversely, given the predecessor function we can recover the tree.

So suppose we are given a graph G and we want to find a spanning tree in G. Choose a vertex u in G and set $D = \{u\}$. Define $\text{pr}(u) = \emptyset$.

Now suppose we are given a subset D and a partially completed predecessor function. If $D \neq V(G)$, look for an edge that joins a vertex in D to a vertex not in D. If there is none, then D determines an empty cut and we deduce that G is not connected. If there is an edge vw where $v \in D$ and $w \notin D$, then add w to D and define $\text{pr}(w) = v$. If $D = V(G)$, then our predecessor function describes a spanning tree in G.

Algorithm 5.4.1. *To find a spanning tree of a graph G: Select any vertex r of G as the initial subgraph D, with* $\mathrm{pr}(r) = \emptyset$. *At each stage, find an edge in G that joins a vertex u of D to a vertex v not in D. Add vertex v and edge $\{u, v\}$ to D, with* $\mathrm{pr}(v) = u$. *Stop when there is no such edge.*

Claim:

If $|V(D)| = |V(G)|$ when the algorithm terminates, then D is a spanning tree of G. If $|(D)| < |V(G)|$ when the algorithm terminates, then G is not connected and so, from Theorem 5.2.1, G has no spanning tree.

Proof:

We begin by using mathematical induction on the number of iterations to show that the subgraphs D produced by the algorithm are subtrees of graph G.

Basis Case: Initially, D is a tree with 1 vertex.

Induction Hypothesis: For $k \geq 0$, assume that the subgraph D produced in the k-th iteration is a tree with $k+1$ vertices and k edges.

Inductive Step: If the algorithm terminates in the $(k+1)$st iteration, then there is nothing to prove. Otherwise, the $(k+1)$st iteration produces a subgraph E by adding vertex v and edge $\{u, v\}$ to D. Therefore E has $k+2$ vertices and $k+1$ edges. Since D is tree, there is a path from u to every vertex in D. Since u, v is a path from u to v, there is a path from u to every vertex in E. Therefore, E is connected by Theorem 4.8.2. By Corollary 5.2.2, E is a tree. The iteration ends by redefining D to be the subtree E. By Mathematical Induction, D is a subtree of G in every iteration.

If D has p vertices when the algorithm terminates, then D is a spanning tree because it includes all the vertices of G, as required.

If D has fewer that p vertices when the algorithm terminates, then $V(D)$ is a nonempty, proper subset of $V(G)$. Since the algorithm has terminated, no edge joins a vertex u in D to a vertex v not in D. Therefore, the cut induced by $V(D)$ is empty. This implies that G is not connected by Theorem 4.8.5, as required. ▌

Spanning trees have the important property of being connected. If u and v are vertices of a connected graph, then in any spanning tree there is a unique path joining u and v. Hence spanning trees provide a nice “small" structure by which one can search all the vertices in a graph using only edges of the tree. This is the reason for the parent function $\mathrm{pr}(v) = u$, in which vertex v is referred to as a **child** of u. In a diagram we display this information by placing an arrow on the edge between u and v, pointing from the child v to the parent u, as in Figure 5.3. The initial vertex r is called the root vertex, and has no parent. One can now easily recover paths in the tree. For vertex $v \neq r$, there is a unique positive integer k such that $\mathrm{pr}^k(v) = r$, and the path from v to r is

$$v, \mathrm{pr}(v), \mathrm{pr}^2(v), \ldots, \mathrm{pr}^k(v) = r$$

A path between two vertices u, v can be recovered from Corollary 4.6.3, using the paths from u and v to the root r. Alternatively, we examine parents from u and v until we find a common "ancestor". A spanning tree with this extra structure, provided by the parent functions (the pointers, or the arrows on the diagram) is often called a **search tree**. If $\mathrm{pr}^k(v) = r$ we say that the **level** of v is k, and write level $(v) = k$. We define level (r) to be 0.

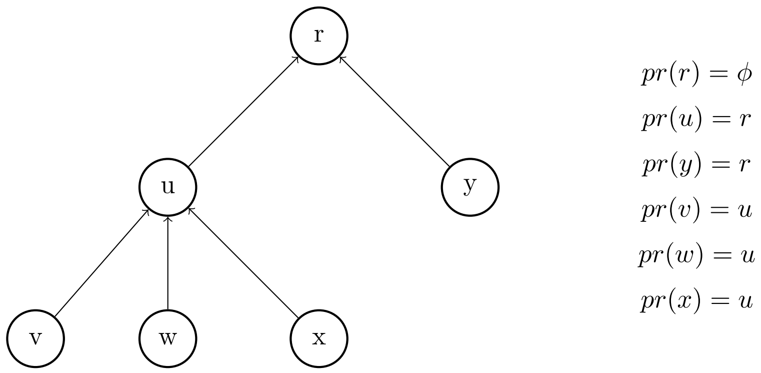


Figure 5.3: A search tree

In Figure 5.3, r is at level 0, u and y are at level 1, while v, w and x are at level two.

In Algorithm 5.4.1, we looked for edges incident with a vertex in the tree and a vertex not in the tree. Thus it is convenient at each stage to say that a vertex in the tree is **exhausted** if it is not adjacent to a vertex outside the tree. Of course, if a vertex is exhausted at any stage, then it will remain exhausted at all later stages. In Algorithm 5.4.1, we can ignore edges incident with exhausted vertices, since the only possible edges that will allow the tree to increase in size must be incident with an unexhausted vertex in the tree. Now we consider a refinement of Algorithm 5.4.1 called **breadth-first search**, in which the unexhausted vertex u at each stage is chosen in a special way.

Algorithm 5.4.2. *Breadth-first search. Follow Algorithm 5.4.1 with the following refinement: At each stage consider the unexhausted vertex u that joined the tree earliest among all unexhausted vertices (called the* **active** *vertex), and choose an edge incident with this vertex and a vertex v not in the tree.*

A **breadth-first search tree** is any spanning tree that is created by applying breadth-first search to a connected graph.

One way to implement breadth-first search is to use a “queue", that is, a first-in, first-out list for vertices in the tree, where vertices are placed at the end of the queue when they are first added to the tree. At each stage the vertex at the head of the queue is examined. If it is exhausted, it is removed from the queue. Otherwise, it is the active vertex.

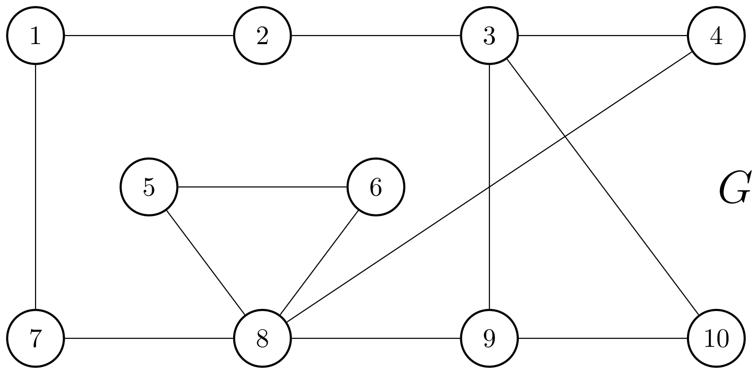


Figure 5.4: Graph *G*

In Figure 5.5, we illustrate an example of breadth-first search, applied to the connected graph *G*. We begin the breadth-first search tree *B* by (arbitrarily) selecting vertex 9 as the root, with pr(9) = ∅, and thus the queue then consists of vertex 9 alone. Vertex 9 becomes active, and we add vertices 3, 8, 10 to *B*, at level 1, in that (arbitrary) order, with pr(3) = pr(8) = pr(10) = 9, and the queue becomes 9, 3, 8, 10 (ordered with the head of the queue on the left, and each new vertex joining the queue on the right). Now vertex 9 is exhausted, so it is removed from the queue, and vertex 3 becomes active. Next, vertices 4 and 2 are added to *B*, with pr(4) = pr(2) = 3, and so vertices 4 and 2 are at level 2, and the queue becomes 3, 8, 10, 4, 2. Vertex 3 is now exhausted, so it is removed from the queue, vertex 8 becomes active, and we add vertices 7, 5, 6 to *B* in that order, so the queue becomes 8, 10, 4, 2, 7, 5, 6. Now vertices 8, 10, 4 are all exhausted, so the queue becomes 2, 7, 5, 6, and vertex 1 is added to *B*, at level 3. Finally, all vertices of *G* are now in *B*, so all vertices in the queue are exhausted, and we stop. In Figure 5.5, we have placed the vertices at each level from left to right, in the order that they joined the breadth-first search tree *B*. As a summary, note that the order in which the vertices of *G* joined *B* is 9, 3, 8, 10, 4, 2, 7, 5, 6, 1.

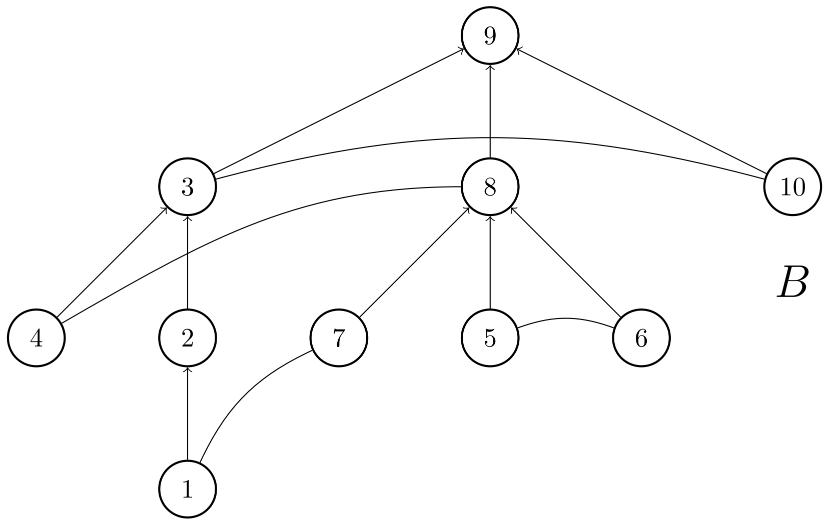


Figure 5.5: A re-drawing of G, with the edges of a breadth-first search tree B indicated with arrows

In Figure 5.5, the edges of B have an arrow to specify the parent. The edges of G that are not contained in the breadth-first search tree B (called **non-tree** edges) have been added to the drawing of B, without arrows. This gives a re-drawing of G that will illustrate the primary property of breadth-first search, given below. First we need a preliminary lemma.

Lemma 5.4.3. *The vertices enter a breadth-first search tree in non-decreasing order of level.*

Proof: We prove this by induction on the number of vertices in the tree at each stage of the algorithm. The first vertex in the tree is the root vertex, with level 0, and the result is true for this first stage.

Now we make the induction hypothesis, that the result is true for the first m vertices in the tree, $m \geq 1$, and consider the next vertex v, that joins the tree at stage $m+1$. Now $\text{pr}(v) = u$, where u is active when v joins the tree, and $\text{level}(v) = \text{level}(u) + 1$. Consider any other non-root vertex x in the tree at stage $m+1$. Then $\text{pr}(x) = y$, and $\text{level}(x) = \text{level}(y) + 1$. But either $y = u$ or y is active before u. In the latter case, y joined the tree before u, so by the induction hypothesis, in either case we have $\text{level}(y) \leq \text{level}(u)$. Thus we have

$$\text{level}(v) = \text{level}(u) + 1 \geq \text{level}(y) + 1 = \text{level}(x),$$

so level$(v) \geq$ level(x) for all other vertices x in the tree at stage $m+1$, so the result is true at stage $m+1$.

Hence, the result is true by mathematical induction. ▌

For example, it is easy to check that Lemma 5.4.3 holds for the breadth-first search tree B given in Figure 5.5.

This result allows us to establish the following important fact about breadth-first search.

Theorem 5.4.4. *(The primary property of breadth-first search.)*
In a connected graph with a breadth-first search tree, each non-tree edge in the graph joins vertices that are at most one level apart in the search tree (of course each tree edge joins vertices that are exactly one level apart).

Proof: Suppose that vertices u and v are joined by an edge, and without loss of generality, that u joins the tree before v. Thus u is active before v, and there are two cases:
Case 1. v is in the tree when u first is active. Then u and v are joined by a non-tree edge, and pr$(v) = w$, where w joined the tree before u. Thus level$(w) \leq$ level(u), by Lemma 5.4.3, so we have

$$\text{level}(v) = \text{level}(w) + 1 \leq \text{level}(u) + 1.$$

But also level$(v) \geq$ level(u), by Lemma 5.4.3, so we conclude in this case that u and v are joined by a non-tree edge with

$$\text{level}(u) \leq \text{level}(v) \leq \text{level}(u) + 1,$$

and the result is true in this case.
Case 2. v is not in the tree when u first is active. Then v will be added to the tree, with pr$(v) = u$ and level (v) = level $(u) + 1$, and the edge between u and v is a tree edge. ▌

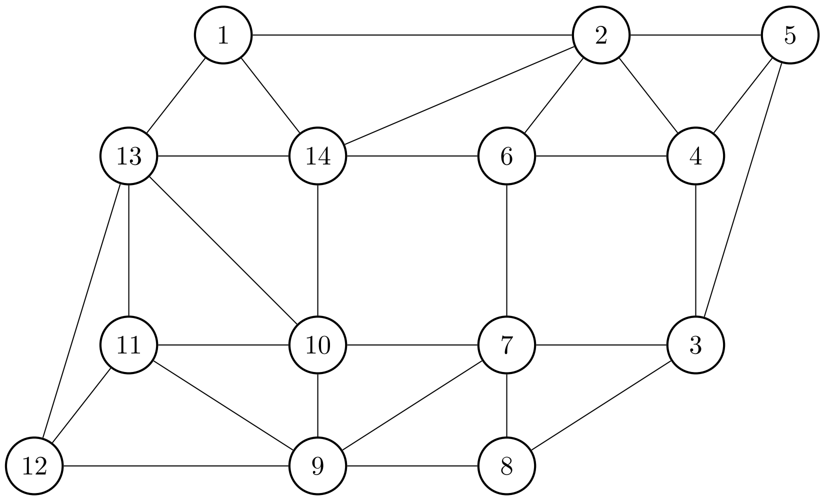
For example, it is easy to check that Theorem 5.4.4 holds for graph G and breadth-first search tree B in Figure 5.5. Here, non-tree edges $\{3,10\}$, $\{5,6\}$ join vertices at the same level, and $\{4,8\}, \{1,7\}$ join vertices one level apart.

Problem Set 5.4

1. Prove that if Algorithm 5.4.1 terminates with subgraph D containing fewer than p vertices, then D is a spanning tree for the component of G containing the initially chosen vertex.

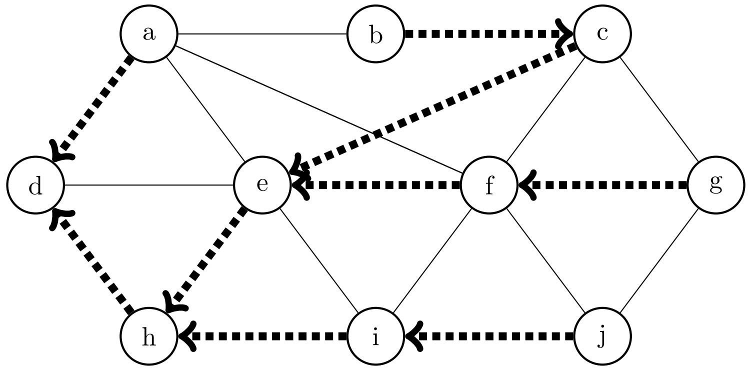
2. Consider a graph G with $V(G) = \{3, 4, 5, \ldots, 25\}$ and with $\{p, q\} \in E(G)$ if and only if either $p|q$ or $q|p$. (Definition: $p|q$ if and only if $q = pr$ for some integer r.) Give a spanning forest of G with the largest number of edges and hence determine the number of components of G.

3. Construct a breadth-first search tree for the graph below, taking vertex labelled 1 as root. When considering the vertices adjacent to the vertex being examined, take them in increasing order of their labels.



4. What graphs have the property that, for a suitable choice of root, the breadth-first search algorithm yields a tree in which all vertices (except the root) are at level 1?

5. Explain why the search tree, rooted at d and indicated by dotted lines in the graph below is not a breadth-first search tree.



6. Prove that if a non-tree edge joins vertices u and v in adjacent levels, say level $(v) = \text{level}\,(u) + 1$, then the parent of v is at the same level as u, and was active *before* u.

5.5 Applications of Breadth-First Search

Theorem 5.3.2 provides a nice characterization of bipartite graphs. Moreover, the ideas in Section 5.3 do provide an algorithm to find a bipartition or to find an odd cycle. However, breadth-first search provides a compact and efficient algorithm, as given in the proof of the following result. We refer to this proof as **constructive** because it allows us to construct directly either an odd cycle or a bipartition. Of course, for a graph that is not connected, we would carry this out by applying the construction for each component (and an odd cycle in any component would demonstrate that the graph is not bipartite).

Theorem 5.5.1. *A connected graph G with breadth-first search tree T has an odd cycle if and only if it has a non-tree edge joining vertices at the same level in T.*

Proof: (i) Suppose G has a non-tree edge joining distinct vertices u, v at the same level in T. Then the paths from u and v to the root vertex first meet at a vertex (possibly the root vertex) which is m levels less than u and v, for some $m \geq 1$. Then the path in T between u and v has length $2m$, and together with the non-tree edge $\{u, v\}$ this gives a cycle of length $2m + 1$ in G, so G has an odd cycle. (ii) If G has no non-tree edge joining vertices at the same level, then G is bipartite, with bipartition into sets of vertices $A = \{v \in V(G) : \text{level}(v) \text{ is odd }\}$, $B = \{v \in V(G) : \text{ level}(v) \text{ is even}\}$. All edges in G are incident with vertices whose levels differ by one; hence one level must be even and one must be odd, so every edge of G is incident with one vertex in A and one vertex in B. But, from Lemma 5.3, this means that G has no odd cycles. ∎

To find an odd cycle in a connected graph G, we need not construct the entire breadth-first search tree and then look for an edge between vertices at the same level. Instead, we can look for these edges when the tree is being built. At each stage, when we examine the vertex at the head of the queue to see if it is exhausted, we check if there is an edge joining it to a vertex already in the tree, at the same level.

Theorem 5.5.1 illustrates an important application of breadth-first search dealing with parity of cycle lengths. We now consider an application of breadth-first search dealing with shortest paths.

Theorem 5.5.2. *The length of a shortest path from u to v in a connected graph G is equal to the level of v in any breadth-first search tree of G with u as the root.*

Proof: From the primary property of breadth-first search trees we know that no edge of G joins vertices that are more than one level apart. If vertex v is at level k in a breadth-first search tree rooted at u, then there is a path (in the tree) of length k from u to v. There can be no shorter path from u to v in G, since such a path would have to contain an edge joining vertices more than one level apart. ∎

The length of the shortest path between two vertices is often called the **distance** between the vertices. Theorem 5.5.2 implies that we can determine the distance between vertex u and any other vertex in a connected graph G by finding a breadth-first search tree of G rooted at u.

Problem Set 5.5

1. Describe a general procedure to determine a maximal bipartite subgraph of any connected graph G. (A proof that your procedure works is not necessary.)

2. The **diameter** of a graph is the largest of the distances between the pairs of vertices in the graph. Let G be a connected graph of diameter three with exactly 20 vertices at distance three from a given vertex v. Prove that G has some spanning tree T with exactly 20 vertices of degree one at level three.

3. (a) Describe an algorithm to determine the diameter of a graph.

   (b) Prove that this algorithm works.

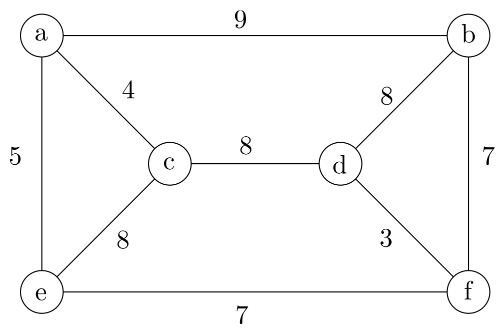
4. Let m, n be integers with $m, n \geq 1$. Let G be a graph (connected) with m vertices at distance n from a given vertex v. Prove that G has a spanning tree with at least m vertices of degree 1.

5. Suppose that a connected graph G has a breadth-first search tree T for which every non-tree edge joins vertices at equal levels. Prove that every cycle of G contains an even number of tree edges.

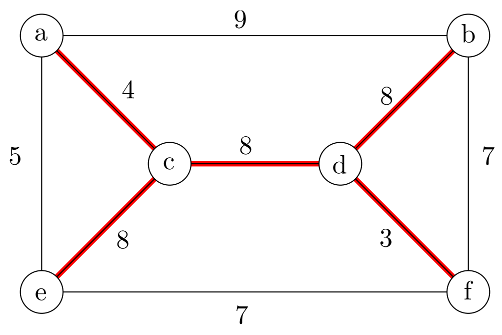
5.6 Minimum Spanning Tree

In the minimum spanning tree (MST) problem, we are given a connected graph G and a weight function on the edges $w : E(G) \to \mathbb{R}$. The goal is to find a spanning tree in G whose total edge weight is minimized.

For example, in the graph to the left, we have a connected graph with the edges being labelled with their weights. In the graph to the right, we have a spanning tree of total weight 31. But is there another spanning tree of smaller weight?



Edge-weighted connected graph G



A spanning tree of weight 31.

This problem is useful in the design of various networks, such as computer networks, road networks, and power grids. The edge weights would represent the costs of building between the two locations. A minimum spanning tree would represent the least amount of cost required to completely connect every location.

There are several efficient algorithms that can solve the MST problem. We present Prim's algorithm here. The idea of Prim's algorithm is that we start with a vertex, and iteratively grow the tree one edge at a time. Each time we grow the tree, we increase the total weight as small as possible.

Prim's algorithm:

1. Let v be an arbitrary vertex in G, let T be the tree consists of just v.

2. While T is not a spanning tree of G...

    (a) Look at all the edges in the cut induced by $V(T)$.

(b) Let $e = uv$ be an edge with the smallest weight in the cut (where $u \in V(T), v \notin V(T)$).

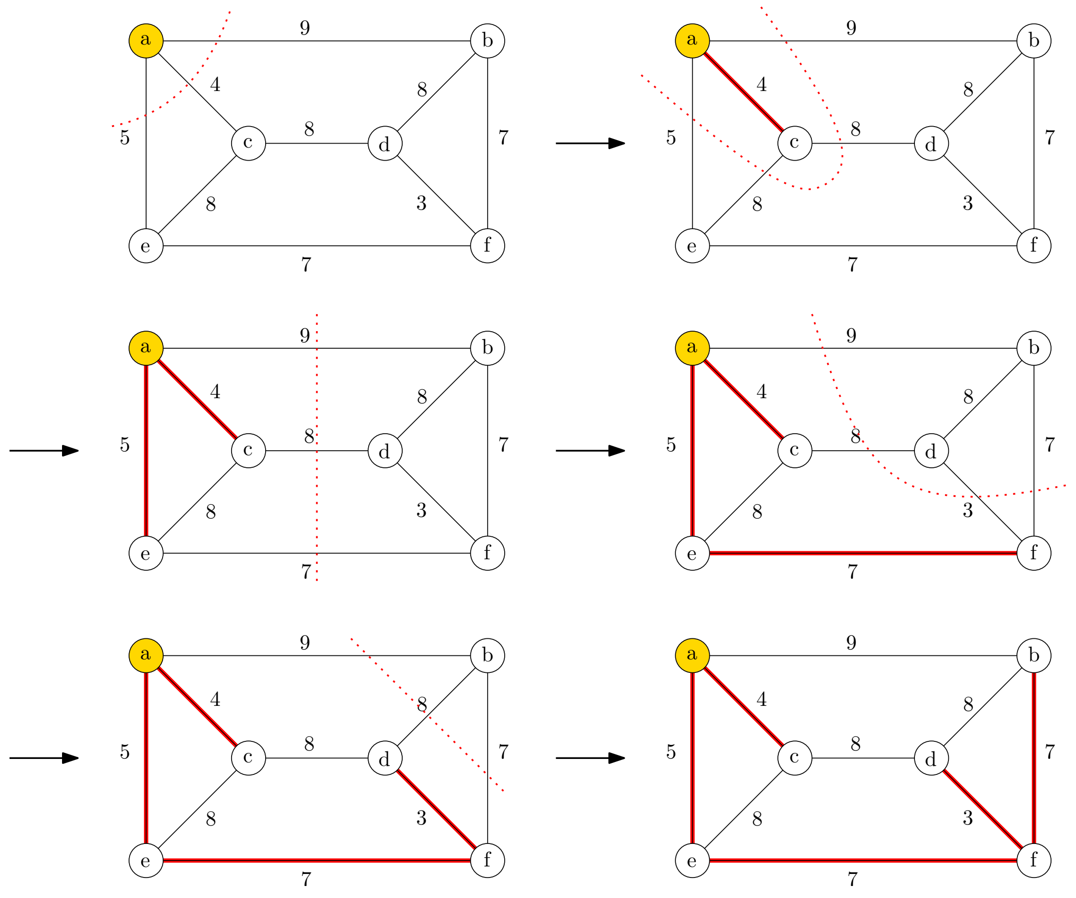
(c) Add v to $V(T)$ and add e to $E(T)$.

Note that in step 2, since T is not a spanning tree of G, $V(T)$ is a non-empty proper subset of $V(G)$. Therefore, by Theorem 4.8.5, the cut induced by $V(T)$ is non-empty, so it is possible to pick an edge from the cut in step 2(b).

Here is an illustration of the algorithm using the example above. We start with the vertex a. In the first iteration, we look at the edges in the cut induced by $\{a\}$, which are $\{ab, ac, ae\}$. Since ac has the smallest weight, we add it to the tree along with vertex c.

In the second iteration, we look at the edges in the cut induced by $\{a, c\}$, which are $\{ab, cd, ce, ae\}$. Since ae has the smallest weight, we add it to the tree along with the vertex e.

We repeat this process until we have a spanning tree. We claim that the tree we produce is a minimum spanning tree.



Prim's algorithm is a greedy algorithm, meaning at each step in the process, we pick the edge that is "best" for our problem. But how do we know that it will always produce a tree of minimum weight? This requires the following proof. For any graph H, we will use the notation $w(H) = \sum_{e \in E(H)} w(e)$.

Theorem 5.6.1. *Prim's algorithm produces a minimum spanning tree for G.*

Proof: Let $T_1, T_2, \dots, T_n$ be the trees produced by the algorithm at each step, where the order of selection of the edges is $e_1, e_2, \dots, e_{n-1}$ (i.e. you get T_{i+1} by adding e_i to T_i). We will prove by induction on k that there exists a MST containing T_k as a subgraph.

Base case: For $k = 1$, it is just a vertex, so every MST contains T_1 as a subgraph.

Induction hypothesis: Assume that there exists a MST containing T_k as a subgraph.

Induction step: We need to prove that there is a MST containing T_{k+1} as a subgraph.

Let T^* be a MST that contains T_k as a subgraph, which is assumed by the induction hypothesis. If T^* also contains e_k, then T^* contains T_{k+1} as a subgraph, and we are done. If not, then $e_k \not\in E(T^*)$. This means that $T^* + e_k$ contains a unique cycle C. Now $C - e_k$ is a path between the two endpoints of e_k, one of which is in $V(T_k)$ and the other is not. Therefore, there is at least one edge e' in $C - e_k$ in the cut induced by $V(T_k)$. By Theorem 5.2.3, $T' = T^* + e_k - e'$ is also a spanning tree.

In Prim's algorithm, when we picked e_k, it is an edge in the cut induced by $V(T_k)$ of minimum weight. So $w(e') \geq w(e_k)$. If $w(e') > w(e_k)$, then $w(T') = w(T^*) + w(e_k) - w(e') < w(T^*)$, which is not possible since T^* is a minimum spanning tree. Therefore, $w(e') = w(e_k)$, which means that $w(T') = w(T^*)$. So T' is also a minimum spanning tree, which contains all edges in T_{k+1}. Therefore, T' is the tree we are looking for.

This induction tells us that there is a MST that contains T_n, meaning T_n must equal to the MST. Hence the algorithm produces a MST. ▮

Chapter 6

Codes

6.1 Vector Spaces and Fundamental Cycles

Let $S(G)$ be the set of all spanning subgraphs of the graph G. Let the element of $S(G)$ with no edges be denoted by Z.

The field $GF(2)$ of integers modulo 2 contains two elements, 0 and 1, whose elements are added and multiplied modulo 2 (e.g. $0 \oplus 0 = 1 \oplus 1 = 0$, $1 \oplus 0 = 0 \oplus 1 = 1$, $0 \cdot 0 = 0 \cdot 1 = 1 \cdot 0 = 0$, $1 \cdot 1 = 1$). For any $H \in S(G)$, we define multiplication of H by an element of $GF(2)$ as follows:

$$0 \cdot H = Z, \;\; 1 \cdot H = H.$$

Addition of elements of $S(G)$ is also defined.

Definition 6.1.1. *For $H_1, H_2 \in S(G)$ the* **modulo 2 sum** *of H_1 and H_2, denoted by $H_1 \oplus H_2$, is the element of $S(G)$ whose edge set consists of all the edges of G that are in H_1 or H_2* but not in both *(the "symmetric difference" of $E(H_1)$ and $E(H_2)$).*

The set of all spanning subgraphs of G forms a vector space under the operation of mod 2 sum.

Theorem 6.1.2. *The set $S(G)$ is a vector space over $GF(2)$.*

Proof: Look up the definition of a vector space in your linear algebra text. You will find that the scalar multiplication and vector addition defined above must satisfy a list of axioms. These are all easy to verify.

For example, for any $H \in S(G)$,

$$Z \oplus H = H \oplus Z = H,$$

so Z is the “zero" element in $S(G)$. Also

$$H \oplus H = Z,$$

so each element of $S(G)$ has an additive inverse, namely itself. We have defined

$$0 \cdot H = Z,$$

and have, for instance,

$$1 \cdot H \oplus 1 \cdot H = H \oplus H = Z = 0 \cdot H = (1 \oplus 1) \cdot H.$$

Proving that all the axioms of a vector space are satisfied by $S(G)$ is left as an exercise. ▮

Suppose that the edges of G are $e_1, e_2, \ldots, e_q$, and let A_i be the element of $S(G)$ that contains edge e_i, and no other edges, for $i = 1, 2, \ldots, q$.

Theorem 6.1.3. $\{A_1, A_2, \ldots A_q\}$ *forms a basis for* $S(G)$.

Proof: The graphs $A_1, A_2, \ldots A_q$ are all contained in $S(G)$. To prove that they form a basis for $S(G)$, it is sufficient to prove that they (i) span $S(G)$ and (ii) form a linearly independent set.

(i) The elements of $S(G)$ are uniquely specified by their edges. For $H \in S(G)$, if H contains edges $e_{i_1}, e_{i_2}, \ldots, e_{i_k}$, then H can be written as the linear combination

$$H = \alpha_1 A_1 \oplus \cdots \oplus \alpha_q A_q$$

where $\alpha_{i_1} = \alpha_{i_2} = \cdots = \alpha_{i_k} = 1$, and all other α's are equal to 0. Thus $\{A_1, A_2, \ldots, A_q\}$ spans $S(G)$.

(ii) Suppose $\beta_1, \ldots, \beta_q$ are such that

$$\beta_1 A_1 \oplus \cdots \oplus \beta_q A_q = Z.$$

If $\beta_j = 1$ for some j, then the graph on the LHS contains edge e_j. But the graph on the RHS contains no edges, so for equality to hold, we must have $\beta_j = 0$ for all $j = 1, \ldots, q$. Thus $\{A_1, A_2, \ldots, A_q\}$ is a linearly independent set.

▮

This means that $S(G)$ is a vector space of dimension q and contains 2^q elements. Now we look at a subspace of $S(G)$.

Definition 6.1.4. *A graph in which all degrees are even non-negative integers is called an* **even** *graph.*

Let $C(G)$ be the set of even spanning subgraphs of a graph G, so $C(G)$ is a subset of $S(G)$.

Theorem 6.1.5. *The set $C(G)$ forms a vector space over $GF(2)$. (This is a subspace of $S(G)$.)*

Proof: Clearly $Z \in C(G)$, so $C(G)$ is not empty. Thus, to prove that $C(G)$ is a subspace of $S(G)$, it is sufficient to prove that (i) $C(G)$ is closed under scalar multiplication and (ii) $C(G)$ is closed under addition.

(i) Suppose $H \in C(G)$. Then $1 \cdot H = H \in C(G)$, and $0 \cdot H = Z \in C(G)$, since Z has all vertex degrees equal to zero, which is an even non-negative integer.

(ii) Suppose $H_1, H_2 \in C(G)$, and let $d_1(v)$ and $d_2(v)$ denote the degree of v in H_1 and H_2, respectively, for all $v \in V(G)$. Let $m(v)$ be the number of edges incident with v that are contained in both H_1 and H_2. Then the degree of v in $H_1 \oplus H_2$ is

$$(d_1(v) - m(v)) + (d_2(v) - m(v)) = d_1(v) + d_2(v) - 2m(v),$$

which is even, since both $d_1(v)$ and $d_2(v)$ are even, and non-negative, since $d_1(v) \geq m(v)$ and $d_2(v) \geq m(v)$. Thus $H_1 \oplus H_2 \in C(G)$.

▌

We have proved that $C(G)$ is a subspace of $S(G)$, so $C(G)$ is a vector space itself.

In order to find a basis for $C(G)$, we consider a result about trees.

Theorem 6.1.6. *Let T be a spanning tree of a connected graph G. If e is an edge of G that is not in T, then $T + e$ contains a unique cycle. (This cycle contains edge e.)*

Proof: Suppose $e = \{u, v\}$. Since T contains no cycles, any cycle in $T + e$ must contain edge e, so it must consist of e together with a path from u to v in T. But from the Lemma 5.1.3 there is a unique such path, and so $T + e$ contains a unique cycle, which passes through e. ▌

Edges e as described in the above result are called **non-tree** edges. If G has q edges and p vertices, then T has $p-1$ edges, so G has $q-(p-1)=q-p+1$ non-tree edges. Suppose that the non-tree edges are $e_1, e_2, \ldots, e_{q-p+1}$. Let C_i be the spanning subgraph of G whose edges are the edges of the unique cycle in $T+e_i$, for $i=1,\ldots,q-p+1$. Then C_i is called a **fundamental cycle**, and $\{C_1, C_2, \ldots, C_{q-p+1}\}$ is the set of fundamental cycles of G determined by T.

Lemma 6.1.7. *For a fixed spanning tree T of a connected graph G, no two elements of $C(G)$ contain exactly the same set of non-tree edges.*

Proof: Consider $H \in C(G)$ with no non-tree edges. Then H is a spanning subgraph of T, and is thus a spanning forest of G. If any component of H has more than one vertex, then there are at least two vertices of degree 1 in that component, by Theorem 5.1.8. But H is an even graph, so it can have no vertices of degree 1. Thus all components of H are isolated vertices, so H must be equal to Z. Hence Z is the unique elment in $C(G)$ with no non-tree edges.

Suppose that $H_1, H_2 \in C(G)$, where H_1 and H_2 contain exactly the same set of non-tree edges. Then $H_1 \oplus H_2$ has no non-tree edges. But $H_1 \oplus H_2 \in C(G)$, so $H_1 \oplus H_2 = Z$, since Z is the unique element of $C(G)$ with no non-tree edges. This gives $H_1 = H_2$ and the result follows. ▌

Theorem 6.1.8. $\{C_1, C_2, \ldots, C_{q-p+1}\}$ *forms a basis for $C(G)$, where G is connected.*

Proof: We have $C_i \in C(G)$, for $i=1,\ldots q-p+1$, since the vertex degrees in C_i are either 2 (for vertices on the cycle in $T+e_i$) or 0 (for the remaining vertices). Thus, to prove that $\{C_1, \ldots, C_{q-p+1}\}$ forms a basis for $C(G)$, it is sufficient to prove that (i) it spans $C(G)$ and (ii) it is a linearly independent set.

(i) From Lemma 6.1.7, the elements of $C(G)$ are uniquely specified by their non-tree edges. For $H \in C(G)$, where H contains non-tree edges $e_{i_1}, e_{i_2}, \ldots, e_{i_m}$, then H can be written as the linear combination

$$H = \alpha_1 C_1 \oplus \cdots \oplus \alpha_{q-p+1} C_{q-p+1}$$

where $\alpha_{i_1} = \alpha_{i_2} = \cdots = a_{i_m} = 1$ and all other α's are equal to 0. Thus $\{C_1, \ldots, C_{q-p+1}\}$ spans $C(G)$.

(ii) Suppose $\beta_1, \ldots, \beta_{q-p+1}$ such that

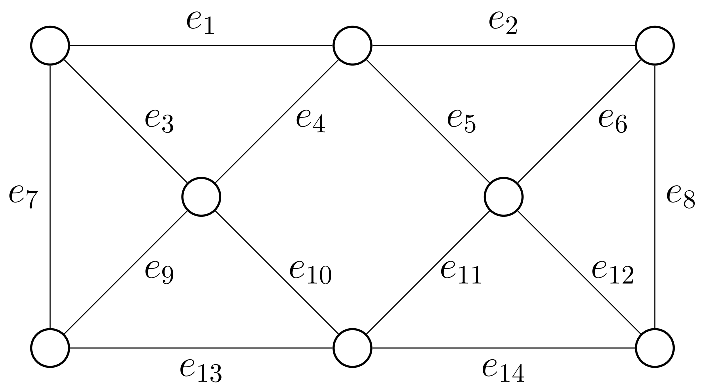
$$\beta_1 C_1 \oplus \cdots \oplus \beta_{q-p+1} C_{q-p+1} = Z.$$

If $\beta_j = 1$ for some j, then the graph on the LHS contains non-tree edge e_j, since C_j is the only fundamental cycle containing non-tree edge e_j. But the graph on the RHS contains no edges, so for equality to hold, we must have $\beta_j = 0$ for all $j = 1, \dots, q-p+1$. Thus $\{C_1, \dots, C_{q-p+1}\}$ is a linearly independent set.

∎

This means that $C(G)$ has dimension $q-p+1$ and contains 2^{q-p+1} elements. The cycles of G (together with some isolated vertices) are elements of $C(G)$ and thus can be uniquely expressed as a mod 2 sum of fundamental cycles. We call $C(G)$ the **cycle space** of G. The number $q-p+1$ is known as the **cyclomatic** number of G.

Problem Set 6.1



1. (a) Find the fundamental cycles corresponding to the spanning tree T with edges

$$e_1, e_2, e_5, e_7, e_{10}, e_{11}, e_{14}$$

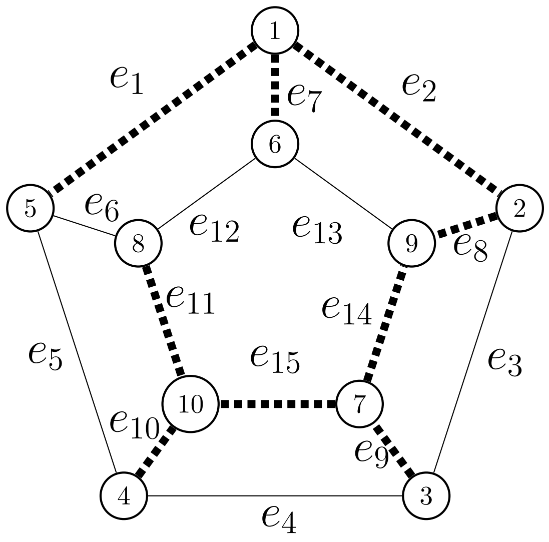
in the above graph. (List their edges.)

(b) Express the spanning even subgraphs with the following edge sets as modulo 2 sums of fundamental cycles from (a):

(i) $E(H_1) = \{e_1, e_2, e_4, e_5, e_7, e_8, e_{10}, e_{11}, e_{13}, e_{14}\}$

(ii) $E(H_2) = \{e_3, e_7, e_9, e_{11}, e_{12}, e_{14}\}$

2. (a) Let T be the spanning tree given by dotted lines in the graph below. Write down the edges in each of the fundamental cycles determined by T.



(b) Express the cycle whose edges are $e_1, e_7, e_{13}, e_{14}, e_{15}, e_{10}$ and e_5, as a modulo 2 sum of fundamental cycles from (a).

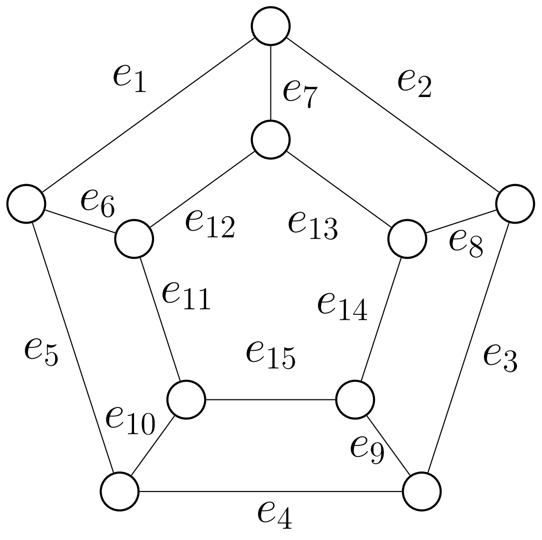
3. In the graph shown below, let T be the spanning tree with edges

$$e_1, e_5, e_6, e_8, e_9, e_{10}, e_{13}, e_{14}, e_{15}.$$

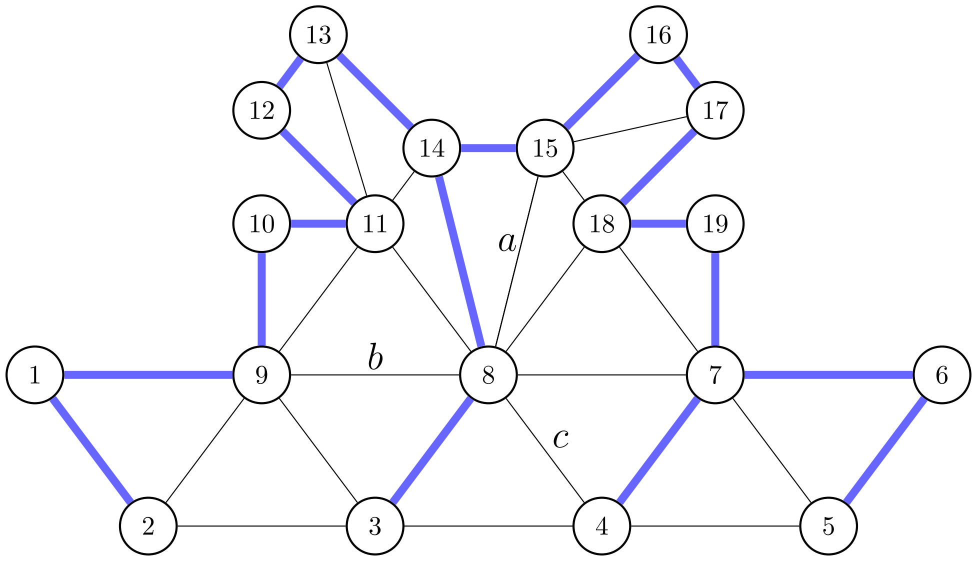
Write down the fundamental cycles determined by T. Also, express the cycle C, whose edges are

$$e_2, e_7, e_{12}, e_{11}, e_{15}, e_{14}, e_8$$

as a linear combination of fundamental cycles.



4. (a) In the graph below, a spanning tree has been constructed. Find the fundamental cycles C_a, C_b and C_c that are determined by the edges a, b and c.



(b) Find also the subgraph $C_a \oplus C_b \oplus C_c$.

5. G is a connected graph with a spanning tree T. For each statement below, either give a proof or find a counterexample.

(a) If G is bipartite, then every fundamental cycle of T must have even length.

(b) If every fundamental cycle of T has even length, then every cycle of G must have even length.

(c) If every fundamental cycle of T has length divisible by 3, then every cycle of G must have length divisible by 3.

6. Prove that a graph which is connected, with 13 vertices and 18 edges, must have at least 6 distinct cycles. Can you find such a graph with 6 edge-disjoint cycles (edge-disjoint means that no edge may belong to more than one of the cycles)?

7. Prove that an edge belongs to $H_1 \oplus H_2 \oplus \cdots \oplus H_k$ if, and only if, it belongs to an odd number of the H_i's where $H_1, H_2, \ldots, H_k \in S(G)$.

8. What is the dimension of $C(G)$ if G has m components, for arbitrary $m \geq 1$?

9. Let $\mathbf{F}(G)$ be the set of spanning subgraphs with an even number of edges of a graph G. (The spanning subgraph containing no edges belongs to $\mathbf{F}(G)$. **Note:** $\mathbf{F}(G)$ is **<u>not</u>** the set of even spanning subgraphs of G.)

   (a) If $F_1, F_2 \in \mathbf{F}(G)$, prove that $F_1 \oplus F_2 \in \mathbf{F}(G)$, where $F_1 \oplus F_2$ is the modulo 2 sum of F_1 and F_2.

   (b) Part (a) implies that $\mathbf{F}(G)$ is a vector space over $GF(2)$. Find a basis for $\mathbf{F}(G)$, and prove that it is a basis. What is the dimension of $\mathbf{F}(G)$, in terms of p and q, the number of vertices and edges in G?

6.2 Graphical Codes

We now explore a nice application of the cycle space of a graph.

Let $G = (V, E)$, $E = \{e_1, \ldots, e_q\}$ be a fixed connected graph. The **characteristic vector** $(x_1, \ldots, x_q)$ of subgraph (V, F) has $x_i = 1$ if $e_i \in F$ and $x_i = 0$ otherwise for $1 \leq i \leq q$. Suppose that $\underline{x}$ and $\underline{y}$ are characteristic vectors of two even subgraphs of G. How many nonzero elements does $\underline{x} \oplus \underline{y}$ contain? It may contain none, if $\underline{x} = \underline{y}$. But if $\underline{x} \neq \underline{y}$, then $\underline{x} \oplus \underline{y}$ is an even subgraph which necessarily contains a cycle; then $x \oplus \underline{y}$ must have at least three nonzero entries.

Definition 6.2.1. *The* **Hamming distance** *between two binary vectors $\underline{x}$ and $\underline{y}$ of the same length is the number of 1's in $\underline{x} \oplus \underline{y}$.*

We can equally well speak about the Hamming distance between subgraphs, using their characteristic vectors.

We have essentially proved the following:

Lemma 6.2.2. *In the cycle space of a graph G, any two distinct vectors have Hamming distance at least three.*

Now let us see how to use this observation. Suppose that two people want to communicate over a "channel" (such as a telephone line), capable of sending $0,1$-sequences, but this channel on rare occasions introduces an error by transmitting a 0 when a 1 was intended, or vice versa.

The two people, say Alice and Bob, can agree on a graph $G = (V,E)$ and a labelling $E = \{e_1,\dots,e_q\}$ of its edges, and further agree only to transmit as messages the characteristic vectors of even subgraphs of G. Suppose that Alice sends a message m to Bob and that Bob receives a message m'; both m and m' are binary vectors of length q. Bob can check whether m' is an even subgraph easily, but can he be sure that the message m' is in fact the message m? If m' is not an even subgraph, Bob is absolutely sure that $m' \neq m$ (naturally, assuming that Alice is playing by the rules and transmitting an even subgraph). But if m' *is* an even subgraph, Bob can only be sure that *either*

- m' is indeed m; or
- the channel introduced at least *three* errors (since m' and m must be at Hamming distance at least three by Lemma 6.2.2).

If errors on the channel are indeed rare, then Bob can reasonably suppose that, if m' is even, it is indeed the message sent by Alice.

Bob can actually do better than this. If only one error was introduced (i.e. $m \oplus m'$ has only one '1' in it), then there is a *unique* even subgraph closest to m. For suppose that $m_1 \oplus m'$ and $m_2 \oplus m'$ both have only one 1; then $(m_1 \oplus m') \oplus (m_2 \oplus m') = m_1 \oplus m_2$ has at most two 1's, and by Lemma 6.2.2, we must have $m_1 = m_2$. So, in particular, m is the **unique** closest vector to m' if only one error was made. Of course, this does not tell us how to *correct* the single error, just that a single error has a unique correction; more on this later.

Let us formalize this. Let C be a set of binary vectors of length q forming a vector space of dimension d; then C has 2^d vectors in it. The **distance** t of C is the minimum Hamming distance between any two distinct vectors of C. Such a set C is called a **binary** (q,d,t)**-error correcting code**, or just a (q,d,t)**-code**. Vectors in C are called **codewords**. The distance of a code is of fundamental importance in its use:

Lemma 6.2.3. *If fewer than t errors are made in the transmission of a codeword, then the received message is either the original codeword, or it is not a codeword at all.*

Lemma 6.2.4. *If at most $\lfloor \frac{t-1}{2} \rfloor$ errors are made in the transmission of a codeword m, the Hamming distance between the received message m' and a codeword is minimum for the unique codeword m.*

Prove these two lemmas for yourself.

Lemma 6.2.5. *The cycle space of a connected graph G on p vertices and q edges is a*
$(q, q-p+1, t)$-code for some $t \geq 3$.

This code is called the **even graphical code** of G. What is the distance of such a code?

If G has a cycle of length ℓ, this cycle has Hamming distance ℓ from the void graph Z. So evidently the girth g of G is an upper bound on the distance. Can two even subgraphs S_1 and S_2 have Hamming distance less than g? Consider $S_1 \oplus S_2$. If $S_1 \neq S_2$, then $S_1 \oplus S_2$ is an even subgraph, and hence it contains a cycle. But all cycles have at least g edges, and so

Lemma 6.2.6. *The distance of the even graphical code of G equals the girth of G.*

As an aside, let's observe that we can extend our definition of cycle space to multigraphs in which repeated edges are permitted. If we choose the two vertex multigraph with q edges between the vertices, the cycle space is just the set of all binary vectors of length q with an even number of 1's. This is the standard **even parity** code.

We are left with a significant question: correcting errors. Suppose that an even subgraph M is transmitted, but that a subgraph S is received which is *not* even. Now $S = M \oplus E$, where E marks the positions in which a transmission error was made; equivalently, $M = S \oplus E$. Both E and M are unknown to us; however, we do know *all* of the valid codewords, and when G has girth g, we assume that at most $\lfloor \frac{g-1}{2} \rfloor$ errors were introduced. Under this assumption, E cannot contain a cycle (why?). To find E, let's observe that a vertex v has odd degree in S if and only if it has odd degree in E (for it surely has even degree in M). So E is a subgraph of G with a specified set of odd degree vertices and as few edges as possible. There is only one candidate for E, since if $S \oplus E_1$ and

$S \oplus E_2$ are codewords and $E_1 \neq E_2$, $E_1 \oplus E_2$ is a nonzero codeword — so we have a contradiction unless more than $\lfloor \frac{g-1}{2} \rfloor$ errors were made. When the number of errors is “small", E can easily be found by brute-force techniques; when larger, algorithms for weighted matching in graphs can be used (this material is covered in C&O 450).

(Aside: Essentially what we do in the general case is to identify the set W of odd degree vertices of S. There must be an even number $2s$ of them by the handshake theorem. We choose a labelling of the vertices in W as $x_1, \ldots, x_s$, $y_1, \ldots, y_s$ so that the sum of the distances between x_i and y_i for $i = 1, \ldots, s$ is *minimum* over all labellings of the vertices. Then find paths $P_1, \ldots, P_s$ where P_i is a shortest path between x_i and y_i. Determine $E = P_1 \oplus P_2 \oplus \cdots \oplus P_s$. Finally compute $M = S \oplus E$ to correct the errors, if any.)

One topic of interest is to determine which graphs give “good" codes (large distance, large dimension and small length). These amount to requiring “large" girth, “large" cyclomatic number, and “small" number of edges. The Petersen graph gives a $(15, 6, 5)$-code, for example. Finding the best codes (and even the best graphical codes) is a challenging problem, discussed in much more detail in CO 331.

Problem Set 6.2

1. Produce all codewords in the even graphical code of K_4. Determine the length, distance and dimension.

2. Produce all codewords in the even graphical code of $K_{3,3}$. Determine the length, distance and dimension.

3. Prove that every connected graph on an even number of vertices has a spanning subgraph in which all vertices have odd degree. (Hint: consider modulo 2 sums of paths.)

4. If G is a connected graph on an even number of vertices, Problem 3 ensures that G has an odd subgraph S. Consider the vector space $C(G)$ of even subgraphs and let $C^+(G) = C(G) \cup \{H \oplus S : \ H \in C(G)\}$. Is $C^+(G)$ a vector space under modulo 2 sum? Compute the length, distance and dimension of the corresponding code.

5. Using Problem 4, extend the $(15, 6, 5)$-code from the Petersen graph to a $(15, 7, 5)$-code. Describe all codewords.

Chapter 7

Planar Graphs

7.1 Planarity

Definition 7.1.1. *A graph G is* **planar** *if it has a drawing in the plane so that its edges intersect only at their ends, and so that no two vertices coincide. The actual drawing is called a* **planar embedding** *of G, or a* **planar map**.

For example, the 3-cube, which we previously considered in Figure 4.12, is a planar graph, with a planar embedding given in Figure 7.1. A planar graph may have a number of essentially different embeddings.

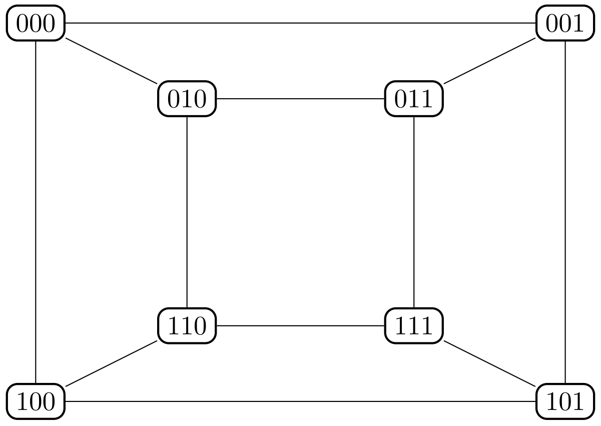


Figure 7.1: A planar embedding of the 3-cube

It is clear that a graph is planar if and only if each of its components is planar. So it is often sufficient to consider only connected planar graphs and connected planar embeddings.

A planar embedding partitions the plane into connected regions called **faces**; one of these regions, called the outer face, is unbounded. For example, the planar embedding given in Figure 7.2 has 4 faces, identified as f_1, f_2, f_3, f_4 in the diagram. In this case, the outer face is f_4.

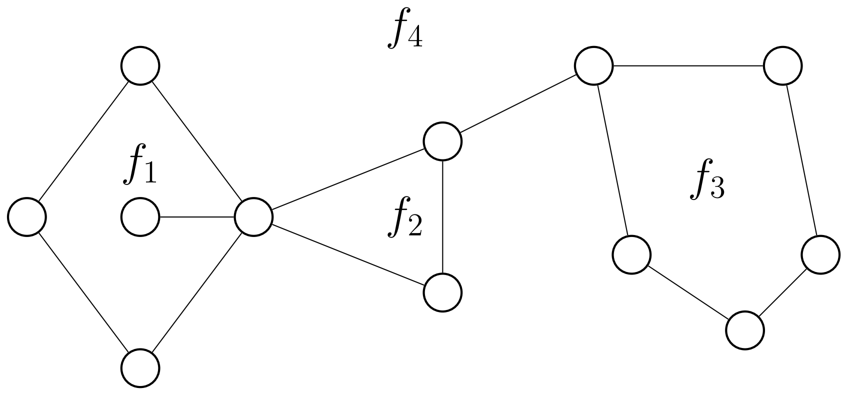


Figure 7.2: A planar embedding with 4 faces

Consider a planar embedding of a connected graph G. The subgraph formed by the vertices and edges in a face is called the **boundary** of the face. We say that two faces are **adjacent** if they are incident with a common edge. Assume, for the moment, that G is connected. As one moves around the entire perimeter of a face f, one encounters the vertices and edges in a fixed order, say

$$W_f = (v_0, e_1, v_1, e_2, v_2, \dots, v_{n-1}, e_n, v_n)$$

where $v_n = v_0$. This sequence is a closed walk of the graph G, and we call it the **boundary walk** of face f. (The boundary walk can start at any vertex, and can proceed around the perimeter in either a clockwise or counterclockwise direction.) The number of edges in the boundary walk W_f is called the **degree** of the face f. For example, in Figure 7.2,

$$\deg(f_1) = 6, \ \deg(f_2) = 3, \ \deg(f_3) = 5, \ \deg(f_4) = 14.$$

In Figure 7.1, all faces have degree 4. Note that a bridge of a planar embedding is incident with just one face, and is contained in the boundary walk of that face twice, once for each side. Thus a bridge contributes 2 to the degree of the face with which it is incident. On the other hand, if e is an edge of a cycle of an embedding, e is incident with exactly two faces, and is contained in the boundary walk of each face precisely once.

Every edge in a tree is a bridge, so a planar embedding of a tree T has a single face of degree $2|E(T)| = 2|V(T)| - 2$.

In what follows we may use 's' for the number of faces in a planar embedding.

Theorem 7.1.2. *If we have a planar embedding of a connected graph G with faces $f_1, \dots, f_s$, then*

$$\sum_{i=1}^{s} \deg(f_i) = 2|E(G)|.$$

Proof: Each edge has two sides, and when we sum the degrees of the faces we are counting the edges twice, once for each side. ∎

Note the similarity between Theorem 7.1.2 and Theorem 4.3.1. This theorem is colloquially known as the **Faceshaking Lemma** or the **Handshaking Lemma for Faces**. We shall make a direct link between these results later when we consider the dual of a planar embedding.

Corollary 7.1.3. *If the connected graph G has a panar embedding with f faces, the average degree of a face in the embedding is $\frac{2|E(G)|}{f}$.*

So far our discussion deals with planar embeddings that are *connected.* For a planar embedding of a disconnected graph, there could be faces whose boundaries lie on several components, and a closed walk around the boundary is not possible. For such faces, we alternatively define their degrees to be the sum of the lengths of the boundaries around each component. For example, in the embedding in Figure 7.3, the face f is incident with 3 components of the graph. The boundary walks around these 3 components have lengths 5, 4, and 2, so the degree of face f is 11.

Note that each edge is still counted twice among all boundaries walks, so Theorem 7.1.2 also holds for disconnected graphs.

7.2 Euler's Formula

There are often a number of completely different planar embeddings of a planar graph. However, every planar embedding of a given connected planar graph has the same number of faces, a fact that we can deduce from the following result, called **Euler's Formula**.

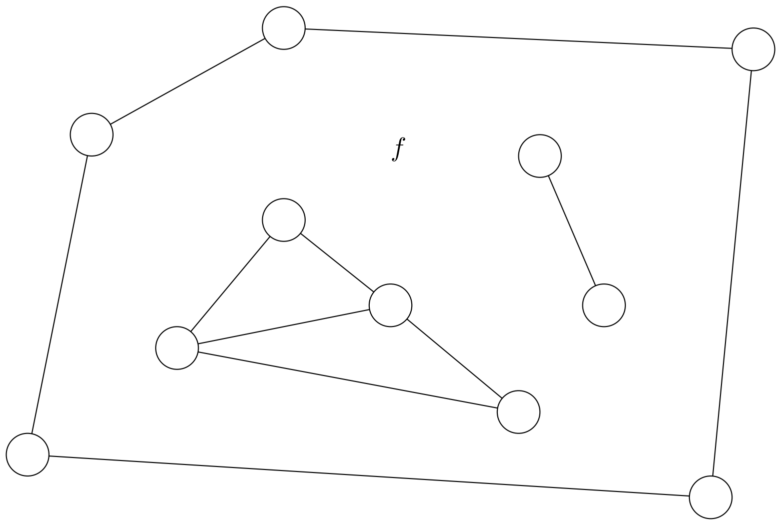


Figure 7.3: A planar embedding of a disconnected graph

Theorem 7.2.1. *(Euler's Formula) Let G be a connected graph with p vertices and q edges. If G has a planar embedding with f faces, then*

$$p - q + f = 2.$$

Proof: For each positive integer p, we prove this result by induction on q. Since G is a connected, it has a spanning tree and so $q \geq p - 1$.

As a tree has no cycles, any planar embedding of a tree has just one face, and the theorem holds.

So assume $q > p - 1$, and assume inductively that Euler's formula is true for any connected graph on p vertices with fewer than q edges. Suppose that we have a planar embedding of G with f faces. Since $q \geq p$ we see that G is not a tree and therefore it has an edge $e = \{u, v\}$ that is not a bridge. Then we also have a planar embedding of $G \setminus e$ (the graph we get from G by deleting the edge e). Since $G \setminus e$ has p vertices and $q - 1$ edges and is connected, it follows by induction that if it has f_1 faces, then

$$p - (q - 1) + f_1 = 2$$

and therefore $f_1 = q + 1 - p$. If we put e back into our drawing, it divides a face into two. So the embedding of G has one more face than that of $G \setminus e$. Hence the number of faces in the embedding of G is $q + 2 - p$ and then

$$|V(G)| - |E(G)| + q + 2 - p = p - q + q + 2 - p = 2.$$

∎

As an example of Euler's Formula, consider the connected planar embedding in Figure 7.2. In this case there are 12 vertices, 14 edges and 4 faces and $12 - 14 + 4 = 2$, as expected.

7.3 Stereographic Projection

There is a lack of symmetry among the faces of a planar map—one of the faces is unbounded. Symmetry can be achieved, making all the faces bounded, by considering embeddings on the surface of a sphere rather than on the plane. The main result in this section is the following:

Theorem 7.3.1. *A graph is planar if and only if it can be drawn on the surface of a sphere.*

Any drawing on the plane can be converted to a drawing on the sphere via **stereographic projection**. Let the sphere be tangent to the plane at point A, and let B be antipodal to A on the sphere. In stereographic projection, the image of each point x on the plane is the unique point x' on the surface of the sphere that lies on the line between x and B. This is illustrated in Figure 7.4.

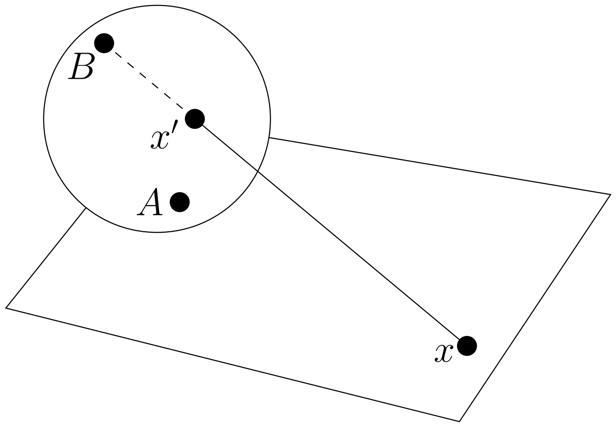


Figure 7.4: Stereographic projection

If we apply stereographic projection to a planar embedding, we obtain a drawing of the graph on the surface of the sphere (with no edges crossing), in which all faces are bounded, and B is in the interior of a face.

On the other hand, given any face f of an embedding G on the sphere, stereographic projection provides a way to obtain a planar embedding H in which the outer (unbounded) face corresponds to f—turn the sphere so that point B lies in face f, and then project the embedding G to the plane to get H. If we redraw an embedding so that a different face becomes the outside face, we consider this to be the same as the original embedding. Roughly speaking, we have the same graph, and the same faces, and the same faces are incident with the the same edges, so they are essentially the same embedding. In particular the number of faces of degree i in two embeddings related in this way will be equal.

A graph may have a number of essentially different planar embeddings. Figure 7.5 exhibits two embeddings of a planar graph. In the first embedding there are two faces of degree three and two of degree five; in the second, there are two faces of degree three, one of degree four and one of degree six. It is reassuring to note that in both embeddings there are four faces (so Euler is happy) and the sum of the faces degrees is 16 in both embeddings, as it should be by Theorem 7.1.2.

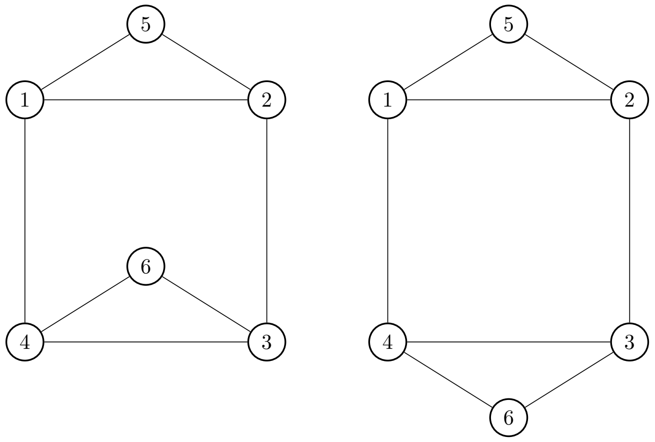


Figure 7.5: Embeddings of a Planar Graph

Problem Set 7.3

1. Prove that every planar embedding has either a vertex of degree at most 3 or a face of degree 3.
2. Prove that each of the graphs shown in Figure 7.6 is planar, by exhibiting a planar embedding.

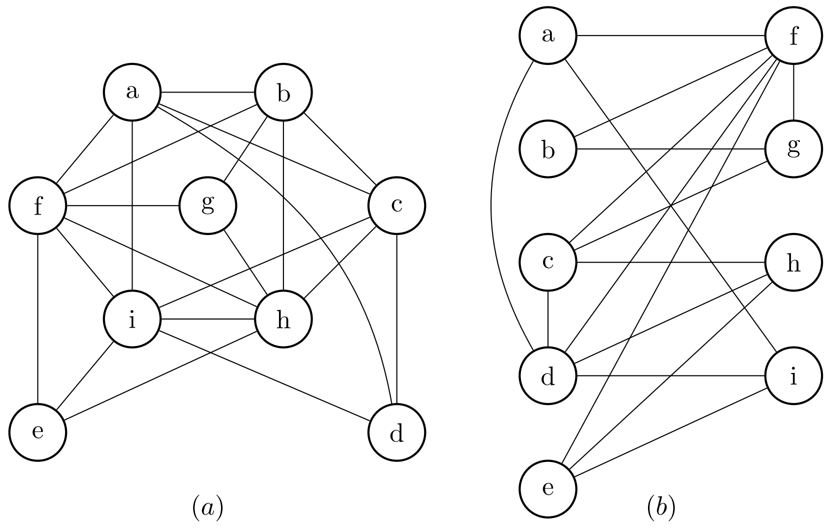


Figure 7.6: Planarity exercises

3. Let $n \geq 3$ be an integer. Suppose that a convex n-gon is drawn in the plane, and then each pair of nonadjacent corner points is joined by a straight line through the interior. Suppose that no 3 of these lines through the interior meet at a common point in the interior. (Figure 7.7 shows such a drawing with $n = 6$.)

   Let f_n be the number of regions into which the interior of the n-gon is divided by this process. (So $f_3 = 1,\ f_4 = 4,\ f_5 = 11$.) Use Euler's Formula to find f_n. (Hint: any set of 4 corner points of the n–gon uniquely determines a pair of intersecting lines in the interior.)

7.4 Platonic Solids

Consider the two geometric solids in Figure 7.8; the cube and the tetrahedron. These polyhedra exhibit a great deal of symmetry. In particular, the faces have the same degree and the vertices have the same degree. We call all such polyhedra **platonic solids**. Surprisingly, there are just five platonic solids: the tetrahedron, the cube, the octahedron, the dodecahedron, and the icosahedron. In this section we will outline a proof of this remarkable fact.

From each platonic solid, we can obtain a planar embedding in which all vertices have the same degree $d \geq 3$ and all faces have the same degree $d^* \geq 3$;

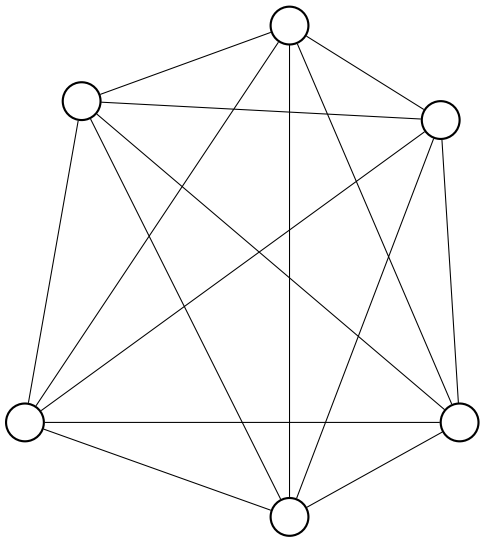


Figure 7.7: A convex 6-gon with diagonals

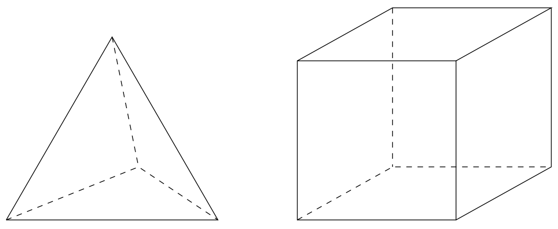


Figure 7.8: The tetrahedron and the cube

see Figure 7.9. We call a graph **platonic** if it admits a planar embedding in which each vertex has the same degree $d \geq 3$ and each face has the same degree $d^* \geq 3$. We will show that the only platonic graphs are those given in Figure 7.9, from which it is easy to deduce that there are just five platonic solids.

Theorem 7.4.1. *There are exactly five platonic graphs.*

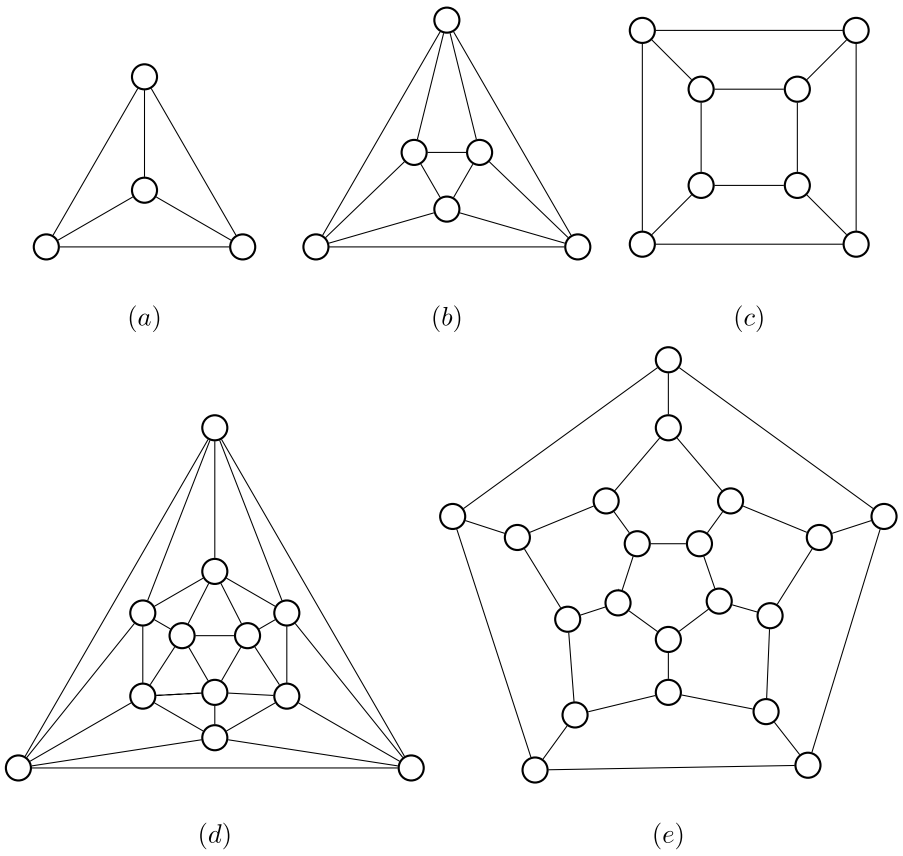


Figure 7.9: (a) the tetrahedron; (b) the octahedron; (c) the cube;(d) the icosahedron; (e) the dodecahedron

We require the following lemma.

Lemma 7.4.2. *Let G be a planar embedding with p vertices, q edges and s faces, in which each vertex has degree $d \geq 3$ and each face has degree $d^* \geq 3$. Then (d, d^*) is one of the five pairs*

$$\{(3,3),\ (3,4),\ (4,3),\ (3,5)\ (5,3)\}.$$

Proof: Assume $p = |V(G)|$, $q = |E(G)|$ and that there are exactly f faces in the embedding. From Euler,

$$p - q + f = 2.$$

Since G is regular with degree d, we have $2q = dp$ by Theorem 4.3.1. Since each face of the embedding has degree d^*, we have $2q = d^* f$ by Theorem 7.1.2. Hence $p = 2q/d$ and $f = 2q/d^*$. So we can write Euler's equation as

$$2 = \frac{2q}{d} - q + \frac{2q}{d^*} = q\left(\frac{2}{d} - 1 + \frac{2}{d^*}\right).$$

We rewrite this in turn as

$$\frac{2}{d} + \frac{2}{d^*} = 1 + \frac{2}{q}. \tag{7.4.1}$$

The basic idea now is to note that the right side of this equality is greater than 1, while the left is struggling to reach 1.

If $d = 3$ and $d^* \geq 6$, then

$$\frac{2}{d} + \frac{2}{d^*} \leq \frac{2}{3} + \frac{2}{6} = 1.$$

If $d = 4$ and $d^* \geq 4$, then

$$\frac{2}{d} + \frac{2}{d^*} \leq \frac{2}{4} + \frac{2}{4} = 1.$$

If $d = 5$ and $d^* \geq 4$, then

$$\frac{2}{d} + \frac{2}{d^*} \leq \frac{2}{5} + \frac{2}{4} = \frac{18}{20} < 1.$$

Finally, if $d \geq 6$ and $d^* \geq 3$, then

$$\frac{2}{d} + \frac{2}{d^*} \leq \frac{2}{6} + \frac{2}{3} = 1.$$

It follows that (d, d^*) must be one of the five pairs in the statement of lemma. ∎

Lemma 7.4.3. *If G is a platonic graph with p vertices, q edges and f faces, where each vertex has degree d and each face degree d^*, then*

$$q = \frac{2dd^*}{2d + 2d^* - dd^*}$$

and $p = 2q/d$ and $f = 2q/d^*$.

Proof: In proving the previous lemma we saw that

$$2 = q\left(\frac{2}{d} - 1 + \frac{2}{d^*}\right).$$

and the formula for q follows at once from this. We also saw that $2q = pd = fd^*$, which yields the other two claims. ∎

To prove Theorem 7.4.1, we look at the five possible values for (d, d^*) given in the previous lemma, and show that in each case there is a unique planar embedding with the required parameters. We will consider the first two cases; the remaining three cases are left as exercises.

Case 1: $d = 3$, $d^* = 3$.

Thus $q = \frac{2\cdot3\cdot3}{2\cdot3+2\cdot3-3\cdot3} = 6$, $p = \frac{2\cdot6}{3} = 4$, and $s = \frac{2\cdot6}{3} = 4$. Note that K_4 is the only graph having 4 vertices and 6 edges.

Case 2: $d = 3$, $d^* = 4$.

Thus $q = \frac{2\cdot3\cdot4}{2\cdot3+2\cdot4-3\cdot4} = 12$, $p = \frac{2\cdot12}{3} = 8$, and $s = \frac{2\cdot12}{4} = 6$. Consider a planar embedding G with these parameters. Since each face has degree 4, no vertex can be repeated in a boundary walk, so each face boundary is a 4–cycle. Let $C = (v_1, v_2, v_3, v_4, v_1)$ be the boundary of one of the faces of G. If v_1 is adjacent to v_3 then (v_2, v_3, v_1, v_2) is part of some boundary walk. This contradicts that each face is bounded by a 4–cycle. Therefore, v_1 is not adjacent to v_3, and, similarly, v_2 is not adjacent to v_4. Since each vertex has degree 3, v_i has a unique neighbour u_i not in C, for $i = 1, 2, 3, 4$. Note that u_1, v_1, v_2, u_2 is part of the boundary walk of some face, so $u_1 \neq u_2$ and u_1u_2 is an edge. Similarly u_2u_3, u_3u_4, and u_4u_1 are also edges. If $u_1 = u_3$ then u_4 is only incident with two faces, namely $(u_4, v_4, v_1, u_1, u_4)$ and $(u_4, v_4, v_3, u_1, u_4)$. This contradicts that u_4 has degree 3, and, hence, $u_1 \neq u_3$. By symmetry, $u_2 \neq u_4$. Therefore G is the 3–cube.

Problem Set 7.4

1. Show that there is a unique planar embedding in which each vertex has degree 4 and each face has degree 3.

2. Show that there is a unique planar embedding in which each vertex has degree 3 and each face has degree 5.

3. Show that there is a unique planar embedding in which each vertex has degree 5 and each face has degree 3.

7.5 Nonplanar Graphs

In order to prove that a graph is planar, we can find a planar embedding. How would we prove that a graph is nonplanar? In this section we will see that in some case we can use Euler's formula to do this.

We need one technical result before we can start.

Lemma 7.5.1. *If G contains a cycle, then in a planar embedding of G, the boundary of each face contains a cycle.*

Proof: Since G has a cycle, it has more than one face.

Therefore, every face f is adjacent to at least one other face, say g.

Let $e_1 = \{v_0, v_1\}$ be an edge that is incident with both f and g. Let H be the component in the boundary of face f containing the edge e_1. Let

$$W_f = (v_0, e_1, v_1, e_2, v_2, \ldots, v_{n-1}, e_n, v_0)$$

be the boundary walk of H. Since the edge e_1 is incident with both f and g, it is contained in W_f precisely once.

The edge e_1 is not a bridge of H because

$$(v_1, e_2, v_2, \ldots, v_{n-1}, e_n, v_0)$$

is a walk from v_1 to v_0 in $H - e_1$. Therefore, by Theorem 4.10.3, H contains a cycle. This establishes the result. ▌

Lemma 7.5.2. *Let G be a planar embedding with p vertices and q edges. If each face of G has degree at least d^*, then $(d^* - 2)q \le d^*(p-2)$.*

Proof: We first deal with the case when G is connected. Let $f_1, f_2, \ldots, f_s$ be the faces of G. Thus, applying Theorem 7.1.2, we have

$$2q = \sum_{i=1}^{s} \deg(f_i) \ge \sum_{i=1}^{s} d^* = d^* s.$$

Now, by Euler's Formula, $s = q + 2 - p$. So,

$$2q \ge d^* s = d^*(q + 2 - p) = d^* q + 2d^* - d^* p.$$

Therefore, $(d^* - 2)q \le d^*(p-2)$.

If G is not connected, then we obtain G' as follows: For each face f in the embedding incident with at least 2 components, pick two of these components, and pick one vertex from each component that is incident with f. Add an edge joining these two vertices. Since we may draw the edge in f, the resulting embedding is still planar. Also, the number of components is reduced by 1. Repeat this process until we have a connected planar embedding G'. Since the degree of a face does not decrease in this process, every face of G' still has degree at least d^*. Therefore, the inequality holds for G'. But G has fewer edges than G', so the inequality also holds for G. ▌

Lemma 7.5.2 relies on a given planar embedding, since it involves the face degrees. We would like similar inequalities for *planar graphs*. The following lemma allows us to relate face degrees to the lengths of cycles, which are independent of the embedding.

Note that a graph with p vertices can have as many as $p(p-1)/2$ edges, a quadratic function of p. However, this is not the case for planar graphs. The following theorem shows that the number of edges in a planar graph is bounded by a linear function in the number of vertices p. (Note that if we allow multiple edges or loops, fixing p does not bound the number of edges; for example, the graph in Figure 7.10 has $p = 2$ vertices and q edges for any positive integer q.)

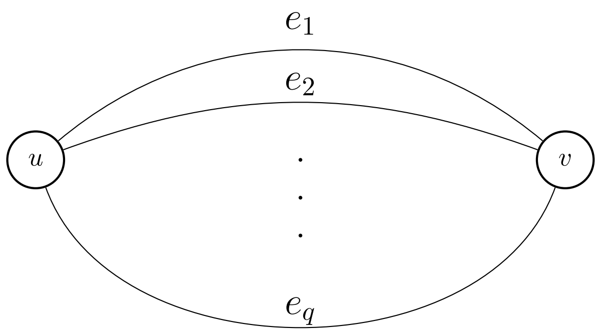


Figure 7.10: Multiple edges in a planar multigraph

Theorem 7.5.3. *In a planar graph G with $p \geq 3$ vertices and q edges, we have*

$$q \leq 3p - 6.$$

Proof: If G does not contain a cycle, them G is a forest. By Corollary 5.1.6, $q \leq p - 1 \leq 3p - 6$ whenever $p \geq 3$.

If G contains a cycle, then by Lemma 7.5.1, every face boundary contains a cycle. Since each cycle in G has length at least three, each face has degree at least 3. By Lemma 7.5.2,

$$q \le 3(p-2)/(3-2) = 3p-6.$$

▮

Theorem 7.5.3 gives an inequality that holds for all planar graphs. Therefore, if a graph does not satisfy the inequality, then it is not planar. We illustrate this idea below on K_5.

Corollary 7.5.4. *K_5 is a not planar.*

Proof: We have

$$|E(K_5)| = \binom{5}{2} = 10.$$

But $3p-6 = 15-6 = 9$, so for K_5 we have

$$q = 10 > 9 = 3p-6,$$

and the inequality $q \le 3p-6$ does not hold. We conclude from Theorem 7.5.3 that K_5 is not a planar graph. ▮

Notice that Theorem 7.5.3 is only a necessary condition for a graph to be planar. If a graph satisfies $|E(G)| \le 3|V(G)|-6$, it does **not** follow that it is planar. As an example, consider the graph of Figure 7.11. Since it has K_5 as a subgraph, it cannot be planar, but $|E(G)| = 11 < 12 = 3|V(G)|-6$.

Corollary 7.5.5. *A planar graph has a vertex of degree at most five.*

Proof: This is true if $|V(G)| \le 2$. If $p = |V(G)| \ge 3$ and $q = |E(G)|$ then by Theorem 7.5.3,

$$q \le 3p-6$$

and therefore

$$\frac{2q}{p} \le 6 - \frac{12}{p}.$$

This shows that the average degree of a vertex in G is less than six, and therefore G contains a vertex of degree at most five. ▮

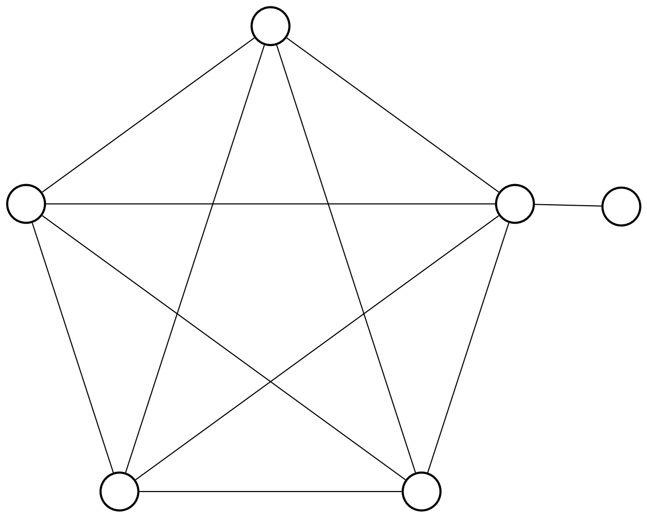


Figure 7.11: A nonplanar graph with $q \le 3p - 6$

After a few attempts to make a planar embedding of $K_{3,3}$, you will conclude that it also is not planar. However, neither Theorem 7.5.3 nor Corollary 7.5.5 implies that $K_{3,3}$ is not planar. (You should check this.) Nevertheless, one can repeat the proof of Theorem 7.5.3 with an additional observation to obtain a result strong enough to prove that $K_{3,3}$ is not planar.

Theorem 7.5.6. *In a bipartite planar graph G with $p \ge 3$ vertices and q edges, we have*

$$q \le 2p - 4.$$

Proof: If G does not contain a cycle, then G is a forest. By Corollary **??**, $q \le p - 1 \le 2p - 4$ whenever $p \ge 3$.

If G contains a cycle, then by Lemma 7.5.1, every face boundary contains a cycle. Since G is bipartite, it does not have cycles of length 3. Hence each cycle in G has length at least 4, and each face has degree at least 4. By Lemma 7.5.2,

$$q \le 4(p-2)/(4-2) = 2p - 4.$$

∎

Lemma 7.5.7. *$K_{3,3}$ is not planar.*

Proof: We have

$$|E(K_{3,3})| = 3 \cdot 3 = 9.$$

But $2p-4 = 2\cdot 6 - 4 = 8$, so for $K_{3,3}$ we have

$$q = 9 > 8 = 2p - 4$$

and the inequality $q \leq 2p-4$ does not hold. We conclude from Theorem 7.5.6 that $K_{3,3}$ is not a planar graph. ▮

7.6 Kuratowski's Theorem

We have been able to prove by counting arguments that two graphs, K_5 and $K_{3,3}$, are not planar. We are going to see how we can use graphs based on K_5 and $K_{3,3}$ to certify that a graph is not planar.

First we need some language to properly state our result.

An **edge subdivision** of a graph G is obtained by applying the following operation, independently, to each edge of G: replace the edge by a path of length 1 or more; if the path has length $m > 1$, then there are $m-1$ new vertices and $m-1$ new edges created; if the path has length $m = 1$, then the edge is unchanged. For example, Figure 7.12 shows a graph H, and an edge subdivision G of H.

Note that the operation of edge subdivision does not change planarity: if G is a planar graph, then all edge subdivisions of G are planar; if G is nonplanar, then all edge subdivisions of G are nonplanar. Similarly, note that if a graph G has a nonplanar subgraph, then G is nonplanar.

From these observations, we can immediately conclude that if a graph G has a subgraph isomorphic to an edge subdivision of $K_{3,3}$ or K_5, then G is nonplanar. One of the most famous results of graph theory, known as **Kuratowski's Theorem**, establishes that the converse of this result is also true.

Theorem 7.6.1. *A graph is not planar if and only if it has a subgraph that is an edge subdivision of K_5 or $K_{3,3}$.*

(We omit the proof of this result as it is beyond the scope of our current study.)

Note that Kuratowski's Theorem has the following surprising consequence. Suppose we begin with a nonplanar graph G and do the following operation as long as it is possible. Delete a vertex v or an edge e whose deletion leaves a nonplanar graph. When this process ends, what will the final graph $G^{'}$ be?

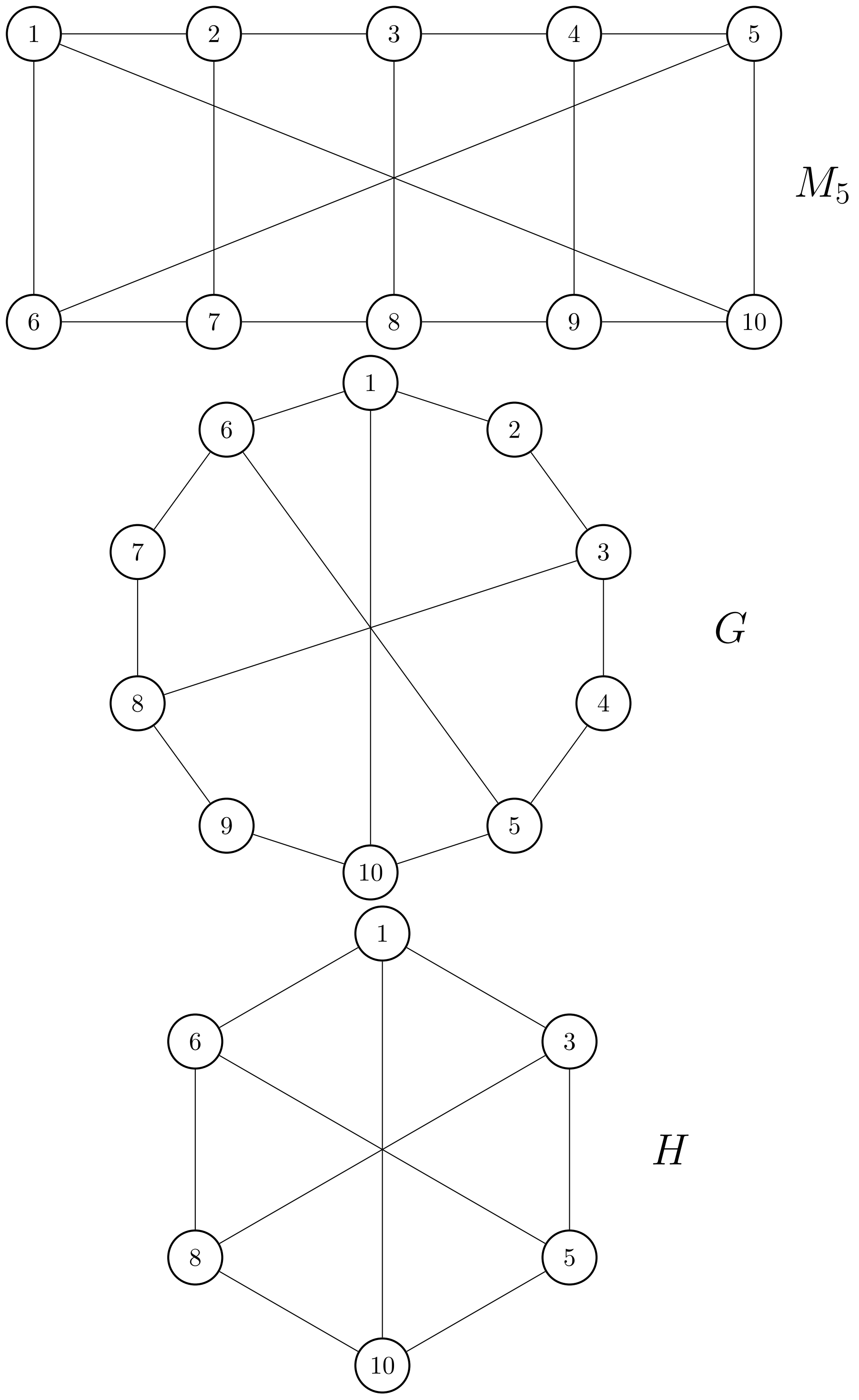


Figure 7.12: Edge subdivisions and an example of Kuratowski's Theorem

According to Kuratowski's Theorem, G' *must be* an edge subdivision of K_5 or $K_{3,3}$!

For example, in Figure 7.12, the graph M_5 has subgraph G, and G is an edge subdivision of H. Also, H is actually $K_{3,3}$, where the vertices in the two vertex classes are $\{1,5,8\}$ and $\{3,6,10\}$. Thus we conclude from Kuratowski's Theorem that M_5 is nonplanar. Figure 7.12 illustrates one convenient strategy when looking for an edge subdivision of $K_{3,3}$ or K_5: first find a long cycle in the graph – in this case the cycle has 10 vertices; then find edges or paths (subdivided edges) across the cycle – in this case edges {1, 10} and {5, 6}, together with any one of edges $\{2,7\},\{3,8\},\{4,9\}$, would create an edge subdivision of $K_{3,3}$.

Note that M_5 has 10 vertices, 15 edges and girth 4, so our counting results are not strong enough to prove that M_5 is not planar.

Problem Set 7.6

1. For each of the graphs in Figure 7.13, determine if it is planar. Prove your conclusion in each case.

2. Let G be a connected planar graph with p vertices and q edges and girth k. Show that
$$q \leq \frac{k(p-2)}{k-2}.$$
Show also that if equality holds, all faces of G have degree k.

3. Prove that the Petersen graph is nonplanar, without using any form of Kuratowski's theorem.

4. (a) Prove that the Petersen graph is nonplanar by Kuratowski's Theorem, finding a subgraph that is an edge subdivision of $K_{3,3}$.

   (b) Show that there exist two edges of the Petersen graph whose deletion leaves a planar graph.

5. Prove that the n-cube is not planar when $n \geq 4$, without using any form of Kuratowski's theorem.

6. (a) Prove that the 4-cube is nonplanar by Kuratowski's Theorem, finding a subgraph that is an edge subdivision of $K_{3,3}$.

   (b) Show that there exist four edges of the 4-cube whose deletion leaves a planar graph.

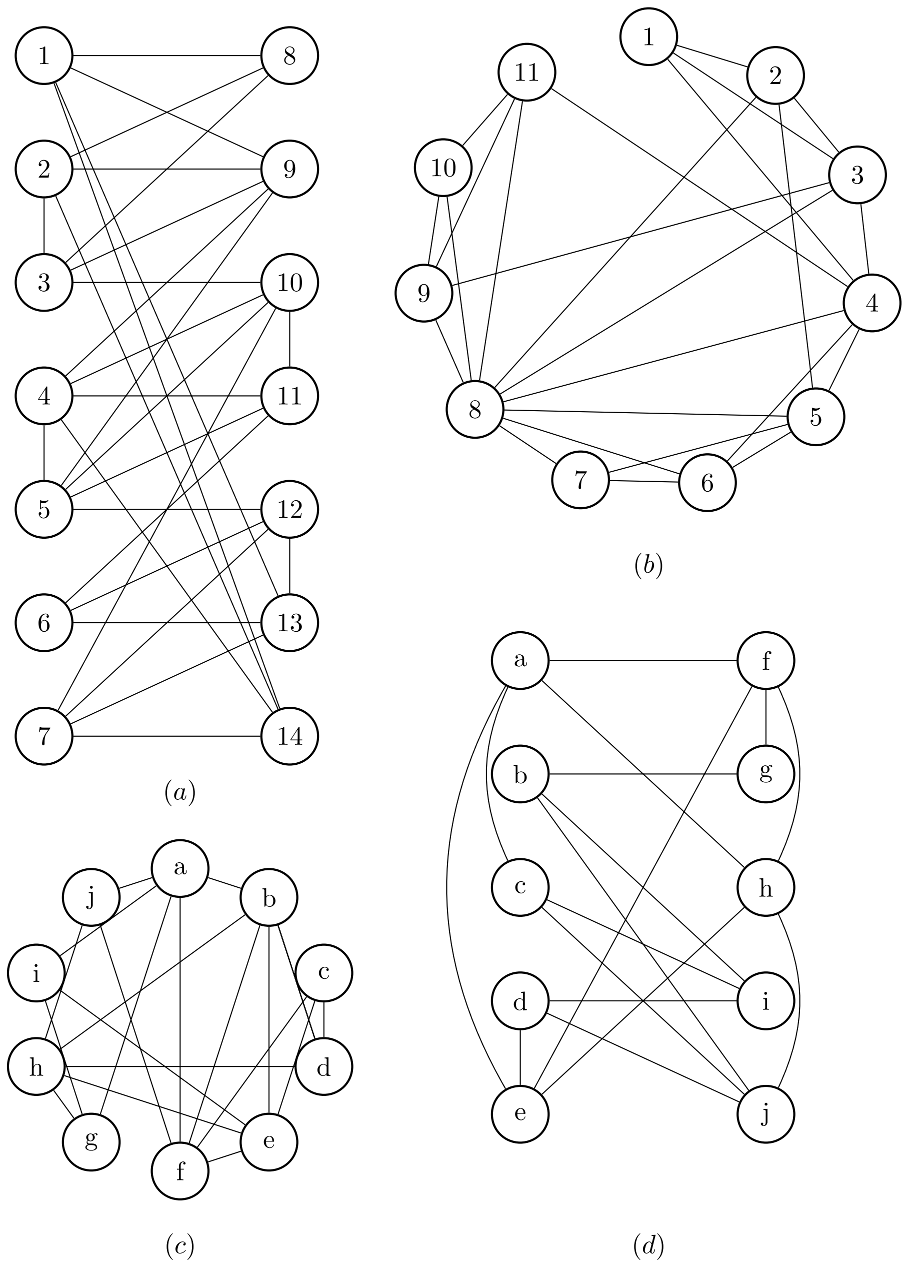


Figure 7.13:

(c) Show that no matter which 3 edges are deleted from a 4-cube, the resulting graph is not planar.

7. Consider the graph G_n with $V(G_n) = \{3, 4, \dots, n\}$ and $\{u, v\} \in E(G_n)$ if and only if $u|v$ or $v|u$.

   (a) Find the largest value k such that G_k is planar and give a planar embedding of G_k.

   (b) Prove that G_{k+1} is not planar using Kuratowski's Theorem.

   (c) Find $n < 50$ so that G_n has K_5 as a subgraph.

   (d) Find $n < 40$ so that G_n has $K_{3,3}$ as a subgraph.

8. Consider the prime graph B_n defined in Problem 11 of Problem Set 4.4.

   (a) Prove that B_8 is planar.

   (b) Using Kuratowski's Theorem, prove that B_9 is not planar.

9. Prove that every planar bipartite graph G has a vertex of degree at most three.

10. Let G denote the graph below. (You may assume, without proof, that G has girth six.)

    (a) Let H be any graph obtained from G by deleting two edges. Prove that H is not planar.

    (b) Prove that there exist three edges that can be deleted from G so that the resulting graph is planar.

11. Prove that every planar graph having girth at least six has a vertex of degree at most two. Prove that this is false if the girth is five.

12. Prove that if G is a planar graph in which every vertex has degree at least five, then $|V(G)| \geq 12$. Find such a graph with $|V(G)| = 12$.

13. (a) Prove that the complement of the 3-cube is nonplanar.

    (b) Prove that if G has $p \geq 11$ vertices, then at least one of G and $\bar{G}$ is nonplanar.

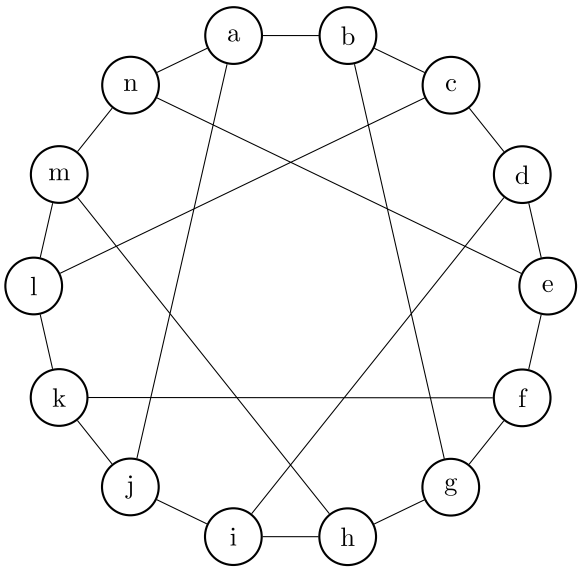


Figure 7.14: Planarity exercise.

7.7 Colouring and Planar Graphs

Definition 7.7.1. *A k-**colouring** of a graph G is a function from $V(G)$ to a set of size k (whose elements are called **colours**), so that adjacent vertices always have different colours. A graph with a k-colouring is called a k-**colourable** graph.*

For example, Figure 7.15 gives a 4-colouring of a graph. The colours in this case are the integers $1,2,3,4$, and the colour assigned (by the function) to each vertex is written beside the vertex in the diagram.

Theorem 7.7.2. *A graph is 2-colourable if and only if it is bipartite.*

Proof: If a graph has bipartition (A,B), then we obtain a 2-colouring by assigning one colour to all vertices in A, and the second colour to all vertices in B; no edge joins two vertices of the same colour since (A,B) is a bipartition (so all edges join a vertex in A to a vertex in B).

Conversely, if a graph has a 2-colouring, then let the vertices of one colour be set A, and the vertices of the second colour be set B. Then (A,B) is a bipartition since all edges join vertices of different colours. ∎

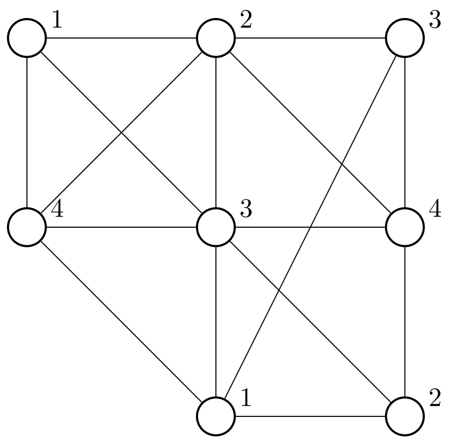


Figure 7.15: A 4-colouring of a graph

It follows that we have a description of all the 2-colourable graphs — they are precisely the graphs in which all cycles have even length. However, 3-colourable graphs are not well understood. There is no simple description known, and graph theorists believe that none exists.

Similarly, whereas there is an efficient algorithm to determine whether a graph is 2-colourable (Theorem 5.5.1), graph theorists believe that no efficient algorithm exists to determine whether a graph is 3-colourable.

Theorem 7.7.3. *K_n is n-colourable, and not k-colourable for any $k < n$.*

Proof: Any graph on n vertices is n-colourable; assign different colours to the vertices, so no two vertices have the same colour. K_n is not k-colourable for any $k < n$, since such a colouring would assign the same colour to some pair of vertices. But all pairs of vertices are adjacent in K_n, so we would have assigned the same colour to adjacent vertices. Thus K_n has no k-colouring for $k < n$. ▌

Theorem 7.7.4. *Every planar graph is* 6*-colourable.*

Proof: The proof is by induction on the number p of vertices. First note that all graphs on one vertex are 6-colourable, so the result is true for $p = 1$.

For the induction hypothesis, assume the result is true for all planar graphs on $p \leq k$ vertices, where $k \geq 1$.

Now consider a planar graph G on $p = k + 1$ vertices. From Corollary 7.5.5, G has a vertex v with $\deg(v) \leq 5$. Suppose we remove vertex v, and all edges incident to v, from G, and call the resulting graph G'. Then G' has k vertices, and is a planar graph (all subgraphs of a planar graph are planar). Thus we can apply the induction hypothesis to G', so G' is 6-colourable. Now find a 6-colouring of G'. There are at most 5 vertices in G' that are adjacent to v in G, so these vertices are assigned at most 5 different colours in the 6-colouring of G'. Thus there is at least one of the 6 colours remaining. Assign one of these remaining colours to v, so that v has a different colour from all of its adjacent vertices in G. Thus we have a 6-colouring of G, and the result is true for $p = k+1$. We have now proved that the result is true by mathematical induction. ▌

Let us proceed to the Five-Colour Theorem. The proof relies on the notion of edge-contraction which we now define.

Definition 7.7.5. *Let G be a graph and let $e = \{x, y\}$ be an edge of G. The graph G/e obtained from G by* **contracting** *the edge e is the graph with vertex set $V(G) \setminus \{x, y\} \cup \{z\}$, where z is a new vertex, and edge set*

$$\{\{u, v\} \in E(G) : \{u, v\} \cap \{x, y\} = \emptyset\} \cup \{\{u, z\} : u \notin \{x, y\}, \{u, w\} \in E(G) \text{ for some } w \in \{x, y\}\}.$$

Intuitively, we can think of the operation of contracting e as allowing the "length" of e to decrease to 0, so that the vertices x and y are identified into a new vertex z. Any other vertex that was adjacent to one (or both) of x and y is adjacent to z in the new graph G/e. See Figure 7.16 for an example.

If G has p vertices and q edges then G/e has $p-1$ vertices and at most $q-1$ edges.

Remark: G/e is planar whenever G is.

The converse of this remark is not true; G/e may be planar when G is non-planar. See Figure 7.7 for an example.

Now we are ready to prove the Five-Colour Theorem.

Theorem 7.7.6. *Every planar graph is 5-colourable.*

Proof: The proof is by mathematical induction on the number of vertices p of a planar graph.

For any graph G having one vertex, G is 5-colourable.

Induction Hypothesis: Assume that every planar graph on $p \le k$ vertices is 5-colourable, where $k \ge 1$.

Let G be any planar graph on $p = k+1$ vertices.

*Case 1: Graph **G** has a vertex of degree 4 or less.*

Proceeding as in the 6-colour theorem above, it can be shown that G is 5-colourable.

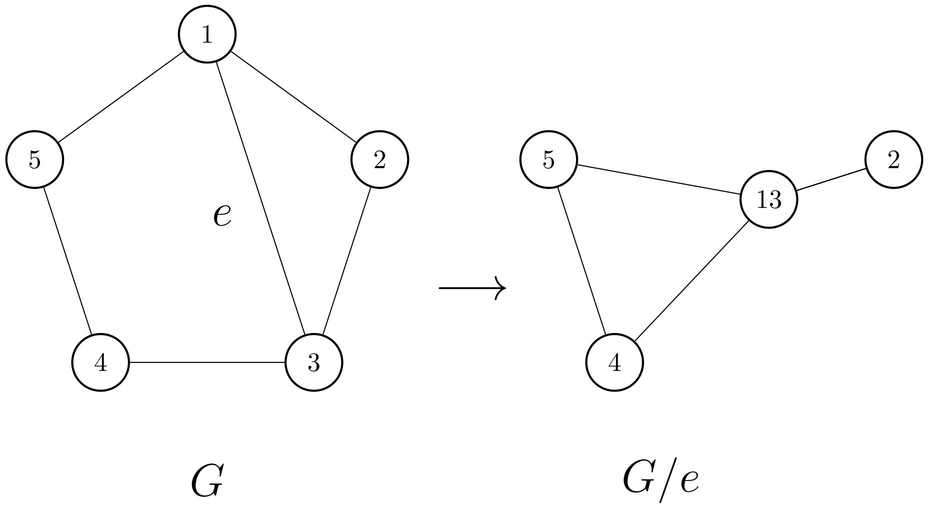


Figure 7.16: Contraction

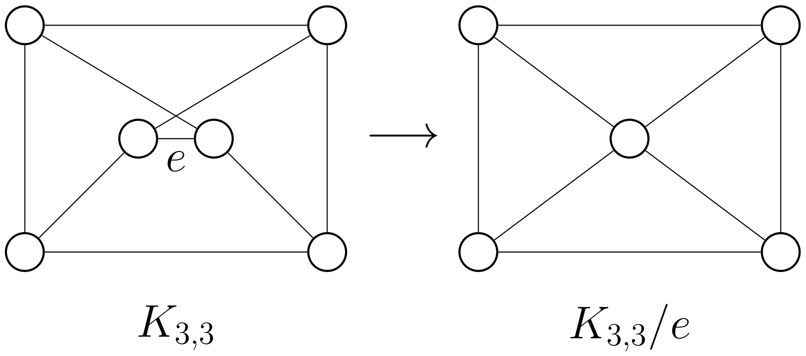


Figure 7.17: Contraction

*Case 2: Graph **G** has no vertex of degree 4 or less.*
Then, by Corollary 7.5.5, G has a vertex, say v, of degree 5.

If the 5 vertices joined to v are mutually adjacent, then G has a subgraph isomorphic to K_5. This is impossible since G is planar by hypothesis. Hence, at least 2 neighbours of v are not adjacent, say a and b.

Contract edge $e = \{a, v\}$ to get graph $H = G/e$, and label the new vertex v. Contract edge $f = \{b, v\}$ of H to get graph $K = H/f$, and label the new vertex v. Since G is planar, so are H and K; furthermore, K has $k-1$ vertices. Therefore K is 5-colourable by the induction hypothesis.

Use the 5-colouring of K to colour all the vertices of G except for a, b, and v. Since a and b are not adjacent in G, they can both accept the same colour. Colour both vertices a and b with the colour assigned to vertex v in K. Hence, at most four distinct colours appear on the five neighbours of v. Colour v with one of the absent colours. Then we have a valid 5-colouring of G.

In both cases, G has a 5-colouring.

Therefore, by mathematical induction it follows that every planar graph has a 5-colouring. ∎

The most famous result of graph theory is the “**Four colour theorem**". The known proofs are by mathematical induction, but involve many hundreds of cases, and use computer verification for cases. Its statement is as follows.

Theorem 7.7.7. *Every planar graph is* 4*-colourable.*

It is important to note that the converse of this result does not hold. For example, from Theorem 7.7.3 we know that K_5 is not 4-colourable, so it is correct to deduce from the Four Colour Theorem that K_5 is not planar. However, from Theorem 7.7.2 we know that $K_{3,3}$ is 2-colourable (and thus 4-colourable), but we know already from Corollary 7.5.7 that $K_{3,3}$ is not planar.

7.8 Dual Planar Maps

Note that our colourings are colourings of vertices, with the restriction that vertices joined by an edge have different colours. However, most descriptions of the Four Colour Theorem refer to colouring of regions, say countries in a map, with the restriction that regions with a common boundary have different colours.

In fact, these are completely equivalent results, as we show now. Given a connected planar embedding G, the **dual** G^* is a planar embedding constructed as follows: G^* has one vertex for each face of G. Two vertices of G^* are joined by an edge whenever the corresponding faces of G have an edge in common (one side for each face), and the edge in G^* is drawn to cross this common boundary edge in G. For example, Figure 7.18 illustrates the construction of G^* from a planar embedding G. Note that the faces of G^* now correspond to the vertices of G.

There are a number of things to note about the relationship between G and G^*. First, a face of degree k in G becomes a vertex of degree k in G^*, and a vertex of degree j in G becomes a face of degree j in G^*. Thus, Theorems 4.3.1 and 7.1.2 are the same result for planar embeddings, since Theorem 4.3.1 for G becomes Theorem 7.1.2 for G^* and vice versa. This connection between G and G^* is even stronger, since $(G^*)^*$ and G are the same graph.

Note that a bridge in G gives an edge in G^* between a vertex and itself (such an edge is called a loop), and more than one edge between two faces in G gives more than one edge between a pair of vertices (these are called, together, a multiple edge). Thus G^* may be a multigraph rather than a graph.

These complications aside, it is now clear that the Four Colour Theorem for colouring vertices in planar graphs is equivalent to the Four Colour Theorem for colouring faces in planar embeddings, via duality.

Problem Set 7.8

1. (a) Prove that every planar graph without a triangle (that is, a cycle of length three) has a vertex of degree three or less.

   (b) Without using Theorem 7.7.7, prove that every planar graph without a triangle is 4-colourable.

2. Prove that if G is a planar graph with girth at least six, then G is 3-colourable. (You may use the result of Problem 11 in Problem Set 7.6.)

3. Let G be a connected planar embedding with p vertices and q edges, and suppose that the dual graph G^* is isomorphic to G.

   (a) Prove that $q = 2p - 2$.

   (b) Give an example of such a graph with six vertices.

4. Let G be a connected planar embedding in which every face has even degree. Prove that G is a bipartite graph.

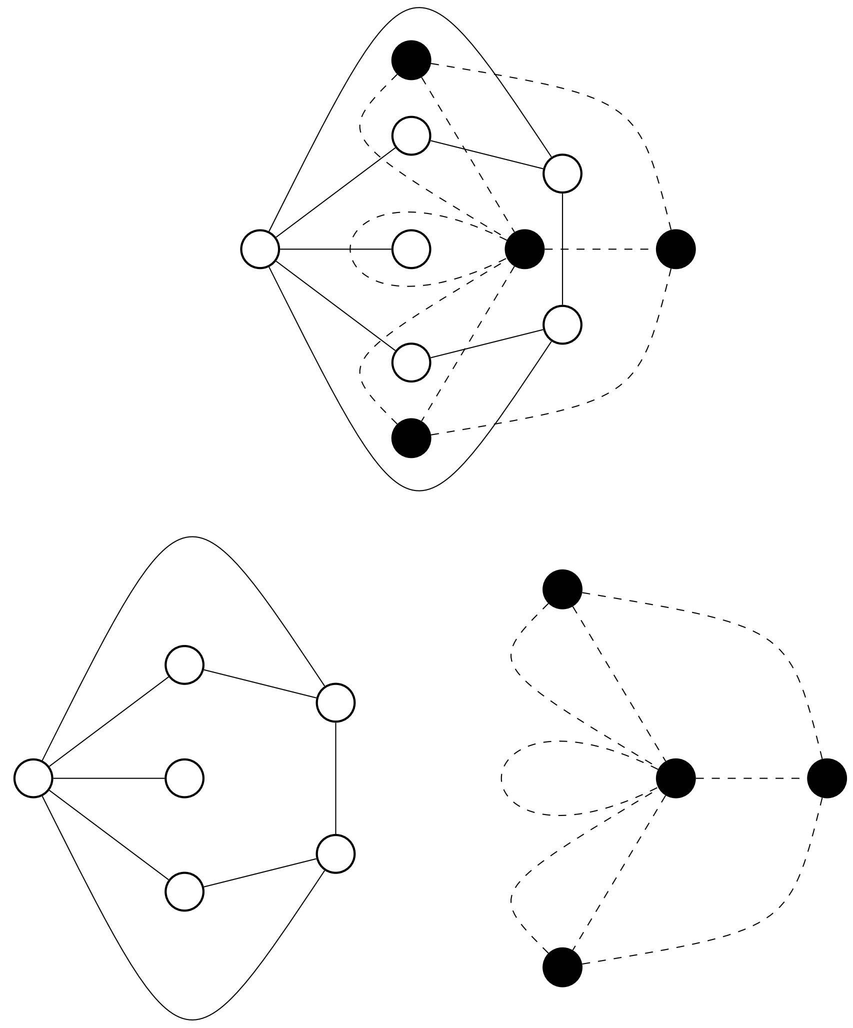


Figure 7.18: The dual of a planar embedding

5. Find a planar bipartite 3-regular graph with 14 vertices.

6. Show that a planar graph with $p > 2$ vertices and $2p-3$ edges is not 2-colourable.

7. Show that a graph with $2m$ vertices and $m^2 + 1$ edges is not 2-colourable.

8. Show that K_5 can be obtained by contracting five edges of the Petersen graph. Hence deduce from the nonplanarity of K_5 and the remark following the definition of edge-contraction, that the Petersen graph is nonplanar.

Chapter 8

Matchings

8.1 Matching

A **matching** in a graph G is a set M of edges of G such that no two edges in M have a common end. (Another way to express the condition is that in the spanning subgraph of G with edge set M, every vertex has degree at most 1.) So M *matches* certain pairs of adjacent vertices—hence the name. The thick edges in Figure 8.1 form a matching. We say that a vertex v of G is **saturated** by M, or that M **saturates** v, if v is incident with an edge in M. Of course, every graph has a matching; for example the empty set $\emptyset$ is always a matching. The question we will be most interested in is to find a largest matching in G, called a **maximum matching** of G. In Figure 8.1 the matching M indicated there has size 3, and therefore is not a maximum matching, since it is easy to find a matching of size 4. A special kind of maximum matching is one having size $p/2$, that is, one that saturates every vertex, called a **perfect matching**. Of course, not every graph has a perfect matching.

We will be concentrating on matching problems for bipartite graphs. Here is a way to restate the problem in case G is bipartite.

Job assignment problem. We are given a set A of workers and a set B of jobs, and for each job, the set of workers capable of doing the job. We want to assign as many jobs as possible to workers able to do them, but each worker is to be assigned to at most one job, and each job is to be assigned to at most one worker.

For example, suppose that $A = \{a, b, c, d, e, f\}$ and $B = \{g, h, i, j, k, l\}$, and the lists of workers that can do the jobs are:

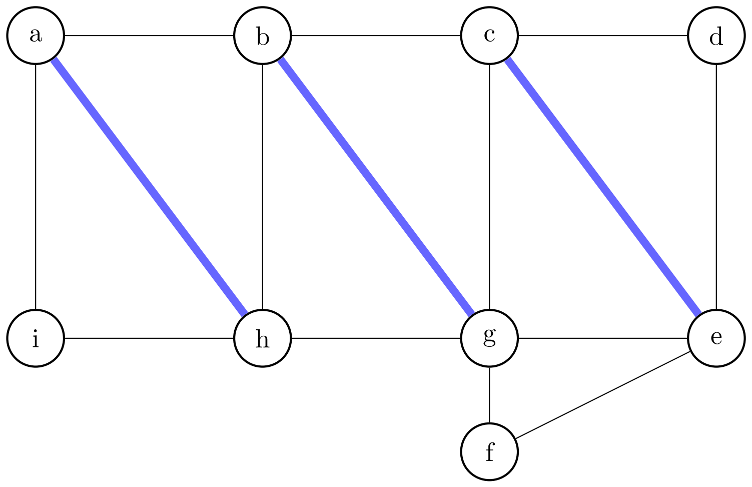


Figure 8.1: G with matching M

For g : c, e
For h : a, c
For i : a, b, c, d, f
For j : c, e
For k : c, e
For l : b, d, f

Why is the job assignment problem equivalent to the maximum matching problem of bipartite graphs? From the data of the job assignment problem we can construct a bipartite graph G with vertex set $A \cup B$ and with $u \in A$ adjacent to $v \in B$ if and only if worker u can do job v. (The graph corresponding to the sample data above is shown in Figure 8.2.) Conversely, given a bipartite graph with bipartition A, B we make a worker for each element of A and a job for each element of B and declare worker u to be able to do job v if and only if $\{u, v\} \in E(G)$. The condition that jobs be assigned to workers that can do them, means that an assignment is a set of edges. The condition that each worker be assigned to at most one job and that each job be assigned to at most one worker, corresponds to the condition that the assigned edges form a matching.

If we have a matching M of G, certain kinds of paths are useful for obtaining a larger matching. We say that a path $v_0 v_1 v_2 \ldots v_n$ is an **alternating path** with respect to M if one of the following is true:

$\{v_i, v_{i+1}\} \in M$ if i is even and $\{v_i, v_{i+1}\} \notin M$ if i is odd
$\{v_i, v_{i+1}\} \notin M$ if i is even and $\{v_i, v_{i+1}\} \in M$ if i is odd.

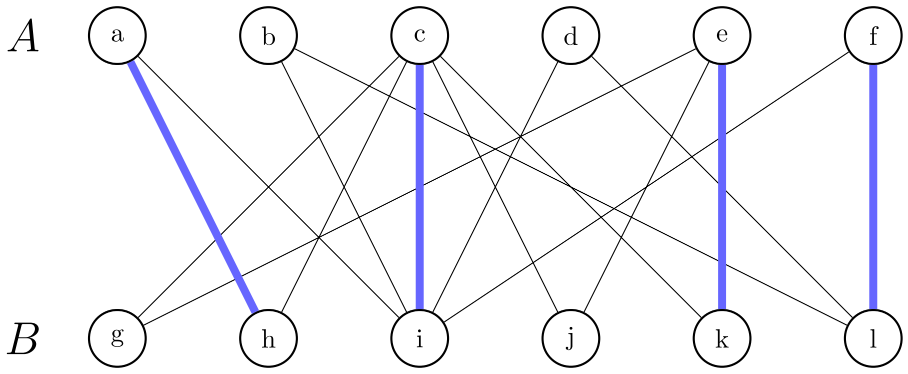


Figure 8.2: A bipartite example

That is, edges of the path are alternately in and not in M. In the graph of Figure 8.1, with respect to the matching indicated there, the following are examples of alternating paths: (i) $ahbg$, (ii) $iahgb$, (iii) $iahbgf$. An **augmenting path** with respect to M is an alternating path joining two distinct vertices neither of which is saturated by M. The path (iii) above is an augmenting path in Figure 8.1. Note that augmenting paths have odd length because they begin and end with nonmatching edges.

Lemma 8.1.1. *If M has an augmenting path, it is not a maximum matching.*

Proof: Let P be an augmenting path $v_0 v_1 v_2 \dots v_n$. Then $\{v_i, v_{i+1}\} \in M$ if i is odd, and $\{v_i, v_{i+1}\} \notin M$ if i is even. Moreover, n must be odd. So there are fewer edges of P in M than not in M. If we replace the edges of M that are in P by the other edges of P, we get a matching M' that is larger than M. ▌

8.2 Covers

A **cover** of a graph G is a set C of vertices such that every edge of G has at least one end in C. In Figure 8.1 $\{a, h, g, c, e\}$ is a cover. It is easy to find large covers, just as it is easy to find small matchings. For example, in any graph G, $V(G)$ is a cover. Also, if G is bipartite with bipartition A, B, then A is a cover, and so is B. A very useful observation about matchings and covers is the following.

Lemma 8.2.1. *If M is a matching of G and C is a cover of G, then $|M| \le |C|$.*

Proof: For each edge $\{u, v\}$ of M, u or v is in C. Moreover, for two different edges of M, any vertices of C they saturate must be different, since M is a matching. Therefore, $|M| \leq |C|$. ▌

Sometimes we can use a cover to prove that a matching is maximum.

Lemma 8.2.2. *If M is matching and C is a cover and $|M| = |C|$, then M is a maximum matching and C is a minimum cover.*

Proof: Let M' be any matching. Then by Lemma 8.2.1

$$|M'| \leq |C| = |M|.$$

It follows that M is a maximum matching. Now let C' be any cover. Then by Lemma 8.2.1

$$|C'| \geq |M| = |C|.$$

It follows that C is a minimum cover. ▌

We will show that if G is bipartite, then it is always possible to find such M and C, that is, that the maximum size of a matching is the minimum size of a cover. This is *König's Theorem*, the subject of the next section.

Problem Set 8.2

1. Show that a tree has at most one perfect matching.

2. How many perfect matchings are there in K_n? How many in $K_{m,n}$?

3. How many perfect matchings has the graph L_n of Figure 8.3? (There are n vertical edges.) (Hint: where a_n denotes the number of perfect matchings of L_n, find a recurrence relation for a_n.)

4. Show that for $n \geq 1$, the n-cube has a perfect matching.

5. Show that the 64 squares of a chessboard can be covered with 32 dominoes, each of which covers two adjacent squares.

6. Show that if two opposite corner squares of a chessboard are removed, then the resulting board cannot be covered with 31 dominoes.

7. Let G be a graph with even number of vertices. Prove that if G has a Hamilton cycle, then G has a perfect matching.

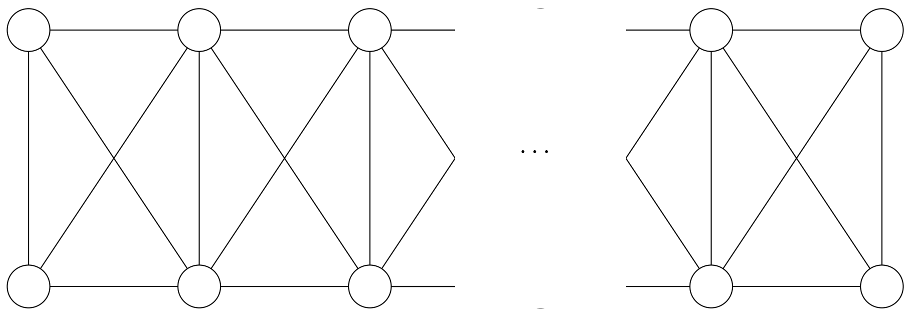


Figure 8.3: L_n

8. In the previous problem, suppose in addition that G is bipartite, with bipartition A, B. Let $u \in A$, $v \in B$, and let H denote the graph obtained from G by deleting u and v and their incident edges. Prove that H has a perfect matching.

9. Show that if two squares of the chessboard having opposite colours are removed, then the resulting board can be covered by 31 dominoes. Hint: Use the previous exercise.

10. Consider the prime graph B_n introduced in Problem 11 of Problem Set 4.4. Use induction on n to show that, if n is even, B_n has a perfect matching. You may use without proof the fact that there is a prime number between k and $2k$ for $k \geq 2$.

11. Prove that C is a cover of G if and only if $V(G) \setminus C$ is a set of pairwise nonadjacent vertices.

12. Show that it is not always true that there exist a matching M and a cover C of the same size.

13. Let N be a matrix. We want to find a largest set of non-zero entries of N such that no two are in the same row or in the same column. Formulate this problem as one of finding a maximum matching in a bipartite graph.

14. In the previous problem interpret the meaning of a cover of the bipartite graph in terms of the matrix N.

15. Find a bipartite graph G with bipartition A, B where $|A| = |B| = 5$, and having the following properties. Every vertex has degree at least 2, the total number

of edges is 16, and G has no perfect matching. Why does your graph not have a perfect matching?

16. Suppose that for some $n \geq 1$, graph G with p vertices satisfies $p = 2n$ and $deg(v) \geq n$ for every vertex v. Prove that G has a perfect matching. (Hint: Prove that if M is a matching that is not perfect, then there exists an augmenting path of length 1 or 3.)

17. Suppose that M is a matching of G that is not contained in any larger matching, and that M' is a maximum matching of G. Prove that $|M'| \leq 2|M|$.

8.3 König's Theorem

The main result about matching in bipartite graphs is the following theorem of König.

Theorem 8.3.1. *(König's Theorem) In a bipartite graph the maximum size of a matching is the minimum size of a cover.*

Though minimum covers have the same size as maximum matchings in bipartite graphs, this is not true in general. For example, a minimum cover in K_p has size $p-1$ and a maximum matching has size $\lfloor \frac{p}{2} \rfloor$.

Let A, B be a bipartition of G, and let M be a matching of G. In the following **XY-construction** we use alternating paths to define sets X, Y, and show that they have certain properties. This will allow us to prove König's Theorem, and also to give an efficient algorithm to find a maximum matching.

Let X_0 be the set of vertices in A not saturated by M and let Z denote the set of vertices in G that are joined by to a vertex in X_0 by an alternating path. If $v \in Z$ we use $P(v)$ to denote an alternating path that joins v to X_0. Now define:

(a) $X = A \cap Z$.

(b) $Y = B \cap Z$.

For example, in the graph of Figure 8.2 we have

$$X_0 = \{b, d\}, \quad X = \{a, b, c, d, e, f\}, \quad Y = \{g, h, i, j, k, l\}.$$

As a second example, consider the same graph but with a different matching, shown in Figure 8.4. Then we have

$$X_0 = \{d\}, \quad X = \{d, b, f\}, \quad Y = \{i, l\}.$$

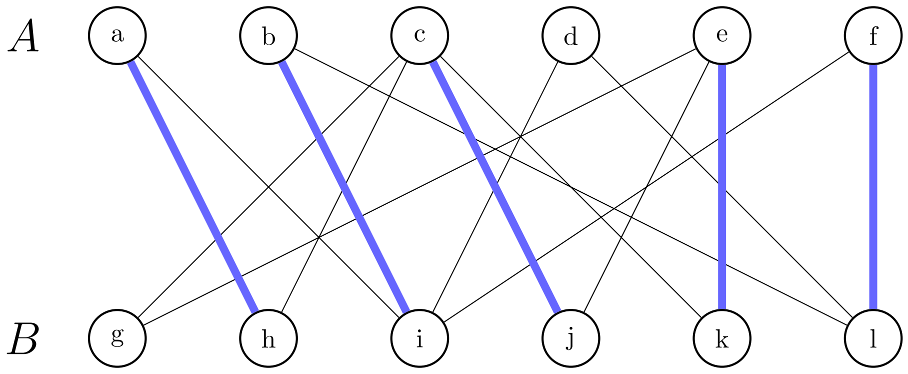


Figure 8.4: A second bipartite example

Notice that any alternating path $P(v)$ has even length if $v \in X$ and odd length if $v \in Y$. Since the first edge of any alternating path beginning at a vertex in X_0 is not a matching edge, and every second edge is a matching edge, it follows that

- If $v \in X$, then the last edge of $P(v)$ is in M (this is true vacuously if $v \in X_0$).
- If $v \in Y$, then the last edge of $P(v)$ is not in M.

One more easy observation: If w is a vertex of an alternating path $P(v)$ from X_0 to $v \in Z$, then $w \in Z$.

Lemma 8.3.2. *Let M be a matching of bipartite graph G with bipartition A, B, and let X and Y be as defined above. Then:*

(a) There is no edge of G from X to $B \setminus Y$;

(b) $C = Y \cup (A \setminus X)$ is a cover of G;

(c) There is no edge of M from Y to $A \setminus X$;

(d) $|M| = |C| - |U|$ where U is the set of unsaturated vertices in Y;

(e) There is an augmenting path to each vertex in U.

Proof: Suppose that (a) is false, and let $u \in X$, $v \in B \setminus Y$, $\{u, v\} \in E(G)$. Then adding v to the even-length alternating path $P(u)$ from X_0 to u, gives us an odd-length alternating path to v, which implies that $v \in Y$, a contradiction.

For (b), the only edges of G that are not incident with an element of C are those from X to $B \setminus Y$. However, no such edges exist by part (a), so C is a cover.

Now suppose that (c) is false, and let $u \in Y$, $v \in A \setminus X$, $\{u,v\} \in M$. Then adding v to the odd-length alternating path $P(u)$ from X_0 to u, gives us an even-length alternating path to v, which implies that $v \in X$, a contradiction.

For (d), from part (c), every edge of M joins a vertex in Y to a vertex in X, or joins a vertex in $A \setminus X$ to a vertex in $B \setminus Y$. The number of edges of the first type is $|Y| - |U|$. Also, by the fact that $X_0 \subseteq X$, every vertex in $A \setminus X$ is saturated by M, so the number of edges of the second type is $|A \setminus X|$. It follows that

$$|M| = |Y| - |U| + |A \setminus X| = |C| - |U|.$$

Finally, (e) is easy—if such a vertex v exists, then $P(v)$ is an augmenting path. ∎

We can check that the conclusions of Lemma 8.3.2 are satisfied in the two examples above. Notice in particular, that in Figure 8.2 we have $U = \{g, j\}$, and, for example, augmenting paths to g and j are given by $bicg$ and $bicj$. If we use the latter path to get a larger matching, then we get the matching of Figure 8.4. In Figure 8.4, there are no unsaturated vertices in Y, so $U = \emptyset$.

Proof of Theorem 8.3.1: Let M be a maximum matching of G. Then from Lemma 8.1.1 and part (e) of Lemma 8.3.2, U must be the empty set, so $|U| = 0$. Therefore, from parts (b) and (d) of Lemma 8.3.2, $C = Y \cup (A \setminus X)$ is a cover of G with $|C| = |M|$, and the result follows immediately from Lemma 8.2.2. ∎

Notice in the example of Figure 8.4 that there is no unsaturated vertex in Y. The construction of the proof of König's Theorem then gives the cover $C = Y \cup (A \setminus X) = \{a, c, e, i, l\}$. It has size 5, and therefore shows that the matching shown there is maximum.

Problem 8.3.3. *Let G be a bipartite graph with bipartition A, B, where $|A| = |B| = n$. Prove that if G has q edges, then G has a matching of size at least q/n.*

Solution: Notice that what is needed, is to prove that the maximum size of a matching is at least q/n. By König's Theorem, it is enough to show that the minimum size of a cover is at least q/n. Suppose that C is a cover of G. There can be at most n edges incident to any element of C, so there can be at most $n|C|$ edges incident with one or more elements of C. But C is a cover, so *every* edge must be incident with one or more elements of C. Therefore, $n|C| \geq q$, or $|C| \geq q/n$. Since every cover contains at least q/n vertices, therefore, the minimum size of a cover is at least q/n, and we are done. ∎

An algorithm for maximum matching in bipartite graphs

The XY-construction and Lemma 8.3.2 essentially provides an algorithm to find a maximum matching:

Step 1. Begin with any matching M.
Step 2. Construct X and Y.
Step 3. If there is an unsaturated vertex v in Y, find an augmenting path $P(v)$ ending at v, use it to construct a larger matching M', and replace M by M'. Then go to Step 2.
Step 4. If every vertex in Y is saturated, stop. M is a maximum matching, and $C = Y \cup (A \setminus X)$ is a cover of minimum size.

Now we give a more explicit way to construct X and Y and, at the same time, the alternating paths $P(v)$ that define them. It is very much like breadth-first search—we find the elements of X, Y in levels. We find first the vertices that are reachable by alternating paths beginning in X_0 and having 0 edges (this is just X_0, of course), then those that are reachable by alternating paths having 1 edge, then those reachable by alternating paths having 2 edges, et cetera. The vertices v in the next level are exactly those (whose level has not yet been assigned) that are joined by an edge to a vertex u in the previous level. The only additional rule is that, when we are creating an even level, that edge $\{u, v\}$ must be a matching edge. Just as in breadth-first search, when a vertex v is recognized as an element of X or of Y, we assign it a parent $\mathrm{pr}(v)$, which is the vertex u in the previous level from which we reached v. When we find an unsaturated vertex v in Y, then we can trace the path $v\mathrm{pr}(v)\mathrm{pr}^2(v)\dots w$, where $w \in X_0$, and this is an augmenting path. There is no need to construct the rest of X and Y in this case—we can immediately use the augmenting path to get a bigger matching. Here is a statement of the resulting algorithm. In it, we use $\hat{X}$ to represent the set of elements of X that we have found so far, and similarly for $\hat{Y}$. Initially, $\hat{X} = X_0$ and $\hat{Y} = \emptyset$.

Bipartite matching algorithm

Step 0. Let M be any matching of G.
Step 1. Set $\hat{X} = \{v \in A : v \text{ is unsaturated}\}$, set $\hat{Y} = \emptyset$, and set $\mathrm{pr}(v)$ to be undefined for all $v \in V(G)$.
Step 2. For each vertex $v \in B \setminus \hat{Y}$ such that there is an edge $\{u, v\}$ with $u \in \hat{X}$, add v to $\hat{Y}$ and set $\mathrm{pr}(v) = u$.
Step 3. If Step 2 added no vertex to $\hat{Y}$, return the maximum matching M and the minimum cover $C = \hat{Y} \cup (A \setminus \hat{X})$, and stop.

Step 4. If Step 2 added an unsaturated vertex v to $\hat{Y}$, use pr values to trace an augmenting path from v to an unsaturated element of $\hat{X}$, use the path to produce a larger matching M', replace M by M', and go to Step 1.
Step 5. For each vertex $v \in A \setminus \hat{X}$ such that there is an edge $\{u, v\} \in M$ with $u \in \hat{Y}$, add v to $\hat{X}$ and set $\mathrm{pr}(v) = u$. Go to Step 2.

Here is an example of the application of the algorithm. Consider the graph of Figure 8.5 with the matching M indicated there. Take $A = \{1, 2, 3, 4\}$.

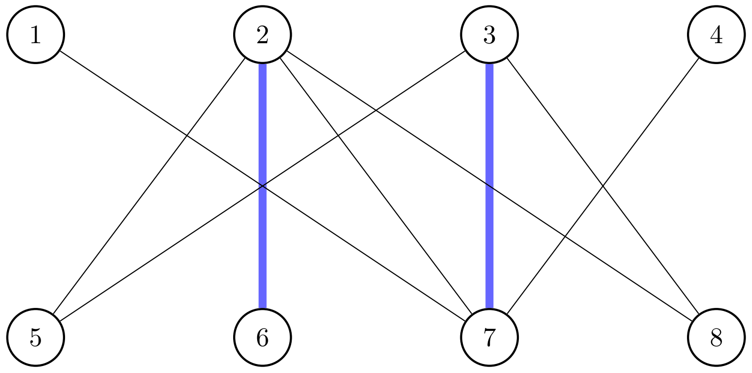


Figure 8.5: Example of the algorithm

Initially, $\hat{X} = \{1,4\}$ and $\hat{Y} = \emptyset$. Then 7 is added to $\hat{Y}$ with $\mathrm{pr}(7) = 1$. Then 3 is added to $\hat{X}$ with $\mathrm{pr}(3) = 7$. Then 5 and 8 are added to $\hat{Y}$ with $\mathrm{pr}(5) = \mathrm{pr}(8) = 3$. Since 5 (for example) is unsaturated we have the augmenting path 5371. This gives the new matching $M = \{\{2,6\}, \{3,5\}, \{1,7\}\}$.

Beginning again (with no parent values), we have $\hat{X} = \{4\}$, $\hat{Y} = \emptyset$. Then 7 is added to $\hat{Y}$ with $\mathrm{pr}(7) = 4$. Then 1 is added to $\hat{X}$ with $\mathrm{pr}(1) = 7$. Now nothing can be added to $\hat{Y}$ and so the algorithm terminates with the current matching M and the cover $C = (A \setminus \hat{X}) \cup \hat{Y} = \{2, 3, 7\}$.

Problem Set 8.3

1. Let G be a bipartite graph, and let Δ be the largest degree of any vertex of G. Prove that G has a matching of size at least q/Δ. Also, show that this is false in general if G is not bipartite.

2. Let n be a positive integer. Construct a bipartite graph with bipartition A, B, where $|A| = |B| = n$, for which the size of a maximum matching is less than

$(q+1)/n$. (In other words, show that the value q/n is the best possible in Problem 8.3.3.)

3. (Difficult) Let G be a bipartite graph with bipartition A, B, where $|A| = |B| = n$, and suppose that every vertex of G has degree at least $\delta < n$. Prove that G has a matching of size at least the minimum of n and $(q-\delta^2)/(n-\delta)$.

4. Construct a bipartite graph with bipartition A, B, where $|A| = |B| = 8$, and having minimum degree 2, for which the size of a maximum matching is less than $(q-3)/6$. (This shows that the value $(q-\delta^2)/(n-\delta)$ in the previous exercise, cannot be increased.)

5. Find a maximum matching and a minimum cover in the graph of Figure 8.6, by applying the algorithm, beginning with the matching indicated.

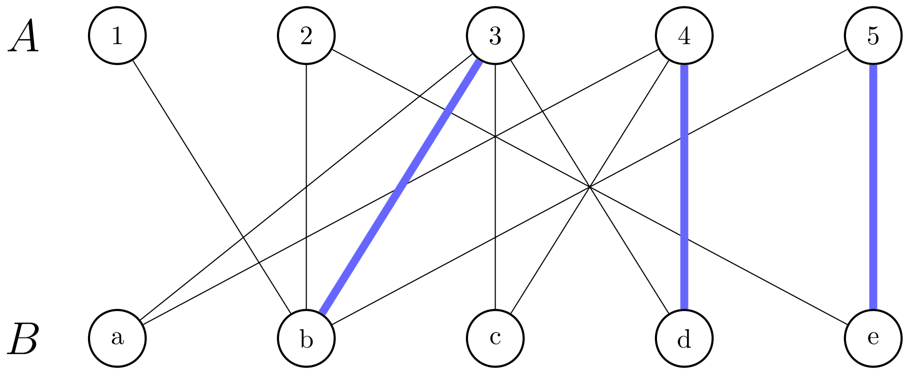


Figure 8.6: Matching exercise

6. Find a maximum matching and a minimum cover in the graph of Figure 8.7, by applying the algorithm, beginning with the matching indicated.

7. Find a maximum matching and a minimum cover in the graph of Figure 8.8, by applying the algorithm, beginning with the matching of size 18 consisting of all the edges oriented from northwest to southeast.

8. Let G be bipartite with bipartition A, B. Suppose that C and C' are both covers of G. Prove that $\hat{C} = (A \cap C \cap C') \cup (B \cap (C \cup C'))$ is also a cover of G.

9. In the previous exercise, prove that if C and C' are minimum covers, then so is $\hat{C}$.

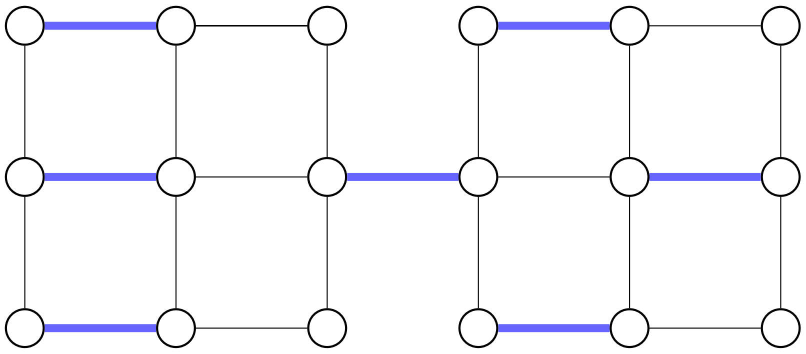


Figure 8.7: Matching exercise

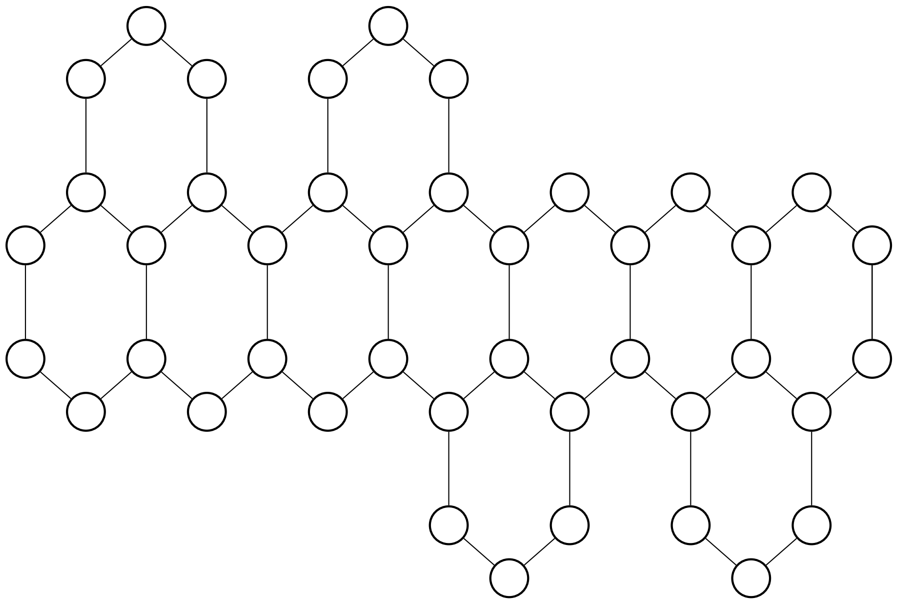


Figure 8.8: Matching exercise

8.4 Applications of König's Theorem

If A, B is a bipartition of G, no matching of G can have size bigger than $|A|$. It is interesting to determine whether there is one of exactly this size. (A perfect matching has this property, but if $|A| < |B|$, there will be no perfect matchings.) This problem was raised in a different context by Hall, and the resulting theorem (although it follows from König's Theorem) is called *Hall's Theorem*.

Consider the subset D of A. If its elements are to be saturated by a matching, there must be at least $|D|$ distinct elements of B that are adjacent in G to at least one element in D, since the matching edges incident to the elements of D must have distinct ends. For example, in Figure 8.4, the subset $D = \{a, b, d, f\}$ of A is joined by edges of the graph only to the three vertices h, i, l of B. It follows that there can be no matching saturating all of D, and therefore there can be no matching saturating all of A. Let us define, for any subset D of vertices of a graph G the **neighbour set** $N(D)$ of D to be $\{v \in V(G) :$ there exists $u \in D$ with $\{u, v\} \in E(G)\}$. Then if there is a matching saturating A, we must have at least $|D|$ elements in $N(D)$. Hall proved that, if every subset D satisfies this condition, then there will exist such a matching. As we indicated above, we can prove this using König's Theorem.

Theorem 8.4.1. *(Hall's Theorem) A bipartite graph G with bipartition A, B has a matching saturating every vertex in A, if and only if every subset D of A satisfies*

$$|N(D)| \geq |D|.$$

Proof: First, suppose that G has a matching M saturating every vertex in A. Then for any subset D of A, $N(D)$ contains the other end of the edge of M incident with v for every $v \in D$, and these vertices must all be distinct, so $|N(D)| \geq |D|$.

We prove the contrapositive of the "if" part of Hall's Theorem; namely, if there is no matching that saturates every vertex of A, then $|N(D)| < |D|$ for some subset $D \subseteq A$.

By hypothesis, there is no matching that saturates every vertex of A. Then, by König's Theorem, there exist a maximum matching M and a minimum covering C such that $|C| = |M| < |A|$. The sets A, B, C partition the vertices of G into 4 subsets $A \cap C, A \setminus C, B \cap C, B \setminus C$ as shown in Figure 8.9. (Remark: some of these subsets may be empty; for example, if $C = B$, then $B \setminus C = A \cap C = \emptyset$.) Since C is a cover, no edge joins a vertex of $B \setminus C$ to a vertex of $A \setminus C$, so $N(A \setminus C) \subseteq B \cap C$.

Let $D = A \setminus C$. Then

$$|N(D)| \le |B \cap C| = |C| - |A \cap C| < |A| - |A \cap C| = |A \setminus C| = |D|.$$

This completes the proof of the "if" part of Hall's Theorem ▌

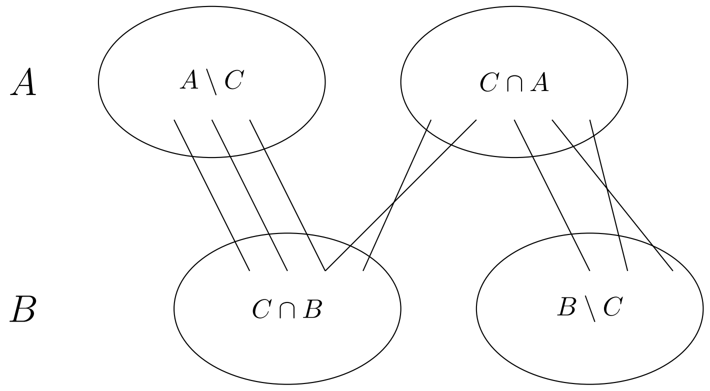


Figure 8.9: Proof of Hall's Theorem

Note that, if there is no matching saturating all vertices in A, the maximum matching algorithm will find a set $D \subseteq A$ such that $|N(D)| < |D|$. Namely, at termination of the algorithm, we take $D = A \setminus C$, where C is the cover $Y \cup (A \setminus X)$. Therefore $D = X$ and $N(D) = Y$.

8.5 Systems of Distinct Representatives

The problem that led Hall to prove Theorem 8.4.1 can be described as follows. Suppose there are several interest groups in some population, for example, bridge players, socialists, football fans, and so on. We want to find a representative for each of these groups, perhaps to serve on some decision-making body. There are two rules, first, that a representative must belong to the group she represents, and second, that no one can represent two different groups. More formally, given a collection $Q_1, Q_2, \ldots, Q_n$ of subsets of a finite set Q, a **system of distinct representatives (SDR)** for the collection is a sequence (or n-tuple) $(q_1, q_2, \ldots, q_n)$ of n distinct elements of Q such that

$$q_i \in Q_i \text{ for } i = 1, 2, \ldots, n.$$

For example, suppose that $Q_1 = \{b,d\}$, $Q_2 = \{a,b,c,d\}$, $Q_3 = \{b,c\}$, $Q_4 = \{b,d\}$. Then we can choose $q_1 = b$, $q_2 = a$, $q_3 = c$, $q_4 = d$. As a second example, suppose that $Q_1 = \{b,d\}$, $Q_2 = \{a,b,c,d\}$, $Q_3 = \{b\}$, $Q_4 = \{b,d\}$. Then it is not hard to convince ourselves that there is no SDR. One way to do so, is to observe that the three subsets Q_1, Q_3, Q_4 have only 2 elements (b and d) among them, and yet an SDR would have to have 3 different elements from those 3 subsets. More generally, if we have a subcollection of the given collection of subsets, that have among them fewer elements than the number of subsets of the subcollection, then there cannot exist an SDR. Hall proved (see Corollary 8.5.1 below) that this is the only thing that can go wrong, that is, that if no such nasty subcollection exists, then there does exist an SDR.

What do SDR's have to do with matching? Given the collection $Q_1, Q_2, \ldots, Q_n$ of subsets of Q, we can construct the bipartite graph with bipartition $A = \{1,2,\ldots,n\}$, $B = Q$ having the following adjacencies: vertices $k \in A$ and $b \in B$ are adjacent if and only if $b \in Q_k$. (For example, the graph associated with the first example above is shown in Figure 8.10.) Now, corresponding to each SDR $(q_1, q_2, \ldots, q_n)$, the edges $\{1, q_1\}$, $\{2, q_2\}, \ldots, \{n, q_n\}$ in G make up a matching saturating A. Conversely, for each matching of G saturating A, say $M = \{\{1,b_1\},\{2,b_2\},\ldots,\{n,b_n\}\}$, the sequence $(b_1, b_2, \ldots, b_n)$ is an SDR of $Q_1, Q_2, \ldots, Q_n$. In summary, the collection $Q_1, Q_2, \ldots, Q_n$ has an SDR if and only if graph G has a matching saturating every vertex in A.

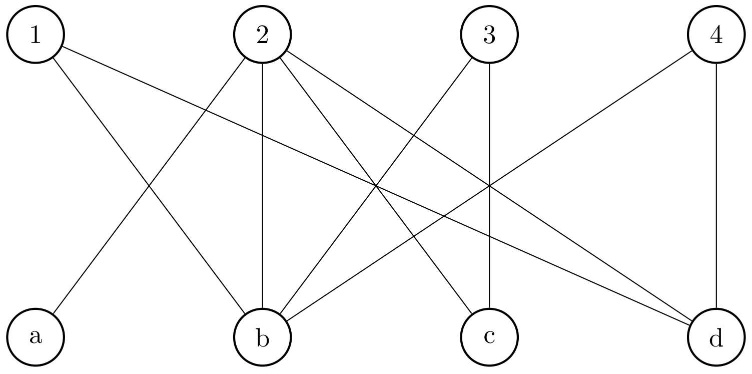


Figure 8.10: Graph constructed from first SDR example

Moreover, the condition described above for an SDR to exist, corresponds exactly to the requirement that in the graph, every subset D of A have $|N(D)| \geq |D|$. Thus the following theorem is nothing but a restatement of Theorem 8.4.1 in the language of SDR's.

Corollary 8.5.1. *(Hall's SDR Theorem) The collection $Q_1, Q_2, \ldots, Q_n$ of subsets of the finite set Q has an SDR if and only if, for every subset J of $\{1, 2, \ldots, n\}$, we have*

$$\left|\bigcup_{i\in J} Q_i\right| \geq |J|.$$

8.6 Perfect Matchings in Bipartite Graphs

We can use Hall's Theorem to obtain a condition for a bipartite graph to have a perfect matching.

Corollary 8.6.1. *A bipartite graph G with bipartition A, B has a perfect matching if and only if $|A| = |B|$ and every subset D of A satisfies*

$$|N(D)| \geq |D|.$$

Proof: Clearly, if $|A| \neq |B|$, then G has no perfect matching. On the other hand, if $|A| = |B|$, then G has a perfect matching if and only if it has a matching saturating every vertex in A, and then the result follows from Hall's Theorem. ▮

Another application is to regular bipartite graphs. We can show that these always have perfect matchings, since it is easy to show that they always satisfy the condition of Corollary 8.6.1. This result will be used in the next section, when we discuss edge-colouring.

Theorem 8.6.2. *If G is a k-regular bipartite graph with $k \geq 1$, then G has a perfect matching.*

Proof: Let A, B be a bipartition of G. Then since every edge has one end in A and the other in B, we have $\sum_{v\in A} deg(v) = \sum_{v\in B} deg(v)$. It follows that $k|A| = k|B|$, and therefore, since $k > 0$, that $|A| = |B|$. Now let $D \subseteq A$. Then since every edge incident with a vertex in D has its other end in $N(D)$, we have

$$\sum_{v\in D} deg(v) \leq \sum_{v\in N(D)} deg(v).$$

It follows that $k|D| \leq k|N(D)|$, and therefore (again, since $k > 0$) that $|N(D)| \geq |D|$. Now by Corollary 8.6.1, G has a perfect matching. ▮

Note: Theorem 8.6.2 works even if G contains multiple edges.

Problem Set 8.6

1. For a bipartition A, B of the graph of Problem Set 8.3, #6, find a set $D \subseteq A$ such that $|N(D)| < |D|$.

2. In the graph of Problem Set 8.3, #7, find a set D of vertices such that $|N(D)| < |D|$.

3. Let G be a bipartite graph with bipartition A, B, let M be a matching of G, and let $D \subseteq A$. Prove that $|M| \leq |A| - |D| + |N(D)|$.

4. Let G be a bipartite graph with bipartition A, B. Prove that the maximum size of a matching of G is the minimum, over subsets D of A, of $|A| - |D| + |N(D)|$.

5. Let G be a bipartite graph with bipartition A, B such that $|A| = |B|$, and for every proper nonempty subset D of A, we have $|N(D)| > |D|$. Prove that for every edge $e \in E(G)$ there is a perfect matching containing e.

6. Check that the proof of Theorem 8.6.2 works even for bipartite multigraphs. (Note that a bipartite multigraph can have multiple edges, but is not allowed to have loops.)

7. A deck of playing cards is arranged in a rectangular array of four rows and thirteen columns. Prove that there exist thirteen cards, no two in the same column and no two of the same value.

8. Find a 3-regular graph having no perfect matching. (Such a graph must be nonbipartite.)

9. Show how to find in a bipartite graph, a largest set of mutually nonadjacent vertices.

10. Let N be a matrix. Let us define the *size* of a submatrix N' of N to be the number of rows of N' plus the number of columns of N'. How could one find a maximum size submatrix of N with the property that each of its entries is 0?

11. Prove Theorem 8.6.2 directly from König's Theorem.

12. Let G be a bipartite graph with bipartition A, B where $|A| = |B| = 2n$. Suppose that, for every subset $X \subseteq A$ and every subset $X \subseteq B$ such that $|X| \leq n$, $|N(X)| \geq |X|$. Prove that G has a perfect matching.

8.7 Edge-colouring

An **edge k-colouring** of a graph G is an assignment of one of a set of k colours to each edge of G so that two edges incident with the same vertex are assigned different colours. Consider the set of edges M that are assigned a particular colour. Then for any vertex v, there can be at most one edge of M incident with v. That is, M is a matching, and so an edge k-colouring of G amounts to a partitioning of the edges of G into k matchings. Figure 8.11 shows an edge 3-colouring of a graph, where the colours are indicated by different ways of drawing the edges.

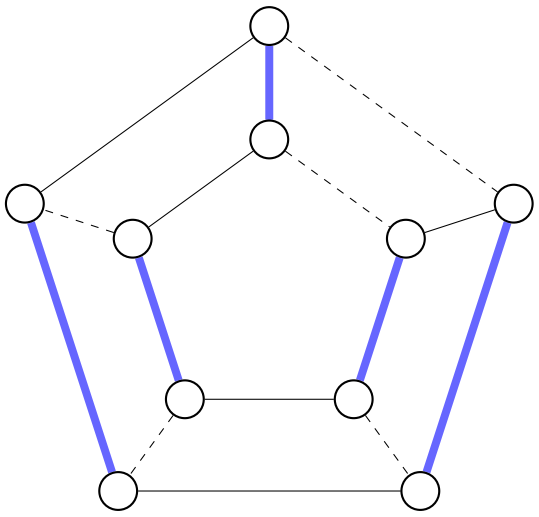


Figure 8.11: A graph with an edge 3-colouring

Obviously, we need at least $deg(v)$ different colours at each vertex v. Therefore, for G to have an edge k-colouring, we must have $k \geq \Delta$, where Δ denotes the maximum degree of vertices of G. The main result of this section is that for bipartite graphs this bound can be achieved.

Theorem 8.7.1. *A bipartite graph with maximum degree Δ has an edge Δ-colouring.*

Lemma 8.7.2. *Let G be a bipartite graph having at least one edge. Then G has a matching saturating each vertex of maximum degree.*

(Note that, in the special case of regular bipartite graphs, all of the vertices have maximum degree, and the lemma says that there must be a perfect matching. In fact, we already have seen this special case—it is Theorem 8.6.2.)

Proof: Let A, B be a bipartition of G and $K = \{v \in V : deg(v) = \Delta\}$. Let M be a maximum matching that leaves as few elements of K unsaturated as possible. If M saturates all elements of K, we are finished. Otherwise, we may assume, by interchanging A with B if necessary, that there is a vertex in $A \cap K$ that is not saturated by M. Apply the XY-construction, except that we define X_0 to consist only of the unsaturated vertices in $A \cap K$. Then Y contains no unsaturated vertex, since M is a maximum matching.

Suppose there is a vertex $w \in X$ having degree less than Δ. Consider the alternating path $P(w)$. It has even length. If we replace the edges of M that are in $P(w)$ by those that are not, we get another maximum matching, but one that leaves fewer elements of K unsaturated, a contradiction to the choice of M. So every vertex in X has degree Δ.

By the construction and the fact that M is maximum, for every $u \in Y$ there is an edge of M joining u to some vertex in X. Since X contains at least one unsaturated vertex, $|X| > |Y|$. Moreover, again by the construction, there is no edge from X to $B \setminus Y$, so $N(X) \subseteq Y$. Therefore,

$$\Delta|Y| < \Delta|X| = \sum_{v \in X} deg(v) \le \sum_{v \in Y} deg(v) \le \Delta|Y|,$$

a contradiction. Therefore, there are no vertices in K not saturated by M, and we are done. ▌

Proof of Theorem 8.7.1: The result can be proved by induction on Δ. First, if $\Delta = 0$, then G has no edges, and has an edge 0-colouring. Now suppose $m \ge 1$ and assume that every bipartite graph having $\Delta < m$ has an edge Δ-colouring. Let G be a bipartite graph with $\Delta = m$. By Lemma 8.7.2, G has a matching M saturating all the vertices of degree m. Delete M from G to obtain the graph H. Then H is bipartite and has $\Delta = m-1$, and so by the induction hypothesis, it has an edge $(m-1)$-colouring. If we use this colouring on G, and use one additional colour for the edges of M, then we have an edge m-colouring of G. The result is proved by induction. ▌

Note that the proof of Lemma 8.7.2 suggests a modification of the matching algorithm which will efficiently find a matching saturating all vertices of maximum degree. Therefore, since the proof of Theorem 8.7.1 just requires us to find and delete such a matching and repeat, there is an efficient algorithm for finding an edge Δ-colouring of a given bipartite graph.

8.8 An Application to Timetabling

We have sets of instructors and courses, and for each instructor, the list of courses she teaches. We need to schedule the courses into as few timetable slots as possible, so that no instructor is required to be in two places at the same time. We have the additional requirement that two sections of the same course cannot be given in the same slot. (This may be because we want courses to be available at several different times.)

Let us construct the bipartite graph G having a vertex for each instructor and a vertex for each course, with an edge joining a course vertex to an instructor-vertex if that instructor teaches that course. (If an instructor teaches more than one section of the same course, this would be a multigraph.) The set of edges of the graph corresponds to all of the classes that must be scheduled. Now suppose that we have arranged a schedule, and consider the set of classes taught in a particular slot. These cannot involve any instructor more than once, nor can they involve the same course more than once. That is, they correspond to a matching of G. If we take the slots to correspond to colours, then our problem is one of colouring the edges of G with the minimum number of colours.

Since G is bipartite, we know from Theorem 8.7.1 that the minimum number of slots needed is the maximum degree of the graph G constructed from the data. This is the largest number of sections of a single course, or the largest number of classes taught by a single instructor, whichever is larger. We also know how to solve the problem algorithmically, and actually find a schedule that achieves this minimum.

But suppose now that there is also a classroom limitation, so that at most m classes can be assigned to the same slot. Then of course we cannot find a schedule having fewer than q/m slots, where q is the total number of classes to be scheduled (and is the number of edges of the graph G above). Our problem now becomes:

Bounded edge-colouring problem: What is the smallest number of colours needed to edge-colour a bipartite graph G, given that no colour can be assigned to more than m edges?

The answer turns out to be the smallest integer that is at least q/m and also is at least Δ. This follows from the next result.

Theorem 8.8.1. *Let G be a graph with q edges, and suppose k, m are positive integers such that* *(a) G has an edge k-colouring;*

(b) $q \le km$.

Then G has an edge k-colouring in which every colour is used at most m times.

Proof: Suppose that the colouring does not already have the desired property, so that some colour, say red, is used at least $m+1$ times. If every other colour is used at least m times, then

$$q \ge m+1+m(k-1) > km,$$

a contradiction. So there exists a second colour, say blue, that is used at most $m-1$ times.

Now the red edges and the blue edges form disjoint matchings M and M' of G. Consider the spanning subgraph H of G having edgeset $M \cup M'$. Then every vertex of this graph has degree 0 or 1 or 2. Therefore, each component of H consists of a path or a cycle with edges alternately in M and M'. Moreover, any cycle is even. Since $|M| > |M'|$, there must exist a component of H containing more edges of M than of M'. Such a component must consist of an augmenting path for M'. If we use that path to make M' larger by one and to make M smaller by one, then we have a new edge k-colouring of G such that red is used fewer times, and blue is still used at most m times. We can repeat this argument until we find a colouring with the desired property. ▌

Now we can solve the bounded edge-colouring problem for bipartite graphs by combining Theorems 8.7.1 and 8.8.1.

Corollary 8.8.2. *In a bipartite graph G, there is an edge k-colouring in which each colour is used at most m times if and only if*

(a) $\Delta \le k$, and

(b) $q \le km$.

Problem Set 8.8

1. Show that the number of colours required to edge-colour a nonbipartite graph G can exceed Δ.

2. Prove that every subgraph of an edge k-colourable graph is edge k-colourable.

3. Prove that a k-regular bipartite graph has an edge k-colouring, using Theorem 8.6.2.

4. Show that the Petersen graph cannot be edge 3-coloured.

5. Show that Theorem 8.8.1 does not work for vertex-colouring. That is, show that if a graph can be k-coloured and $mk \geq p$, it is not necessarily true that there is a k-colouring in which each colour is used at most m times.

6. An n by n *permutation matrix* is a matrix having one 1 and $n-1$ 0's in every row and in every column. Let N be an n by n matrix such that every row and every column contains k 1's and $n-k$ 0's. Prove that N is the sum of k permutation matrices.

7. State an algorithm for finding in a bipartite graph a matching saturating all vertices of maximum degree.

Index